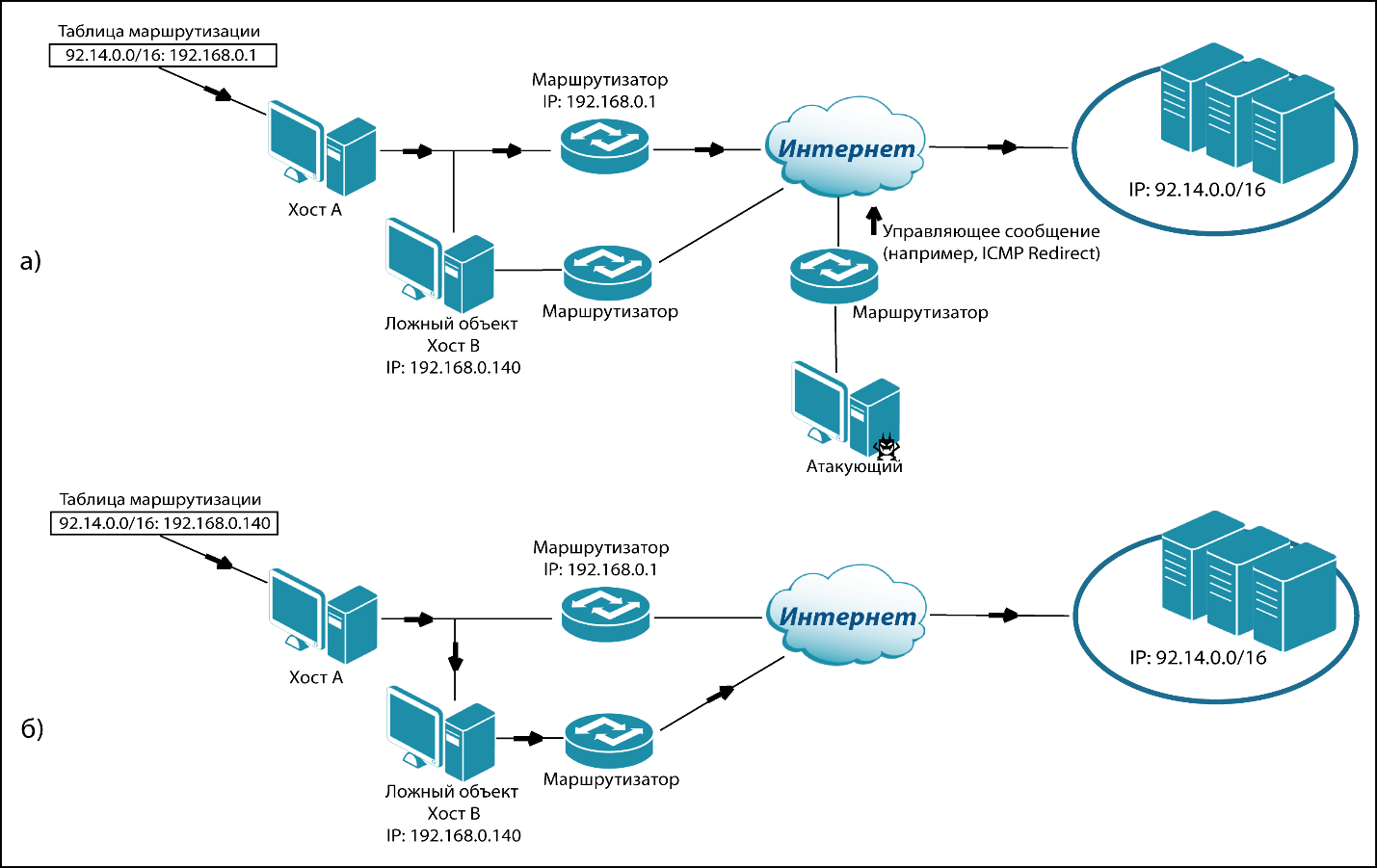
**Анализатор трафика, или сниффер (от англ. to sniff — нюхать)** — программа или устройство для перехвата и анализа сетевого трафика (своего и/или чужого).

Принцип работы



Сниффер может анализировать только то, что проходит через его сетевую карту. Внутри одного сегмента сети Ethernet все пакеты рассылаются всем машинам, из-за этого возможно перехватывать чужую информацию. Использование коммутаторов (switch, switch-hub) и их грамотная конфигурация уже является защитой от прослушивания. Между сегментами информация передаётся через коммутаторы. Коммутация пакетов — форма передачи, при которой данные, разбитые на отдельные пакеты, могут пересылаться из исходного пункта в пункт назначения разными маршрутами. Так что если кто-то в другом сегменте посылает внутри него какие-либо пакеты, то в ваш сегмент коммутатор эти данные не отправит.

Перехват трафика может осуществляться:

обычным «прослушиванием» сетевого интерфейса (метод эффективен при использовании в сегменте концентраторов (хабов) вместо коммутаторов (свитчей), в противном случае метод малоэффективен, поскольку на сниффер попадают лишь отдельные фреймы);

подключением сниффера в разрыв канала;

ответвлением (программным или аппаратным) трафика и направлением его копии на сниффер (Network tap);

через анализ побочных электромагнитных излучений и восстановление таким образом прослушиваемого трафика;

через атаку на канальном (2) (MAC-spoofing) или сетевом (3) уровне (IP-spoofing), приводящую к перенаправлению трафика жертвы или всего трафика сегмента на сниффер с последующим возвращением трафика в надлежащий адрес.

Применение

В начале 1990-х широко применялся хакерами для захвата пользовательских логинов и паролей, которые в ряде сетевых протоколов передаются в незашифрованном или слабозашифрованном виде. Широкое распространение хабов позволяло захватывать трафик без больших усилий в больших сегментах сети практически без риска быть обнаруженным.

Снифферы применяются как в деструктивных, так и в благих целях. Анализ прошедшего через сниффер трафика позволяет:

Обнаружить паразитный, вирусный и закольцованный трафик, наличие которого увеличивает загрузку сетевого оборудования и каналов связи (снифферы здесь малоэффективны; как правило, для этих целей используют сбор разнообразной статистики серверами и активным сетевым оборудованием и её последующий анализ).

Выявить в сети вредоносное и несанкционированное ПО, например, сетевые сканеры, флудеры, троянские программы, клиенты пиринговых сетей и другие (это обычно делают при помощи специализированных снифферов — мониторов сетевой активности).

Перехватить любой незашифрованный (а порой и зашифрованный) пользовательский трафик с целью получения паролей и другой информации.

Локализовать неисправность сети или ошибку конфигурации сетевых агентов (для этой цели снифферы часто применяются системными администраторами)

Поскольку в «классическом» сниффере анализ трафика происходит вручную, с применением лишь простейших средств автоматизации (анализ протоколов, восстановление TCP-потока), то он подходит для анализа лишь небольших его объёмов.

Противодействие

Снизить угрозу сниффинга пакетов можно с помощью таких средств, как:

Криптография

Антиснифферы

Коммутируемая инфраструктура

**Зачем это делать ?**

Пример 1. **Анализ трафика.**  
Пользователи вашей сети пользуются вашим прокси-сервером. Вы можете увидеть на какие сайты заходят пользователи, запрещать дальнейшие переходы на эти сайты.

Пример 2. **Сбор данных.**  
Ваши пользователи пользуются через вас некоторыми веб-ресурсами. Например, они вводят vin-номер своего автомобиля на сайте дилера авто и получают в ответ данные этого автомобиля. Вы можете сохранять эти данные в свою базу данных.

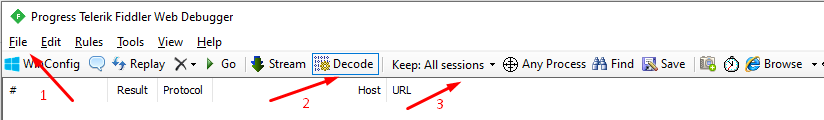
Пример 3. **Подмена HTTP-пакетов.**  
Вам нужно изменить для ваших пользователей внешний вид сайта. Вы можете изменить стили сайта, скрывать любые элементы, добавить свои элементы, вырезать определенные слова или заменить их на другие слова, изменить картинку сайта на любую свою.

Пример 4. **Подмена POST-данных.**  
Вам нужно подправить данные передаваемые на веб-сервер через POST-запрос. Существует множество информации передаваемой в POST-запросах. Пример: отправка логина/пароля на сервер в процессе авторизации. Или онлайн тест отправляет на сервер результаты вашего теста.

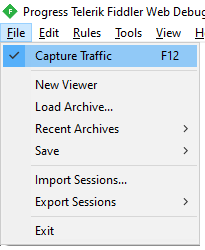
**Установка Fiddler**

1. Переходим на [https://www.telerik.com/download/fiddler](http://personeltest.ru/aways/www.telerik.com/download/fiddler), скачиваем Fiddler Classic.
2. Установка простая и быстрая.
3. Запускаем программу.

**Настройка Fiddler**



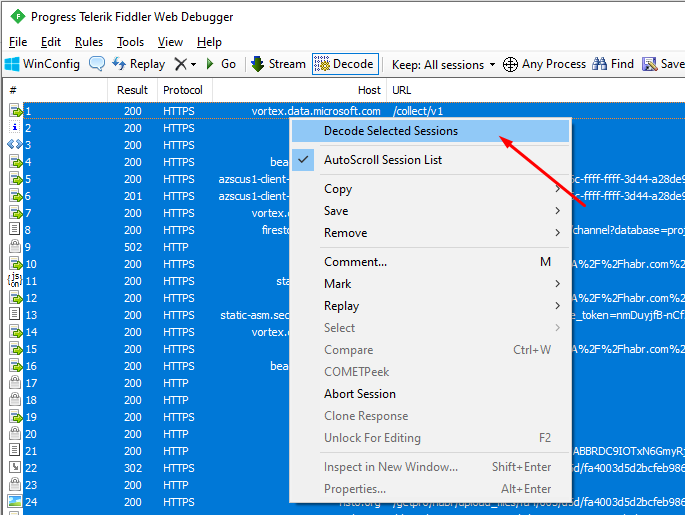
В меню File есть опция "*Capture Traffic*". По умолчанию опция включена. Это означает что Fiddler прописывает в реестре Windows себя в качестве прокси-сервера. Браузеры Internet Explorer, Edge, Chrome используют данную настройку, а значит HTTP-пакеты от этих браузеров пойдут через Fiddler.



Если опция "*File -> Capture Traffic*" выключена, то Fiddler перестает работать как системный прокси-сервер и перехватывает только те пакеты, которые идут непосредственно на адрес Fiddler. Это может быть когда вы настроили ваше приложение или браузер сами для работы через ip/port Fiddler. По умолчанию Fiddler слушает на порту 127.0.0.1:8888

Опция "*Keep: All sessions*".  
В данном режиме Fiddler не очищает журнал собранных HTTP-пакетов. Если требуется продолжительная работа Fiddler, то при большой нагрузке этих пакетов будет очень много и Fiddler скушает всю доступную оперативную память компьютера. Чтобы этого не случилось переключите в режим "Keep: 100 sessions".

Опция "*Decode*".  
По умолчанию выключена. В процессе анализа собранных пакетов рекомендуется включить чтобы пакеты автоматически декодировались. Либо можно выделить собранные пакеты через Ctrl+A, вызвать меню нажатием правой кнопки мыши по выделенным пакетам и нажать "Decode Selected Sessions".



**Основные настройки**

Переходим в "*Tools -> Options...*".  
  
**Вкладка "HTTPS".**  
После установки Fiddler не собирает HTTPS-трафик, это необходимо включить. Ставим галочку в опции "*Decrypt HTTPS traffic*". После этого Fiddler сгенерирует самоподписанный сертификат и спросит хотите ли установить данный сертификат. Отвечаем да.  
  
Опция "*Ignore server certificate errors (unsafe)*" - сразу можно не включать. На некоторых порталах бывают ошибки сертификатов, но это редко. Как увидите так включите )  
Настройка протоколов. По умолчанию стоит значение "<client>;ssl3;tls1.0". Советую сразу установить значение на "<client>;ssl3;tls1.0;tls1.1;tls1.2". После изменения настроек необходимо перезапустить программу чтобы настройки вступили в силу.

Изображение выглядит как текст

Автоматически созданное описание

Кнопка "*Actions*":

Изображение выглядит как текст

Автоматически созданное описание

"*Trust Root Certificate*" - если сгенерированный Fiddler сертификат вы не установили после включения опции "Decrypt HTTPS traffic", то можно это сделать здесь.

"*Export Root Certificate to Desktop*" - если вы планируете использовать Fiddler как прокси-сервер локальной сети, то на каждом устройстве пользователя необходимо установить сгенерированный выше сертификат. С помощью этой опции сохраняете сертификат на ваш рабочий стол.

"*Reset All Certificates*" - в некоторых случаях необходимо сгенерировать новый сертификат взамен старого. В этом случае сбрасываем все Fiddler-сертификаты и генерируем новый сертификат.

**Вкладка "Connections".**  
Здесь устанавливаем на каком порту Fiddler работает как прокси-сервер. Порт по умолчанию "8888".

"*Allow remote computers to connect*" - включаем опцию чтобы Fiddler начал принимать подключения от других компьютеров.

"*Act as system proxy on startup*" - по умолчанию опция включена. Если включена, то при запуске опция "File -> Capture Traffic" включена.

Изображение выглядит как текст

Автоматически созданное описание

После изменения данных настроек необходимо перезапустить программу чтобы настройки вступили в силу.

**Вкладка "Gateway".**Здесь устанавливаем куда Fiddler отправляет входящие пакеты, какой прокси использует.

"*Use System Proxy (recommended)*" - использование системного прокси из реестра текущего пользователя.

"*Manual Proxy Configuration*" - возможность задать вручную прокси-сервер.

"*No proxy*" - задаем что выход в Интернет напрямую, без использования прокси.

После изменения данных настроек необходимо перезапустить программу чтобы настройки вступили в силу.

Изображение выглядит как текст

Автоматически созданное описание

**Установка сертификатов на Windows устройствах**

После того как сгенерированный сертификат скопирован на рабочий стол этот сертификат необходимо установить на каждое устройство которое будет использовать данный Fiddler в качестве прокси-сервера.

Для установки сертификата используем консоль управления MMC: в коммандной строке вводим команду "mmc".

В меню файл выбираем "*Добавить или удалить оснастку*". Из доступных оснасток выбираем "*Сертификаты*" и с помощью кнопки "*Добавить*" выбираем данную оснастку. Нажимаем "Ок" и выбираем "*учетной записи компьютера*". Это нужно чтобы открыть сертификаты которые установлены для всего компьютера, а затем установить сертификат Fiddler именно в это хранилище. Если открыть сертификаты "*моей учетной записи пользователя*", то после установки сертификата Fiddler в это хранилище другие пользователи данного компьютера не смогут подключиться к Fiddler.

Изображение выглядит как текст

Автоматически созданное описание

Установку сертификата производим в "Доверенные корневые центры сертификации".

Изображение выглядит как текст

Автоматически созданное описание

Если ваши компьютеры находятся в домене, то используйте инструменты домена для установки сертификата каждому пользователю или на каждый компьютер сети.

**Анализ трафика**

В процессе работы Fiddler сниффит все HTTP-запросы и их обычно много. Для поиска необходимых запросов можно использовать фильтры. Правой кнопкой мыши выбираем лишний запрос, выбираем "*Filter Now*" и "*Hide '...'*" чтобы скрыть запросы к данному домену. Можно удалять вручную выделенные запросы используя кнопку "*Delete*".

Изображение выглядит как текст

Автоматически созданное описание

Кроме использования фильтров можно искать отдельный текст в теле запросов/ответов: "*Ctrl+F*" для открытия меню поиска. Найденные запросы подсвечиваются по умолчанию желтым цветом.

Изображение выглядит как текст

Автоматически созданное описание

**Изменение данных запросов**

В Fiddler существует инструмент "*Fiddler ScriptEditor*" (Редактор скриптов) для создания правил модификации трафика. Запуск редактора скриптов через "*Ctrl+R*" или выбора пункта меню "*Rules -> Customize Rules...*".

В редакторе скриптов есть два основных метода: "*OnBeforeRequest*" и "*OnBeforeResponse*":

"*OnBeforeRequest*" - выполнение скриптов в этом методе происходит перед отправкой пакетов на веб-сервер.

"*OnBeforeResponse*" - выполнение скриптов в этом методе происходит после получения ответа от веб-сервера.

Ниже приведены примеры скриптов с указанием в каком методе их расположить.

**Задача 1: Запрет сайта**

Запрещаем переход на адрес сайта содержащий строку.

// OnBeforeRequestif (oSession.uriContains("//ya.ru/")) oSession.host = 'access.denied';

**Задача 2: Запрет загрузки ресурса**

Запрещаем загрузку "*.svg*" файлов для заданного адреса сайта.

// OnBeforeRequestif (oSession.uriContains("yastatic.net") && oSession.url.EndsWith(".svg")){oSession.host = 'na.derevnu.dedushke';}// или// OnBeforeRequestif (oSession.uriContains("yastatic.net") && oSession.url.EndsWith(".svg")){oSession.responseBodyBytes = new byte[0];}// OnBeforeResponseif (oSession.uriContains("yastatic.net") && oSession.url.EndsWith(".svg")){oSession.ResponseBody = null;}

**Задача 3: Переадресация запроса**

Переадресация запроса на адрес сайта содержащий строку.

// OnBeforeRequestif (oSession.uriContains("//ya.ru/")) oSession.url = "yandex.ru"

**Задача 4: Сбор данных**

Пользователи подключаются через данный прокси-сервер и делают в браузерах некоторые запросы вида "*https://myhost.ru?key=abcd&vin=VF38BLFXE81078232&lang=ru*". Задача записать в базу данных событие поиска и передать значение vin-номера. Данный скрипт создает файлы с названием включающем vin-номер. Кроме скрипта необходимо создать утилиту/службу, которая раз в заданный интервал читает каталог "*C:\vinsearch\*" и записывает данные в базу данных.

// OnBeforeResponseif(oSession.uriContains("https://myhost.ru?key=") && oSession.uriContains("&vin=")){oSession.utilDecodeResponse();// поиск позиции индекса начала vin-номераvar startVin = oSession.url.IndexOf("vin=") + 4;// поиск позиции индекса конца vin-номераvar endVin = oSession.url.IndexOf("&", startVin); // поиск подстроки зная индекс начала и индекс конца подстрокиvar vin = oSession.url.Substring(startVin, endVin - startVin);// создание файла с именем типа vin\_текущееЗначениеМиллисекунд.txtoSession.SaveResponseBody("C:\\vinsearch\\" + vin + "\_" + DateTime.Now.Millisecond + ".txt");}

**Задача 5: Изменить текст в ответе**

В данном примере меняем текст "*Иванов*" на "*Петров*".

// OnBeforeResponseif (oSession.uriContains("https://myhost.ru")){oSession.utilDecodeResponse(); oSession.utilReplaceInResponse('Иванов','Петров');}

**Задача 6: Заменить ресурс веб-портала на локальный ресурс**

Заменим картинку веб-портала на картинку расположенною на локальном диске.

// OnBeforeResponseif (oSession.uriContains("/Static/app/img/world.svg")){oSession.LoadResponseFromFile("c:/scripts/lang.png");}

**Задача 7: Изменение свойств HTML-объектов**

Например, есть картинка с заданными размерами в HTML и нужно эти размеры изменить.

// OnBeforeResponseoSession.utilReplaceInResponse('/Static/app/img/world.svg" height="15" width="15" style="height: 15px','/Static/app/img/world.svg" height="1" width="1" style="height: 1px');

**Задача 8: Скрыть элементы по className меняя css-файлы**

В данном примере скрываем элементы зная их className в css-файле добавляя свойство "*visibility: hidden;*"

// OnBeforeResponseoSession.utilDecodeResponse();oSession.utilReplaceInResponse("#header\_Area\_Right {", "#header\_Area\_Right { visibility: hidden; ");

**Задача 9: Заставить страницу открываться в текущем окне**

Пример: существует JavaScript, который открывает ссылку в новом окне. Нужно сделать чтобы ссылка открывалась в текущем окне.

// OnBeforeResponseif (oSession.uriContains("myhost.ru") && oSession.uriContains(".js")){oSession.utilDecodeResponse();oSession.utilReplaceInResponse("window.open(url, '\_blank', option);", "window.open(url);");}

**Задача 10: Выполнение скриптов для определенных IP**

В данном примере меняем текст "*Иванов*" на "*Петров*" только для IP = "*192.168.0.100*"

// OnBeforeResponseif (oSession.clientIP == '192.168.0.100'){oSession.utilDecodeResponse();oSession.utilReplaceInResponse('Иванов','Петров');}

**Задача 11: Меняем css-стили портала**

Css-файлы веб-портала можно сохранить на локальном диске, отредактировать и настроить скрипт отдавать стили с локального диска, а не с портала.

// OnBeforeResponseif (oSession.uriContains("/banner.css")){oSession.LoadResponseFromFile("c:/scripts/banner.css");}

**Задача 12: Запрет PUT-команды и аналогичных**

Запрет команды по ее типу: "*PUT*", "*DELETE*", etc.

// OnBeforeRequestif (oSession.HTTPMethodIs("PUT") && oSession.uriContains("https://myhost.ru/")){oSession.host = 'access.denied';}

**Задача 13: Изменение тела POST-запроса**

Изменить тело POST-запроса для заданного портала. При авторизации на данном портале вне зависимости от введенных пользователем данных на веб-портал отправятся данные из скрипта.

// OnBeforeRequestif (oSession.uriContains("https://myhost.ru/") && oSession.RequestMethod == "POST"){oSession.utilSetRequestBody("username=xxx&password=yyy");}

**Задача 14: Меняем заголовки HTTP-пакета**

Заголовки пакетов можно легко редактировать: удалять, добавлять, изменять.

// OnBeforeRequest// Удалить заголовок с именем 'User-Agent'oSession.oRequest.headers.Remove("User-Agent");// Добавить заголовок 'xxx' со значением 'yyy'oSession.oRequest.headers.Add("xxx", "yyy");// Изменить значение заголовка с именем 'User-Agent' на значение 'xxx' oSession.oRequest.headers["User-Agent"] = "xxx";

**Задача 15: Меняем Cookie**

Работа с Cookie: добавление, удаление, редактирование

// OnBeforeRequest - добавить в запрос CookieoSession.oRequest["Cookie"] = (oSession.oRequest["Cookie"] + ";mycookie=xxx");// OnBeforeRequest - изменить значение Cookie 'JSESSIONID' на 'xxx'oSession.oRequest['Cookie'] = oSession.oRequest['Cookie'].Replace("JSESSIONID=","ignoredCookie=") + ";JSESSIONID=xxx";// OnBeforeRequest - удалить Cookie 'JSESSIONID'oSession.oRequest['Cookie'] = oSession.oRequest['Cookie'].Replace("JSESSIONID=","ignoredCookie=");