Homework2

February 26, 2017

1 MSDS 7349 Data and Network Security

1.0.1 Homework Basic Security

By: Kyle Killion

Sources : * Violent Python: A Cookbook for Hackers, Forensic Analysts, Penetration Testers and Security Engineers. By : TJ O'Connor https://docs.google.com/file/d/0B-F3NpsEIXCYcDZaUXhfdXlFM1k/edit

1.1 Table of Contents:

# Exercise 1: UNIX Password Cracker

- The Password for **Victim** was found: Hash found in password list = **HX9LLTdc/jiDE** salt = **HX** [+] Found password : **egg**
  [+] Matches Hash sig: **HX9LLTdc/jiDE**

- No Password was found for the Root which means we don't have the Password in our dictionary.

- The salt value can be taken from the first two characters in the Hash Signature: $sv = hash[0:2]$

```
In [41]: import numpy as np
         import pandas as pd
         from passlib.hash import des_crypt



         def testPwd(hash):
             sv = hash[0:2]
             print('\nhash found in password list =',hash,' salt =',sv,'\n')
             dtFile = open('HW2dictionary.txt','r')
             for word in dtFile.readlines():
                 word = word.strip('\n')
                 cryptword = des_crypt.hash(word,salt=sv)
                 if cryptword == hash:
```

```
                print('\n[+] Found password :', word, '\n[+] Matches Hash Sig
            else:
                print('[-]no match for', word, cryptword)

    pwdFile = open('HW2passwords.txt','r')

    for line in pwdFile.readlines():
        if ':' in line:
            user = line.split(':')[0]

            hv = line.split(':')[1].strip(' ')
            print('\n[*] Crackin PWD for :', user)

            if len(hv) == 13:
                checkthis = 'True'
                testPwd(hv)
            else:
                checkthis = 'False'
```

```
[*] Crackin PWD for : victim

hash found in password list = HX9LLTdc/jiDE  salt = HX

[-]no match for apple HXJintBqUVCEY
[-]no match for orange HX6dAZy7TqQE.

[+] Found password : egg
[+] Matches Hash Sig : HX9LLTdc/jiDE

[-]no match for lemon HXCtvQhLXGgZg
[-]no match for grapes HXtZSWbomS0xQ
[-]no match for secret HXXxJi0n6Huro
[-]no match for strawberry HXENul7GkUdlM
[-]no match for password HXHE1BtHtA3N2

[*] Crackin PWD for : root

hash found in password list = DFNFxgW7C05fo  salt = DF

[-]no match for apple DFWnBavAebFoM
[-]no match for orange DFLxqyp4Kja72
[-]no match for egg DFV1s9aC7EHaw
[-]no match for lemon DFbvAVNj8uv2A
[-]no match for grapes DFslWvHbYHibg
[-]no match for secret DF2DT.ZFW5s96
[-]no match for strawberry DFfSzRsQPaTcc
[-]no match for password DFkS2oDEJxgmY
```

# Exercise 2 : Zip File Password Cracker
The Help file was very useful due to running into a Python 3.2 + Issue. It helped me find a workaround.

```
In [8]: help('zipfile')
```

```
Help on module zipfile:

NAME
    zipfile - Read and write ZIP files.

DESCRIPTION
    XXX references to utf-8 need further investigation.

CLASSES
    builtins.Exception(builtins.BaseException)
        BadZipFile
        LargeZipFile
    builtins.object
        ZipFile
            PyZipFile
        ZipInfo

    class BadZipFile(builtins.Exception)
     |  Common base class for all non-exit exceptions.
     |
     |  Method resolution order:
     |      BadZipFile
     |      builtins.Exception
     |      builtins.BaseException
     |      builtins.object
     |
     |  Data descriptors defined here:
     |
     |  __weakref__
     |      list of weak references to the object (if defined)
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from builtins.Exception:
     |
     |  __init__(self, /, *args, **kwargs)
     |      Initialize self.  See help(type(self)) for accurate signature.
     |
     |  __new__(*args, **kwargs) from builtins.type
     |      Create and return a new object.  See help(type) for accurate signature.
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from builtins.BaseException:
     |
     |  __delattr__(self, name, /)
     |      Implement delattr(self, name).
     |
     |  __getattribute__(self, name, /)
     |      Return getattr(self, name).
     |
     |  __reduce__(...)
     |      helper for pickle
     |
     |  __repr__(self, /)
     |      Return repr(self).
     |
     |  __setattr__(self, name, value, /)
     |      Implement setattr(self, name, value).
     |
     |  __setstate__(...)
     |
     |  __str__(self, /)
     |      Return str(self).
     |
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from builtins.BaseException:
     |
     |  __cause__
     |      exception cause
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args

    BadZipfile = class BadZipFile(builtins.Exception)
     |  Common base class for all non-exit exceptions.
     |
     |  Method resolution order:
```

```
 |      BadZipFile
 |      builtins.Exception
 |      builtins.BaseException
 |      builtins.object
 |
 |  Data descriptors defined here:
 |
 |  __weakref__
 |      list of weak references to the object (if defined)
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from builtins.Exception:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __new__(*args, **kwargs) from builtins.type
 |      Create and return a new object.  See help(type) for accurate signature.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from builtins.BaseException:
 |
 |  __delattr__(self, name, /)
 |      Implement delattr(self, name).
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      helper for pickle
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setattr__(self, name, value, /)
 |      Implement setattr(self, name, value).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from builtins.BaseException:
```

```
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class LargeZipFile(builtins.Exception)
 |  Raised when writing a zipfile, the zipfile requires ZIP64 extensions
 |  and those extensions are disabled.
 |
 |  Method resolution order:
 |      LargeZipFile
 |      builtins.Exception
 |      builtins.BaseException
 |      builtins.object
 |
 |  Data descriptors defined here:
 |
 |  __weakref__
 |      list of weak references to the object (if defined)
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from builtins.Exception:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __new__(*args, **kwargs) from builtins.type
 |      Create and return a new object.  See help(type) for accurate signature
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from builtins.BaseException:
 |
 |  __delattr__(self, name, /)
 |      Implement delattr(self, name).
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      helper for pickle
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
 |  __setattr__(self, name, value, /)
 |      Implement setattr(self, name, value).
 |
 |  __setstate__(...)
 |
 |  __str__(self, /)
 |      Return str(self).
 |
 |  with_traceback(...)
 |      Exception.with_traceback(tb) --
 |      set self.__traceback__ to tb and return self.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from builtins.BaseException:
 |
 |  __cause__
 |      exception cause
 |
 |  __context__
 |      exception context
 |
 |  __dict__
 |
 |  __suppress_context__
 |
 |  __traceback__
 |
 |  args

class PyZipFile(ZipFile)
 |  Class to create ZIP archives with Python library files and packages.
 |
 |  Method resolution order:
 |      PyZipFile
 |      ZipFile
 |      builtins.object
 |
 |  Methods defined here:
 |
 |  __init__(self, file, mode='r', compression=0, allowZip64=True, optimize=-1)
 |      Open the ZIP file with mode read 'r', write 'w', exclusive create 'x',
 |      or append 'a'.
 |
 |  writepy(self, pathname, basename='', filterfunc=None)
 |      Add all files from "pathname" to the ZIP archive.
 |
 |      If pathname is a package directory, search the directory and
 |      all package subdirectories recursively for all *.py and enter
 |      the modules into the archive.  If pathname is a plain
 |      directory, listdir *.py and enter all modules.  Else, pathname
 |      must be a Python *.py file and the module will be put into the
 |      archive.  Added modules are always module.pyc.
 |      This method will compile the module.py into module.pyc if
 |      necessary.
 |      If filterfunc(pathname) is given, it is called with every argument.
 |      When it is False, the file or directory is skipped.
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from ZipFile:
 |
 |  __del__(self)
 |      Call the "close()" method in case the user forgot.
 |
 |  __enter__(self)
 |
 |  __exit__(self, type, value, traceback)
 |
 |  __repr__(self)
 |      Return repr(self).
 |
 |  close(self)
 |      Close the file, and for mode 'w', 'x' and 'a' write the ending
 |      records.
 |
 |  extract(self, member, path=None, pwd=None)
 |      Extract a member from the archive to the current working directory,
 |      using its full name. Its file information is extracted as accurately
 |      as possible. `member' may be a filename or a ZipInfo object. You can
 |      specify a different directory using `path'.
 |
 |  extractall(self, path=None, members=None, pwd=None)
 |      Extract all members from the archive to the current working
 |      directory. `path' specifies a different directory to extract to.
 |      `members' is optional and must be a subset of the list returned
 |      by namelist().
 |
 |  getinfo(self, name)
 |      Return the instance of ZipInfo given 'name'.
 |
 |  infolist(self)
 |      Return a list of class ZipInfo instances for files in the
 |      archive.
 |
 |  namelist(self)
 |      Return a list of file names in the archive.
 |
 |  open(self, name, mode='r', pwd=None)
 |      Return file-like object for 'name'.
 |
 |  printdir(self, file=None)
 |      Print a table of contents for the zip file.
 |
 |  read(self, name, pwd=None)
 |      Return file bytes (as a string) for name.
 |
 |  setpassword(self, pwd)
 |      Set default password for encrypted files.
 |
 |  testzip(self)
 |      Read all the files and check the CRC.
 |
 |  write(self, filename, arcname=None, compress_type=None)
 |      Put the bytes from filename into the archive under the name
 |      arcname.
 |
 |  writestr(self, zinfo_or_arcname, data, compress_type=None)
 |      Write a file into the archive.  The contents is 'data', which
 |      may be either a 'str' or a 'bytes' instance; if it is a 'str',
 |      it is encoded as UTF-8 first.
 |      'zinfo_or_arcname' is either a ZipInfo instance or
 |      the name of the file in the archive.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from ZipFile:
 |
 |  __dict__
 |      dictionary for instance variables (if defined)
 |
 |  __weakref__
 |      list of weak references to the object (if defined)
 |
 |  comment
 |      The comment text associated with the ZIP file.
 |
 |  ----------------------------------------------------------------------
 |  Data and other attributes inherited from ZipFile:
 |
 |  fp = None
```

```
class ZipFile(builtins.object)
 |  Class with methods to open, read, write, close, list zip files.
 |
 |  z = ZipFile(file, mode="r", compression=ZIP_STORED, allowZip64=True)
 |
 |  file: Either the path to the file, or a file-like object.
 |        If it is a path, the file will be opened and closed by ZipFile.
 |  mode: The mode can be either read 'r', write 'w', exclusive create 'x',
 |        or append 'a'.
 |  compression: ZIP_STORED (no compression), ZIP_DEFLATED (requires zlib),
 |               ZIP_BZIP2 (requires bz2) or ZIP_LZMA (requires lzma).
 |  allowZip64: if True ZipFile will create files with ZIP64 extensions when
 |              needed, otherwise it will raise an exception when this would
 |              be necessary.
 |
 |  Methods defined here:
 |
 |  __del__(self)
 |      Call the "close()" method in case the user forgot.
 |
 |  __enter__(self)
 |
 |  __exit__(self, type, value, traceback)
 |
 |  __init__(self, file, mode='r', compression=0, allowZip64=True)
 |      Open the ZIP file with mode read 'r', write 'w', exclusive create 'x',
 |      or append 'a'.
 |
 |  __repr__(self)
 |      Return repr(self).
 |
 |  close(self)
 |      Close the file, and for mode 'w', 'x' and 'a' write the ending
 |      records.
 |
 |  extract(self, member, path=None, pwd=None)
 |      Extract a member from the archive to the current working directory,
 |      using its full name. Its file information is extracted as accurately
 |      as possible. `member' may be a filename or a ZipInfo object. You can
 |      specify a different directory using `path'.
 |
 |  extractall(self, path=None, members=None, pwd=None)
 |      Extract all members from the archive to the current working
 |      directory. `path' specifies a different directory to extract to.
 |      `members' is optional and must be a subset of the list returned
 |      by namelist().
 |
 |  getinfo(self, name)
 |      Return the instance of ZipInfo given 'name'.
 |
 |  infolist(self)
 |      Return a list of class ZipInfo instances for files in the
 |      archive.
 |
 |  namelist(self)
 |      Return a list of file names in the archive.
 |
 |  open(self, name, mode='r', pwd=None)
 |      Return file-like object for 'name'.
 |
 |  printdir(self, file=None)
 |      Print a table of contents for the zip file.
 |
 |  read(self, name, pwd=None)
 |      Return file bytes (as a string) for name.
 |
 |  setpassword(self, pwd)
 |      Set default password for encrypted files.
 |
 |  testzip(self)
 |      Read all the files and check the CRC.
 |
 |  write(self, filename, arcname=None, compress_type=None)
 |      Put the bytes from filename into the archive under the name
 |      arcname.
 |
 |  writestr(self, zinfo_or_arcname, data, compress_type=None)
 |      Write a file into the archive.  The contents is 'data', which
 |      may be either a 'str' or a 'bytes' instance; if it is a 'str',
 |      it is encoded as UTF-8 first.
 |      'zinfo_or_arcname' is either a ZipInfo instance or
 |      the name of the file in the archive.
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  __dict__
 |      dictionary for instance variables (if defined)
 |
 |  __weakref__
 |      list of weak references to the object (if defined)
 |
 |  comment
 |      The comment text associated with the ZIP file.
 |
 |  ----------------------------------------------------------------------
 |  Data and other attributes defined here:
 |
 |  fp = None

class ZipInfo(builtins.object)
 |  Class with attributes describing each file in the ZIP archive.
 |
 |  Methods defined here:
 |
 |  FileHeader(self, zip64=None)
 |      Return the per-file header as a string.
 |
 |  __init__(self, filename='NoName', date_time=(1980, 1, 1, 0, 0, 0))
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __repr__(self)
 |      Return repr(self).
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  CRC
 |
 |  comment
 |
 |  compress_size
 |
 |  compress_type
 |
 |  create_system
 |
 |  create_version
 |
 |  date_time
 |
 |  external_attr
 |
 |  extra
 |
 |  extract_version
 |
 |  file_size
 |
 |  filename
 |
 |  flag_bits
 |
 |  header_offset
 |
 |  internal_attr
 |
 |  orig_filename
 |
 |  reserved
 |
 |  volume

error = class BadZipFile(builtins.Exception)
 |  Common base class for all non-exit exceptions.
 |
 |  Method resolution order:
 |      BadZipFile
 |      builtins.Exception
 |      builtins.BaseException
 |      builtins.object
 |
 |  Data descriptors defined here:
 |
 |  __weakref__
 |      list of weak references to the object (if defined)
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from builtins.Exception:
 |
 |  __init__(self, /, *args, **kwargs)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __new__(*args, **kwargs) from builtins.type
 |      Create and return a new object.  See help(type) for accurate signature
 |
 |  ----------------------------------------------------------------------
 |  Methods inherited from builtins.BaseException:
 |
 |  __delattr__(self, name, /)
 |      Implement delattr(self, name).
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __reduce__(...)
 |      helper for pickle
 |
 |  __repr__(self, /)
 |      Return repr(self).
 |
     |  __setattr__(self, name, value, /)
     |      Implement setattr(self, name, value).
     |
     |  __setstate__(...)
     |
     |  __str__(self, /)
     |      Return str(self).
     |
     |  with_traceback(...)
     |      Exception.with_traceback(tb) --
     |      set self.__traceback__ to tb and return self.
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors inherited from builtins.BaseException:
     |
     |  __cause__
     |      exception cause
     |
     |  __context__
     |      exception context
     |
     |  __dict__
     |
     |  __suppress_context__
     |
     |  __traceback__
     |
     |  args

FUNCTIONS
    is_zipfile(filename)
        Quickly see if a file is a ZIP file by checking the magic number.

        The filename argument may be a file or file-like object too.

DATA
    ZIP_BZIP2 = 12
    ZIP_DEFLATED = 8
    ZIP_LZMA = 14
    ZIP_STORED = 0
    __all__ = ['BadZipFile', 'BadZipfile', 'error', 'ZIP_STORED', 'ZIP_DEF...

FILE
    c:\anaconda3\lib\zipfile.py
```

Known Bug for Python 3.2+ : http://bugs.python.org/issue9170
Source : http://stackoverflow.com/questions/7483138/python-the-zipfile-module-doesnt-seem-to-work-with-passwords
“The feature request is now WONTFIXed. It is unlikely Python will support this in the future. – Kevin Nov 6 ’15 at 23:34”

1.1.1 Python 3.5 WorkAround:

```
In [61]: import zipfile

         zippy = zipfile.ZipFile('evil.zip', 'r')

         try:
             pwdString = "secret".encode('UTF-8')
             extAdam = zippy.extract(member='evil/note_to_adam.txt', pwd=pwdString)

             count = 0
             for i in extAdam:
                 try:
                     print(zippy.filelist[count])
                     count += 1
                 except:
                     pass


         except Exception as e:
             print(str(e))
             pass

         print("=======================================================")
         print("")

         f = open(extAdam, 'r')
         for line in f.readlines():
             print(line)
         f.close()
```

```
<ZipInfo filename='evil/evil.jpg' compress_type=deflate filemode='-rw-r--r--' file_
<ZipInfo filename='evil/note_to_adam.txt' compress_type=deflate filemode='-rw-r--r-
=======================================================

Sorry, you are too late - she ate the apple.

---------

[Image downloaded from http://farm3.staticflickr.com/2422/4424308439_7bd9e833d3_z.j
```

2 Dictionary attack utilized against the Zip File

$[+] Password : secret$

```
In [52]: dtFile = open('HW2dictionary.txt','r')

         for line in dtFile.readlines():
             pwd = line.strip('\n')

             try:
                 pwdString2 = pwd.encode('UTF-8')
                 zippy.extract(member='evil/note_to_adam.txt', pwd=pwdString2)
                 print('[+] Password : ', pwd)
             except Exception as e:

                 print('[-]', str(e).split(',')[0],')', pwd)
                 pass

         dtFile.close()
```

```
[-] ('Bad password for file' ) apple
[-] ('Bad password for file' ) orange
[-] ('Bad password for file' ) egg
[-] ('Bad password for file' ) lemon
[-] ('Bad password for file' ) grapes
[+] Password :  secret
[-] ('Bad password for file' ) strawberry
[-] ('Bad password for file' ) password
```

# Exercise 3 : Port Scanner

```
In [77]: import optparse
         import socket as s

         def cnxnScan(tgtHost, tgtPort):
             """ This tests for a Connection"""

             try:
                 cnxnSoc = s.socket(s.AF_INET, s.SOCK_STREAM)
                 cnxnSoc.connect((tgtHost, tgtPort))

                 print('[+] %d/tcp OPEN' % tgtPort)
                 cnxnSoc.close()

             except Exception as e:
                 print('[=] %d/tcp CLOSED' % tgtPort, str(e))
```

```
def prtScan(tgtHost, tgtPorts):
    """ This Scans thru the Ports"""

    try:
        tarIP = s.gethostbyname(tgtHost)

    except Exception as e:
        print(str(e), '\n[=] Unknown Host %s' % tgtHost)

    try:
        tgtName = s.gethostbyaddr(tarIP)
        print('\n[+] Scan Results for : ', tgtName[0])

    except:
        print('\n[+] Scan Results for : ', tarIP)

    s.setdefaulttimeout(5)

    for i in tgtPorts:
        print("Scanning port : ", i)
        cnxnScan(tgtHost, int(i))



hosts = ['www.google.com','www.facebook.com','www.linkedin.com','www.yahoo
        'www.youtube.com' ,'199.181.132.249', '205.251.242.54']

ports = [21,22,80, 443]

for host in hosts:
    print('\n', host)
    prtScan(host, ports)

print("\n#########    FIN   ##########")
```

```
 www.google.com

[+] Scan Results for :  tl-in-f104.1e100.net
Scanning port :  21
[=] 21/tcp CLOSED [WinError 10061] No connection could be made because the target m
Scanning port :  22
[=] 22/tcp CLOSED timed out
Scanning port :  80
[=] 80/tcp CLOSED [WinError 10061] No connection could be made because the target m
```

```
Scanning port :  443
[=] 443/tcp CLOSED [WinError 10061] No connection could be made because the target

 www.facebook.com

[+] Scan Results for :  edge-star-mini-shv-01-dft4.facebook.com
Scanning port :  21
[=] 21/tcp CLOSED [WinError 10061] No connection could be made because the target m
Scanning port :  22
[=] 22/tcp CLOSED timed out
Scanning port :  80
[=] 80/tcp CLOSED [WinError 10061] No connection could be made because the target m
Scanning port :  443
[=] 443/tcp CLOSED [WinError 10061] No connection could be made because the target

 www.linkedin.com

[+] Scan Results for :  108-174-10-10.fwd.linkedin.com
Scanning port :  21
[=] 21/tcp CLOSED [WinError 10061] No connection could be made because the target m
Scanning port :  22
[=] 22/tcp CLOSED timed out
Scanning port :  80
[=] 80/tcp CLOSED [WinError 10061] No connection could be made because the target m
Scanning port :  443
[=] 443/tcp CLOSED [WinError 10061] No connection could be made because the target

 www.yahoo.com

[+] Scan Results for :  ir1.fp.vip.ir2.yahoo.com
Scanning port :  21
[=] 21/tcp CLOSED [WinError 10061] No connection could be made because the target m
Scanning port :  22
[=] 22/tcp CLOSED timed out
Scanning port :  80
[=] 80/tcp CLOSED [WinError 10061] No connection could be made because the target m
Scanning port :  443
[=] 443/tcp CLOSED [WinError 10061] No connection could be made because the target

 www.youtube.com

[+] Scan Results for :  sa-in-f91.1e100.net
Scanning port :  21
[=] 21/tcp CLOSED [WinError 10061] No connection could be made because the target m
Scanning port :  22
[=] 22/tcp CLOSED timed out
Scanning port :  80
[=] 80/tcp CLOSED [WinError 10061] No connection could be made because the target m
```

```
Scanning port :  443
[=] 443/tcp CLOSED [WinError 10061] No connection could be made because the target

 199.181.132.249

[+] Scan Results for :  199.181.132.249
Scanning port :  21
[=] 21/tcp CLOSED [WinError 10061] No connection could be made because the target m
Scanning port :  22
[=] 22/tcp CLOSED [WinError 10061] No connection could be made because the target m
Scanning port :  80
[=] 80/tcp CLOSED [WinError 10061] No connection could be made because the target m
Scanning port :  443
[=] 443/tcp CLOSED [WinError 10061] No connection could be made because the target

 205.251.242.54

[+] Scan Results for :  205.251.242.54
Scanning port :  21
[=] 21/tcp CLOSED timed out
Scanning port :  22
[=] 22/tcp CLOSED [WinError 10061] No connection could be made because the target m
Scanning port :  80
[=] 80/tcp CLOSED [WinError 10061] No connection could be made because the target m
Scanning port :  443
[=] 443/tcp CLOSED [WinError 10061] No connection could be made because the target
#########    FIN   ##########
```

3 NMAP of OS and ports of LocalHost

Starting Nmap 7.40 (https://nmap.org) at 2017-02-26 14:30 Central Standard Time
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000060s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 90 closed ports

PORT	STATE	SERVICE	VERSION
135/tcp	open	msrpc	Microsoft Windows RPC
445/tcp	open	microsoft-ds	Microsoft Windows 7 - 10 microsoft-ds (workgroup: HALAMERICA)
3306/tcp	open	mysql	MySQL 5.7.15-log

PORT	STATE	SERVICE	VERSION
3389/tcp	open	ms-wbt-server	Microsoft Terminal Service
4899/tcp	open	radmin	Famatech Radmin 3.X (Radmin Authentication)
8888/tcp	open	http	Tornado httpd 4.4.1
49152/tcp	open	unknown	
49153/tcp	open	unknown	
49154/tcp	open	unknown	
49155/tcp	open	unknown	

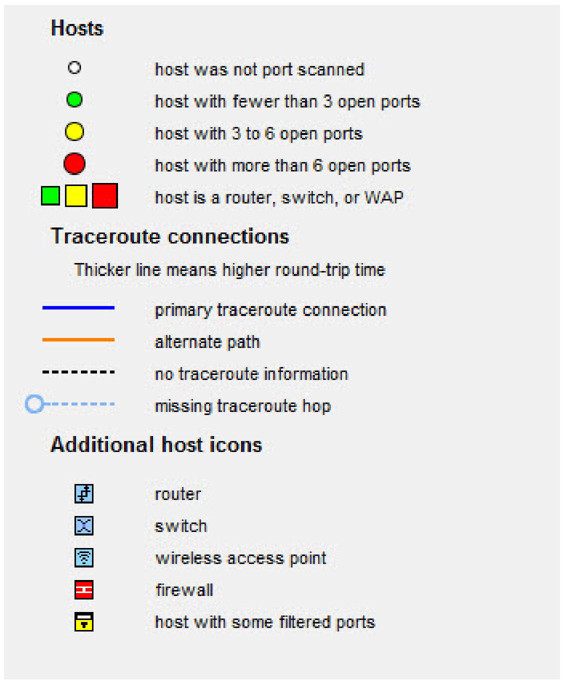
Device type: general purpose
Running: Microsoft Windows 7|8.1
OS CPE: cpe:/o:microsoft:windows_7 cpe:/o:microsoft:windows_8.1:r1
OS details: Microsoft Windows 7 or 8.1 R1
Network Distance: 0 hops
Service Info: Host: ENAUS00073002; OS: Windows; CPE: cpe:/o:microsoft:windows
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 38.73 seconds

4 NMAP analysis of host FACEBOOK behind the HALLIBURTON network

This was interesting

```
In [83]: from IPython.display import Image
         Image(filename='LegendNmap.jpg')
```

```
Out[83]:
```



Hosts

○	host was not port scanned
🟢	host with fewer than 3 open ports
🟡	host with 3 to 6 open ports
🔴	host with more than 6 open ports
🟩🟨🟥	host is a router, switch, or WAP

Traceroute connections

Thicker line means higher round-trip time

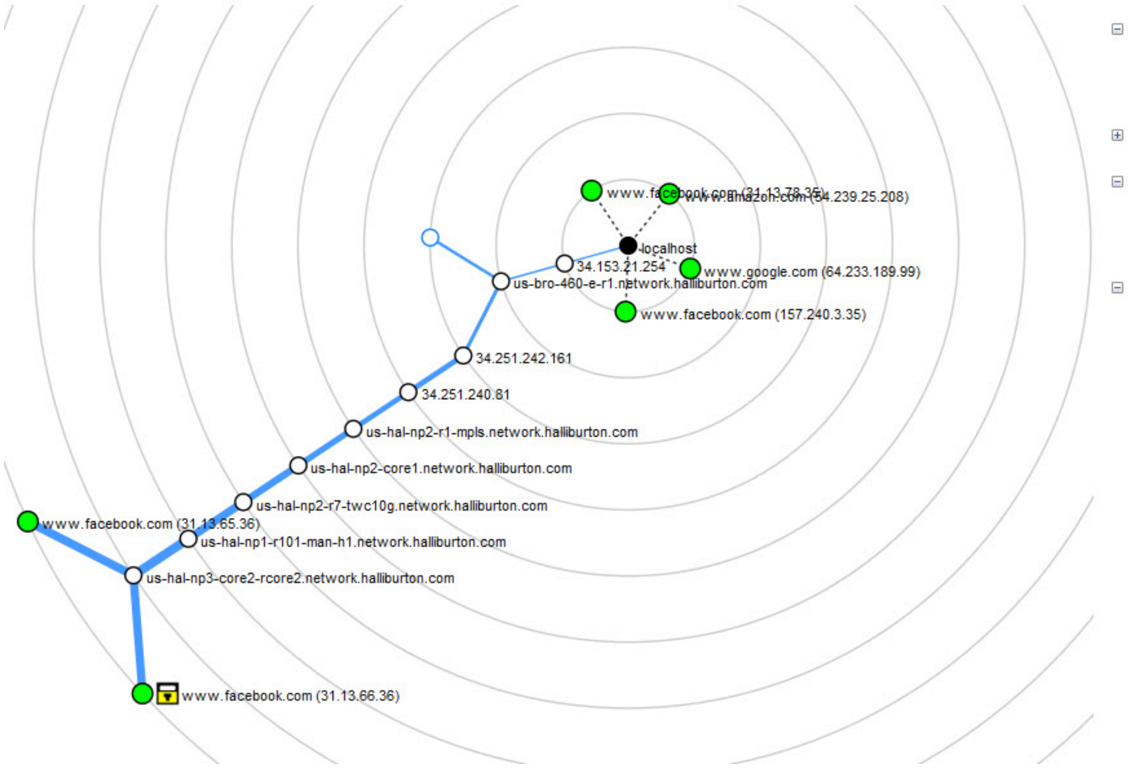
──	primary traceroute connection
──	alternate path
--------	no traceroute information
○-------	missing traceroute hop

Additional host icons

▣	router
▣	switch
▣	wireless access point
▣	firewall
▣	host with some filtered ports

```
In [82]: Image(filename='NMAP_scan_map.jpg')
```

```
Out[82]:
```


www.facebook.com (31.13.78.35)
www.amazon.com (54.239.25.208)
localhost
34.153.21.254
www.google.com (64.233.189.99)
us-bro-460-e-r1.network.halliburton.com
www.facebook.com (157.240.3.35)
34.251.242.161
34.251.240.81
us-hal-np2-r1-mpls.network.halliburton.com
us-hal-np2-core1.network.halliburton.com
us-hal-np2-r7-twc10g.network.halliburton.com
www.facebook.com (31.13.65.36)
us-hal-np1-r101-man-h1.network.halliburton.com
us-hal-np3-core2-rcore2.network.halliburton.com
www.facebook.com (31.13.66.36)

```
Starting Nmap 7.40 ( https://nmap.org ) at 2017-02-26 12:55 Central Standard Time
NSE: Loaded 143 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 12:55
Completed NSE at 12:55, 0.00s elapsed
Initiating NSE at 12:55
Completed NSE at 12:55, 0.00s elapsed
Initiating Ping Scan at 12:55
Scanning www.facebook.com (31.13.66.36) [4 ports]
Completed Ping Scan at 12:55, 0.81s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 12:55
Completed Parallel DNS resolution of 1 host. at 12:55, 0.03s elapsed
Initiating SYN Stealth Scan at 12:55
Scanning www.facebook.com (31.13.66.36) [65535 ports]
SYN Stealth Scan Timing: About 19.11% done; ETC: 12:57 (0:02:11 remaining)
SYN Stealth Scan Timing: About 37.62% done; ETC: 12:57 (0:01:41 remaining)
SYN Stealth Scan Timing: About 56.24% done; ETC: 12:57 (0:01:11 remaining)
SYN Stealth Scan Timing: About 77.09% done; ETC: 12:57 (0:00:36 remaining)
Completed SYN Stealth Scan at 12:57, 157.51s elapsed (65535 total ports)
Initiating Service scan at 12:57
Initiating OS detection (try #1) against www.facebook.com (31.13.66.36)
Initiating Traceroute at 12:57
Completed Traceroute at 12:57, 3.03s elapsed
Initiating Parallel DNS resolution of 10 hosts. at 12:57
Completed Parallel DNS resolution of 10 hosts. at 12:57, 0.16s elapsed
NSE: Script scanning 31.13.66.36.
Initiating NSE at 12:57
Completed NSE at 12:57, 0.02s elapsed
Initiating NSE at 12:57
Completed NSE at 12:57, 0.00s elapsed
Nmap scan report for www.facebook.com (31.13.66.36)
Host is up (0.021s latency).
Other addresses for www.facebook.com (not scanned): 2a03:2880:f113:83:face:b00c:0:25de
rDNS record for 31.13.66.36: edge-star-mini-shv-02-dft4.facebook.com
Not shown: 65532 closed ports
PORT STATE SERVICE VERSION
22/tcp filtered ssh
123/tcp filtered ntp
3389/tcp filtered ms-wbt-server
Device type: WAP|load balancer|firewall|webcam|router
Running: Asus embedded, Cisco embedded, Cisco PIX OS 8.X, D-Link embedded, Linksys
embedded, Palo Alto embedded, Planet embedded, Vodafone embedded
OS CPE: cpe:/h:asus:rt-53n cpe:/o:cisco:pix_os:8.0 cpe:/h:dlink:dcs-
6620g cpe:/h:linksys:befsr41 cpe:/h:paloalto:pa-500 cpe:/h:planet:wap-1950
cpe:/h:vodafone:easybox_802
Too many fingerprints match this host to give specific OS details
```

```
Network Distance: 12 hops
TRACEROUTE (using port 256/tcp)
HOP RTT ADDRESS
1 2.00 ms 34.153.21.254
2 3.00 ms us-bro-460-e-r1.network.halliburton.com (34.38.147.254)
3 . . .
4 7.00 ms us-bro-460-e-r1.network.halliburton.com (34.38.147.254)
5 10.00 ms 34.251.242.161
6 16.00 ms 34.251.240.81
7 16.00 ms us-hal-np2-r1-mpls.network.halliburton.com (34.251.240.82)
8 19.00 ms us-hal-np2-core1.network.halliburton.com (34.36.248.1)
9 23.00 ms us-hal-np2-r7-twc10g.network.halliburton.com (34.36.248.98)
10 26.00 ms us-hal-np1-r101-man-h1.network.halliburton.com (10.250.0.25)
11 29.00 ms us-hal-np3-core2-rcore2.network.halliburton.com (10.192.2.21)
12 26.00 ms edge-star-mini-shv-02-dft4.facebook.com (31.13.66.36)
NSE: Script Post-scanning.
Initiating NSE at 12:57
Completed NSE at 12:57, 0.00s elapsed
Initiating NSE at 12:57
Completed NSE at 12:57, 0.00s elapsed
Read data files from: C:Files (x86)
OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 168.19 seconds

    Raw packets sent: 65616 (2.889MB) | Rcvd: 65558 (2.623MB)
```