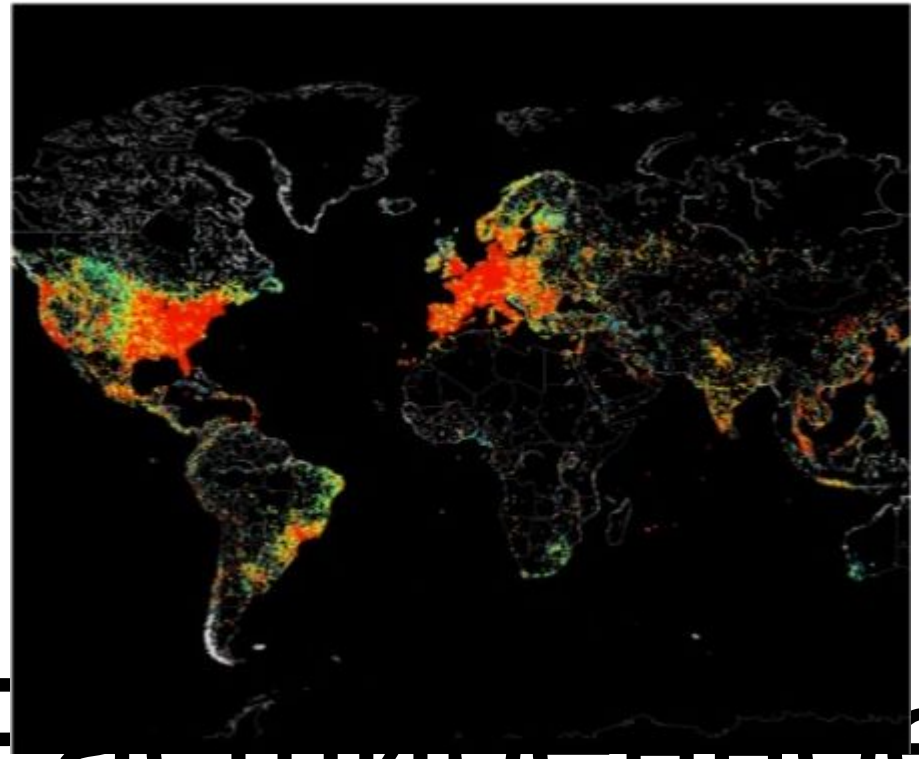


Расширенное руководство по использованию Shodan

shodan-manual.com

Содержание

Вступление

Shodan – это поисковый движок для устройств, подключённых к интернету. Веб-поисковики, например, Google или Bing, отлично подходят поиска сайтов на просторах интернета. Но что делать, если вам нужно найти не какой-либо сайт, а компьютеры, работающие с конкретным программным обеспечением (например, Apache)? Или, к примеру, вам нужно узнать, какая версия Microsoft IIS наиболее популярна на данный момент? Или вы хотите получить информацию о количестве анонимных ftp-серверов? Возможно, появилась новая уязвимость, и вам необходимо понять, сколько хостов она может заразить? Обычные поисковики не смогут дать вам ответы на эти вопросы.

Всё о данных

Баннер

Основная единица данных, собираемая Shodan - это баннер. Баннер в данном контексте – это текстовая информация, которая описывает службу, используемую на устройстве. Для веб-серверов это будет хэдер, возвращаемый как отклик, для Telnet же это будет экран входа.

Содержание баннера напрямую зависит от вида службы. Например, вот типичный HTTP-баннер:

```
HTTP/1.1 200 OK
Server: nginx/1.1.19
Date: Sat, 03 Oct 2015 06:09:24 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 6466
Connection: keep-alive
```

Баннер, приведённый примерном выше, говорит о том, что на целевом устройстве запущено программное обеспечение web-сервера **nginx** версии **1.1.19**. Чтобы продемонстрировать, как могут выглядеть разные баннеры, ниже приведён баннер для протокола промышленной системы управления Siemens S7:

```
Copyright: Original Siemens Equipment
PLC name: S7_Turbine
Module type: CPU 313C
Unknown (129): Boot Loader       A
Module: 6ES7 313-5BG04-0AB0 v.0.3
Basic Firmware: v.3.3.8
Module name: CPU 313C
Serial number of module: S Q-D9U883642013
Plant identification:
Basic Hardware: 6ES7 313-5BG04-0AB0 v.0.3
```


Протокол Siemens S7 даёт отклик в виде совершенно другого баннера. Он предоставляет информацию о прошивке, ее серийном номере и достаточно большое количество подробных данных, описывающих само устройство.

Вам необходимо будет решить, какие именно службы вас интересуют, чтобы продуктивно пользоваться поиском Shodan, поскольку баннеры очень сильно различаются между собой.

Внимание: Shodan ищет именно баннеры, а не хосты. Это значит, что если на одном IP-адресе запущено много служб, каждая из них будет представлена, как самостоятельный результат.

Метаданные устройства

Помимо баннеров, Shodan также умеет захватывать метаданные устройства – геолокацию, имя хоста, ОС и так далее (см. приложение А). Большую часть метаданных можно найти через основной сайт Shodan, однако некоторые области поиска доступны только пользователям API разработчика.

IPv6

На октябрь 2015 года Shodan собирает миллионы баннеров в месяц от устройств, доступных на IPv6. Эти цифры по-прежнему ничтожны по сравнению с сотнями миллионов баннеров, собранных на IPv4, но ближайшие годы количество баннеров, безусловно, будет расти.

Сбор данных

Режим сбора данных

Сканеры Shodan работают в режиме 24/7 и обновляют базу данных в реальном времени. Вы получаете последний снимок интернета каждый раз, как посещаете официальный веб-сайт Shodan.

Распространение

Сканеры представлены в странах по всему миру, включая:

- США (Восточное и Западное Побережье)
- Китай
- Исландия
- Франция
- Тайвань
- Вьетнам
- Румыния
- Чехия

Данные собираются со всего мира, чтобы предотвратить географические смещения. К примеру, многие американские администраторы систем блокируют все китайские диапазоны IP. Распределение сканеров Shodan по всему миру гарантирует, что любая блокировка не будет влиять на сбор данных.

Рандомизация

Базовый алгоритм сканеров:

1. Генерация случайного IPv4 адреса
2. Генерация случайного порта для проверки списка портов, понимаемых Shodan
3. Проверка случайного IPv4 адреса на случайном порте и захват баннера
4. Возвращение к пункту 1

Это означает, что агенты не занимаются сканированием инкрементных сетевых диапазонов. Сканирование выполняется в полностью случайном режиме – таким образом обеспечивается единообразный охват всего Интернета и предотвращается возможность предвзятой оценки данных в любой момент.

Углублённое описание работы с SSL

В настоящее время протокол SSL становится все более важным аспектом в обслуживании и потреблении Интернет-контента, поэтому вполне логично, что Shodan дополняет собираемые данные каждой службы информацией о поддержке SSL. Баннеры для SSL-сервисов, к примеру, HTTPS, включают в себя не только сертификат SSL, но и многое другое. Вся собранная информация о SSL, про которую пойдёт речь ниже, сохраняется в свойствах **ssl** в баннере (см. приложение A и приложение E).

Проверка на уязвимости

Heartbleed

Если служба оказывается уязвимой для Heartbleed, то баннер будет содержать два дополнительных параметра. **opts.heartbleed** содержит в себе необработанный отклик от тестирования службы на уязвимость к Heartbleed. Стоит обратить внимание, что для данного тестирования сканеры вытягивают лишь небольшое количество данных, которого достаточно, чтобы подтвердить, что служба подвержена влиянию Heartbleed, но не хватает для захвата секретных ключей. Сканеры также добавляют **CVE-2014-0160** в список **opts.vulns**, если устройство уязвимо. В противном же случае, если устройство не подвержено данной уязвимости, будет добавлена запись «**!CVE-2014-0160**». Если запись в **opts.vulns** имеет префикс ! или - тогда служба не уязвима для данной CVE.

```
{
      "opts": {
            "heartbleed": "... 174.142.92.126:8443 - VULNERABLE\n",
            "vulns": ["CVE-2014-0160"]
      }
}
```


Также у Shodan есть возможность поиска по данным уязвимости. Например, чтобы выполнить поиск устройств в США, подверженных влиянию Heartbleed, используется такая строка:

```
country:US vuln:CVE-2014-0160
```

FREAK

Если служба поддерживает шифры EXPORT, то сканеры добавляют «**CVE-2015-0204**» в свойство **opts.vulns**:

```
"opts": {
      "vulns": ["CVE-2015-0284"]
}
```

Logjam

Сканеры делают попытки подключиться к службе SSL, используя эфемерные шифры Диффи-Хеллмана, и если соединение происходит успешно, то будет сохранена следующая информация:

```
"dhparams": {
      "prime": "bbbc2dcad84674907c43fcf580e9...",
      "public_key": "49858e1f32aefe4af39b28f51c...",
      "bits": 1024,
      "generator": 2,
      "fingerprint": "nginx/Hardcoded 1024-bit prime"
}
```

Версия SSL

Когда браузер подключается к службе SSL, он будет «обсуждает» с сервером версию SSL и шифрование, которое будет использоваться. В результате этого «обсуждения», будет выбрана определенная версию SSL, например, TLSv1.2, которая будет использоваться в дальнейшем это для связи.

Сканеры Shodan начинают тестирование SSL с выполнения стандартного запроса, описанного выше. Однако далее они пытаются напрямую подключиться к серверу, используя определенные версию SSL. Другими словами, сканеры будут пытаться подключиться к серверу с помощью SSLv2, SSLV3, TLSv1.0, TLSv1.1 и TLSv1.2 напрямую, для определения всех поддерживаемых службой SSL версий. Собранная информация будет доступна к просмотру в поле **ssl.versions**:

```
{
      "ssl": {
            "versions": ["TLSv1", "SSLv3", "-SSLv2", "-TLSv1.1", "-TLSv1.2"]
      }
}
```

Если перед версией стоит - (дефис), то устройство не поддерживает эту версию SSL. Если же дефиса нет, то версия поддерживается. К примеру, сервер расписанный выше, поддерживает следующие версии:

```
TLSv1
```

```
SSLv3
```

И не поддерживает:

```
SSLv2
TLSv1.1
TLSv1.2
```

Информацию о версии также можно просмотреть через API website/. Например, поисковый запрос ниже даст отклик в виде всех служб (HTTPS, POP3 с SSL и так далее) которые позволяют соединения с SSLv2:

```
ssl.version:sslv2
```

Следование по цепочке

Цепочка сертификатов - это список сертификатов SSL от корневого сертификата до сертификата конечного пользователя. Баннер для SSL-служб включают в себя свойство **ssl.chain**, которое включает все SSL-сертификаты цепочки в сертификаты, сериализованные в формат PEM.

За пределами основных функций

Для большинства служб сканеры пытаются произвести анализ основного текста баннера и вычленить любую полезную информацию. Несколько ярких примеров – захват названий групп в MongoDB, скриншоты, получаемые из службы удаленных рабочих столов и сохранение списка пиров Bitcoin. Имеются две продвинутых техники анализа данных, которые используются Shodan, которые я хотел бы выделить особо:

Веб-компоненты

В этом случае сканеры пытаются определить технологию, которая была использована для создания сайта. Для http- и https-модулей, для разбивки сайта на веб-компоненты используются хэдеры и HTML. Полученная информация сохраняется в свойстве **http.components**. Данное свойство является словарём технологий, где ключ - это название технологии (к примеру, jQuery), а значение – это еще один словарь со свойствами категорий. Свойство категории - это список категорий, которые связаны с определённой технологией. Например:

```
"http": {
...
```



```
        "components": {
                "jQuery": {
                        "categories": ["javascript-frameworks"]
                },
                "Drupal": {
                        "categories": ["cms"]
                },
                "PHP": {
                        "categories": ["programming-languages"]
                }
        },
        ...
},
```

В примере выше, свойство **http.components** указывает, что сайт работает на CMS Drupal, которая использует jQuery и PHP. Shodan REST API обеспечивает поиск информации через фильтр **http.component** и 2 фасета* – **http.component** и **http.component_category**. Чтобы получить доступ к полному списку всех возможных значений компонентов/категорий, необходимо использовать различные фасеты. К примеру, чтобы получить полный список всех возможных категорий, используйте следующую команду shodan:

*фасет – грань, одна из сторон рассматриваемого объекта, ограниченная совокупность однородных значений по некоторому классификационному признаку.

```
$ shodan stats --facets http.component_category:1000 http
      Top 47 Results for Facet: http.component_category
      javascript-frameworks 8,982,996
      web-frameworks 1,708,503
      programming-languages 1,409,763
      font-scripts 1,288,397
```

Каскадность

В случае если баннер даёт отклик в виде информации о пирах или каким-либо другим образом получает информацию о другом IP-адресе, на котором запускается служба, сканеры пытаются выполнить захват баннера с этого IP или службы. К примеру, стандартный порт для mainline DHT (используется Bittorrent) – 6881. Баннер такого DHT-узла будет выглядеть так:

```
DHT Nodes
97.94.250.250       58431
158.77.37.22 34149
113.181.97.227      63579
252.246.184.188 36408
83.145.107.53       52158
77.232.167.126      52716
25.89.240.146       27179
147.23.120.228      59074
85.58.200.213       27422
180.214.174.82      36937
241.241.187.233 60339
166.219.60.135      3297
149.56.67.21 13735
107.55.196.179      8748
```

Ранее сканер захватывал бы вышеупомянутый баннер и только потом двигался далее. При включенной каскадности, сканер будет запускать запрос на захват баннера для всех пиров. В примере выше, сканер запускает сканирование для IP 54.70.96.157 на порту 61770, используя граббер баннеров dht, IP 85.82.92.188 будет просканирован на порту 42155 и так далее. То есть, одиночное сканирование IP может привести к каскаду сканирований, если первичные данные, полученные при сканировании, содержат информацию о других потенциальных хостах.

Чтобы отслеживать взаимосвязь между первоначальным запросом на сканирование и любыми дочерними/каскадными запросами, мы представляем 2 новых свойства:

- _shodan.id: уникальный идентификатор баннера. Это свойство гарантированно существует, если запрос на каскадирование может быть запущен из службы, хотя это означает, что любой такой запрос будет выполнен успешно.
- _shodan.options.referrer: предоставляет уникальный идентификатор для баннера, ставшего причиной создания текущего баннера. То есть, такой баннер является «родителем» текущего.

Веб-интерфейсы

Самый простой путь получить доступ к данным, собранным с помощью Shodan – это веб-интерфейсы. Почти каждый из них позволяет вам ввести поисковый запрос, так что сперва стоит поговорить о них:

Поисковые запросы

По умолчанию, поисковый запрос просматривает только основной текст баннера и не выполняет поиск метаданных в нём. К примеру, если вы вводите запрос «Google», то результаты будут включать только те записи, в баннере которых есть текст «Google»; вовсе не обязательно, что результатом будет отклик в виде диапазона сети Google.

Как мы видим выше, поиск по тексту "Google" выдаёт результат в виде множества Google Search Appliances, купленными организациями и получившими доступ в интернет; в результатах нет служб серверов гугла.

Shodan будет пытаться найти результаты, которые соответствуют **всем условиям поиска**. Это означает, что между каждым значением в поиске присутствует + или AND. Например, запрос «apache + 1.3» эквивалентен «apache 1.3».

Чтобы произвести поиск метаданных, необходимо использовать **фильтры поиска**.

Фильтры поиска

Фильтры - это специальные ключевые слова, используемые Shodan для сужения результатов поиска на основе метаданных службы или устройства. Фильтры вводятся в таком формате:

```
filtername:value
```

Внимание: между двоеточием и значением фильтра не должно быть пробела.

Для использования значения с пробелом, нужно заключить его в кавычки. К примеру, чтобы найти все интернет-устройства, расположенные в Сан-Диего, запрос будет такой:

```
city:"San Diego"
```

Некоторые фильтры позволяют указать несколько значений, разделённых запятой. Чтобы найти устройства, на которых запущен Telnet на портах 23 и 1023, выполните следующий запрос:

```
port:23, 1023
```

Если фильтр не допускает использование запятой в значении (**port, hostname, net**), то такой фильтр даёт вам возможность предоставить несколько значений.

Также можно использовать фильтры, чтобы исключить некоторые результаты путём помещения знака "-" перед фильтром. Чтобы найти все устройства, не находящиеся в Сан-Диего, запись будет такая:

```
-city:"San Diego"
```

Во многих ситуациях, гораздо проще исключить результаты. Например, следующий поисковый запрос использует **hash:0**, чтобы показать службы на порте 8080, в которых основной текстовый баннер не является пустым:

```
port:8080 -hash:0
```

Каждый баннер на Shodan имеет числовое свойство **hash**. Для пустых баннеров это значение будет равно нулю. Если вам нужно найти устройство с коротким, статичным баннером, то **hash** фильтр предоставит хороший способ для их точного определения.

Shodan поддерживает массу фильтров, вот некоторые из них:

Имя фильтра	Описание	Пример
category	Доступные категории: ics, malware	

city	Название города	
country	Полное название страны	
net	Показывает только результаты из предоставленного диапазона IP (в CIDR-нотации)	net:190.30.40.0/24
org	Сужает результаты до основанных на организации, которой принадлежит IP	org:"Verizon Wireless"

Смотри приложение B для изучения полного списка доступных поисковых фильтров.

Поисковый движок Shodan

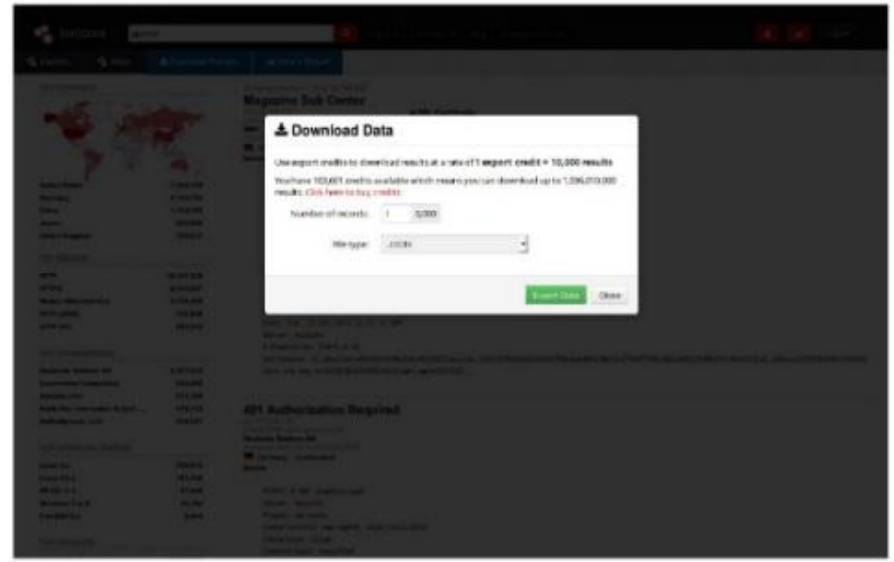
Основной интерфейс для доступа к данным, собранным Shodan, это его собственный поисковый движок, расположенный по адресу https://www.shodan.io

По умолчанию, поисковый запрос будет производить поиск по данным, собранным за последние 30 дней. Это большое отличие от старого сайта (shodanhq.com), который по умолчанию производил поиск по всей базе данных Shodan. Это означает, что все результаты, полученные с сайта – свежие и предоставляют вам точную карту интернета на нынешний момент.

В дополнение к поиску, сайт также предоставляет следующие возможности:

Скачивание данных

После осуществления поиска по запросу, вверху появится кнопка скачивания данных (**Download Data**). При нажатии на неё вы сможете выбрать формат скачивания данных – **JSON, CSV** или **XML**.



Формат **JSON** генерирует файл, в котором каждая строка содержит полный баннер и все сопутствующие метаданные, собранные Shodan. Этот формат предпочтителен, поскольку сохраняется вся информация, которая вообще доступна. Также этот формат совместим с клиентом командной строки Shodan, что даёт вам возможность загружать данные с сайта Shodan, а затем обрабатывать их далее с помощью терминала.

При сохранении в формате **CSV**, результатом будет файл, содержащий IP-адрес, порт, сам баннер, организацию и имена хостов этого баннера. Информация в нём неполная из-за ограничений в формате CSV. Этот формат рекомендован к использованию, если вам нужна только основные данные результатов поиска, или если вам нужна возможность быстро загрузить скачанный файл во внешние инструменты, к примеру, Excel.

Формат **XML** - это уже устаревший способ сохранения результатов поиска. С ним сложнее работать и он потребляет больше места чем JSON, что делает его субоптимальным для большинства ситуаций.

Загрузка данных требует монет экспорта (**export credits**), одноразовой валюты, доступной к приобретению на сайте. Они никоим образом не связаны с API-интерфейсом Shodan и не обновляются автоматически каждый месяц. Одна монета может быть использована для загрузки результатов числом до 10,000.

Файлы данных, сгенерированные на сайте, могут быть найдены в разделе загрузок сайта (Downloads). Вы можете перейти в этот раздел с помощью кнопки  в правом верхнем углу.

Генерация (создание) отчёта

Сайт даёт возможность сгенерировать отчёт, основанный на поисковом запросе. Этот отчёт содержит в себе графики/таблицы, предоставляющие вам большую карту распределения результатов по интернету. Эта опция бесплатна и доступна абсолютно всем.

Когда вы генерируете отчёт, вы даёте Shodan задачу сделать снимок поискового запроса и предоставить общий обзор. После генерации отчёта, он не меняется и не обновляется при добавлении новых данных в Shodan. Также это значит, что вы можете генерировать отчёт раз в месяц и отслеживать изменения, сравнивая его с предыдущими отчётами.

Нажимая на кнопку  в верхнем правом углу, вы получите список отчётов, сформированных ранее.

Общие поисковые запросы

Поиск определенных устройств требует знаний о программном обеспечении, запускаемом на них, и их реакции на захват баннеров через интернет. К счастью, можно использовать опыт и знания сообщества, используя каталог поиска на Shodan. Люди могут описать, пометить и поделиться их поисковыми запросами с другими

пользователями. Если у вас есть желание начать работу с Shodan, общие поисковые запросы будут вам хорошим подспорьем на первых этапах использования системы.

Внимание: общие поисковые запросы видны всем посетителям сайта. Не делитесь запросами, которые вы бы не желали показывать другим людям.

Пример: поиск нестандартных служб

Основной реакцией, которую я вижу, когда речь заходит о устройствах, выставленных в Интернете, является нечто вроде:

> [-] XDRosenheim 1 point 3 days ago
> And this is why my server is whitelisted, password protected and not on port 25565. I don't like data miners...

*И ЭТО ЯВЛЯЕТСЯ ПРИЧИНОЙ ТОГО, ЧТО МОЙ СЕРВЕР В БЕЛЫХ ЛИСТАХ, ЗАЩИЩЕН ПАРОЛЕМ И НЕ НАХОДИТСЯ НА ПОРТУ 25565. Я НЕ ЛЮБЛЮ ДОБЫТЧИКОВ ДАННЫХ…

Собственно, идея запуска этой службы (в данном случае это Minecraft) на нестандартном порту – это хороший способ оставаться необнаруженным. В кругах людей, занимающихся безопасностью, этот способ также известен как «безопасность через незаметность» и считается крайне неэффективным и устаревшим. Что наиболее ужасно – это может дать хозяину сервера или устройства чувство ложной защищённости. Например, давайте посмотрим на людей, запускающих OpenSSH на нестандартном порту. Чтобы это сделать, вводим следующую команду:

```
product:openssh - port:22
```

Фильтр **product** используется, чтобы показать только серверы OpenSSH, а **–port:22** исключает все результаты со стандартного порта SSH (22). Чтобы получить более понятный обзор результатов поиска, сгенерируем отчёт:

Top Countries

1.	United States	139,969
2.	Australia	59,493
3.	Germany	24,584
4.	Brazil	24,405
5.	China	15,123
6.	France	14,708
7.	Russian Federation	11,065
8.	United Kingdom	10,692
9.	Poland	8,496
10.	Canada	7,484

Полученный отчёт также даст нам понимание, какие нестандартные порты используются чаще всего:

1. **2222**: 323,930
2. **5000**: 47,439
3. **23**: 13,482
4. **26**: 7,569
5. **5555**: 6,856
6. **9999**: 6,286
7. **82**: 6,046
8. **2323**: 3,622
9. **6666**: 2,735
10. **3333**: 2,644

Эти числа выглядят не очень случайными, не так ли? Прямо сейчас вы должны понять, что ваш случайный выбор нестандартного порта может быть не так уж и уникален. Порт 2222 столь же распространён, как и порт 8080 для HTTP, а также это стандартный порт для Kippo honeypot, хотя я сильно сомневаюсь, что столько людей запускают honeypot одновременно. Следующий по популярности порт – это 5000, который отличается от шаблона остальных портов (не симметричное или повторяющееся число). И примерно в этот же момент я понял, что Австралия – вторая страна по количеству запусков OpenSSH на нестандартных портах. Я присмотрелся к этой стране, и что же я увидел? Количество нестандартных портов SSH (5000) примерно равно количеству стандартных – 54,000 и 68,000 соответственно. Изучив некоторые баннеры, полученные из Австралии, можно понять, что есть отпечаток SSH, общий для многих их них:

```
5b:a2:5a:9a:91:28:68:9c:92:2b:9e:bb:7f:7c:2e:06
```

Похоже, что австралийский интернет-провайдер BigPond устанавливает и настраивает сетевое оборудование, которое не только запускает OpenSSH на порту 5000 (вероятнее всего, для удаленного управления), но также имеет те же SSH-ключи, которые установлены на этом оборудовании. Устройства же в австралийских сетях также запускают старую версию OpenSSH, которая была выпущена ещё 4 сентября 2007 года. Нет никакой гарантии, что запуск OpenSSH на стандартном порту сделал бы их более

защищёнными, но эти 54 тысячи устройств составляют 25% от общего числа OpenSSH серверов версии 4.7 в интернете (самая популярная версия OpenSSH – 5.3).

Карты Shodan

Сайт Карты Shodan предоставляют доступ к исследованию результатов поиска вместо главного сайта, построенного на текстовой основе. Он отображает до 1,000 результатов единовременно; и во время того, как вы приближаете или отдаляете карту, уточняет поисковый запрос для той области, на которой вы сфокусированы.

Все поисковые фильтры, работающие на основном сайте, также работают и здесь.

Стили карт

Существует несколько вариантов представления карт, доступных к просмотру, на ваше предпочтение. Нажмите кнопку  около строки поиска для получения списка этих вариантов.

Спутник (Satellite)

Спутник без легенды (Satellite without labels)

Светлый вариант отображения улиц (Streets light)

Темный вариант отображения улиц (Streets dark)

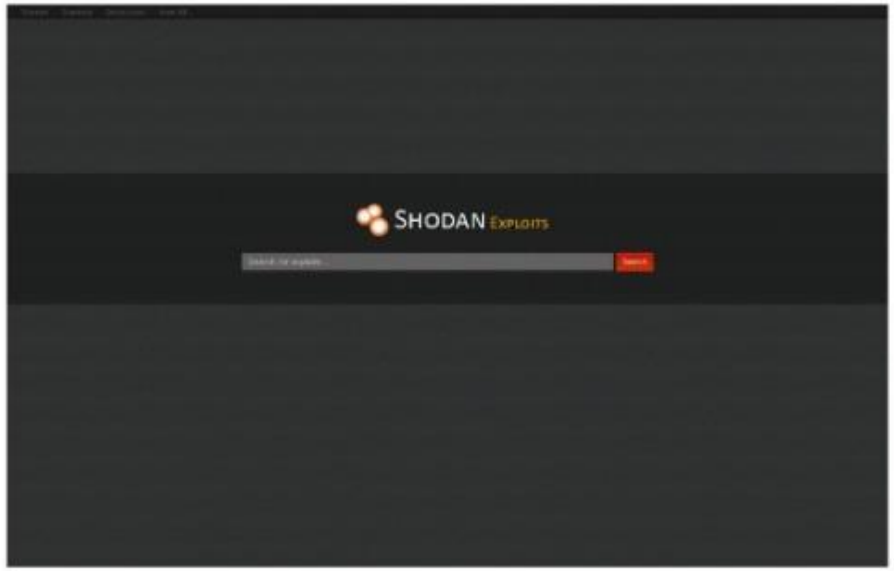
Зелёный вариант отображения улиц (Streets Green)

Красный вариант отображения улиц (Streets red)

И вариант вида пиратской карты ☺ (Pirate)

Уязвимости Shodan

Сайт Уязвимостей Shodan собирает уязвимости и эксплойты с CVE, Exploit DB и Metasploit чтобы сделать их доступными для поиска через веб-интерфейс.



Поисковые фильтры, доступные на этом сайте, отличаются от фильтров на прочих проектах Shodan, так как целью разработчиков было сделать их максимально простыми в использовании, где это возможно.

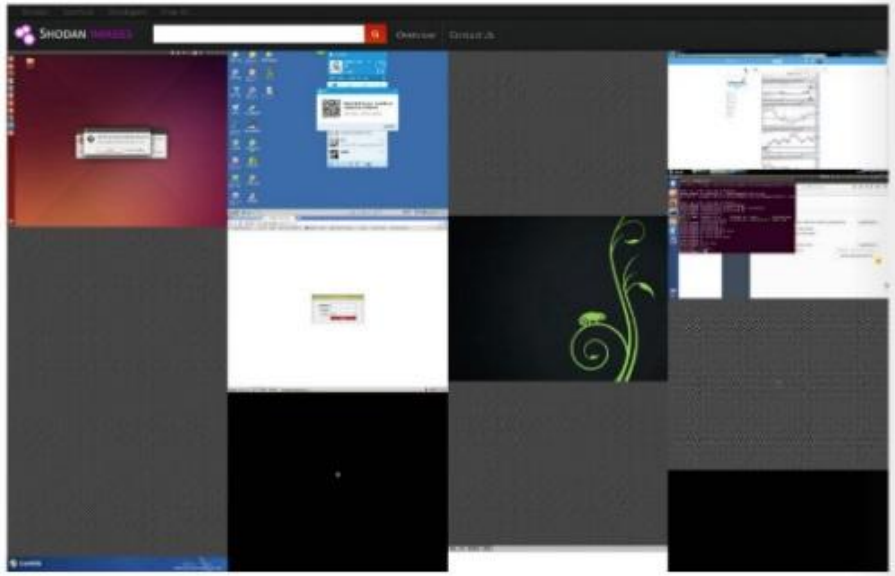
Внимание: по умолчанию, на сайте уязвимостей поиск происходит в полной информации эксплойта, включая метаданные. Этим проект отличается от Shodan, который ищет только в тексте баннера, если не указаны дополнительные фильтры.

Доступны следующие поисковые фильтры:

Название	Определение
author	Автор уязвимости/эксплойта
description	Описание
platform	Целевая платформа (php, windows, linux…)
type	Тип эксплойта (remote, dos…)

Изображения Shodan

Для того, чтобы с лёгкостью просмотреть все скриншоты, собираемые Shodan, можно посетить сайт изображений Shodan. Этот проект – это достаточно дружественный интерфейс фильтра **has_screenshot**.



Строка поиска наверху использует тот же синтаксис, что и поисковый движок Shodan. Наиболее результативно будет использовать эту строку для фильтрации по организации или блоку IP-адресов (netblock). Однако, можно также использовать фильтрацию по типу изображений.

Данные изображений собираются из 5 разных источников:

- VNC
- Удалённый рабочий стол (RDP)
- RTSP
- Веб-камеры
- X Windows

Каждый источник изображений получается из отделного порта/службы и, соответственно, имеет свой баннер. Это значит, что если вы хотите видеть изображения только с веб-камер, вам нужно искать так:

```
HTTP
```

Для поиска VNC необходимо использовать запрос **RFB**, а для поиска **RTSP** просто ввести RTSP.

Также можно искать изображения, используя основной сайт Shodan или карты Shodan по поисковому запросу **has_screenshot:true**.\

Практические упражнения: Сайты Shodan

Упражнение 1

Найдите сайт 4SICS, используя Shodan.

Подсказка: смотри приложение В для списка поисковых фильтров.

Упражнение 2

Сколько служб VNC в интернете позволяют анонимный доступ?

Упражнение 3

Сколько IP в Швеции уязвимы к Heartbleed и до сих пор поддерживают SSLv3?
Сколько IP уязвимы к Heartbleed в вашей организации?

Упражнение 4

Найдите все промышленные системы управления в вашем городе.

Упражнение 5

Какое средство удалённого администрирования (RAT) наиболее популярно в США?

Внешние инструменты.

Интерфейс командной строки Shodan

Начало работы

Интерфейс командной строки shodan поставляется вместе с официальной библиотекой Python для Shodan, из чего следует, что если вы используете последнюю версию библиотеки, у вас уже есть доступ к CLI. Для установки нового инструмента, просто выполните команду:

```
easy_install shodan
```

Когда установка инструмента завершится, нужно его инициализировать вашим ключом API:

```
shodan init YOUR_API_KEY
```

Посетите https://account.shodan.io чтобы получить API ключ для вашего аккаунта.

alert

Команда **alert** даёт вам возможность создавать, перечислять, очищать и удалять оповещения сети.

convert

Эта команда конвертирует сжатый файл JSON, сгенерированный Shodan в другой файловый формат. На данный момент поддерживается конвертация в **kml** и **csv**.

count

Команда оказывает количество результатов для поискового запроса

```
$ shodan count microsoft iis 6.0
5360594
```

download

Осуществление поиска по Shodan и скачивание результатов в файл, где каждая строка — это баннер JSON (см. приложение А).

По умолчанию, будут сохранены только 1,000 результатов. Если вам требуется большее количество, обратите своё внимание на флаг `--limit`.

Команда download – это та команда, которую вы будете использовать наиболее часто для получения результатов из Shodan, так как она позволяет и скачать их, и в дальнейшем обработать, используя команду parse. Поскольку пролистывание результатов использует ваши query credits, будет разумно всегда сохранять результаты, чтобы не тратить кредиты на те же действия позже.

```
$ shodan download microsoft-data microsoft iis 6.0
Search query:                   microsoft iis 6.0
Total number of results:        5310596
Query credits left:             100000
Output file:                    microsoft-data.json.gz
  [#######------------------------------]   20%  00:00:20
```

host

Просмотр информации о хосте, например его географическом расположении, открытых портов и организации, владеющей IP-адресом.

```
$ shodan host 189.201.128.250
```

```
189.201.128.250
Hostnames:              customer-250.xertix.com
City:                   Mexico
Country:                Mexico
Organization:           Metro Net, S.A.P.I. de C.V.
Number of open ports:   1
Vulnerabilities:        Heartbleed

Ports:
    443 Fortinet FortiGate 50B or FortiWifi 80C firewall http config
        |-- SSL Versions: SSLv3, TLSv1, TLSv1.1, TLSv1.2
        |-- Diffie-Hellman Parameters:
                Bits:           1024
                Generator:      2
                Fingerprint:    RFC2409/Oakley Group 2
```

honeyscore

Проверка, является ли IP-адрес honeypot'ом, выдающим себя за промышленную систему управления.

```
$ shodan honeyscore 41.231.95.212
```

info

Получение основной информации о вашем тарифе API, включающей в себя количество запросов и сканирований, оставшихся вам в этом месяце.

```
$ shodan info
Query credits available: 5102
Scan credits available: 249
```

myip

Команда показывает ваш IP-адрес, который выходит в интернет.

```
$ shodan myip
199.30.49.210
```

parse

Команда parse используется для анализа файла, полученного командой download. Она позволяет вам отфильтровать поля, интересующие вас, конвертировать JSON в CSV, и дружелюбна во взаимодействии с прочими скриптами.

Команда ниже показывает IP-адрес, порт и организацию в формате CSV для данных Microsoft-IIS, загруженных ранее:

```
$ shodan parse --fields ip_str,port,org --separator , microsoft-data.json.gz
```

```
216.28.245.171,80,Web Force Systems,
103.41.16.147,80,
218.244.142.211,80,China Network Information Center,
81.22.98.166,80,Kriter Internet Hiz.Ltd.Sti.,
75.149.30.138,443,Comcast Business Communications,
23.108.235.233,80,Nobis Technology Group, LLC,
207.57.69.157,8080,Verio Web Hosting,
66.129.113.13,80,Peak 10,
168.143.6.120,8080,Verio Web Hosting,
218.0.3.56,80,China Telecom Ningbo,
104.202.81.231,80,
98.191.178.20,443,Cox Communications,
108.186.164.90,80,Peg Tech,
23.105.63.236,80,Nobis Technology Group, LLC,
67.227.184.237,8443,Smash Data Design,
107.163.173.34,80,
185.22.198.84,80,Nexto SAS,
72.29.22.40,80,Cybercon,
216.119.64.188,80,CrystalTech Web Hosting,
104.221.145.60,80,
198.171.51.81,8080,Verio Web Hosting,
209.10.173.10,443,Quality Technology Services, N.J., LLC,
```


scan

Команда scan обеспечивает доступ к нескольким подкомандам, наиболее важной из которых является submit, позволяющей вам осуществлять сканирование через Shodan.

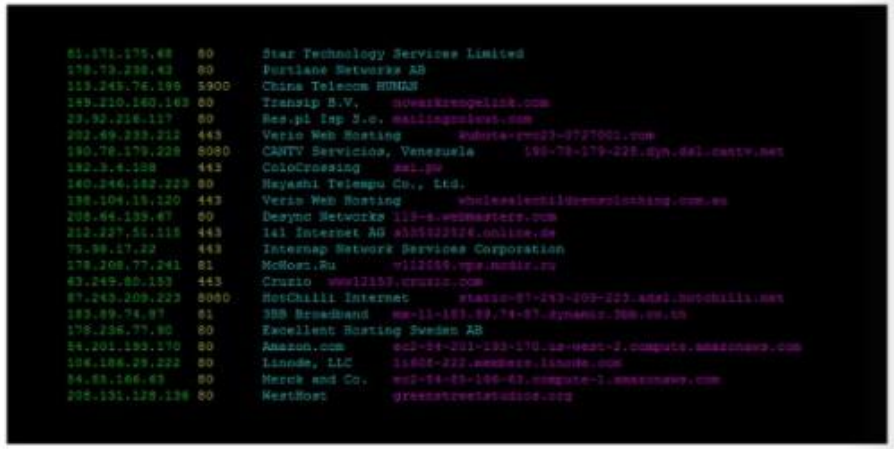
```
$ shodan scan submit 202.69.165.20
```

```
achillean@demo:~$ shodan scan submit 202.69.165.20

Starting Shodan scan at 2015-07-24 04:14 (100000 scan credits left)

202.69.165.20
  Country                  Philippines
  City                     Pampanga
  Organization             ComClark Network & Technology Corp.

  Open Ports:
    80/tcp
    443/tcp
    902/tcp    VMware Authentication Daemon (1.10)
```

search

Эта команда позволяет вам производить поиск по Shodan и просматривать результаты в терминале. По умолчанию она показывает IP, порт, имена хостов и данные. Можно использовать параметр `--fields` для отображения тех полей баннера, в которых вы заинтересованы.

```
$ shodan search --fields ip_str,port,org,hostnames microsoft iis 6.0
```



stats

Команда stats позволяет вам прописать фасеты для поискового запроса.

Например, следующая команда показывает страны, в которых расположено наибольшее количество веб-серверов Apache:

```
$ shodan stats --facets country apache
Top 10 Results for Facet:  country
US                         8,336,729
DE                         4,512,172
CN                         1,470,434
JP                         1,093,699
GB                         832,221
NL                         684,432
FR                         667,871
CA                         581,630
RU                         324,698
BR                         266,788
```

stream

Данная команда предоставляет доступ к просмотру в настоящем времени потока данных, собираемых сканерами Shodan.


```
achillean@demo:~$ shodan stream --help
Usage: shodan stream [OPTIONS]

  Stream data in real-time.

Options:
  --color / --no-color
  --fields TEXT          List of properties to output.
  --separator TEXT       The separator between the properties of the search
                         results.
  --limit INTEGER        The number of results you want to download. -1 to
                         download all the data possible.
  --datadir TEXT         Save the stream data into the specified directory as
                         .json.gz files.
  --ports TEXT           A comma-separated list of ports to grab data on.
  --quiet                Disable the printing of information to the screen.
  --streamer TEXT        Specify a custom Shodan stream server to use for
                         grabbing data.
  -h, --help             Show this message and exit.
```

Команда поддерживает много различных флагов, однако действительно важно упомянуть лишь 3 из них:

--datadir

Флаг **--datadir** позволяет вам уточнить папку, в которую будет сохраняться поток данных. Файлы, сгенерированные в директории **--datadir**, будут иметь следующий формат имён:

```
YYYY-MM-DD.json.gz
```

К примеру, файл будет называться «2016-01-15.json.gz». Каждый день, новый файл будет автоматически генерироваться по мере приёма потока данных, пока вы его не отключите. Следующая команда скачивает все данные с потока в реальном времени и сохраняет их в директории /var/lib/shodan/:

```
shodan stream --datadir /var/lib/shodan/
```

--limit

Флаг **--limit** указывает, сколько результатов будет скачано. По умолчанию, команда stream продолжает работу бесконечно, пока вы не выйдете из программы. Однако, если вы заинтересованы в сборе только образца данных, то данный флаг гарантирует сбор небольшого количества записей. Для примера:

```
shodan stream --limit 100
```

Эта команда присоединится к потоку данных, выведет первые 100 полученных результатов и затем прекратит работу.

--ports

Этот флаг принимает как значения разделённый запятыми список портов, чтобы позволить вам принимать в реальном времени только данные от этих портов. Следующая команда будет принимать баннеры, собранные от портов 80 или 8080:

```
shodan stream --ports 80,8080
```

Пример: анализ сети

В основном Shodan используется, чтобы получить представление о том, что запущено в вашей общедоступной сети. Инструмент командной строки Shodan может помочь вам быстро понять, с чем вы имеете дело. Для лучшего понимания мы, к примеру, рассмотрим диапазон 78.13.0.0/16. Чтобы начать, давайте посмотрим, сколько служб имеют выход в интернет:

```
$ shodan count net:78.13/16
4363
```

Команда count предоставит нам общее количество баннеров, которые Shodan собрал из подсети 78.13/16. На момент написания руководства, это было 4,363 результата. Команда дает нам понимание того, насколько велик объём публичной сети организации, но не раскрывает информацию о том, для чего предназначены найденные службы. Таким образом, следующим нашим шагом будет распределение открытых портов сети:

```
$ shodan stats --facets port net:78.13/16
Top 10 Results for Facet: port
7547                            1,192
80                                543
443                               264
8080                              191
1900                              147
53                                122
49152                              83
81                                 64
22                                 61
21                                 38
```

10 наиболее часто встречающихся портов это неплохая отправная точка, но в идеале мы хотим получить полную картину распределения. Чтобы это сделать, мы уточним максимальное количество фасетов, которые должно быть возвращено откликом:

```
$ shodan stats --facets port:100000 net:78.13/16
Top 1060 Results for Facet: port
7547                            1,192
80                                543
443                               264
8080                              191
1900                              147
53                                122
49152                              83
81                                 78
22                                 78
21                                 59
5060                               55
1723                               49
554                                48
3128                               36
5555                               33
8443                               31
8000                               28
8081                               25
5000                               23
82                                 21
```



```
6881                               19
8089                               17
500                                16
83                                 16
37777                              14
88                                 13
5353                               12
4500                               12
5001                               10
...
```

Мы получили 1060 уникальных портов, которые были обнаружены открытыми в сети. Был сделан запрос на огромное максимальное количество фасетов (10,000), несмотря на то, что это гораздо больше чем примерно 300 портов, которые насканировал Shodan. На данном этапе мы имеем небольшое количество областей, которые можно исследовать дальше. Во-первых, наиболее часто встречающийся порт 7547, который используется модемами для обновления их настроек, и который появлялся в новостях в связи с вопросами безопасности. Также имеются много веб-серверов, работающих на нестандартных портах (8080, 81, 82, 8443 и так далее) которые тоже следует изучить. Например, это веб-сервера, на которых работают службы на нестандартных портах:

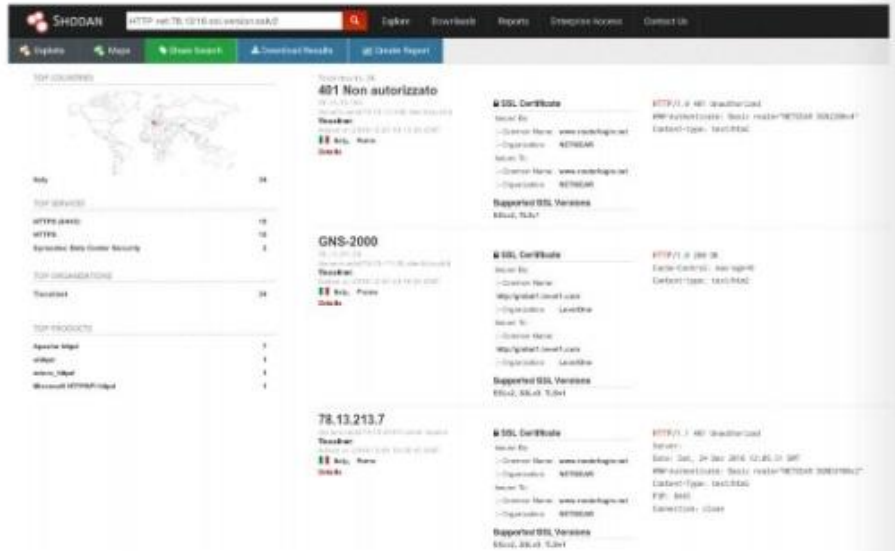
```
$ shodan stats --facets product "HTTP net:78.13/16 -port:80,443"
Top 10 Results for Facet: product
Apache httpd                       39
micro_httpd                        22
GoAhead-Webs httpd                 21
nginx                              18
Netwave IP camera http config      16
Boa HTTPd                              13
uc-httpd                                5
Allegro RomPager                    4
uhttpd                                  3
mt-daapd DAAP                           2
```

> Внимание: поисковый запрос заключён в кавычки чтобы предотвратить Bash от восприятия **-port** как флага команды **shodan**.

Другой необходимый вопрос – это понимание, как используется SSL в сети. Для этого мы можем использовать SSL-тестирование Shodan, которое выполняется автоматически для всех служб, поддерживающих SSL (HTTPS, POP3, IMAP и проч.) Чтобы начать работу, давайте посмотрим, какие версии SSL/TLS наиболее часто используются веб-серверами:

```
$ shodan stats --facets ssl.version HTTP net:78.13/16
Top 5 Results for Facet:
ssl.version
tlsv1                             283
tlsv1.2                           198
tlsv1.1                           187
sslv3                              88
sslv2                              34
```

Хорошие новости для нас заключаются в том, что в основном серверы используют TLS1.0 и выше. Но, есть несколько устройств, которые поддерживают древний, устаревший SSLv2.



По всей видимости, эти устройства NetGear являются основными пользователями SSLv2-совместимых служб в этой сети.

Пример: исследование Telnet

Давайте предположим, что мы хотим провести исследование устройств в интернете, использующих Telnet. Для начала, мы можем соединить все ранее упомянутые команды в следующую:

```
mkdir telnet-data
shodan stream --ports 23,1823,2323 --datadir telnet-data/ --limit 10000
```

Сперва, создаём папку с именем **telnet-data** для хранения данных Telnet. Затем делаем запрос на 10,000 записей (**--limit 10000**) из потока данных на стандартных Telnet портах (**--ports 23, 1023, 2323**) и сохраняем результаты в предварительно созданный каталог (**--datadir telnet-data/**).

Аддон Maltego

Maltego – это приложение с открытым исходным кодом, используемое для обнаружения и обработки данных. Оно позволяет вас визуально исследовать и сопоставлять данные из множества источников.

Аддон Shodan для Maltego предоставляет 2 новых сущности (entities) – Service и Exploit и 5 новых преобразований (transforms):

- searchShodan
- searchShodanbyDomain
- searchShodanbyNetblock
- toShodanHost
- searchExploits

Плагины браузера

Существуют плагины, как на Chrome, так и на Firefox, которые позволяют вам увидеть, какие службы используются сайтом.

Практические упражнения: интерфейс командной строки

Упражнение 1

Скачайте IP-адреса, уязвимые к Heartbleed в Швеции и Норвегии, используя CLI.

Отфильтруйте результаты для Швеции и сохраните их в отдельном файле.

> Внимание: распакуйте файл и изучите необработанные данные, чтобы увидеть, как выглядит необработанный отклик на тестирование на Heartbleed.

Упражнение 2

Скачайте 1,000 последних баннеров, используя поток данных в реальном времени, и определите их локацию, используя Google Maps

> Подсказка: shodan convert

Упражнение 3

Напишите скрипт для скачивания списка известных вредоносных IP-адресов и заблокируйте любой исходящий на них трафик.

> Подсказка: iptables -A OUTPUT -d x.x.x.x -j DROP

API Разработчика

Shodan предоставляет API-интерфейс разработчика (https://developer.shdan.io/api), который используется для программного доступа к собираемой информации. Все веб-сайты и инструменты, включая основной сайт Shodan, работают от API. Все, что можно сделать через веб-сайт, может быть с не меньшим успехом выполнено с помощью вашего собственного кода.

API делится на 2 части: REST API и Streaming API. API REST предоставляет способы поиска по Shodan, поиска хостов, получения полной информации о запросах и множество вспомогательных методов, упрощающих разработку. Streaming API предоставляет необработанный поток данных в режиме реального времени, который собирает Shodan в данный момент. Существует несколько потоков данных, на которые можно подписаться, но эти данные не доступны для включения в поиск или для другого взаимодействия. Они используются самой системой Shodan.

> Внимание: Только пользователи с подпиской на API могут получить доступ к Streaming API.

Ограничения использования

Есть три ограничения на использование API, зависящие от вашего тарифного плана:

1. Поиск. Чтобы ограничить число поисков, которые могут быть выполнены в течение месяца, Shodan использует query credits. Один query credit используется при поиске с использованием фильтров, либо, когда вы переходите с первой страницы результатов поиска. К примеру, если вы ищете «apache», кредиты не используются, ели же запрос будет вида «apache country:us», используется 1 кредит. Так же, один кредит будет израсходован, если вы перейдёте на вторую страницу любого из вышеуказанных поисков.
2. Сканирование. API сканирования по запросу использует scan credits, чтобы ограничить количество хостов, которые вы можете попросить Shodan просканировать. На каждый запрос будет тратиться один кредит.
3. Сетевые оповещения. Число IP-адресов, за которыми можно следить, используя оповещения, ограничено вашей подпиской на API. Только пользователи с платными тарифами. Также, вы не можете настроить больше 100 оповещений на аккаунт.

> Внимание: query credits и scan credits восстанавливаются в начале каждого месяца.

Представление фасетов

Фасеты предоставляют сводную информацию о конкретной области интересующего вас баннера. Фильтры позволяют сузить результаты поиска, в то время как фасеты, напротив, позволяют увидеть результаты издалека, формируя большую картину. К примеру, основной сайт Shodan использует фасеты, чтобы предоставить статистическую информацию в левой части результатов поиска:

TOP COUNTRIES

United States	430,835
Germany	139,912
China	111,143
Russian Federation	101,453
Costa Rica	84,542

TOP ORGANIZATIONS

Abdicar Communications, S.A.	59,590
OVH SAS	33,674
Cogent Communications	27,988
Korea Telecom	27,682
Thorn Communications	19,095

К использованию доступен большой список фасетов (см. приложение C), и, используя API, вы можете контролировать интересующие вас аспекты. Например, поиск port:22 и фасет ssh.fingerprint в результате предоставят вам информацию, какие SSH-отпечатки чаще всего встречаются в Интернете.
Фасеты часто являются отправной точкой для исследования проблем в масштабе всемирной сети, например, дублирование ключей SSH, ошибки недобросовестных хостинг-провайдеров или бреши в безопасности сети на всей территории страны.

На данный момент фасеты можно использовать только из API и интерфейса командной строки Shodan.

Начало работы

Все примеры будут показаны в Python, и предполагается, что у вас есть доступ к командной строке, хотя существуют библиотеки и клиенты Shodan, доступные на других языках.

Для установки библиотеки Shodan для Python, выполните следующую команду:

```
easy_install shodan
```

Если у вас уже установлена библиотека, и вы хотите её обновить, используйте команду:

```
easy_install -U shodan
```

Инициализация

Первая вещь, которую обязательно сделать – это инициализация API Shodan:

```
import shodan
api = shodan.Shodan('YOUR API KEY')
```

Где YOUR API KEY – это API ключ вашего аккаунта, который вы можете получить здесь:

https://account.shodan.io

Поиск

Теперь, когда наш объект API настроен, мы готовы к проведению поиска:

```
# Wrap the request in a try/ except block to catch errors
try:
	# Search Shodan
	results = api.search('apache')

	# Show the results
	print 'Results found: %s' % results['total']
	for result in results['matches']:
		print 'IP: %s' % result['ip_str']
		print result['data']
		print ''
except shodan.APIError, e:
	print 'Error: %s' % e
```

Выполняя код, в первую очередь мы используем метод `Shodan.search()` для объекта `api`, получая в ответ словарь с результатами. Затем мы выводим общее количество полученных результатов, и, как завершающий шаг, мы возвращаемся к началу результатов и выводим на экран их IP и баннер. Каждая страница результатов поиска содержит до 100 результатов.

Эта функция может выдать нам намного больше информации. Чуть ниже я приведу сокращенный пример словаря, который мы получаем при использовании `Shodan.search`:

```
{
      'total': 8669969,
      'matches': [
            {
                  'data': 'HTTP/1.0 200 OK\r\nDate: Mon, 08 Nov 2010 05:09:59
                  GMT\r\nSer...',
                  'hostnames': ['pl4t1n.de'],
                  'ip': 3579573318,
                  'ip_str': '89.110.147.239',
                  'os': 'FreeBSD 4.4',
                  'port': 80,
                  'timestamp': '2014-01-15T05:49:56.283713'
            },
            ...
      ]
}
```

Смотри приложение A для того, чтобы просмотреть полный список свойств, которые может содержать баннер.

Внимание: по умолчанию, некоторые большие поля баннер, таких как «html», усекаются, чтобы уменьшить использование пропускной способности. Если вы хотите получать всю информацию, просто отключите этот параметр, введя `minify=False`. Например, следующий запрос для поиска анонимных VNC-служб гарантирует, что вы получите как ответ на неё максимально подробную информацию:

```
results = api.search('has_screenshot:true', minify=False)
```

Также рекомендуется объединять все запросы API в предложении try/except, поскольку любая ошибка будет вызывать исключения и остановку работы. Но, чтобы не перегружать нас информацией, я не буду в это вдаваться.

Скрипт выше предоставит нам результаты только с первой страницы. Чтобы мы могли увидеть вторую или последующие страницы, необходимо использовать параметр `page` при выполнении запроса:

```
results = api.search('apache', page=2)
```

Или же, если вы хотите пролистать все возможные результаты, есть один способ, который сделает вашу жизнь проще. Это `search_cursor()`:

```
for banner in api.search_cursor('apache'):
      print(banner['ip_str']) # Print out the IP address for each banner\
```

Важно: этот параметр будет давать отклик только в виде баннеров, и не позволяет использовать фасеты. Используйте его только для пролистывания результатов.

Поиск на хосте

Чтобы определить, что Shodan может найти на конкретном IP-адресе, мы можем использовать функцию `Shodan.host()`:

```
# Lookup the host
host = api.host('217.140.75.46')

# Print general info
print """
      IP: %s
      Organization: %s
      Operating System: %s
""" % (host['ip_str'], host.get('org', 'n/a'), host.get('os', 'n/a'))

# Print all banners
for item in host['data']:
      print """
            Port: %s
            Banner: %s
      """ % (item['port'], item['data'])
```

По умолчанию, Shodan будет выводить только ту информацию по хосту, которая собрана совсем недавно. Чтобы просмотреть полную историю IP-адреса, используйте параметр `history`. Например, запрос:

```
host = api.host('217.140.75.46', history=True)
```

Выдаст в ответ все баннеры, включая службы, которые больше не активны на данном хосте.

Сканирование

Shodan сканирует интернет как минимум раз в месяц, но если вам необходимо запросить его просканировать сеть немедленно, вы можете сделать это, используя возможность API «сканирование по требованию» (on-demand scanning).

В отличие от сканирования инструментом, такого как, к примеру, NMap, сканирование Shodan происходит асинхронно. Это значит, что после того, как вы отправляете запрос в Shodan, вы не получите результаты немедленно. На этом этапе разработчик решает, как именно собирать информацию: исследуя данные IP-адреса, выполняя поиск по Shodan либо подключаясь к живому потоку данных. Интерфейс командной строки Shodan создаёт временное оповещение сети после инициализации сканирования, а затем ждёт результатов, которые придут в потоке данных.

```
scan = api.scan('198.20.69.0/24')
```

Также, возможно сделать запрос на сканирование списка сетей, предоставив список адресов в CIDR-нотации:

```
scan = api.scan(['198.20.49.30', '198.20.74.0/24'])
```

После приёма запроса на сканирование, API вернёт следующую информацию:

```
{
	'id': 'R2XRT5HH6X67PFAB',
	'count': 1,
	'credits_left': 5119
}
```

Инструмент API предоставит уникальный идентификатор (`id`), который вы можете использовать в целях слежения, общее число (`count`) IP-адресов, принятых на сканирование, и остаток scan credits (`credits_left`).

Поток данных в реальном времени

Streaming API это служба, основанная на HTTP, которая выдаёт в реальном времени отклик в виде потока данных, собираемых Shodan. Никакие возможности поиска здесь не доступны, это лишь лента, состоящая из всех данных, которые собирают сканеры.

Например, это скрипт, который выведет поток баннеров от устройств, уязвимых для FREAK (CVE-2015-0204):

```
def has_vuln(banner, vuln):
	if 'vulns' in banner['opts'] and vuln in banner['opts']['vulns']:
		return True
	return False

for banner in api.stream.banners():
	if has_vuln(banner, 'CVE-2015-0204'):
		print banner
```

Чтобы сберечь пространство и пропускную способность, множество свойств в баннере являются опциональными. Чтобы сделать работу с ними более простой, лучше объединить доступ к свойствам в одну функцию. В примере выше, метод `has_vuln()` проверяет, уязвима ли служба к предоставленной CVE или нет.

> Внимание: Тарифы API стандартного плана имеют доступ лишь к 1% потока данных. Полный доступ открыт лишь покупателям лицензии данных (data license).

Сетевые оповещения

Такие оповещения – это лента данных реального времени, собираемая Shodan для определённого сетевого диапазона. Чтобы начать работу с ними, нужно выполнить два шага:

Создание сетевого оповещения

Чтобы создать оповещение, вам нужно предоставить `name` и `network range`. Имя должно быть поясняющим, чтобы вы знали за чем следит данное оповещение, или же причину его создание.

```
alert = api.create_alert('Production network', '198.20.69.0/24')
```


Как и в методе `scan()`, вы так же предоставляете список сетевых диапазонов монитору:

```
alert = api.create_alert('Production and Staging network', [
	'198.20.69.0/24',
	'198.20.70.0/24',
])
```

> Внимание: с помощью сетевых оповещений можно следить только за ограниченным числом IP, и общее число таких оповещений для аккаунта не может превышать 100.

Достаточно полезная вещь при комбинировании оповещений и сканировании – это установить время истечения оповещения:

```
alert = api.create_alert('Temporary alert', '198.20.69.0/24', expires=60)
```

Оповещение, приведённое выше, будет активно 60 секунд, а затем будет отключено, и не может больше использоваться.

После успешного создания оповещения, API выдаст следующее:

```
{
	"name": "Production network",
	"created": "2015-10-17T08:13:58.924581",
	"expires": 0,
	"expiration": null,
	"filters": {
		"ip": ["198.20.69.0/24"]
},
	"id": "EPGWQG5GEELV4799",
	"size": 256
}
```

Подписка

После того, как оповещение было создано, оно готово к использованию в виде потока данных для сети.

```
for banner in api.stream.alert(alert['id']):
	print banner
```

Как и в стандартном потоке данных, метод `alert()` предоставляет итератор, где каждый элемент является баннером, так как он собирается сканерами Shodan. Единственным аргументом, который требует этот метод – это идентификатор оповещения (alert ID), который мы получили при создании сетевого оповещения.

Использование интерфейса командной строки Shodan

Здесь мы коротко изучим, как можно выполнить код Python, используя Shodan CLI. Начнём с удаления всех существующих оповещений:

ВНИМАНИЕ!

Команда clear удалит все оповещения, созданные на аккаунте.

```
$ shodan alert clear
Removing Scan: 198.20.69.0/24 (ZFPSZCYUKVZLUT4F)
Alerts deleted
```

Теперь нужно убедиться, что не осталось оповещений:

```
$ shodan alert list
You haven't created any alerts yet.
```

Теперь – создаём новое оповещение:

```
$ shodan alert create "Temporary alert" 198.20.69.0/24
Successfully created network alert!
Alert ID: ODMD34NFPLJBRSTC
```

Последний шаг – это подписка на оповещение и сохранение данных, полученных от него. Чтобы вывести эти данные потоком, вводим идентификатор оповещения **ODMD34NFPLJBRSTC** в команду **stream**.

```
$ mkdir alert-data
$ shodan stream --alert=ODMD34NFPLJBRSTC --datadir=alert-data
```

Командой выше мы принимаем поток данных оповещения с идентификатором **ODMD34NFPLJBRSTC** и сохраняем их в папку с названием **alert-data**. Каждый день в этой папке будет генерироваться новый файл, содержащий баннеры, собранные за день. То есть, нам не нужно беспокоиться о ротации файлов, команда **stream** сделает это за нас. Через несколько дней папка будет выглядеть так:

```
$ ls alert-data
2016-06-05.json.gz
2016-06-06.json.gz
2016-06-07.json.gz
```

Пример: публичные данные MongoDB

MongoDB – это популярная база данных NoSQL, и в течении длительного времени она не требовала никакой аутентификации. Это привело к тому, что многие экземпляры MongoDB были публично доступны в интернете. Shodan захватывает баннеры этих баз данных, которые содержат множество информации об объектах внутри баз. Ниже мы видим отрывок из такого баннера:

```
MongoDB Server Information
...
{
        "ok": 1.0,
        "tokumxAuditVersion": "unknown",
        "bits": 64,
        "tokukvVersion": "unknown",
```



```
        "tokumxVersion": "2.0.2",
        "javascriptEngine": "V8",
        "version": "2.4.10",
        "versionArray": [
                2,
                4,
                10,
                0
        ],
        "debug": false,
        "compilerFlags": "-fPIC -fno-strict-aliasing -ggdb -Wall -Wsign-compare -Wno\
-unknown-pragmas -Winvalid-pch -pipe -Wnon-virtual-dtor -Woverloaded-virtual -Wn\
o-unused-local-typedefs -fno-builtin-memcmp -O3",
        "maxBsonObjectSize": 16777216,
        "sysInfo": "Linux vps-vivid-x64-04 2.6.32-042stab106.6 #1 SMP Mon Apr 20 14:\
48:47 MSK 2015 x86_64 x86_64 x86_64 GNU/Linux BOOST_LIB_VERSION=1_55",
        "loaderFlags": " ",
        "gitVersion": "unknown"
},
...
```

В общих чертах, баннер состоит из хэдера «MongoDB Server Information», за которым следуют 3 объекта JSON, разделенные запятыми. Баннер будет содержать в себе «authentication enabled», если сервер требует введения учетных данных. Каждый объект JSON содержит различную информацию о базе данных, и я рекомендую вам изучить полный баннер на Shodan (предупреждаю, он очень длинный), выполнив такой запрос:

```
product:MongoDB metrics
```

> Внимание: поисковый термин *metrics* гарантирует, что мы получим только те экземпляры MongoDB, которые не требуют аутентификации.

Теперь можно использовать этот баннер, чтобы определить, какие имена баз данных наиболее популярны, и насколько много данных находится в открытом доступе в интернете! Наш рабочий процесс будет выглядеть так:

1. Скачать все баннеры MongoDB
2. Обработать скачанный файл и вывести 10 наиболее часто встречающихся имён баз данных и общий объём данных

Скачать данные достаточно просто, используя интерфейс командной строки Shodan:

```
shodan download --limit -1 mongodb-servers.json.gz product:mongodb
```

Команда выше даёт установку скачать все результаты (**--limit -1**) для поискового запроса **product:mongodb** в файл с именем **mongodb-servers.json.gz.** Также, вы можете скачать результат команды из раздела Extras книги на Leanpub. Теперь нам нужен простой Python-скрипт для обработки файла данных Shodan. Чтобы простым способом перебрать все результаты в файле, используем метод **shodan.helpers.iterate_files()**:

```
import shodan.helpers as helpers
import sys

# The datafile is the 1st argument to the command
datafile = sys.argv[1]

for banner in helpers.iterate_files(datafile):
```

```
	# Now we have the banner
```

Поскольку каждый баннер – это запись JSON с добавленным хэдером, давайте обработаем баннер в родной словарь Python, используя библиотеку **simplejson**:

```
# Strip out the MongoDB header added by Shodan
data = banner['data'].replace('MongoDB Server Information\n', '').split('\n},\n'\
)[2]

# Load the database information
data = simplejson.loads(data + '}')
```

Последний шаг, что нам остался – это продолжать слежение за общим выложенным объёмом данных и самыми используемыми именами баз данных:

```
total_data = 0
databases = collections.defaultdict(int)

...

	# Then in the loop
	# Keep track of how much data is publicly accessible
	total_data += data['totalSize']

	# Keep track of which database names are most common
	for db in data['databases']:
		databases[db['name']] += 1
```

В Python есть полезный класс **collections.defaultdict**, который автоматически создаёт стандартное значение ключа словаря, если таковый не существует. И нам нужно только получить доступ к свойствам MongoDB-баннера **totalSize** и **databases**, чтобы получить необходимую нам информацию. Наконец, нужно просто вывести актуальные результаты:

```
print('Total: {}'.format(humanize_bytes(total_data)))

counter = 1
for name, count in sorted(databases.iteritems(), key=operator.itemgetter(1), rev\
erse=True)[:10]:
	print('#{}\t{}: {}'.format(counter, name, count))
	counter += 1
```

Этой командой сначала выводится общие количество доступных данных, и мы используем метод **humanize_bytes()**, чтобы конвертировать байты в человекочитаемый формат GB/ MB/ и так далее. Затем мы сортируем **databases** в **обратном** порядке по количеству раз, когда было замечено определенное имя базы данных (**key=operator.itemgetter(1)**) и получаем 10 первых результатов (**[:10]**).

Ниже приведен скрипт целиком, который читает файл данных Shodan и анализирует баннер:


```
import collections
import operator
import shodan.helpers as helpers
import sys
import simplejson

def humanize_bytes(bytes, precision=1):
	"""Return a humanized string representation of a number of bytes.

	Assumes `from __future__ import division`.

	>>> humanize_bytes(1)
	'1 byte'
	>>> humanize_bytes(1024)
	'1.0 kB'
	>>> humanize_bytes(1024*123)
	'123.0 kB'
	>>> humanize_bytes(1024*12342)
	'12.1 MB'
	>>> humanize_bytes(1024*12342,2)
	'12.05 MB'
	>>> humanize_bytes(1024*1234,2)
	'1.21 MB'
	>>> humanize_bytes(1024*1234*1111,2)
	'1.31 GB'
	>>> humanize_bytes(1024*1234*1111,1)
	'1.3 GB'
	"""
	abbrevs = (
		(1<<50L, 'PB'),
		(1<<40L, 'TB'),
		(1<<30L, 'GB'),
		(1<<20L, 'MB'),
		(1<<10L, 'kB'),
		(1, 'bytes')
	)
	if bytes == 1:
		return '1 byte'
	for factor, suffix in abbrevs:
		if bytes >= factor:
			break
	return '%.*f %s' % (precision, bytes / factor, suffix)
total_data = 0
databases = collections.defaultdict(int)
for banner in helpers.iterate_files(sys.argv[1]):
	try:
		# Strip out the MongoDB header added by Shodan
		data = banner['data'].replace('MongoDB Server Information\n',
	'').split(\'\n},\n')[2]

		# Load the database information
		data = simplejson.loads(data + '}')

		# Keep track of how much data is publicly accessible
		total_data += data['totalSize']

		# Keep track of which database names are most common
		for db in data['databases']:
			databases[db['name']] += 1
	except Exception, e:
		pass

print('Total: {}'.format(humanize_bytes(total_data)))
```

```
counter = 1
for name, count in sorted(databases.iteritems(), key=operator.itemgetter(1), rev\
erse=True)[:10]:
        print('#{}\t{}: {}'.format(counter, name, count))
        counter += 1
```

Здесь мы видим образец информации, которую выдаёт скрипт:

```
Total: 1.8 PB
#1      local: 85845
#2      admin: 67648
#3      test: 24983
#4      s: 5121
#5      config: 4329
#6      proxy: 2045
#7      research: 2007
#8      seolib_new: 2001
#9      traditional: 1998
#10     simplified: 1998
```


Практические упражнения: Shodan API

Упражнение 1

Напишите скрипт для слежения за сетью, используя Shodan и вышлите оповещения

Упражнение 2

Напишите скрипт для выгрузки последних изображений в папку

> Подсказка: изображения зашифрованы с использованием base64. Python может с лёгкостью декодировать их в двоичный код используя: image_string.decode('base64')

Промышленные системы управления

Если коротко, промышленные системы управления (ICS) - это компьютеры, которые контролируют окружающий мир. Они отвечают за управление кондиционерами в вашем офисе, турбинами на электростанции в вашем городе, освещением в театре или роботами на заводе.

Исследования, проведенные с 2012 по 2014 годы Project SHINE (SHodan INtelligence Extraction), чётко указывают, что в Интернете существует не менее 2 миллионов общедоступных устройств, которые напрямую связаны с ICS. Первый пакет данных, содержащий 500 000 устройств ICS, был отправлен в 2012 году в ICS-CERT. ICS-CERT определил, что примерно 7 200 из 500 000 были поражены. И с требованием увеличить количество связей между всеми устройствами, это число ожидаемо будет расти. Были предприняты попытки защитить эти устройства, отключив их от сети или исправив недостатки, но эта проблема оказалась довольно сложной, и не имеющей простых решений.

Общие аббревиатуры

Перед тем, как перейти к протоколам и информации о том, как найти устройства ICS, полезно будет узнать, что значат некоторые сокращения:

BMS	(Building Management System)	Система управления зданием
DCS	(Distributed Control System)	Распределённая система управления
HMI	(Human Machine Interface)	Человеко-машинный интерфейс
ICS	(Industrial Control System)	Промышленная система управления
PLC	(Programmable Logic Controller)	Программируемый логический контроллер
RTU	(Remote Terminal Unit)	Устройство связи с объектом
SCADA	(Supervisory Control and Data Acquisition)	Приложение ICS (смотри гиперссылку)
VNC	(Virtual Network Computing)	Удалённый доступ к рабочему столу

**прим. переводчика: дал ссылки на все аббревиатуры, для более полного понимания описанных терминов*

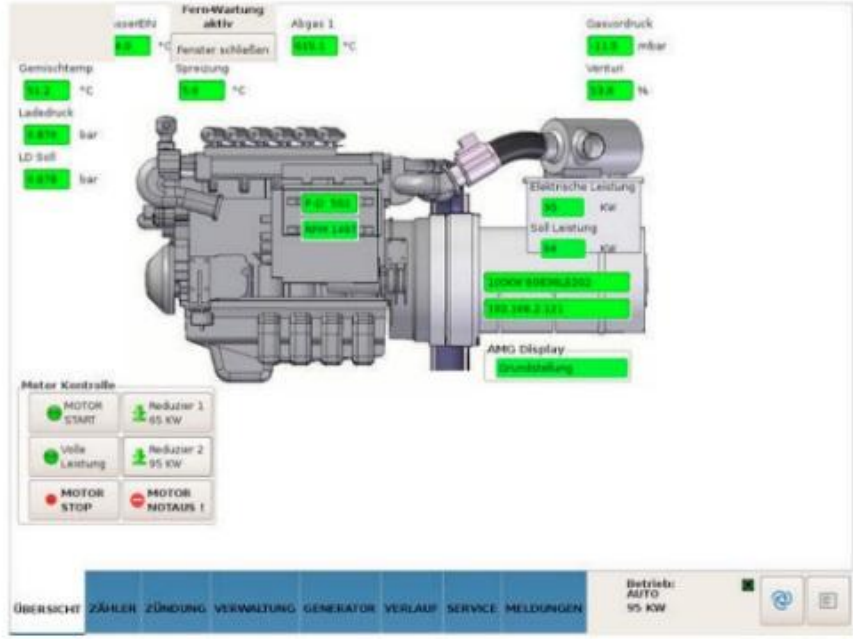
Протоколы

Есть два разных способа идентификации систем управления в интернете:

Протоколы Non-ICS, используемые в окружении ICS

Большинство результатов ICS в Shodan обнаруживаются путем поиска веб-серверов или других часто используемых протоколов, которые напрямую не связаны с ICS, но могут отображаться в их сети.

Например: веб-сервер, работающий на HMI или компьютере под управлением Windows, на котором при подключении к ICS без какой-либо авторизации запускается приложение удаленного рабочего стола. Эти протоколы предоставляют вам визуальный вид ICS, но обычно форма аутентификации всё-таки присутствует.



Изображение выше – это HMI для двигателя выставленный в сеть соединением VNC без аутентификации, найденное с помощью изображений Shodan.

Протоколы ICS

Это необработанные протоколы, которые используются системами управления. Каждый протокол ICS имеет свой собственный уникальный баннер, но у них есть одна общая черта: они не требуют никакой аутентификации. Это означает, что если у вас есть удаленный доступ к промышленному устройству, у вас также есть возможность произвольно считать читать и изменять этот протокол. Однако необработанные протоколы ICS обычно проприетарны и достаточно сложны в разработке. Из этого следует, что проверить, поддерживает ли устройство протокол ICS с использованием Shodan не составит проблемы, но реально взаимодействовать с системой управления – весьма трудная задача. Баннер ниже описывает PLC Siemens S7. Стоит обратить внимание, что он содержит много подробной информации об устройстве, включая его серийный номер и местоположение:

102
tcp
s7

```
Serial number of memory card: MMC 26559C8A
Copyright: Original Siemens Equipment
PLC name: SIMATIC 300
Unknown (129): Boot Loader          A
Module: 6ES7 315-2EG10-0AB0  v.0.2
Basic Firmware: v.2.3.2
Module name: CPU 315-2 PN/DP
Serial number of module: S C-TNR942412005
Plant identification: Kw Termometria Full
Basic Hardware: 6ES7 315-2EG10-0AB0  v.0.2
```

Защита ICS, соединённых с интернетом

Большинство баннеров ICS не содержат информации о месторасположении устройства или о владельце системы управления. Это делает чрезмерно сложным защиту устройства и является одной из основных причин того, что они находятся в сети еще годы после исследований на их уязвимость.

Если вы обнаружите систему управления, которая выглядит крайне важной, принадлежит государству или по другой причине не должна быть онлайн, пожалуйста, сообщите ICS-CERT.

Случаи использования

Доступ к ICS для США

Допустим, вам поручено подготовить краткую презентацию о рисках промышленных систем управления для США. Для начала работы, сначала необходимо получить общее представление о таких устройствах, используя основной сайт Shodan:

https://www.shodan.io/search?query=category%3Aics17

В результате мы получим список всех устройств, использующих протоколы ICS в Интернете. Однако, существует множество веб-серверов и других протоколов (SSH, FTP и так далее), которые работают на тех же портах, что и промышленные системы управления. Значит, следующим шагом нам необходимо будет их отфильтровать:

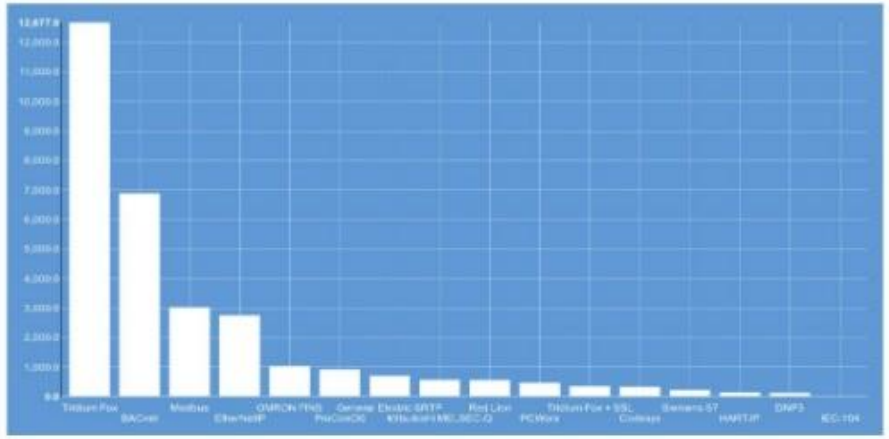
https://www.shodan.io/search?query=category%3Aics+-http+-html+-ssh+-ident18

> Внимание: Если вы обладаете доступом уровня enterprise, вы можете использовать tag:ics вместо указанного выше поискового запроса.

Теперь у нас есть отфильтрованный список устройств с запущенными небезопасными протоколами ICS. Поскольку презентация сфокусирована на устройствах в США, нужно сузить диапазон результатов до американских IP-адресов:

https://www.shodan.io/search?query=category%3Aics+-http+-html+-ssh+-ident+country%3Aus19

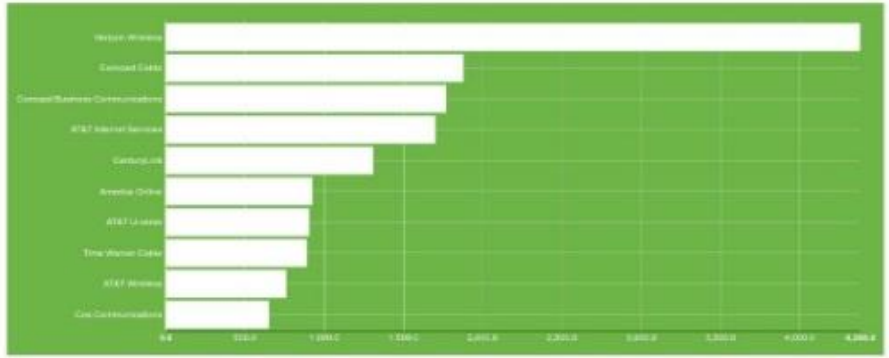
Чтобы получить полный обзор данных и несколько диаграмм для работы с ними, мы можем создать бесплатный отчет. Этот шаг даст нам представление о том, какие протоколы ICS США видны в интернете:



Протокол Tridium Fox, используемый инфраструктурой Niagara, является самым популярным протоколом ICS в США, чуть меньшую популярность имеют BACnet и Modbus. Полученные данные показывают нам, что большинство доступных устройств - это BMS, которые используются в офисах, на фабриках, на стадионах, в аудиториях и прочих учреждениях.

Вышеприведенная диаграмма была сохранена в виде изображения с помощью Nimbus Screen Capture в Firefox; также вы можете использовать плагин Awesome Screenshot Minus для Chrome.

В этом отчёте также прослеживается основная проблема с ICS в интернете: это то, что большинство из них работают в мобильных сетях. Это крайне затрудняет отслеживание и защиту этих устройств.



На этом этапе, данные говорят нам о том, что:

1. Есть как минимум 65,000 ICS в интернете, с доступом к необработанному интерфейсу без аутентификации
2. Примерно половина из них (~31,000) находится в США
3. BMS – это наиболее распространённые системы ICS
4. Большинство систем работают в мобильных сетях

Что читать дальше:

1. Определение устройств, обращённых в интернет с помощью программной информации PLC.
2. Специальная публикация NIST. Руководство по безопасности промышленных систем безопасности.
3. Количественная оценка и визуализация областей атак на промышленные системы.

Определение Honeypot'ов

Honeypots (далее – приманки, *прим. переводчика*) становятся все более используемым и полезным инструментом для понимания целей людей, атакующих интернет-системы. За период своей работы в данной сфере, я видел множество неправильно настроенных приманок при сканировании всемирной сети, и вот несколько советов, которыми я хочу поделиться, чтобы вы могли распознать приманку и избежать ошибок при их настройке.

Что же такое Honeypot?

Это специально настроенное устройство, которое выдаёт себя за другое, предназначенное для отслеживания и записи сетевой активности. В случае систем управления, ICS-приманка – это обычный компьютер, который выдаёт себя за систему управления, к примеру, завод или электростанцию. Приманки используются для сбора информации о злоумышленниках, в том числе о том, на какие сети нацелены их атаки, какие инструменты они используют и прочие полезные сведения, которые помогают максимально укрепить защиту своей сети.

В последние годы приманки также используются для подсчёта количества атак на промышленные системы управления, подключенные к интернету. Крайне важно понять, как правильно настроить приманку, прежде чем приступать к сбору данных. Многие люди делают ошибки при настройке приманок, и я расскажу, как эти ошибки делают присутствие приманки очевидным, и научу отличать реальные системы управления.

Наиболее используемой приманкой, симулирующей промышленные системы управления, является Conpot. Данное программное обеспечение написано на очень высоком уровне и чрезвычайно эффективно при правильной его настройке. В большинстве примеров и обсуждений я буду использовать именно Conpot, но приводимые принципы применимы ко всем приманкам.

Какой смысл в обнаружении приманок?

Данные, генерируемые приманками, так же важны, как и их размещение. Если мы хотим получать действительную информацию, о том, кто атакует системы управления, мы должны собирать данные только из реалистичных приманок. Опытных и умных злоумышленников не получится обмануть плохо настроенной приманкой. Важно иметь определённые знания об общих ошибках при развертывании приманки, чтобы повысить качество собираемых данных.

Настройки по умолчанию

Самая распространенная ошибка при развертывании приманки – это использование стандартных настроек. Такие настройки выдают ответ в виде одного и того же баннера, с одними и теми же серийными номерами, именами PLC и прочей информацией, которая должна разниться на разных IP-адресах.

Я понял, насколько часто встречается эта ошибка после первого сканирования интернета по запросу Siemens S7:

S7 Serial Number Uniqueness

30 процентов серийных номеров в результат присутствуют больше чем в одном баннере. Это не означает, что все дублирующие серийные номера – приманки, но это неплохая отправная точка для проведения исследований.

В случае с S7, наиболее популярный серийный номер, встречающийся в интернете – это 88111222, а это серийный номер по умолчанию для Conpot.

Showing results 1 - 10 of 110

91.229.57.200
FH JOANNEUM Gesellschaft mbH
Added on 2015-12-11 23:26:59 GMT
Austria, Allerheiligen Bei Wildon
Details

```
Location designation of a module:
Copyright: Original Siemens Equipment
Module type: IM151-8 PN/DP CPU
PLC name: Technodrome
Module: v.0.0
Plant identification: Mouser Factory
OEM ID of a module:
Module name: Siemens, SIMATIC, S7-200
Serial number of module: 88111222
```

54.164.128.60
ec2-54-164-128-60.compute-1.amazonaws.com
AMAZON
Added on 2015-12-11 15:00:37 GMT
United States, Ashburn
Details

```
Location designation of a module:
Copyright: Original Siemens Equipment
Module type: IM151-8 PN/DP CPU
PLC name: Technodrome
Module: v.0.0
Plant identification: Mouser Factory
OEM ID of a module:
Module name: Siemens, SIMATIC, S7-200
Serial number of module: 88111222
```

Поиск по серийному номеру делает простым определение расположения устройств Conpot в интернете. Также, не забудьте изменить и другие свойства баннера, не только серийный номер:

52.24.188.77
E.I. du Pont de Nemours and Co.
Added on 2015-11-21 16:03:26 GMT
United States, Wilmington
Details

```
Location designation of a module:
Copyright: Original Siemens Equipment
Module type: CPU 315-2 PN/DP
PLC name: Technodrome
Module: v.0.0
Plant identification: Mouser Factory
OEM ID of a module:
Module name: Siemens, SIMATIC, S7-200
Serial number of module: S C-C4VD66352012
```

Пользователь выше изменил серийной номер на уникальное число, но не изменил ни имя PLC (**Technodrome**), ни заводской идентификатор (**Mouser Factory**). Каждый экземпляр приманки должен иметь уникальные значения для того, чтобы избежать её раскрытия.

История имеет значение

Приманка ложна быть настроена правильно с первого дня своей жизни, иначе история баннера раскроет её. Например:

```
Location designation of a module:
Copyright: Original Siemens Equipment
Module type: IM151-8 PN/DP CPU
PLC name: PG[random.randint(0,1) f
Module: v.0.0
Plant identification: Power Generation One
OEM ID of a module:
Module name: Siemens, SIMATIC, S7-200
Serial number of module: 8675309
```

Сверху мы видим баннер, претендующий на то, чтобы быть Siemens S7 PLC. Однако, была допущена ошибка в шаблоне генерации баннера, и, вместо того, чтобы показывать правильное имя PLC, он показывает шаблонный метод **random.randint(0,1).** Shodan уже проиндексировал этот баннер, и даже если в будущем ошибка будет исправлена, пользователь может посмотреть историю для этого IP и увидеть, что ранее использовался неправильный S7 баннер.

Пример запроса истории IP-адреса в Shodan API:

```
host = api.host('xxx.xxx.xxx.xxx', history=True)
```

Эмулируйте устройства, а не службы

Будьте проще, не пытайтесь эмулировать слишком много служб единовременно. Приманка должна выглядеть как устройство, а большинство настоящих устройств не запускают MongoDB, DNP3, MySQL, Siemens S7, Kamstrup, ModBus, Automated Tank Gauge, Telnet и SSH на одном IP-адресе.

Ports

Подумайте о том, как устройство настроено в реальном мире, и лишь потом приступайте к эмуляции. Не запускайте все возможные службы только потому, что это возможно.

В коде, вы можете использовать число портов в виде метрики:

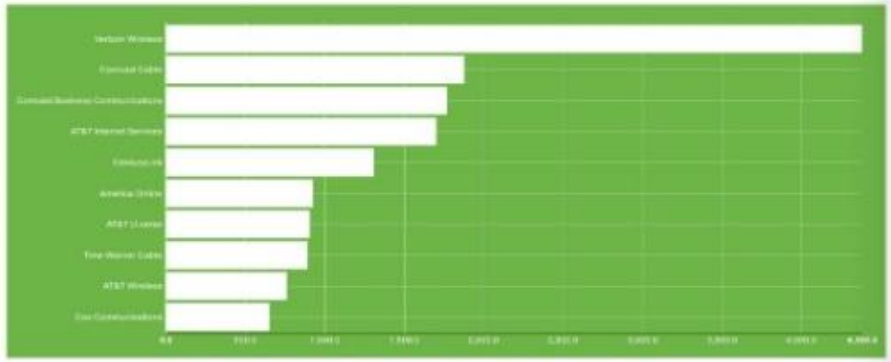
```
# Get information about the host
host = api.host('xxx.xxx.xxx.xxx')

# Check the number of open ports
if len(host['ports']) > 10:
      print('{} looks suspicious'.format(host['ip_str']))
else:
      print('{} has few ports open'.format(host['ip_str']))
```

Расположение, расположение, расположение

Важна не только правильная настройка программного обеспечения, но и расположение приманки в такой сети, в которой может быть система управления. Расположение приманки, эмулирующей Siemens S7 PLC в облачный хостинг Амазона не имеет особого смысла. Ниже я приведу распространённых провайдеров облачных хостингов, которых следует избегать при размещении приманки:

1. Amazon EC2
2. Rackspace
3. Digital Ocean
4. Vultr
5. Microsoft Azure
6. Google Cloud

Для размещения приманки, максимально похожей на реальное устройство, стоит обратить внимание на список самых распространённых ICS в Shodan и выберите публично доступные. В целом, лучшим вариантом будет размещение приманки в IP-пространстве соседствующей ICS. На следующем изображении можно увидеть организации, на которых расположены большинство ICS:



Honeyscore

Я разработал инструмент, который называется Honeyscore, который использует все вышеописанные методы и самообучается для вычисления приманок и определения, является ли IP таковой или нет.

Просто введите IP-адрес устройства и этот инструмент произведет множество проверок чтобы определить – приманка это или нет.

Тэг приманки

Алгоритм самообучения, используемый в Honeyscore, был добавлен в сканеры Shodan, так что теперь они способны определять приманки ICS во время сбора данных. Также, мы добавили определение распространённых приманок в виде веб-приложений и несколько приманок Elastic. Если баннер принадлежит одной из известных приманок, тогда в свойстве **tags** будет прописан тэг **honeypot**. Также сканеры будут предоставлять название обнаруженной приманки в свойствах продукта.

Что читать дальше:

1. Статья Honeypots в Википедии
2. Взлом приманок для развлечения и прибыли (видео)

Приложение A: спецификация баннера

Для получения последнего списка полей баннера, смотрите онлайн-документацию.

Баннер может содержать следующие свойства/поля:

Основные свойства

Название	Описание	Пример
asn	Автономный номер системы	AS4837
data	Главный баннер службы	HTTP/1.1 200...
ip	IP-адрес как переменная	493427495
ip_str	IP-адрес как строка	199.30.15.20
ipv6	IPv6-адрес как строка	2001:4860:4860::8888
port	Порт службы	80
timestamp	Дата и время сбора информации	2014-01-15T05:49:56.283713
hash	Числовой хэш свойства *data*	
hostnames	Список имён хоста для IP	["shodan.io", "www.shodan.io"]
domains	Список имён хоста для домена	["shodan.io"]
link	Тип соединения	Ethernet or modem
location	Геолокация устройства	*Смотри ниже*
opts	Дополнительные/экспериментальные данные, не содержащиеся в главном баннере	
org	Организация, которой присвоен IP	Google Inc.
isp	ISP, отвечающая за пространство IP	Verizon Wireless
os	Операционная система	Linux
uptime	Время активности IP	50
tags	Список тэгов, описывающих устройство	["ics", "vpn"]
transport	Тип транспортного протокола, который использовался для сбора баннера	tcp

Свойства Elastic

Следующие свойства собираются для Elastic (ранее - ElasticSearch)

Название	Описание
elastic.cluster	Основная информация о кластере
elastic.indices	Список индексов, доступный в кластере
elastic.nodes	Список нод/пиров кластера и информация о них

Свойства HTTP(S)

Shodan переходит по перенаправлениям откликов HTTP и записывает всю информацию в баннер. Единственные случаи, когда сканеры не переходят далее, это если HTTP-запрос перенаправляется в зону HTTPS и наоборот.

Название	Описание
http.components	Технология, которая использовалась для создания сайта
http.host	Имя хоста, посланное для захвата HTML
http.html	HTML-контент сайта
http.html_hash	Числовой хэш свойства *http.html*
http.location	Локация последнего HTML-отклика
http.redirects	Список переходов. Каждый редирект имеет три свойства: *host*, *data* и *location*
http.robots	Файл robots.txt сайта
http.server	Хэдер *Server* сайта
http.sitemap	XML карта сайта
http.title	Название сайта

Свойства географического расположения

Следующие свойства – это подсвойства для свойства **location**, находящегося в верхнем уровне записи баннера.

Название	Описание
area_code	Код области расположения устройства
city	Название города
country_code	Код страны из 2 букв
country_code3	Код страны из 3 букв
country_name	Название страны

dma_code	Торговый код области (только США)
latitude	Широта
longitude	Долгота
postal_code	Почтовый индекс
region_code	Код региона

Свойства SMB

Название	Описание
smb.anonymous	Позволяет ли служба анонимные подключения (true/false)
smb.capabilities	Список функций, поддерживаемых службой
smb.shares	Список доступных сетевых ресурсов
smb.smb_version	Версия протокола, использовавшегося для сбора данных
smb.software	Название программы, запускающей службу
smb.raw	Список hex-кодированных пакетов, отправленных сервером; полезен, если вы хотите сами сделать парсинг SMB

Свойства SSH

Название	Описание
ssh.cipher	Шифр, который использовался во время обмена данными
ssh.fingerprint	Отпечаток устройства
ssh.kex	Список алгоритмов обмена ключами, поддерживаемых сервером
ssh.key	SSH-ключ сервера
ssh.mac	Алгоритм кода аутентификации сообщений

Свойства SSL

Если служба завёрнута в SSL, то Shodan производит дополнительное тестирование и показывает результаты в следующих свойствах:

Название	Описание
ssl.acceptable_cas	Список сертификатов, поддерживаемых сервером
ssl.cert	Проанализированный SSL сертификат

ssl.cipher	Предпочитаемый шифр для SSL-соединения
ssl.chain	Список SSL-сертификатов от корневого до пользовательского
ssl.dhparams	Параметры Даффи-Хеллмана
ssl.tlsext	Список расширений TLS, поддерживаемых сервером
ssl.versions	Поддерживаемые версии SSL; если значение начинается с «-», тогда служба НЕ поддерживает эту версию

Свойства ISAKMP

Следующие свойства собираются для соединений VPN, использующих протокол ISAKMP (как IKE):

Название	Описание
isakmp.initiator_spi	Hex-зашифрованный индекс параметра безопасности для инициатора
isakmp.responder_spi	Hex-зашифрованный индекс параметра безопасности для отвечающего
isakmp.next_payload	Полезная нагрузка, отправленная после инициации
isakmp.version	Версия протокола
isakmp.exchange_type	Тип обмена
isakmp.flags.encryption	Установка бита шифрования: true или false
isakmp.flags.commit	Установка бита поручения: true или false
isakmp.flags.authentication	Установка бита аутентификации: true или false
isakmp.msg_id	Hex-зашифрованный идентификатор сообщения
isakmp.length	Размер ISAKMP-пакета

Особые свойства

_shodan

Это свойство содержит информацию о способе сбора информации Shodan'ом. Оно отличается от прочих свойств, так как не предоставляет никакой информации об устройстве. Вместо этого, оно даёт информацию о том, какой граббер баннеров использовался для взаимодействия с IP. Это может быть важно для понимания для портов, на которых работает много служб. Например, порт 80 широко известен для веб-серверов, но также он используется различными вредоносными программами для обхода фаервола. Свойство _shodan сообщит вам, использовался ли http-модуль для сбора информации, или некий вредоносный модуль.

Название	Описание
_shodan.crawler	Уникальный идентификатор сканера Shodan

_shodan.id	Уникальный идентификатор баннера
_shodan.module	Название модуля Shodan, использовавшегося сканером Shodan для сбора баннера
_shodan.options	Конфигурационные опции, использовавшиеся во время сбора данных
_shodan.hostname	Имя хоста для использования при отсылке веб-запросов
_shodan.options.referrer	Уникальный идентификатор баннера, запустившего сканирование для этого порта/службы

Пример


```
{
      "timestamp": "2014-01-16T08:37:40.881917",
      "hostnames": [
            "99-46-189-78.lightspeed.tukrga.sbcglobal.net"
      ],
      "org": "AT&T U-verse",
      "guid": "1664007502:75a821e2-7e89-11e3-8686-898808088080",
      "data": "NTP\nxxx.xxx.xxx.xxx:7546\n68.94.157.2:123\n68.94.156.17:123",
      "port": 123,
      "isp": "AT&T U-verse",
      "asn": "AS7018",
      "location": {
            "country_code3": "USA",
            "city": "Atlanta",
            "postal_code": "30328",
            "longitude": -84.3972,
            "country_code": "US",
            "latitude": 33.93350988090091,
            "country_name": "United States",
            "area_code": 404,
            "dma_code": 524,
            "region_code": null
      },
      "ip": 1664007502,
      "domains": [
            "sbcglobal.net"
      ],
      "ip_str": "99.46.189.78",
      "os": null,
      "opts": {
            "raw": "\\x97\\x00\\x03*\\x90\\x03\\x06H\\x00\\x00\\x00\\x00\\x00\\x00\
\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x01G\\x06\\xa7\\x8ec.\\xbdN\\x00\\
\\x00\\x00\\x01\\x1dz\\x07\\x02\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\
\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\
\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\
\\x00q\\x00\\x00\\x00i\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x06XD^\\x9d\\x02c.\\xbdN\\\
x00\\x00\\x00\\x01\\x06{\\x04\\x04\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\\
\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\
\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\
\\x00\\x00q\\x00\\x00\\x00o\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x06YD^\\x9c\\x11c.\\xb\
dN\\x00\\x00\\x00\\x00\\x01\\x06{\\x04\\x04\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\\
x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\\
x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00\\x00",
            "ntp": {
                  "more": false
            }
      }
```


}

Приложение B: список поисковых фильтров

Основные фильтры

Название	Описание	Тип
after	Показывает результаты только после определённой даты (ДД/ММ/ГГГГ)	строка
asn	Автономный номер системы	строка
before	Показывает результаты только до определённой даты (ДД/ММ/ГГГГ)	строка
category	Главный баннер службы	строка
city	Название города	строка
country	Двухбуквенный код страны	строка
geo	Принимает от 2 до 4 параметров. Если 2 параметра: широта, долгота. Если три: широта, долгота, диапазон. Если 4: левая верхняя широта, левая верхняя долгота, правая нижняя широта, правая нижняя долгота.	строка
hash	Хэш свойства data	переменная
hash IPv6	True/False	булево значение
has_screenshot	True/False	булево значение
hostname	Полное имя хоста для устройства	строка
ip	Псевдоним для сетевого фильтра	строка
isp	ISP, отвечающая за блок IP-адресов	строка
net	Сетевой диапазон в CIDR-нотации	строка
org	Организация, которой присвоен блок IP-адресов	строка
os	Операционная система	строка
port	Номер порта для устройства	переменная
postal	Почтовый индекс (только США)	строка
product	Название программы/продукта, предоставляющего баннер	строка
region	Название региона/штата	строка
state	Псевдоним региона	строка
version	Версия продукта	строка
vuln	Идентификатор CVE для уязвимости	строка

HTTP-фильтры

Название	Описание	Тип
http.components	Технология, которая использовалась для создания сайта	строка
http.component_category	Категория веб-компонентов, используемых на сайте	строка
http.html	HTML веб-баннеров	строка
http.html_hash	Хэш HTML сайта	переменная
http.status	Код статуса ответа	переменная
http.title	Название сайта веб-баннера	строка

NTP-фильтры

Название	Описание	Тип
ntp.ip	IP-адреса полученные откликом от команды monlist	строка
ntp.ip_count	Количество IP-адресов в откликом от первой команды monlist	переменная
ntp.more	True/False; есть ли еще IP для сбора командой monlist	булево значение
ntp.port	Порт, используемый IP-адресами из отклика monlist	переменная

**приведу статью с хобра для понимания команды monlist: тык*

SSL-фильтры

Название	Описание	Тип
has_ssl	True/False	булево значение
ssl	Поиск всех данных SSL	строка
ssl.alpn	Протоколы прикладного уровня, такие как HTTP/2 ("h2")	строка
ssl.chain_count	Количество сертификатов в цепочке	переменная
ssl.version	Возможные значения: SSLv2, SSLv3, TLSv1, TLSv1.1, TLSv1.2	строка
ssl.cert.alg	Алгоритм сертификата	строка
ssl.cert.expired	True/False	булево значение
ssl.cert.extension	Имена расширений в сертификате	строка
ssl.cert.serial	Серийный номер как переменная или шестнадцатеричная строка	строка/переменная

ssl.cert.pubkey.bits	Количество бит в публичном ключе	переменная
ssl.cert.pubkey.type	Тип публичного ключа	строка
ssl.cipher.version	SSL-версия предпочитаемого шифра	строка
ssl.cipher.bits	Число бит в предпочитаемом шифре	переменная
ssl.cipher.name	Название предпочитаемого шифра	строка

Telnet-фильтры

Название	Описание	Тип
telnet.option	Поиск всех опций	строка
telnet.do	Сервер запрашивает клиент поддерживать эти опции	строка
telnet.dont	Сервер запрашивает клиент не поддерживать эти опции	строка
telnet.will	Сервер поддерживает эти опции	строка
telnet.wont	Сервер не поддерживает эти опции	строка

Приложение C: поисковые фасеты

Основные фасеты

Название	Описание
asn	Автономный номер системы
city	Полное название города
country	Полное название страны
domain	Домен(ы) для устройства
has_screenshot	Имеет доступные скриншоты
isp	ISP, управляющая блоком IP-адресов
link	Тип сетевого соединения
org	Организация, владеющая блоком IP-адресов
os	Операционная система
port	Номер порта для службы
postal	Почтовый индекс
product	Название программы/продукта баннера
region	Название региона/штата

state	Псевдоним региона
uptime	Время активности хоста в секундах
version	Версия продукта
vuln	Идентификатор CVE для уязвимости

HTTP-фасеты

Название	Описание	Тип
http.component	Название технологии, используемой на сайте	строка
http.component_category	Категория веб-компонентов, используемых на сайте	строка
http.html_hash	Хэш HTML сайта	переменная
http.status	Код статуса ответа	переменная

NTP-фасеты

Название	Описание
ntp.ip	IP-адреса полученные откликом от команды monlist
ntp.ip_count	Количество IP-адресов в откликом от первой команды monlist
ntp.more	True/False; есть ли еще IP для сбора командой monlist
ntp.port	Порт, используемый IP-адресами из отклика monlist

SSH-фасеты

Название	Описание
ssh.cipher	Название шифра
ssh.fingerprint	Отпечаток устройства
ssh.mac	Имя MAC используемого алгоритма (к примеру, hmac-sha1)
ssh.type	Тип ключа аутентификации (к примеру, ssh-rsa)

SSL-фасеты

Название	Описание
ssl.version	Версия SSL поддерживается

ssl.alpn	Протоколы прикладного уровня
ssl.chain_count	Количество сертификатов в цепочке
ssl.cert.alg	Алгоритм сертификата
ssl.cert.expired	True/False; истёк срок действия сертификата или нет
ssl.cert.serial	Серийный номер как переменная
ssl.cert.extension	Имена расширений в сертификате
ssl.cert.pubkey.bits	Количество бит в публичном ключе
ssl.cert.pubkey	Имя типа публичного ключа
ssl.cipher.bits	Число бит в предпочитаемом шифре
ssl.cipher.name	Название предпочитаемого шифра
ssl.cipher.version	SSL-версия предпочитаемого шифра

Telnet-фасеты

Название	Описание
telnet.option	Показать всех опции
telnet.do	Сервер запрашивает клиент поддерживать эти опции
telnet.dont	Сервер запрашивает клиент не поддерживать эти опции
telnet.will	Сервер поддерживает эти опции
telnet.wont	Сервер не поддерживает эти опции

Приложение D: список портов

Порт	Название службы (служб)
7	Echo
11	Systat
13	Daytime
15	Netstat
17	Quote of the day
19	Character generator
21	FTP
22	SSH
23	Telnet
25	SMTP
26	SSH

Порт	Название службы (служб)
4369	EPMD
4443	Symantec Data Center Security
4444	malware
4500	IKE NAT-T (VPN)
4567	Modem web interface
4664	Qasar
4730	Gearman
4782	Qasar
4800	Moxa Nport
4840	OPC UA
4911	Niagara Fox with SSL

37	rdate
49	TACACS+
53	DNS
67	DHCP
69	TFTP, BitTorrent
70	Gopher
79	Finger
80	HTTP, malware
81	HTTP, malware
82	HTTP, malware
83	HTTP
84	HTTP
88	Kerberos
102	Siemens S7
104	DICOM
110	POP3
111	Portmapper
113	identd
119	NNTP
123	NTP
129	Password generator protocol
137	NetBIOS
143	IMAP
161	SNMP
175	IBM Network Job Entry
179	BGP
195	TA14-353a
311	OS X Server Manager
389	LDAP
389	CLDAP
443	HTTPS
443	QUIC
444	TA14-353a, Dell SonicWALL
445	SMB
465	SMTPS
500	IKE (VPN)
502	Modbus
503	Modbus
515	Line Printer Daemon
520	RIP
523	IBM DB2
554	RTSP
587	SMTP mail submission
623	IPMI
626	OS X serialnumbered
636	LDAPS
666	Telnet

4949	Munin
5006	MELSEC-Q
5007	MELSEC-Q
5008	NetMobility
5009	Apple Airport Administration
5060	SIP
5094	HART-IP
5222	XMPP
5269	XMPP Server-to-Server
5353	mDNS
5357	Microsoft-HTTPAPI/2.0
5432	PostgreSQL
5577	Flux LED
5601	Kibana
5632	PCAnywhere
5672	RabbitMQ
5900	VNC
5901	VNC
5938	TeamViewer
5984	CouchDB
6000	X11
6001	X11
6379	Redis
6666	Voldemort database, malware
6667	IRC
6881	BitTorrent DHT
6969	TFTP, BitTorrent
7218	Sierra wireless (Telnet)
7474	Neo4j database
7548	CWMP (HTTPS)
7777	Oracle
7779	Dell Service Tag API
8008	Chromecast
8009	Vizio HTTPS
8010	Intelbras DVR
8060	Roku web interface
8069	OpenERP
8087	Riak
8090	Insteon HUB
8099	Yahoo SmartTV
8112	Deluge (HTTP)
8126	StatsD
8139	Puppet agent
8140	Puppet master
8181	GlassFish Server (HTTPS)
8333	Bitcoin
8334	Bitcoin node dashboard (HTTP)

771	Realport
789	Redlion Crimson3
873	rsync
902	VMWare authentication
992	Telnet (secure)
993	IMAP with SSL
995	POP3 with SSL
1010	malware
1023	Telnet
1025	Kamstrup
1099	Java RMI
1177	malware
1200	Codesys
1234	udpxy
1400	Sonos
1434	MS-SQL monitor
1515	malware
1521	Oracle TNS
1604	Citrix, malware
1723	PPTP
1741	CiscoWorks
1833	MQTT
1900	UPnP
1911	Niagara Fox
1962	PCworx
1991	malware
2000	iKettle, MikroTik bandwidth test
2081	Smarter Coffee
2082	cPanel
2083	cPanel
2086	WHM
2087	WHM
2123	GTPv1
2152	GTPv1
2181	Apache Zookeeper
2222	SSH, PLC5, EtherNet/IP
2323	Telnet
2332	Sierra wireless (Telnet)
2375	Docker
2376	Docker
2379	etcd
2404	IEC-104
2455	CoDeSys
2480	OrientDB
2628	Dictionary
3000	ntop
3260	iSCSI

8443	HTTPS
8554	RTSP
8800	HTTP
8880	Websphere SOAP
8888	HTTP, Andromouse
8889	SmartThings Remote Access
9000	Vizio HTTPS
9001	Tor OR
9002	Tor OR
9009	Julia
9042	Cassandra CQL
9051	Tor Control
9100	Printer Job Language
9151	Tor Control
9160	Apache Cassandra
9191	Sierra wireless (HTTP)
9418	Git
9443	Sierra wireless (HTTPS)
9595	LANDesk Management Agent
9600	OMRON
9633	DarkTrack RAT
9869	OpenNebula
10001	Automated Tank Gauge
10001	Ubiquiti
10243	Microsoft-HTTPAPI/2.0
10554	RTSP
11211	Memcache
12345	malware
17000	Bose SoundTouch
17185	VxWorks WDBRPC
12345	Sierra wireless (Telnet)
11300	Beanstalk
13579	Media player classic web interface
14147	Filezilla FTP
16010	Apache Hbase
16992	Intel AMT
16993	Intel AMT
18245	General Electric SRTP
20000	DNP3
20547	ProconOS
21025	Starbound
21379	Matrikon OPC
23023	Telnet
23424	Serviio
25105	Insteon Hub
25565	Minecraft
27015	Steam A2S server query, Steam RCon

3306	MySQL
3310	ClamAV
3386	GTPv1
3388	RDP
3389	RDP
3460	malware
3541	PBX GUI
3542	PBX GUI
3689	DACP
3702	Onvif
3780	Metasploit
3787	Ventrilo
4000	malware
4022	udpxy
4040	Deprecated Chef web interface
4063	ZeroC Glacier2
4064	ZeroC Glacier2 with SSL
4070	HID VertX/ Edge door controller
4157	DarkTrack RAT

27016	Steam A2S server query
27017	MongoDB
28015	Steam A2S server query
28017	MongoDB (HTTP)
30313	Gardasoft Lighting
30718	Lantronix Setup
32400	Plex
37777	Dahuva DVR
44818	EtherNet/IP
47808	Bacnet
49152	Supermicro (HTTP)
49153	WeMo Link
50070	HDFS Namenode
51106	Deluge (HTTP)
53413	Netis backdoor
54138	Toshiba PoS
55443	McAfee
55553	Metasploit
55554	Metasploit
62078	Apple iDevice
64738	Mumble

Приложение E: Пример SSL-баннера


```
{
    "hostnames": [],
    "title": "",
    "ip": 2928565374,
    "isp": "iWeb Technologies",
    "transport": "tcp",
    "data": "HTTP/1.1 200 OK\r\nExpires: Sat, 26 Mar 2016 11:56:36 GMT\r\nExpire\
s: Fri, 28 May 1999 00:00:00 GMT\r\nCache-Control: max-age=2592000\r\nCache-Cont\
rol: no-store, no-cache, must-revalidate\r\nCache-Control: post-check=0, pre-che\
ck=0\r\nLast-Modified: Thu, 25 Feb 2016 11:56:36 GMT\r\nPragma: no-cache\r\nP3P:\
 CP=\"NON COR CURa ADMa OUR NOR UNI CON NAV STA\"\r\nContent-type: text/html\r\n\
Transfer-Encoding: chunked\r\nDate: Thu, 25 Feb 2016 11:56:36 GMT\r\nServer: sw-\
cp-server\r\n\r\n",
    "asn": "AS32613",
    "port": 8443,
    "ssl": {
        "chain": ["-----BEGIN CERTIFICATE-----\nMIIDszCCApsCBFBTb4swDQYJKoZIhvcN\
AQEFBQAwgZ0xCzAJBgNVBAYTAlVTMREw\nDwYDVQQIEwhWaXJnaW5pYTEQMA4GA1UEBxMHSGVybmRvbj\
ESMBAGA1UEChMJUGFy\nYWxsZWxzMRgwFgYDVQQLEw9QYXJhbGxlbHMgUGFuZWwxGDAWBgNVBAMTD1Bh\
cmFs\nbGVscyBQYW5lbDEhNB8GCSqGSIb3DQEJARYSaW5mb0BwYXJhbGxlbHNuY29tMB4X\nDTEyMDkx\
NDE3NTUyM1oXDTEzMDkxNDE3NTUyM1owgZ0xCzAJBgNVBAYTAlVTMREw\nDwYDVQQIEwhWaXJnaW5pYT\
EQMA4GA1UEBxMHSGVybmRvbjESMBAGA1UEChNJUGFy\nYWxsZWxzMRgwFgYDVQQLEw9QYXJhbGxlbHMg\
UGFuZWwxGDAWBgNVBAMTD1BhcmFs\nbGVscyBQYW5lbDEhMB8GCSqGSIb3DQEJARYSaW5mb0BwYXJhbG\
xlbHNuY29tMIIB\nIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAxc9Vy/qajKtFFnHxGOFPHTxm\
\nSOnsffWBT8fyXnK3h8u041VxvZDh3XkpA+ptg2fWOuIT0TTYuqw+tqiDmg8YTsHy\njcpMFBtXV2cV\
dhKXaS3YYlM7dP3gMmkGmH+ZvCgCYc7L9MIJxYJy6Zeuh67YxEMV\ngiU8mZpvc70Cg5WeW1uBCXtUAi\
jDLsVWnhsV3YuxlweEvkRpAk3EHehKbvgMnEZ5\nQ30QySe0GAqC7bWzKrwsJAOUk/+Js18+3QKb/LmD\
a9cRjtFCTo6hYfPbfHj8RxQh\n4Xmnn/CtZ48wRQTqKXSOG+Zk3OuU7/jX1Gt/jxN6n77673e6uCsggT\
wut/EtNwID\nAQABMA0GCSqGSIb3DQEBBQUAA4IBAQBb/yTy76Ykwr7DBOPAXc766n73OsZizjAt\n1k\
mx7LxgN3X/wFxD53ir+sdOqbPgJl3edrE/ZG9dNl6LhUBbUK+9s6z9QicEfSxo\n4uQpFSywbGGmXInE\
ZmyT4SsOLi/hNgy68f49LO1h6rn/p7QgIKd31g7189ZfFkFb\nRdD49s1l/Cc5Nm4XapUVvmnS91MlPk\
```



```
/OOIg1Lu1rYkuc8sIoZdPbep52H3GaTTjG\nkmO7nUIii0goB7TQ63mU67+NWHAmQQ8CtCDCN49kJyen\
lWFjD6Je2U4q0IFQrxHw\nNy+tquo/n/sa+NV8QOj1gMVcFsLhYm7Z5ZONg0QFX5AL+Eyj/AwZ\n----\
-END CERTIFICATE-----\n"],
        "cipher": {
            "version": "TLSv1/SSLv3",
            "bits": 256,
            "name": "DHE-RSA-AES256-GCM-SHA384"
        },
        "alpn": [],
        "dhparams": {
            "prime": "b10b8f96a080e01dde92de5eae5d54ec52c99fbcfb06a3c69a6a9dca52\
d23b616073e28675a23d189838ef1e2ee652c013ecb4aea906112324975c3cd49b83bfaccbdd7d90\
c4bd7098488e9c219a73724effd6fae5644738faa31a4ff55bccc0a151af5f0dc8b4bd45bf37df36\
5c1a65e68cfda76d4da708df1fb2bc2e4a4371",
            "public_key": "2e30a6e455730b2f24bdaf5986b9f0876068d4aa7a4e15c9a1b9c\
a05a420e8fd3b496f7781a9423d3475f8bedee53f0391aaa95a738c8f0e250a5869a86d41bdb0194\
66dba5c641e4b2b4b82db4cc2d4ea8d9804ec00514f30a4b6ce170b81c3e1ce4b3d17647c8e5b8f6\
65bb7f588100bcc9a447d34d728c3709fd8a5b7753b",
            "bits": 1024,
            "generator": "a4d1cbd5c3fd34126765a442efb99905f8104dd258ac507fd6406c\
ff14266d31266fea1e5c41564b777e690f5504f213160217b4b01b886a5e91547f9e2749f4d7fbd7\
d3b9a92ee1909d0d2263f80a76a6a24c087a091f531dbf0a0169b6a28ad662a4d18e73afa32d779d\
5918d08bc8858f4dcef97c2a24855e6eeb22b3b2e5",
            "fingerprint": "RFC5114/1024-bit MODP Group with 160-bit Prime Order\
Subgroup"
        },
        "versions": ["TLSv1", "-SSLv2", "SSLv3", "TLSv1.1", "TLSv1.2"]
    },
    "html": "\n\t\t<html><head>\n\t\t<meta charset=\"utf-8\">\n\t\t<meta http-eq\
uiv=\"X-UA-Compatible\" content=\"IE=edge,chrome=1\">\n\t\t<title></title>\n\t\t\
<script language=\"javascript\" type=\"text/javascript\" src=\"/javascript/commo\
n.js?plesk_version=psa-11.0.9-110120608.16\"/></script>\n\t\t<script language=\"\
javascript\" type=\"text/javascript\" src=\"/javascript/prototype.js?plesk_versi\
on=psa-11.0.9-110120608.16\"></script>\n\t\t<script>\n\t\t\tvar opt_no_frames = \
false;\n\t\t\tvar opt_integrated_mode = false;\n\t\t</script>\n\t\t\n\t\t</head>\
<body onLoad=\";top.location='/login.php3?window_id=&amp;requested_url=https%3A%\
2F%2F174.142.92.126%3A8443%2F';\"></body><noscript>You will be redirected to the\
new address in 15 seconds... If you are not automatically taken to the new loca\
tion, please enable javascript or click the hyperlink <a href=\"/login.php3?wind\
ow_id=&amp;requested_url=https%3A%2F%2F174.142.92.126%3A8443%2F\" target=\"top\"\
>/login.php3?window_id=&amp;requested_url=https%3A%2F%2F174.142.92.126%3A8443%2F\
</a>.</noscript></html><!--_____\
_____\
_____\
_____IE error page size limitation_____\
_____\
_____\
_____\
_____-->",
    "location": {
        "city": null,
        "region_code": "QC",
        "area_code": null,
        "longitude": -73.5833,
        "country_code3": "CAN",
        "latitude": 45.5,
        "postal_code": "H3G",
        "dma_code": null,
        "country_code": "CA",
        "country_name": "Canada"
    },
    "timestamp": "2016-02-25T11:56:52.548187",
    "domains": [],
    "org": "iWeb Technologies",
    "os": null,
    "_shodan": {
```

```
        "options": {},
        "module": "https",
        "crawler": "122dd688b363c3b45b8e7582622da1e725444808"
    },
    "opts": {
        "heartbleed": "2016/02/25 03:56:45 ([]uint8) {\n 00000000 02 00 74 63 6 \
5 6e 73 75  73 2e 73 68 6f 64 61 6e |..tcensus.shodan|\n 00000010 2e 69 6f 53  \
45 43 55 52  49 54 59 20 53 55 52 56 |.ioSECURITY SURV|\n 00000020 45 59 fe 7a \
 a2 0d fa ed  93 42 ed 18 b0 15 7d 6e |EY.z.....B....}n|\n 00000030 29 98 f6 f  \
8 ce 80 b1 94  b5 4b 47 ac dd 18 aa b9 |).......KG.....|\n 00000040 db 1c 01  \
45 95 10 e0 a2  43 fe 8e ac 88 2f e8 75 |...E....C..../.u|\n 00000050 8b 19 5f \
 8c e0 8a 80 61  56 3c 68 0f e1 1f 73 9e |....aV<h...s.|\n 00000060 61 4f d \
a db 90 ce 84 e3  79 5f 9d 6c a0 90 ff fa |aO......y_.l....|\n 00000070 d8 16  \
e8 76 87 b2 e5 5e  8e 3e a4 45 61 2f 6a 2d |...v...^.>.Ea/j-|\n 00000080 5d 11 \
 74 94 03 3c 5d                           |].t..<]|\n}\n\n2016/02/25 03:56:45\
 174.142.92.126:8443 - VULNERABLE\n",
        "vulns": ["CVE-2014-0160"]
    },
    "ip_str": "174.142.92.126"
}
```

Ответы на упражнения

Веб-сайт

Упражнение 1

```
title:4sics
```

Упражнение 2

```
rfb authentication disabled
```

Упражнение 3

```
vuln:CVE-2014-0160 country:se ssl.version:sslv3
vuln:CVE-2014-0160 org:"your organization"
```

Упражнение 4

```
category:ics city:"your city name"
```

Упражнение 5

```
category:malware country:us
```


Интерфейс командной строки

Упражнение 1

```
shodan download --limit -1 heartbleed-results country:se,no vuln:CVE-2014-0160
shodan parse --filters location.country_code:SE -O heartbleed-sweden heartbleed-\
results.json.gz
```

> Внимание: Аргумент **–filters** производит чувствительный к регистру поиск свойств, выраженных строками, следовательно, код страны Швеции должен быть написан большими буквами.

Упражнение 2

```
mkdir data
shodan stream --limit 1000 --datadir data/
shodan convert data/* kml

# Загрузите KML файл на https://www.google.com/maps/d/
```

Упражнение 3

```
#!/bin/bash

shodan download --limit -1 malware.json.gz category:malware

for ip in `shodan parse --fields ip_str malware.json.gz`
do
        iptables -A OUTPUT -d $ip -j DROP
done
```

Shodan API

Замените значение YOUR_API_KEY вашим ключом API из аккаунта на сайте Shodan.

Упражнение 1

```
#!/usr/bin/env python

# Инициализация Shodan
import shodan
api = shodan.Shodan("YOUR_API_KEY")

# Создайте новое оповещение
alert = api.create_alert('My first alert', '198.20.69.0/24')

try:
        # Подпишитесь на данные созданного оповещения
        for banner in api.stream.alert(alert['id']):
                print banner
except:
        # Произведите очистку при ошибках
        api.delete_alert(alert['id'])
```

Подсказка: Ниже – решение с помощью команд **alert** интерфейса командной строки Shodan:

```
# Создайте оповещение
shodan alert create "My first alert" 198.20.69.0/24

# Подпишитесь на ленту реального времени и сохраните данные в каталог "/tmp"
shodan stream --alerts=all --datadir=/tmp

# После завершения, очистите все оповещения
shodan alert clear
```

Упражнение 2

```
mkdir images
```

Запустите команду выше чтобы создать каталог для сохранения изображений. Затем сохраните следующий код в файле, таком как image-stream.py:

```
#!/usr/bin/env python

import shodan

output_folder = 'images/'
api = shodan.Shodan("YOUR_API_KEY")

for banner in api.stream.banners():
      if 'opts' in banner and 'screenshot' in banner['opts']:
            # На данный момент все изображения в формате JPG
            # Сделать: Используйте MIME-тип для определения расширения файла
            # Сделать: Поддержка результатов IPv6

            # Создайте имя файла, используя его IP-адрес
            filename = '{}/{}.jpg'.format(output_folder, banner['ip_str'])

            # Создайте сам файл
            output = open(filename, 'w')

            # Изображения зашифрованы с использованием base64
            output.write(banner['opts']['screenshot']['data'].decode('base64'))
```