



Хакер - Тотальная слежка в интернете — как за тобой следят и как положить этому конец  
nopaywall



<https://t.me/nopaywall>

[Олег Парамонов](#)

## Содержание статьи

- [На примере](#)
- [Найти и перепрятать](#)
- [Усы, лапы и хвост — вот мои документы!](#)
- [Шапочка из фольги своими руками](#)

В 1993 году журнал «Нью-Йоркер» напечатал знаменитую карикатуру про пса за компьютером. «В интернете никто не знает, что ты собака», — сообщала подпись. Спустя двадцать с лишним лет дела обстоят с точностью до наоборот. В сегодняшнем интернете любая собака знает, кто ты такой, — и порой даже лучше, чем ты сам. Интернет плохо совместим с тайнами, и тайна частной жизни — не исключение. О каждом клике, сделанном в браузере, по определению должны знать две стороны: клиент и сервер. Это в лучшем случае. На самом деле где двое, там и трое, а то и, если взять в качестве примера сайт «Хакера», все двадцать восемь.

## На примере

Чтобы убедиться в этом, достаточно включить встроенные в Chrome или Firefox инструменты разработчика. Согласно им, при загрузке главной страницы хакер.ru браузер совершает 170 запросов. Больше половины этих запросов не имеют ни малейшего отношения к документам, которые расположены на серверах «Хакера». Вместо этого они ведут к 27 различным доменам, принадлежащим нескольким

иностранным компаниям. Именно эти запросы съедают 90% времени при загрузке сайта.

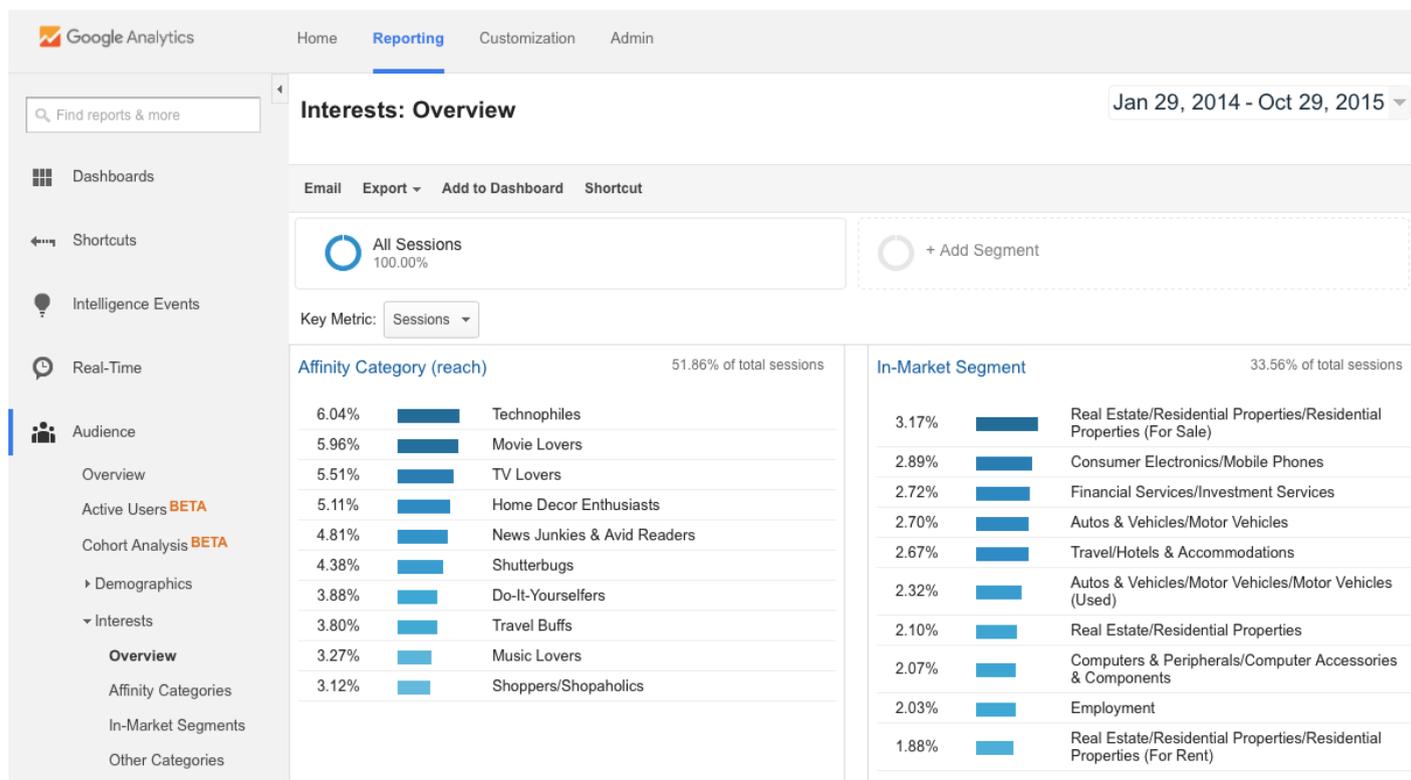
Method	File	Domain	Type	Transferred	Size	Time
200 GET	hi-res-css.122b94d46dce332d66...	s7.addthis.com	js	28,14 KB	91,43 KB	→ 176 ms
200 POST	admin-ajax.php	xakep.ru	html	2,71 KB	11,87 KB	→ 1162 ms
200 POST	admin-ajax.php	xakep.ru	json	0,04 KB	0,04 KB	→ 756 ms
200 GET	show_ads_impl.js	pagead2.googleadsyn...	js	cached	276,63 KB	
200 GET	ads?client=ca-pub-115495164341...	googleads.g.doubl...	html	35,94 KB	97,99 KB	→ 151 ms
200 GET	osd.js	pagead2.googleadsyn...	js	22,24 KB	58,29 KB	→ 35 ms
200 GET	sh.ffb539525be53bf07820b48d.h...	s7.addthis.com	html	cached	68,45 KB	
200 GET	18262075?wmode=5&callback=_y...	mc.yandex.ru	js	0,09 KB	0,09 KB	→ 17 ms
200 GET	O28XaUnlrsP2IR365eB2Pg.woff2	fonts.gstatic.com	woff2	25,50 KB	34,01 KB	→ 36 ms
200 GET	Zd2E9abXLFGSr9G3YK2MsCCeYr...	fonts.gstatic.com	woff2	45,98 KB	61,30 KB	→ 38 ms
200 GET	O28XaUnlrsP2IR365eB2Pg.woff2	fonts.gstatic.com	woff2	25,50 KB	34,01 KB	→ 39 ms
200 GET	Zd2E9abXLFGSr9G3YK2MsCCeYr...	fonts.gstatic.com	woff2	45,98 KB	61,30 KB	→ 41 ms
302 GET	/ads/user-lists/943782719/?fmt=1...	www.google.com	html	0,07 KB	0 KB	→ 78 ms
200 GET	O28XaUnlrsP2IR365eB2Pg.woff2	fonts.gstatic.com	woff2	25,50 KB	34,01 KB	→ 48 ms
200 GET	Zd2E9abXLFGSr9G3YK2MsCCeYr...	fonts.gstatic.com	woff2	45,98 KB	61,30 KB	→ 65 ms
200 GET	runtime.js	www.gstatic.com	js	cached	409,87 KB	
200 GET	300lo.json?53zpz9&colc=144622...	m.addthis.com	js	0,26 KB	0,33 KB	→ 145 ms
200 GET	xakep?callback=me.define&_=62911	b1.malerex.ru	js	6,33 KB	30,22 KB	→ 74 ms
200 GET	expansion_embed.js	pagead2.googleadsyn...	js	cached	150,40 KB	
200 GET	abg.js	tpc.googleadsyndicati...	js	cached	51,65 KB	
200 GET	m_js_controller.js	tpc.googleadsyndicati...	js	cached	59,05 KB	
200 GET	/ads/user-lists/943782719/?fmt=1...	www.google.ru	html	0,07 KB	0,06 KB	→ 87 ms
204 GET	push?client=ca-pub-11549516434...	cm.g.doubleclick.net	html	—	0 KB	→ 317 ms
200 GET	nr-768.min.js	js-agent.newrelic.c...	js	cached	22,09 KB	
200 GET	main.css?v=5.4.0	b1.malerex.ru	css	cached	3,32 KB	
200 GET	O28XaUnlrsP2IR365eB2Pg.woff2	fonts.gstatic.com	woff2	cached	34,01 KB	
200 GET	Zd2E9abXLFGSr9G3YK2MsCCeYr...	fonts.gstatic.com	woff2	cached	61,30 KB	
200 GET	activeview?id=osdim&avi=Bc6...	pagead2.googleadsyn...	gif	0,04 KB	0,05 KB	→ 26 ms
200 GET	CbzsU2oByTYt-pLbi77EkWqPp7C...	pagead2.googleadsyn...	js	cached	9,64 KB	
200 GET	O28XaUnlrsP2IR365eB2Pg.woff2	fonts.gstatic.com	woff2	cached	34,01 KB	
200 GET	Zd2E9abXLFGSr9G3YK2MsCCeYr...	fonts.gstatic.com	woff2	cached	61,30 KB	
200 GET	40195a0833?a=7901801&pl=1446...	bam.nr-data.net	js	0,04 KB	0,04 KB	→ 478 ms
200 GET	collect?v=1&_v=j39&a=172862...	www.google-analyt...	gif	0,03 KB	0,05 KB	→ 117 ms

Что это за домены? Рекламные сети, несколько систем веб-аналитики, социальные сети, платежный сервис, облако Amazon и пара маркетинговых виджетов. Похожий набор, и зачастую даже более обширный, имеется на любом коммерческом сайте. Побочный эффект этого заключается в том, что твои визиты на хакер.ru — никакой не секрет. О них знаем не только мы (это само собой), но и обладатели этих 27 доменов.

Многие из них не просто знают. Они наблюдают за тобой с самым пристальным интересом. Видишь баннер? Он загружен с сервера Doubleclick, крупной рекламной сети, которая принадлежит Google. С его помощью Гугл узнал, что ты побывал на хакер.ru. Если бы баннера не было, он нашел бы другой способ. Те же данные можно извлечь с помощью трекера Google Analytics или через AdSense, по обращению к шрифтам с Google Fonts или к jQuery на CDN Google. Хоть какая-то зацепка найдется на значительной доле страниц в интернете.

Анализ истории перемещений пользователя по интернету помогает Google с неплохой точностью определить его интересы, пол, возраст, достаток, семейное положение и даже

состояние здоровья. Это нужно для того, чтобы точнее подбирать рекламу. Даже незначительное увеличение точности таргетинга в масштабах Google — это миллиарды долларов, но возможны и другие применения. Согласно документам, которые опубликовал Эдвард Сноуден, американские и британские спецслужбы перехватывали трекеры Google для идентификации подозреваемых.



За тобой следят — это факт, с которым нужно смириться. Лучше сосредоточиться на других вопросах. Как они это делают? Можно ли скрыться от слежки? И стоит ли?

## Найти и перепрятать

Для того чтобы следить за человеком, нужно уметь его идентифицировать. Самый простой и хорошо изученный способ идентификации — это cookie. Проблема заключается в том, что он уязвимее всего для атак со стороны поборников privacy. О них знают и пользователи, и даже политики. В Евросоюзе, к примеру, действует закон, вынуждающий сайты предупреждать пользователей о вреде кук. Толку ноль, но сам факт настораживает.

Другая проблема связана с тем, что некоторые браузеры по умолчанию блокируют cookie, установленные третьей стороной — например, сервисом веб-аналитики или рекламной сетью. Такое ограничение можно обойти, прогнав пользователя через цепочку редиректов на сервер третьей стороны и обратно, но это, во-первых, не очень удобно, а во-вторых, вряд ли кого-то спасет в долгосрочной перспективе. Рано или поздно потребуется более надежный метод идентификации.

В браузере куда больше мест, где можно спрятать идентификационную информацию, чем планировали разработчики. Нужна лишь некоторая изобретательность. Например, через свойство DOM `window.name` другим страницам можно передать до двух мегабайт данных, причем в отличие от кук, доступных лишь скриптам с того же домена, данные в `window.name` [доступны](#) и из других доменов. Заменить куки на `window.name` мешает лишь эфемерность этого свойства. Оно не сохраняет значение после завершения сессии.

Несколько лет назад в моду вошло хранение идентификационной информации при помощи так называемых Local Shared Objects (LSO), которые предоставляет Flash. В пользу LSO играли два фактора. Во-первых, в отличие от кук, пользователь не мог их удалить средствами браузера. Во-вторых, если куки в каждом браузере свои, то LSO, как и сам Flash, один для всех браузеров на компьютере. За счет этого можно идентифицировать пользователя, попеременно работающего в разных браузерах. Может сложиться впечатление, что LSO придумали специально для слежки, но это не так. На самом деле разработчики Adobe просто не осознавали, что они делают. Когда до них дошло, что LSO можно использовать в качестве бессмертной вездесущей куки, они поспешили исправить оплошность и добавили программный интерфейс, удаляющий LSO. Современные браузеры вызывают его при очистке кук. Это несколько уменьшило полезность LSO, но трекеры по-прежнему применяют эту технологию.

У LSO есть несколько альтернатив. В первую очередь речь идет о HTML5 Local Storage — хранилище данных, которое встроено во все современные браузеры. Оно также очищается одновременно с куками и, в отличие от LSO, не может служить для слежки за пользователями в других браузерах. Тем не менее исследователи отмечают, что HTML5 Local Storage используется рядом крупных сайтов для резервного хранения идентификационной информации.

Более экзотический вариант — браузерные базы данных IndexedDB и Web SQL Database. Их в той или иной степени поддерживают последние версии Firefox, Chrome и Internet Explorer. Первые случаи практического использования IndexedDB в качестве запасного варианта на случай утраты обычных кук и LSO были замечены в 2014 году на китайских сайтах `weibo.com` и `sina.com.cn`.

В 2011 году сервис аналитики Kissmetrics предпринял попытку спрятать идентификатор пользователя еще глубже — в сам запрос HTTP. Для хранения идентификатора приспособили поля `Etag` и `Last-Modified`, предназначенные для проверки актуальности закешированной версии документа. Обычно при отправке документа клиенту сервер помещает в поле `Etag` его хеш, а в поле `Last-Modified` — дату последнего обновления. Когда документ потребуется снова, браузер сообщит серверу значение `Etag` или `Last-Modified` его закешированной версии. Если они совпадают с известными серверу, тот не станет отправлять данные снова, а вернет статус 304: «документ не изменился».

Бросается в глаза, насколько этот процесс похож на обмен куками. Разница лишь в реакции сервера, но как раз ее-то поменять проще всего. Если в Etag поместить идентификатор пользователя, он будет храниться у клиента до тех пор, пока цела страница в кеше браузера. Там он переживет удаление кук и даже отключение JavaScript. Чтобы полностью избавиться от него, пользователю придется очистить кеш и обнулить историю посещений. Etag и Last-Modified можно использовать для восстановления идентификатора, когда и куки, и данные из LSO или HTML5 Local Storage утрачены.

Кеш и история посещений — это вообще находка для шпиона, и не только из-за Etag. Начнем с того, что идентификатор можно не просто прилагать к закешированному файлу, но и вставлять в него. В этом случае, пока жив кеш, сохранится и идентификатор пользователя. Более хитрый метод использует постоянные редиректы по статусу 301. Сервер выдает этот статус при изменении адреса документа. Браузер запоминает новый адрес и в будущем переходит по нему, минуя старый. Этот механизм применяют для сохранения идентификаторов пользователя.

Вот как это делают: в документ встраивают запрос к невидимой картинке, iframe или скрипту. Если в запросе отсутствует параметр с идентификатором, сервер возвращает постоянный редирект на тот же URL, но уже с идентификатором. Браузер запоминает адрес с идентификатором и в следующий раз вызывает его. Этот метод интересен тем, что с его помощью можно обмениваться идентификатором между доменами, ведь кеш-то общий.

Родственный метод основан на использовании кеша HTTP Strict Transport Security (HSTS). Согласно стандарту HSTS, сервер может с помощью специального поля в запросе рекомендовать браузеру устанавливать для определенных документов защищенное соединение. Браузер сохраняет эти рекомендации в кеше HSTS. При необходимости в них можно зашифровать идентификатор пользователя. Для этого следует встроить в страницу несколько невидимых изображений, при запросе к которым браузер возвращает рекомендацию HSTS. Каждая такая рекомендация рассматривается как один бит идентификатора.

## Усы, лапы и хвост — вот мои документы!

В последнее время набирает популярность другой подход к решению этой проблемы — так называемый фингерпринтинг (от английского слова fingerprint — отпечаток пальца). Фингерпринтинг идентифицирует пользователя не по специальным меткам, сохраненным на его системе, а по уникальным особенностям его браузера, системы и устройства.

Поскольку фингерпринтинг не требует хранения данных на клиенте, его очень трудно заметить и почти невозможно избежать. Если куки действуют лишь в рамках одного домена, уникальные особенности остаются неизменными при посещении различных сайтов. Это значительно упрощает слежку за передвижениями пользователя по интернету. Хуже того, в отличие от кук уникальные особенности нельзя отключить. Усилия пользователя приведут максимум к замене одного набора признаков другим, еще более узнаваемым.

Простейшие методы фингерпринтинга используют в качестве уникальных характеристик IP-адрес, версию браузера и системы, системный язык, разрешение экрана, часовой пояс, показания часов с точностью до миллисекунды и список стандартных шрифтов, установленных на компьютере. При помощи Flash этот список можно дополнить сведениями о подключенных к устройству мыши, клавиатуре, микрофоне, камере и поддержке мультитача.

Незначительные изменения некоторых признаков не мешают опознавать уже знакомого пользователя. Он может воспользоваться другим браузером, переехать в другой часовой пояс или поменять разрешение, но, если не сделать все это одновременно, вероятность идентификации останется высокой.

Существуют и более замысловатые способы фингерпринтинга. Компания AddThis экспериментировала с идентификацией пользователя по особенностям отображения шрифтов. Для этого создается невидимый пользователю canvas, на котором выводится надпись. Хеш последовательности данных о цвете каждого пикселя canvas и становится идентификатором. На то, как именно будет выглядеть надпись, влияет операционная система, установленные шрифты, графическая карта, версия графических драйверов, настройки сглаживания, тип и версия браузера, а также особенности самого дисплея. Тонких отличий предостаточно ([PDF](#)), но на них трудно повлиять — идеальное сочетание для трекинга.

### Windows:

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

### OS X:

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

### Linux:

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

К слову, виджеты AddThis, год назад незаметно проводившие фингерпринтинг каждого посетителя, теперь стоят и на сайте хакер.ru. Но можешь быть спокоен: когда эту компанию поймали за руку, она прекратила эксперимент. Сейчас никакого фингерпринтинга. По крайней мере, заметного.

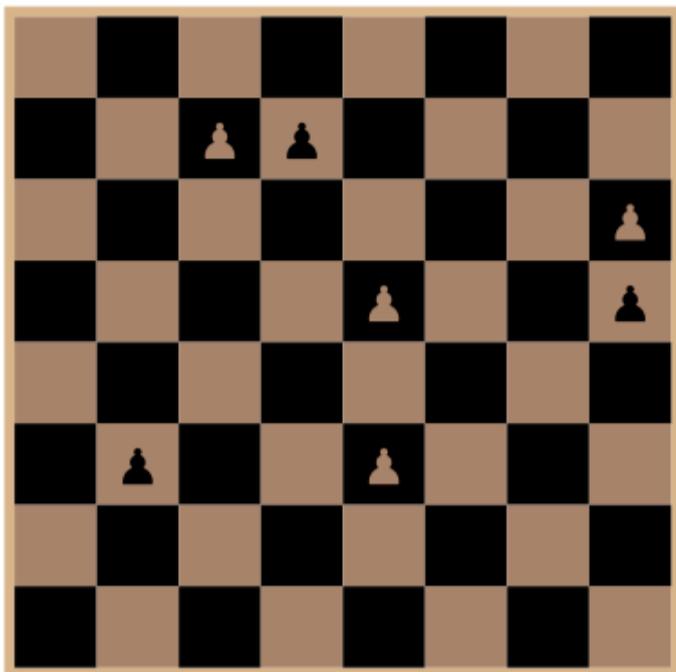
Еще один метод фингерпринтинга анализирует историю посещений. Исследователи показали, что информация о посещении 500 сайтов из заранее определенного набора позволяет точно идентифицировать около 70% пользователей, причем в том случае, если в истории присутствуют социальные сети, речь может идти не просто об идентификации, но и о деанонимизации.

Чтобы определить, посещал ли пользователь тот или иной сайт, есть свои хитрости. Например, можно попытаться загрузить документ с нужного сайта. По скорости отклика будет понятно, есть он в кеше или нет. Можно воспользоваться тем фактом, что ссылки на посещенные сайты отображаются другим цветом. Чтобы выяснить цвет, согдится все тот же canvas. Есть, впрочем, вариант любопытнее: ссылки нетрудно замаскировать под капчу. Тогда пользователь сам выдаст все нужные сведения. Этот способ особенно полезен, когда JavaScript отключен.

Как замаскировать ссылки под капчу? За счет различного оформления посещенных и непосещенных ссылок. Исследователи из университета Карнеги — Меллон, которые в 2011 году предложили эту методику извлечения истории, перечисляют несколько возможностей. Во-первых, можно сделать каждую ссылку отдельным словом и при помощи CSS скрыть посещенные ссылки. Теперь нужно попросить пользователя ввести

текст, который он видит. По пропущенным словам легко определить, на каких сайтах он уже был. Другой вариант капчи представляет собой изображение шахматной доски, на которой расставлены пешки.

Please click on all of the chess pawns.



Каждая пешка — это опять-таки ссылка, и просмотренные ссылки сделаны невидимыми. Пользователь должен кликнуть каждую пешку. Пешки, на которые он не кликнул, соответствуют ссылкам, ведущим на посещенные сайты.

## Шапочка из фольги своими руками

Даже беглого перечисления методов трекинга достаточно для того, чтобы уловить общие мотивы. Во-первых, Flash. Его исчезновение избавит сразу и от LSO, и от утечки сведений об устройстве, которая упрощает фотоприпечатывание. Лучше обойтись без Flash — в 2015 году это просто.

Чтобы предотвратить хранение идентификатора в HTML5 Local Storage и иже с ним, нужно либо избавляться от JavaScript, либо запрещать куки. И то и другое — совсем не безболезненный процесс. Отсутствие кук и JavaScript не только делает невозможным использование современных веб-приложений. Нередко оно ломает совсем безобидные сайты. Страдания неизбежны, и дальше будет только хуже.

Бороться с методами, которые используют кеш? Это гиблое дело. От Etag и Last-Modified еще можно спрятаться за прокси, переписывающим HTTP-запросы, но и только. От кеша редиректов нет спасения. От кеша HSTS тоже, и это, к слову, правильно — его

отсутствие делает браузер уязвимым для атак типа MITM. Частое удаление кеша — это, кажется, единственный выход, но и он далек от идеала.

С фингерпринтингом дело обстоит еще веселее. Борьба с трекерами делает тебя уязвимее для фингерпринтинга. Удалил Flash? Что ж, теперь ты белая ворона. Вас таких меньше процента, и более уникального признака не придумать. С тем же успехом можно прятаться на улице города при помощи накладной бороды, темных очков и большой шляпы. Это не маскировка, а эффективный способ привлечения внимания к своей персоне. Еще Тог поставь, и будет комплект!

Рассчитывать на полную победу над трекерами вряд ли стоит, но создать иллюзию незаметности все же можно. Для начала оговорим выбор браузера. Эппловский Safari отпадает сразу. Он уникален тем, что не выключает куки, локальные хранилища данных и кеш даже в режиме инкогнито. Chrome — хороший браузер, но настоящему параноику следует держаться от него подальше. В Google никогда не скрывали, что собирают и анализируют информацию о пользователях. Остается Firefox. Он не лишен порочных связей с Google и даже пингует его при установке, но какой у нас выбор?

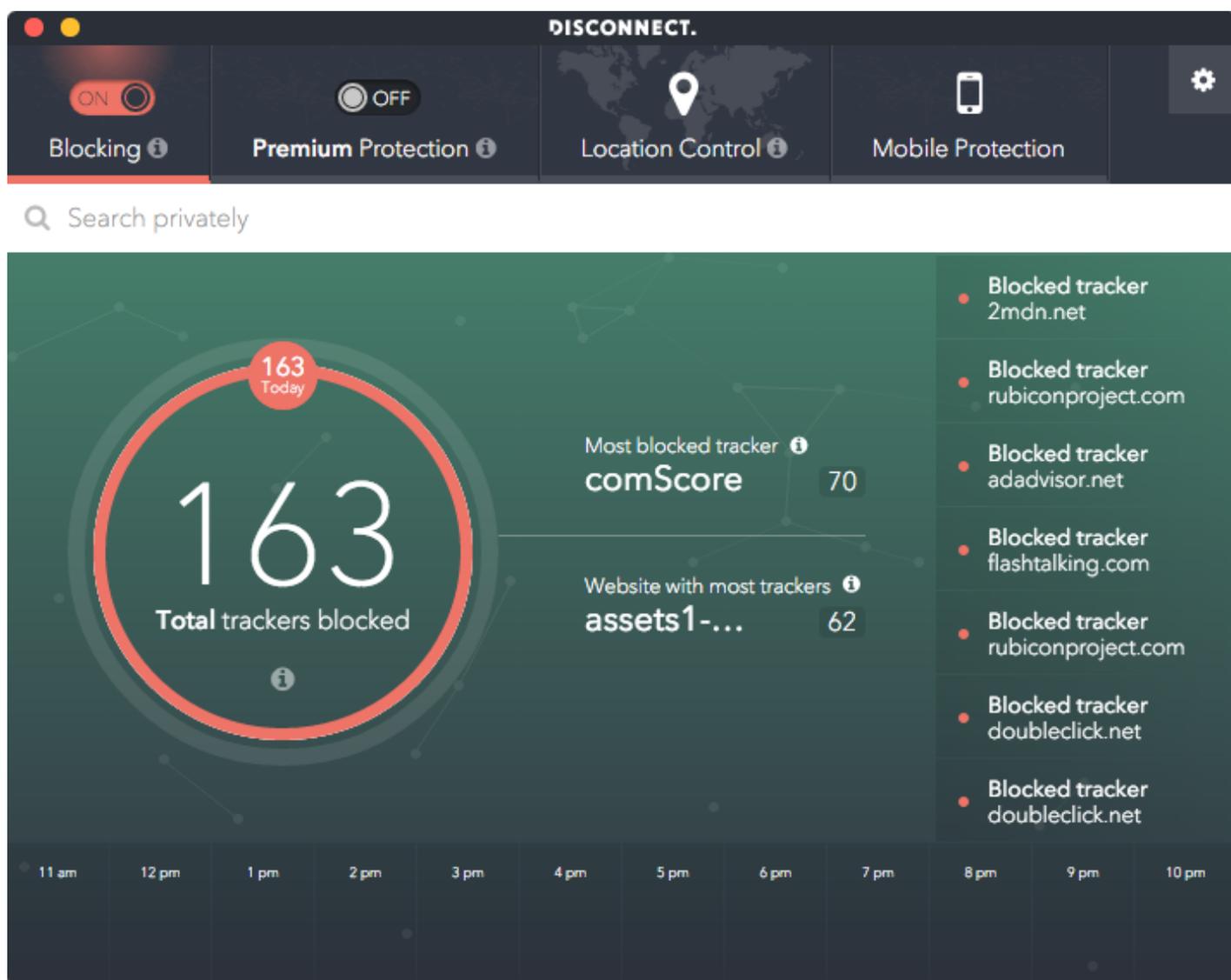
В Firefox встроен блокировщик трекеров и рекламы, но по умолчанию он выключен. Чтобы активировать его, нужно открыть скрытые настройки (about:config) и включить свойство

```
privacy.trackingprotection.enabled
```

. У левого края адресной строки появится новая иконка — маленький щит. Она нужна для того, чтобы избирательно включать или выключать блокировку трекеров именно на данном домене. На хакер.ru при включенной блокировке количество загружаемых файлов падает вдвое, а скорость загрузки вырастает в четыре раза.

Блокировщик трекеров Firefox заимствует черный список у Disconnect — популярного средства блокировки трекеров, которое существует в виде браузерного аддона и приложения для всех популярных платформ. Очевидный недостаток Disconnect в том, что лучше всего он знает трекеры, которые популярны за границей. Мусор из России течет через него, как сквозь решето.

С популярной альтернативой Disconnect — аддоном Ghostery — тоже не все просто. Он эффективно удаляет трекеры, а затем продает информацию о своих пользователях тем самым рекламщикам, которые их ставят. В теории от продажи своих данных можно отказаться, но на практике — какого параноика убедят эти отговорки? Либо приложения, торгующие данными, либо борьба со слежкой — нужно выбрать что-то одно.



Неплохой репутацией пользуется блокировщик рекламы и трекеров uBlock Origin. Он позволяет подписаться на множество черных списков различного происхождения и назначения, в том числе для блокировки рекламы, трекеров, кнопок социальных сетей и вредоносного кода. Им можно заменить и Adblock Plus, и Ghostery.

С помощью аддона [RequestPolicy Continued](#) можно закрутить гайки еще сильнее. Он запрещает любые запросы к другим доменам, если пользователь заранее не внес их в белый список. Аддон [Self-Destructing Cookies](#) уничтожает куки и содержимое локальных хранилищ после завершения сессии. Наконец, старый добрый [NoScript](#) блокирует исполнение JavaScript, Flash и Java и включает только по просьбе пользователя.

Завершив выполнение рекомендаций, перечисленных в прошлом абзаце, стоит задуматься о жизни. Каким будет следующий шаг? Тебя все равно найдут, поэтому лучше не медлить. Беги в тайгу, подальше от NoScript и Flash. Firefox и аддоны — это полумеры и самообман. Против интернета помогут топор и кусачки, против фингерпринтинга — наждачная бумага. Удачи!

Читайте ещё больше платных статей бесплатно: <https://t.me/nopaywall>