Статья Несколько слов об энтропии

xss.is/threads/54022

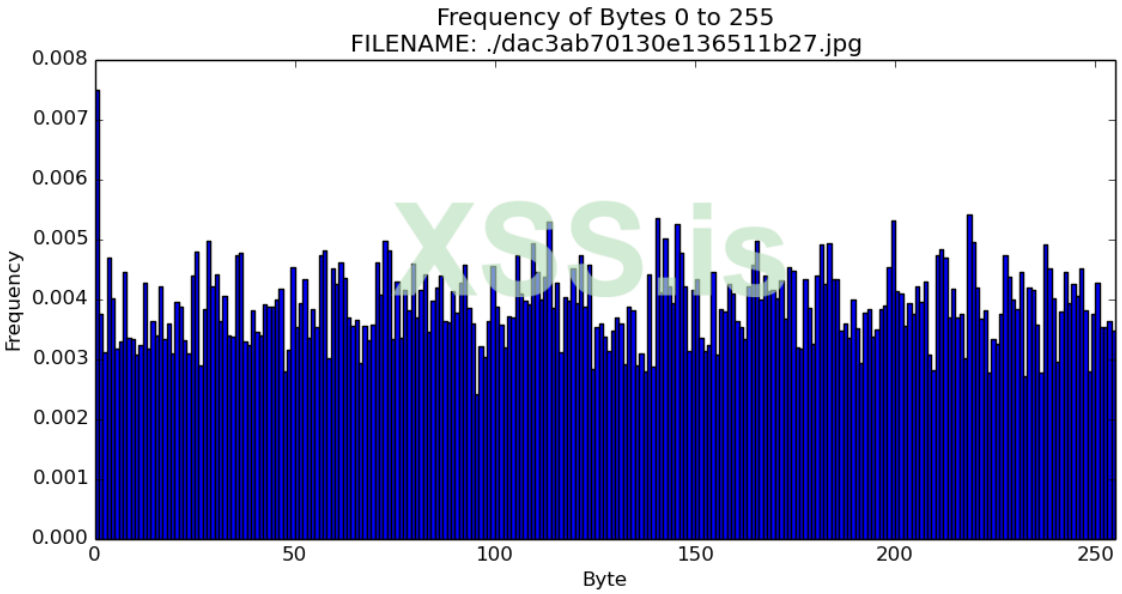
Всем известно что энтропия - это мера хаоса системы, а "информация" (данные) - это мера упорядоченности системы. Используя это свойство можно выполнять качественный анализ данных вне зависимости от их характера, вычисляя функцию частоты распределения: значение этой функции будет принимать большие значения для данных содержащих больше "хаоса" и, соответственно, меньшие значения для данных, содержащих меньше "хаоса".

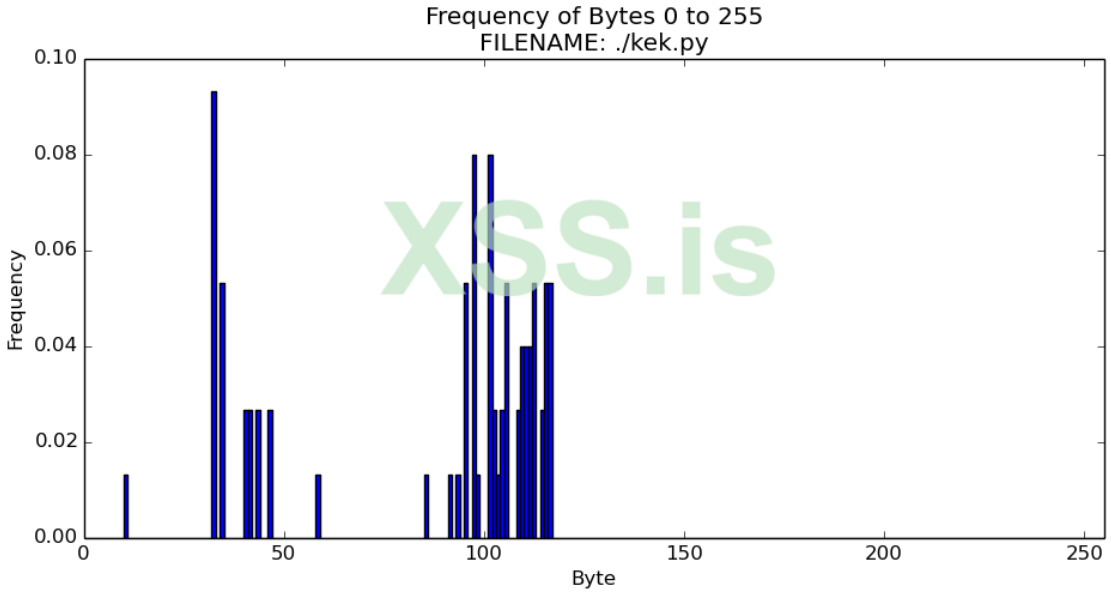
Немного теории

Программа, демонстрирующая описанное выше свойство частоты распределения байтов данных в файле приведена далее (оригинал лежащего в основе кода доступен здесь).
Python:

```
import sys
import math
import os
import numpy as np
import matplotlib.pyplot as plt

if len(sys.argv) != 2:
    print("Usage: " + os.path.basename(__file__) + " filename")
    sys.exit()

f = open(sys.argv[1], "rb")
byteArr = map(ord, f.read())
f.close()
fileSize = len(byteArr)
#print("File size in bytes: " + fileSize)

freqList = []
for b in range(256):
    ctr = 0
    for byte in byteArr:
        if byte == b:
            ctr += 1
    freqList.append(float(ctr) / fileSize)
# print("Frequencies of each byte-character: " + freqList)

# entropy
ent = 0.0
for freq in freqList:
    if freq > 0:
        ent = ent + freq * math.log(freq, 2)
ent = -ent



N = len(freqList)

ind = np.arange(N)  # the x locations for the groups
width = 1.00        # the width of the bars

#fig = plt.figure()
fig = plt.figure(figsize=(11,5),dpi=100)
ax = fig.add_subplot(111)
rects1 = ax.bar(ind, freqList, width)
ax.set_autoscalex_on(False)
ax.set_xlim([0,255])

ax.set_ylabel('Frequency')
ax.set_xlabel('Byte')
ax.set_title('Frequency of Bytes 0 to 255\nFILENAME: ' + sys.argv[1])

plt.show()
```

В результате выполнения программы, для заданного файла, вычисляется значение энтропии, а также строится частотная характеристика.
В качестве наглядного примера можно рассмотреть частотные характеристики файла со сжатием и без сжатия (первый и второй графики соответственно).





Из графиков видно, что у сжатого файла энтропия выше, чем у файла без сжатия. Этот принцип широко *используется при анализе бинарных файлов, позволяя сделать вывод о наличии/отсутствии сжатия в файле* (и его типе).
Помимо анализа бинарных файлов свойство энтропии используется и при анализе обычных файлов: у строк, содержащих секреты (пароли/криптографические ключи/токены и т.п.) энтропия выше чем у обычных строк. Применение энтропии при анализе данных существенно ускоряет этот процесс.

На этом краткий экскурс в теории можно закончить и перейти к практике.

Софт

Софта подобной тематики существует масса (любят велосипеды делать).
Наиболее известным считается truffleHog, в котором реализованы типовые функции для анализа (поиск по заданным шаблонам, поиск по энтропии и т.д.). Имеет большой недостаток - работает только с git-репозиториями, без возможности анализа локальных файлов. Существует доработанная до ума модификация - truffleHog3. Среди доработок: добавили возможность работы с локальными файлами, а также функции фильтрации результатов. Как и оригинал, truffleHog3 также давно не обновлялся.

В настоящее время активно развивается идейный продолжатель truffleHog - rustyHog, превосходящий предшественника как по скорости работы, так и по количеству реализованных фич.
Распространяется rustyHog в виде исходных текстов (компилируется в несколько команд, как под windows, так и под linux/macos), также имеется готовый образ в Docker.

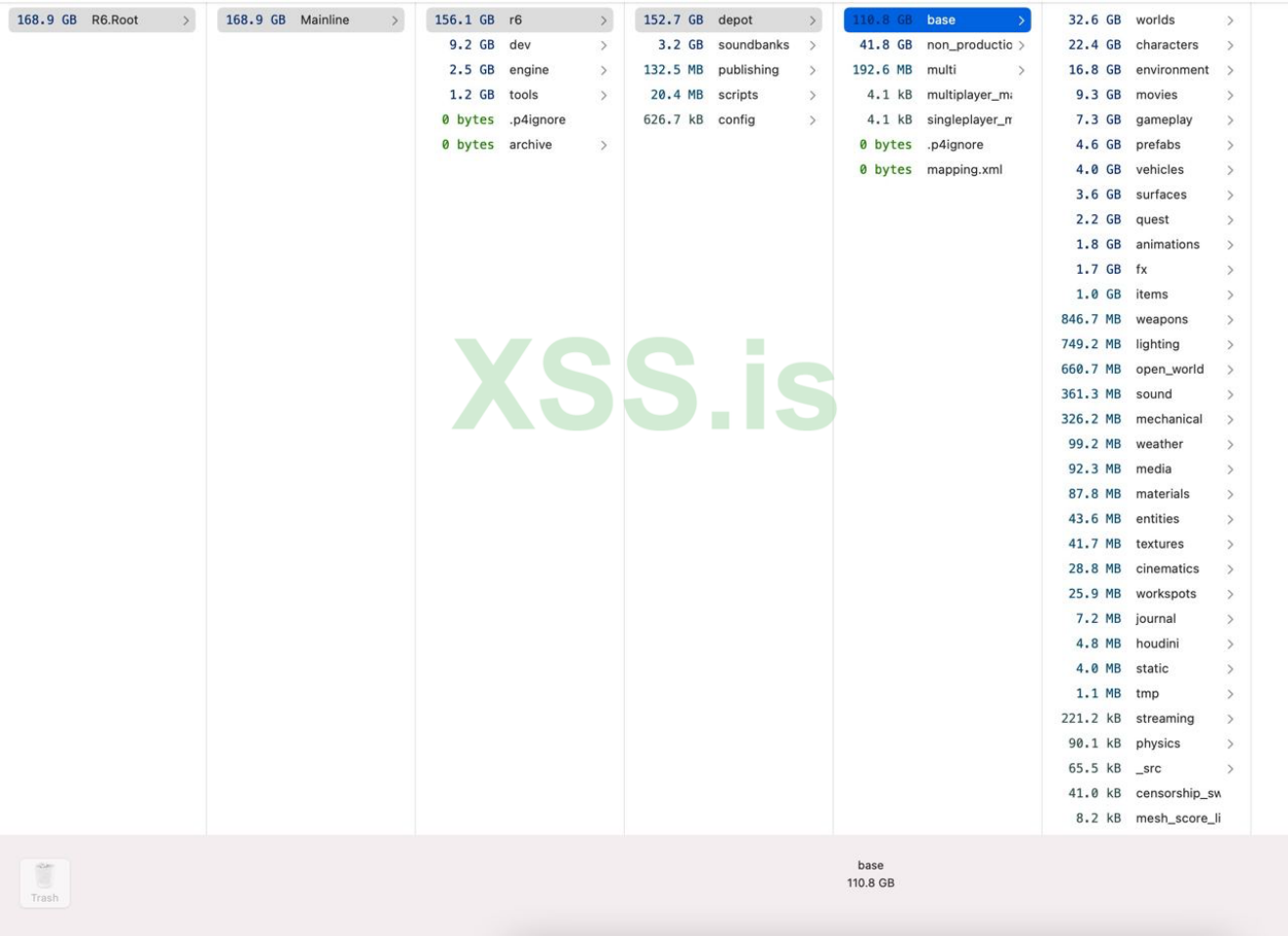
Для удобства использования rustyHog разбит на компоненты по области применения:

- Ankamali Hog: для работы с гугл-диском.
- Berkshire Hog: для работы с S3-инстансами.
- Choctaw Hog: для работы с Git-репозиториями.
- Duroc Hog: для работы с файловой системой.
- Essex Hog: для работы с Confluence.
- Gottingen Hog: Sдля работы с JIRA.

Список параметров у каждой из утилит типовой, пример справки для duroc_hog приведён на рисунке ниже.

Анализ данных

Чтобы продемонстрировать возможности анализа необходим хороший и наглядный пример: слитая в паблик утечка CD-Project отлично для этого подойдёт.
В числе выложенных и доступных архивов для анализа практическую пользу имеет только архив cp.7z, весящий 120 Гб, после распаковки имеем 160 Гб данных (структура каталогов приведена на рисунке далее). С помощью rustyHog найти что-то интересное становится достаточно просто.

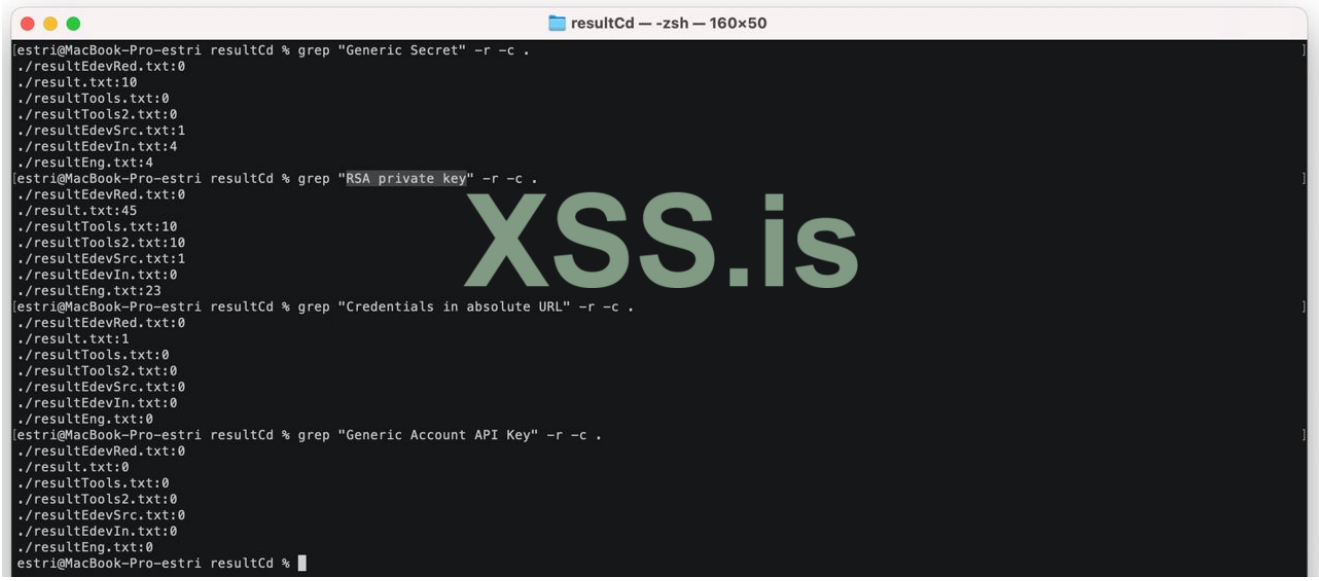


В логе каждая из записей содержит поля, позволяющие однозначно понять что было найдено и где (путь до файла, номер строки, выявленный тип данных). Список используемых по умолчанию полей доступен здесь. По умолчанию он содержит большое количество регулярных выражений, с помощью которых могут быть найдены секреты без применения энтропии.

Результат.
Пример выявленных интересных строк, с использованием энтропии приведён на рисунке ниже. Полностью результаты намеренно не привожу, т.к. анализ этих результатов показал отсутствие какой-либо полезной информации.

```
{
  "stringsFound": [
    "R6flPri9fcebNaBhlzpBdRfMK5Z3KpIhHtmVdiBnaM8Nvd/WHwlqmuLMc3GkL30SgLdTMEZeS1SZ"
  ],
  "path": "./utility/BuildPatchTool/Engine/Content/Certificates/ThirdParty/cacert.pem",
  "reason": "Entropy",
  "linenum": 3274,
  "diff": "R6flPri9fcebNaBhlzpBdRfMK5Z3KpIhHtmVdiBnaM8Nvd/WHwlqmuLMc3GkL30SgLdTMEZeS1SZ\r"
},
{
  "stringsFound": [
    "lDePSHFjIuwXZQeVikvfj8ZaCuWw419eaxGrDPmF60Tp+ARz8un+XJiM9XOva7R+zdRcAitMOeGy"
  ],
  "path": "./utility/BuildPatchTool/Engine/Content/Certificates/ThirdParty/cacert.pem",
  "reason": "Entropy",
  "linenum": 352,
  "diff": "lDePSHFjIuwXZQeVikvfj8ZaCuWw419eaxGrDPmF60Tp+ARz8un+XJiM9XOva7R+zdRcAitMOeGy\r"
},
```


Из полного списка шаблонов регулярных выражений для поиска заслуживают внимания следующие типы секретов: Generic Secret, RSA private key, Credentials in absolute URL, Generic Account API Key (остальные не рассматриваем, т.к. их наличие маловероятно). Результаты выявленных секретов приведены далее на рисунках.




```
},
{
  "stringsFound": [
    "Secret = \"5fe9feaf7ad2ee2080764faf8a5a45461eed231672c709cf8536aade8c0699eb\""
  ],
  "path": "./engine/config/platform/pc/serverconfig.ini",
  "reason": "Generic Secret",
  "linenum": 10,
  "diff": "ClientSecret = \"5fe9feaf7ad2ee2080764faf8a5a45461eed231672c709cf8536aade8c0699eb\"\r"
},
```



```
{
  "stringsFound": [
    "-----BEGIN RSA PRIVATE KEY-----"
  ],
  "path": "./dev/external/libmicrohttpd/libmicrohttpd-0.9.64/src/testcurl/https/tls_test_keys.h",
  "reason": "RSA private key",
  "linenum": 26,
  "diff": "const char ca_key_pem[] = \"-----BEGIN RSA PRIVATE KEY-----\\n\"\r"
```

```
{
  "stringsFound": [
    "ftp://login:pass@bleh.gif"
  ],
  "path": "./dev/external/icu/source/test/testdata/regextst.txt",
  "reason": "Credentials in absolute URL",
  "linenum": 1818,
  "diff": "\"(\\[[Ii][Mm][Gg]\\])(\\S+?)(\\[\\/[Ii][Mm][Gg]\\])\"   G \"<0>[img]ftp://login:pass@bleh.gif[/img]</0>\""
},
```


Итог анализа

В составе "утечки" CD-Project слитой в паблик было выявлено:

- несколько приватных ключей;
- несколько секретов, используемых при сборке-отладке;
- большое количество адресов email разработчиков используемых opensource компонентов;
- очень большая куча мусора.

Ничего из выявленного не представляет ценности как таковой.
Результат в целом соответствует действительности: были слиты части SDK и куски кода, не содержащие какой-либо ценности.
Всё то же, что имеет хоть какой-то вес содержится в архивах под паролем, о чём было написано в readme-файле в торрент раздаче данной "утечки".

readme_ru.txt — Open with TextEdit

```
Привет чумбы

Объявляем сбор средств на благотворительность (кто не верит может пойти нахуй), взамен за каждые собранные 10к$, мы
предоставим по одному паролю за выложенные архивы.

В очереди: thronebreaker.7z, w3.7z, w3rtx.7z, cp.7z
Что внутри? исходны коды и асеты: Кровная вражда: Ведьмак, ведьмак 3 ванила (вроде тот движок с E3), ведьмак 3 некст
ген, киберпанк2077
Пароли мы релизнем на http://qgrjbug5str5yvc37vzzksfitsc6vfuzqqmyt5u6b25o4woaezht4bid.onion/ если упадёт просто в
социалные сети.

Что требуется от вас?
        1. Скачать
        2. Раздать
        3. Задонатить на адреса
                BTC: 1KCxq5Z9SyCBudXfJoCFpsRVydTea7wV7F
                ETH: 0x3fb3a7Fa96C673B247ee19E86beFF3622B1cF353
                DOGE: DQ2UB3CFttqVw7GsHQyLwPBfszkmLV84xh
        4. Рассказать другу

И всем для разогрева, в архивах на SDK ps4 / ps5 / switch / xbox пароль не установлен!

CDPR Данные и отчеты вашей компании, нда, отданы в сми, публично эти данные не будут выложены нами!

Этот слив согласован с покупателем данных CDPR, в обмен на скидку.

П.С.
Илон Маск, мужик, пампани биток 1 апреля плиз.

Wake up samurai, we have a city to burn!!! :D
```

Заключение

В этой статье мы рассмотрели простой способ для быстрого анализа данных, а вот где применять его решать вам.
Спасибо за внимание!

P.S.: Статья написана при содействии club1337, спасибо.
Last edited: Jul 15, 2021