Trevor Hornsby

Archive of Personal Tools for NCL and other CTFs

OSINT

- https://wigle.net/
    - Public info on wifi networks
- Sherlock (PYTHON SCRIPT) - https://github.com/sherlock-project/sherlock
- Geo recon - https://github.com/radioactivetobi/geo-recon
- SSL/ .crt decode - https://certlogik.com/decoder/
- Partial QR Decoder - https://merricx.github.io/qrazybox/
- Chrome session forensics
    - http://lsauer.net/chrome-session-restore/ (pretty bad tbh)
    - https://github.com/JRBANCEL/Chromagnon (pretty good)

Image Metadata/ stego

- http://metapicz.com/#landing
- http://exif.regex.info/exif.cgi
- Digital invisible toolkit - http://diit.sourceforge.net/download.php
- Image magick - https://imagemagick.org/index.php
- Openpuff - https://embeddedsw.net/OpenPuff_Steganography_Home.html
- https://incoherency.co.uk/image-steganography/#unhide
- https://www.rbcafe.com/software/outguess/
- https://github.com/KuroLabs/stegcloak
- http://manpages.ubuntu.com/manpages/bionic/man1/zbarimg.1.html
- Cat, more, strings

Reverse Image Search

- https://tineye.com/
- Google
- https://yandex.com/images/
- https://hostingchecker.com/tools/reverse-image-search/

Crypto

- https://gchq.github.io/CyberChef/
- Binwalk
- https://www.dcode.fr/en
- http://rumkin.com/tools/
- Shift Cipher - https://goto.pachanka.org/crypto/shift-cipher
- https://www.rapidtables.com
- https://crypto.interactive-maths.com/pigpen-cipher.html
- https://www.wfonts.com/font/masonic-cipher
- RSA

  - https://www.alpertron.com.ar/ECM.HTM
  - https://asecuritysite.com/encryption/factors?n=1034776851837418228051242693253376923

```
def egcd(a, b):
      x,y, u,v = 0,1, 1,0
      while a != 0:
      q, r = b//a, b%a
      m, n = x-u*q, y-v*q
      b,a, x,y, u,v = a,r, u,v, m,n
      gcd = b
      return gcd, x, y

def main():

      p = 255097177
      q = 2203439394347318375616311846034251943 0053
      e = 65537
      ct = 1415060907955076984980255543080831671725408472748

      # compute n
      n = p * q

      # Compute phi(n)
      phi = (p - 1) * (q - 1)

      # Compute modular inverse of e
      gcd, a, b = egcd(e, phi)
      d = a

      print( "n:  " + str(d) );

      # Decrypt ciphertext
      pt = pow(ct, d, n)
      print( "pt: " + str(pt) )

if __name__ == "__main__":
      main()
```

      - Then convert to text

Passwords

- John the ripper - https://www.openwall.com/john/doc/
- Hashcat - https://hashcat.net/wiki/

    - Put pcap into hashcat
    - Rules
        - One rule to rule them all
- Crackstation - https://crackstation.net/
- aircrack -ng for wireless - https://www.aircrack-ng.org/documentation.html
- Pcap - https://hashcat.net/cap2hashcat/

Wordlists

- Seclists - https://github.com/danielmiessler/SecLists
- Rockyou - Download
- Wordlister (PYTHON SCRIPT) - https://github.com/4n4nk3/Wordlister
- http://wordlists.assetnote.io/
- Ophcrack (rainbowtables) - https://ophcrack.sourceforge.io/
    - https://ophcrack.sourceforge.io/tables.php
- Scraper Script (use as reference, will not work for every site)

```
import urllib.request, re

# Method to extract facult records from site that is passed to it, returns list of objects where each object contains the values we extract
def Facutly_record_extraction(site):
    # Request HTML source code from site
    req =urllib.request.Request(site,headers={'User-Agent': 'Mozilla/5.0'})
    u2 = urllib.request.urlopen(req)
    website = u2.read().decode('utf-8')

    # Split HTML source by person
    parts = website.split('az-list')

    # Extract data
    for part in parts:

        hero = re.findall('<a href=.*>(.*)</a>', part)
        for i in hero:
            new = i.replace(" ", "")
            re.sub('\([^()]*\)', '', new)
            with open("heroes.txt", "a") as f:
                print(new)
                f.write(new + '\n')

def main():
    faculty = Facutly_record_extraction('https://www.britannica.com/topic/list-of-superheroes-2024795')
    faculty = Facutly_record_extraction('https://superheroes.fandom.com/wiki/List_of_DC_Comics_Characters')
    faculty = Facutly_record_extraction('https://www.marvel.com/comics/characters')
# Calling main method
main()
```

Logs

- Linux
    - cat - https://man7.org/linux/man-pages/man1/cat.1.html
    - grep - https://www.gnu.org/software/grep/manual/grep.html
    - awk - https://opensource.com/article/19/11/how-regular-expressions-awk
- Excel / Google Sheets
    - https://support.microsoft.com/en-us/office/split-a-column-of-text-power-query-5282d425-6dd0-46ca-95bf-8e0da9539662
    - https://edu.gcfglobal.org/en/excel2013/filtering-data/1/
- Splunk - https://www.splunk.com/en_us/download.html

enum/ exploit

- Chr is opposite of ord in python
- https://hoppscotch.io/
- Hex Bit XOR File Decryption

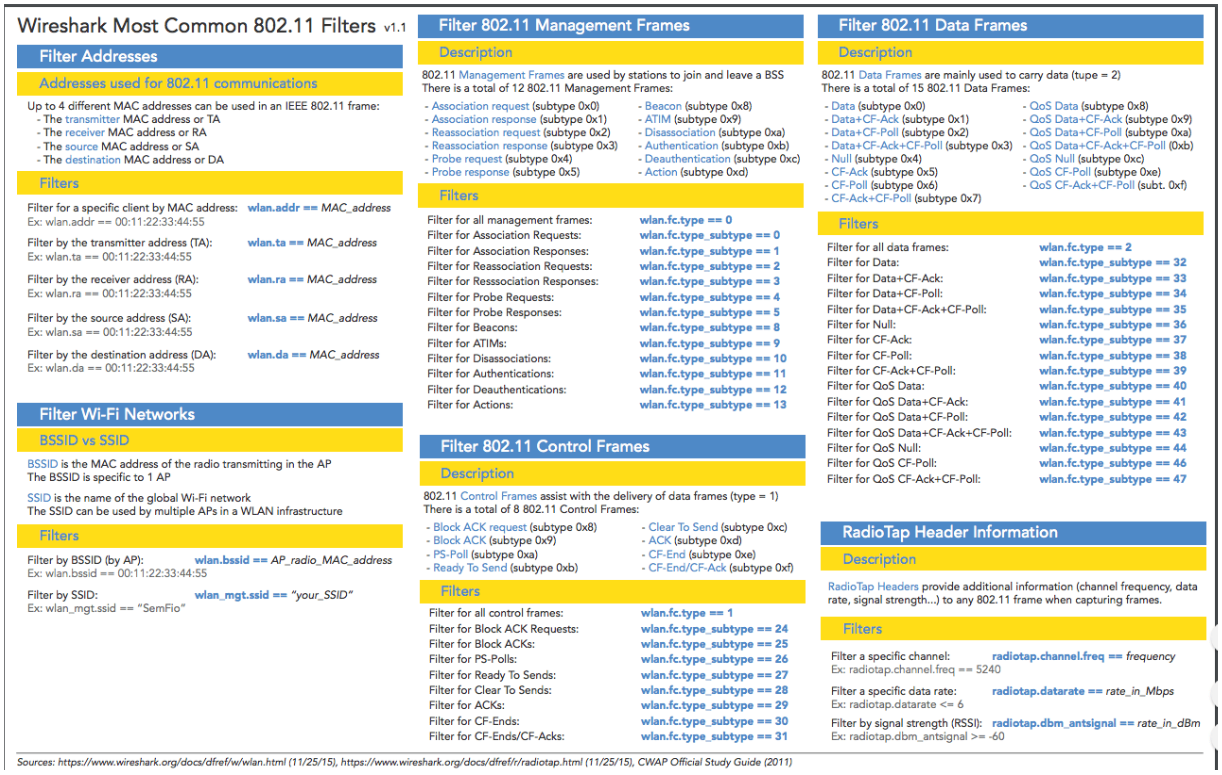
```
key = 0x2c

with open('ransom.png', 'rb') as f:
    img = f.read()

f = open('out.png', 'wb')
count = 0
arr = []
for bit in img:
    arr.append(bit)

count = 0
```

Trevor Hornsby

```
for he in arr:
    if count == 0:

        new = int(arr[0]) ^ key
    else:
        new = int(arr[count]) ^ int(arr[count - 1])

    f.write(bytes.fromhex("b'{:02x}'".format(new)[2:4]))

    count += 1
```

Network

- https://github.com/odedshimon/BruteShark
- https://www.netresec.com/
- https://www.wireshark.org/
- https://hashcat.net/cap2hashcat/



Wireshark Most Common 802.11 Filters v1.1

Filter Addresses

Addresses used for 802.11 communications

Up to 4 different MAC addresses can be used in an IEEE 802.11 frame:
- The transmitter MAC address or TA
- The receiver MAC address or RA
- The source MAC address or SA
- The destination MAC address or DA

Filters

Filter for a specific client by MAC address: Ex: wlan.addr == 00:11:22:33:44:55	**wlan.addr ==** *MAC_address*
Filter by the transmitter address (TA): Ex: wlan.ta == 00:11:22:33:44:55	**wlan.ta ==** *MAC_address*
Filter by the receiver address (RA): Ex: wlan.ra == 00:11:22:33:44:55	**wlan.ra ==** *MAC_address*
Filter by the source address (SA): Ex: wlan.sa == 00:11:22:33:44:55	**wlan.sa ==** *MAC_address*
Filter by the destination address (DA): Ex: wlan.da == 00:11:22:33:44:55	**wlan.da ==** *MAC_address*

Filter Wi-Fi Networks

BSSID vs SSID

BSSID is the MAC address of the radio transmitting in the AP
The BSSID is specific to 1 AP

SSID is the name of the global Wi-Fi network
The SSID can be used by multiple APs in a WLAN infrastructure

Filters

Filter by BSSID (by AP): Ex: wlan.bssid == 00:11:22:33:44:55	**wlan.bssid ==** *AP_radio_MAC_address*
Filter by SSID: Ex: wlan_mgt.ssid == "SemFio"	**wlan_mgt.ssid ==** *"your_SSID"*

Filter 802.11 Management Frames

Description

802.11 Management Frames are used by stations to join and leave a BSS
There is a total of 12 802.11 Management Frames:

- Association request (subtype 0x0)
- Association response (subtype 0x1)
- Reassociation request (subtype 0x2)
- Reassociation response (subtype 0x3)
- Probe request (subtype 0x4)
- Probe response (subtype 0x5)
- Beacon (subtype 0x8)
- ATIM (subtype 0x9)
- Disassociation (subtype 0xa)
- Authentication (subtype 0xb)
- Deauthentication (subtype 0xc)
- Action (subtype 0xd)

Filters

Filter for all management frames:	**wlan.fc.type == 0**
Filter for Association Requests:	**wlan.fc.type_subtype == 0**
Filter for Association Responses:	**wlan.fc.type_subtype == 1**
Filter for Reassociation Requests:	**wlan.fc.type_subtype == 2**
Filter for Resssociation Responses:	**wlan.fc.type_subtype == 3**
Filter for Probe Requests:	**wlan.fc.type_subtype == 4**
Filter for Probe Responses:	**wlan.fc.type_subtype == 5**
Filter for Beacons:	**wlan.fc.type_subtype == 8**
Filter for ATIMs:	**wlan.fc.type_subtype == 9**
Filter for Disassociations:	**wlan.fc.type_subtype == 10**
Filter for Authentications:	**wlan.fc.type_subtype == 11**
Filter for Deauthentications:	**wlan.fc.type_subtype == 12**
Filter for Actions:	**wlan.fc.type_subtype == 13**

Filter 802.11 Control Frames

Description

802.11 Control Frames assist with the delivery of data frames (type = 1)
There is a total of 8 802.11 Control Frames:

- Block ACK request (subtype 0x8)
- Block ACK (subtype 0x9)
- PS-Poll (subtype 0xa)
- Ready To Send (subtype 0xb)
- Clear To Send (subtype 0xc)
- ACK (subtype 0xd)
- CF-End (subtype 0xe)
- CF-End/CF-Ack (subtype 0xf)

Filters

Filter for all control frames:	**wlan.fc.type == 1**
Filter for Block ACK Requests:	**wlan.fc.type_subtype == 24**
Filter for Block ACKs:	**wlan.fc.type_subtype == 25**
Filter for PS-Polls:	**wlan.fc.type_subtype == 26**
Filter for Ready To Sends:	**wlan.fc.type_subtype == 27**
Filter for Clear To Sends:	**wlan.fc.type_subtype == 28**
Filter for ACKs:	**wlan.fc.type_subtype == 29**
Filter for CF-Ends:	**wlan.fc.type_subtype == 30**
Filter for CF-Ends/CF-Acks:	**wlan.fc.type_subtype == 31**

Filter 802.11 Data Frames

Description

802.11 Data Frames are mainly used to carry data (tupe = 2)
There is a total of 15 802.11 Data Frames:

- Data (subtype 0x0)
- Data+CF-Ack (subtype 0x1)
- Data+CF-Poll (subtype 0x2)
- Data+CF-Ack+CF-Poll (subtype 0x3)
- Null (subtype 0x4)
- CF-Ack (subtype 0x5)
- CF-Poll (subtype 0x6)
- CF-Ack+CF-Poll (subtype 0x7)
- QoS Data (subtype 0x8)
- QoS Data+CF-Ack (subtype 0x9)
- QoS Data+CF-Poll (subtype 0xa)
- QoS Data+CF-Ack+CF-Poll (0xb)
- QoS Null (subtype 0xc)
- QoS CF-Poll (subtype 0xe)
- QoS CF-Ack+CF-Poll (subt. 0xf)

Filters

Filter for all data frames:	**wlan.fc.type == 2**
Filter for Data:	**wlan.fc.type_subtype == 32**
Filter for Data+CF-Ack:	**wlan.fc.type_subtype == 33**
Filter for Data+CF-Poll:	**wlan.fc.type_subtype == 34**
Filter for Data+CF-Ack+CF-Poll:	**wlan.fc.type_subtype == 35**
Filter for Null:	**wlan.fc.type_subtype == 36**
Filter for CF-Ack:	**wlan.fc.type_subtype == 37**
Filter for CF-Poll:	**wlan.fc.type_subtype == 38**
Filter for CF-Ack+CF-Poll:	**wlan.fc.type_subtype == 39**
Filter for QoS Data:	**wlan.fc.type_subtype == 40**
Filter for QoS Data+CF-Ack:	**wlan.fc.type_subtype == 41**
Filter for QoS Data+CF-Poll:	**wlan.fc.type_subtype == 42**
Filter for QoS Data+CF-Ack+CF-Poll:	**wlan.fc.type_subtype == 43**
Filter for QoS Null:	**wlan.fc.type_subtype == 44**
Filter for QoS CF-Poll:	**wlan.fc.type_subtype == 46**
Filter for QoS CF-Ack+CF-Poll:	**wlan.fc.type_subtype == 47**

RadioTap Header Information

Description

RadioTap Headers provide additional information (channel frequency, data rate, signal strength...) to any 802.11 frame when capturing frames.

Filters

Filter a specific channel: Ex: radiotap.channel.freq == 5240	**radiotap.channel.freq ==** *frequency*
Filter a specific data rate: Ex: radiotap.datarate <= 6	**radiotap.datarate ==** *rate_in_Mbps*
Filter by signal strength (RSSI): Ex: radiotap.dbm_antsignal >= -60	**radiotap.dbm_antsignal ==** *rate_in_dBm*

Sources: https://www.wireshark.org/docs/dfref/w/wlan.html (11/25/15), https://www.wireshark.org/docs/dfref/r/radiotap.html (11/25/15), CWAP Official Study Guide (2011)

Dakoda's stuff

```
Capture The Flag Resource List
-------------------------------
```

Trevor Hornsby

NCL

---Semi-Annual Colligate Comps, sponsored by College---
NCL Gymnasium -- Pre-Season General Walk Through with solutions
NCL Individual Game -- Harder than Gym but still overall easy
NCL Team Game -- Harder than Individual game, team of 7

Captf

---List of permanent CTF's for practice---
https://captf.com/practice-ctf/

Hack-The-Box

---Basically a more difficult NCL, Not entry level---
hackthebox.eu

Over-The-Wire

---Good introduction---
overthewire.org

Micro Corruption

---Reverse engineering specific, very pretty---
microcorruption.com

TryHackME

---If you are extremely new to cybersecurity, i would suggest starting off with this.---
---They have a whole pathway with basic boxes that teach you basics of different tools.--
https://www.tryhackme.com/

Big CTF Checklist + Tools

Trevor Hornsby

---Just a Big CTF Checklist + Tools---
https://github.com/JohnHammond/ctf-katana
https://github.com/enaqx/awesome-pentest

CTF Challenge Search + Writeups

---Amazing resource for learning harder challenges---
https://ctf.courgettes.club/

Shane's NCL Writeups

---Please use your MTU emails to access it. (Note, some are incomplete because I was not smart enough, or I was too lazy.---
Credit goes to @Shane H
https://drive.google.com/drive/folders/19brnJsSIMOqHz3zF1VqSgCEcfZOyJZhK?usp=sharing
----

CTF Tools

Stenography

Binwalk
Digital Invisible Ink Toolkit
ZSteg
futureboy.us/stegano/decinput.html
manytools.org/hacker-tools/steganography-encode-text-into-image
----

Cryptography

cyberchef -- http://icyberchef.com/
pycrypto -- library for python -- install using -> pip install pycrypto
crackstation

Reverse Engineering\Exploitation and Exploitation

Trevor Hornsby

https://ghidra-sre.org/ --- Basics in E&E Presentation
GDB --- Basics in E&E Presentation

Web Exploitation

Burb
BurpSuite

Wireless access/Network Exploitation

Hashcat
Aircrack-ng
SecList -- https://github.com/danielmiessler/SecLists
