FA2022 Week 12

Forensics

Minh
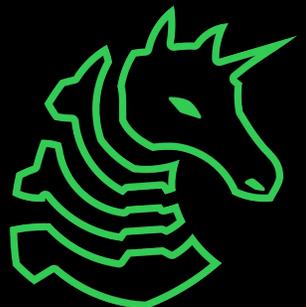
Announcements

- Sunshine CTF 2022
    - Starts this Saturday at 9AM
    - Completely virtual because of fall break
    - We are trying to get top 50 in the world on CTFtime!

- Fall break
    - No meeting this Sunday or next Thursday and Sunday
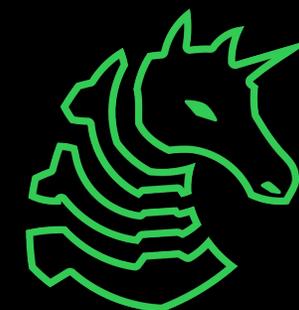
ctf.sigpwny.com

sigpwny{guess_god}
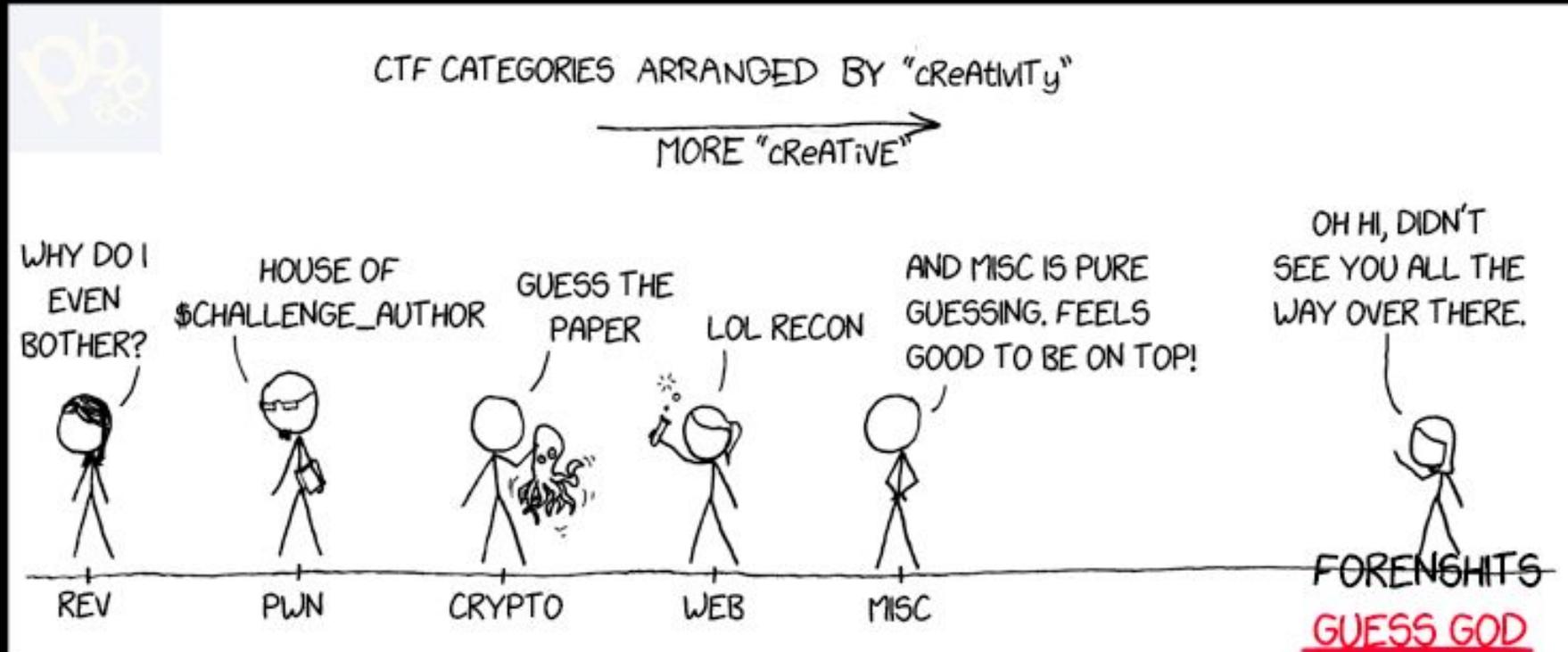
What is Digital Forensics?

- File Forensics
  - Requires understanding of a file format
  - Data can be hidden within the format (steganography)
- Filesystem Forensics
  - We conduct forensics to determine what happened on a system in a holistic manner, usually after some sort of incident
- Memory Forensics
  - Different from traditional filesystem forensics: we are given the contents of RAM in the form of a dump file
  - Given a memory dump, what information can we extract from it?

File Forensics

file

Is a file really what is says it is? Can a file be opened in a different way? Don't always trust the extension!

`file` uses "magic bytes" or a file signature to determine the format of a file (e.g. PNG files always start with `89 50 4E 47`)

```
$ file unknown.txt
unknown.txt: JPEG image data, JFIF standard 1.01, aspect ratio, density 1x1,
 segment length 16, progressive, precision 8, 400x400, components 3
```

Usage:

`file <FILE>`
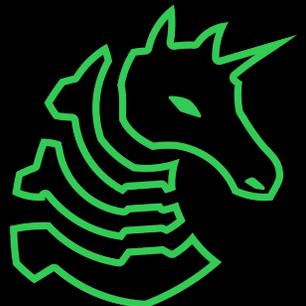
binwalk

Finds files formats appended in a file (like `file` but recursive). For example, some binaries include their assets!

```
$ binwalk cmd.exe

DECIMAL          HEXADECIMAL        DESCRIPTION
--------------------------------------------------------------------------------
0                0x0                Microsoft executable, portable (PE)
235448           0x397B8            Copyright string: "Copyright (c) Microsoft Corpora
253789           0x3DF5D            mcrypt 2.5 encrypted data, algorithm: "o_exit", ke
283472           0x45350            XML document, version: "1.0"
283534           0x4538E            Copyright string: "Copyright (c) Microsoft Corpora
294568           0x47EA8            PNG image, 256 x 256, 8-bit/color RGBA, non-interl
294609           0x47ED1            Zlib compressed data, default compression
```

Usage:

`binwalk <FILE>` - View a list of contained file formats

`binwalk -e <FILE>` - Extract each file format from the file

strings

Lists plaintext strings that exist in a file, useful for binary files

Usage:

`strings <FILE>` - print all strings of length 4 or greater

`strings -n 16 <FILE>` - print all strings of length 16 or greater

```
$ strings -n 16 cmd.exe
!This program cannot be run in DOS mode.
SetThreadUILanguage
Unknown exception
bad array new length
api-ms-win-core-winrt-l1-1-0.dll
ext-ms-win-branding-winbrand-l1-1-0.dll
ext-ms-win-cmd-util-l1-1-0.dll
ext-ms-win-appmodel-shellexecute-l1-1-0.dll
onecore\internal\sdk\inc\wil\opensource\wil\resource.h
WilFailureNotifyWatchers
RtlRegisterFeatureConfigurationChangeNotification
RtlUnregisterFeatureConfigurationChangeNotification
RtlNotifyFeatureUsage
NtQueryWnfStateData
NtUpdateWnfStateData
onecore\internal\sdk\inc\wil/Staging.h
CMD Internal Error %s
Null environment
APerformUnaryOperation: '%c'
APerformArithmeticOperation: '%c'
IsDebuggerPresent
SetConsoleInputExeNameW
RaiseFailFastException
RtlNtStatusToDosErrorNoTeb
RtlDllShutdownInProgress
RtlDisownModuleHeapAllocation
NtQueryInformationProcess
Copyright (c) Microsoft Corporation. All rights reserved.
onecore\base\cmd\StartShellExecServiceProvider.h
onecore\base\cmd\maxpathawarestring.cpp
```

xxd

Prints a hexdump of a file

Good to look for recognizable hex patterns or perform advanced hex manipulation

Usage:

```
xxd <FILE>
```

grep

Search for text matches within a file or recursively in files!

```
grep "text you want to find" <FILE>

grep -R "text you want to find" <DIRECTORY>
```

Combine with other utilities!

```
cat <FILE> | grep "text"

strings <FILE> | grep "text"
```
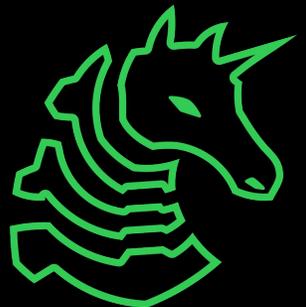
Steganography

It sucks

Steganography

- Hide data in other data

- Inherently guessy during CTFs
  - Try lots of ideas
  - Waste lots of time
  - Use statistical approaches if applicable
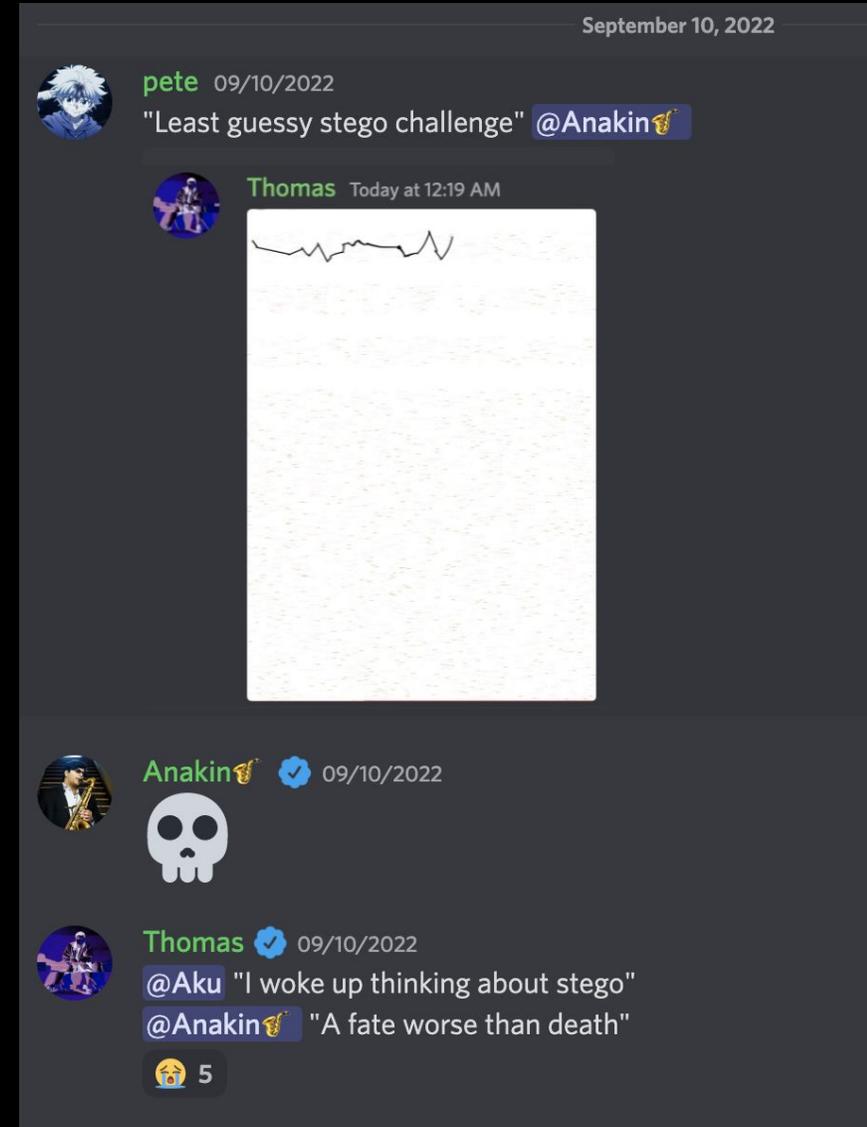
Image Stego - LSB

- LSB (least significant bit) encoding

- Not really useful in the real world, but CTFs love it

- Take the least significant bit (last bit) of each color byte and concatenate all of them to form a message

- Image is mostly visibly unchanged

Image Stego - LSB

RGB(218,150,149)

R = 11011010
G = 10010110
B = 10010101
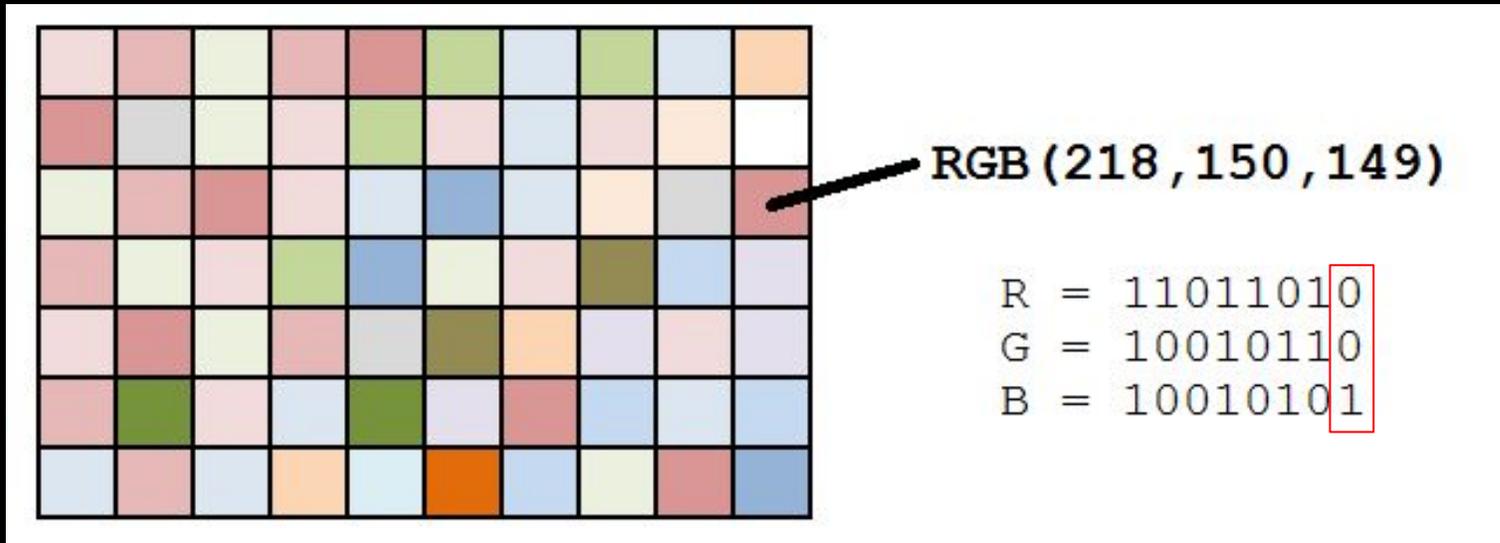
Message = (R & 1) || (G & 1) || (B & 1) = 001...
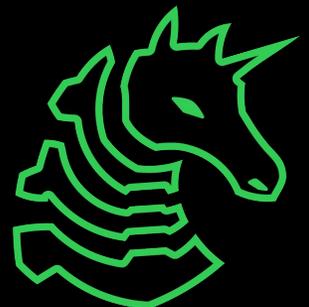
Can you tell the difference?

Original

LSB Encoded

Can you tell the difference?

Original (Red, Bit 0)
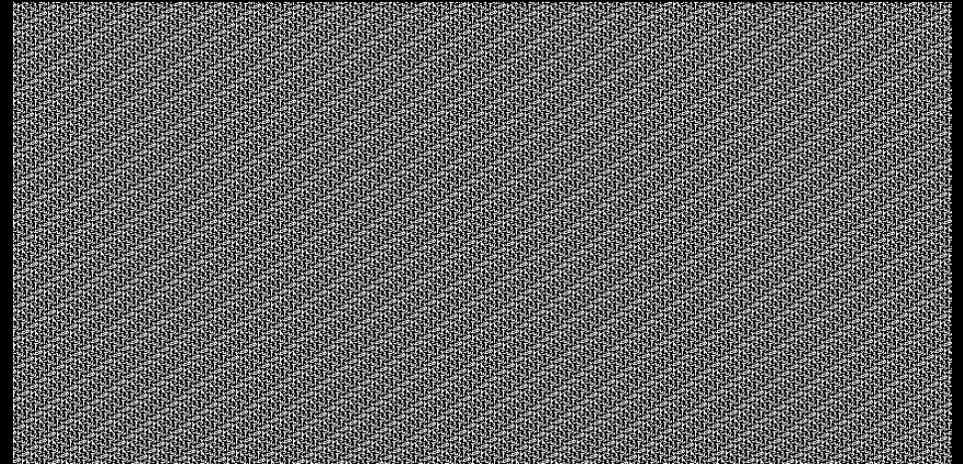
LSB Encoded (Red, Bit 0)

Stegoanalysis tool: https://stegonline.georgeom.net/image
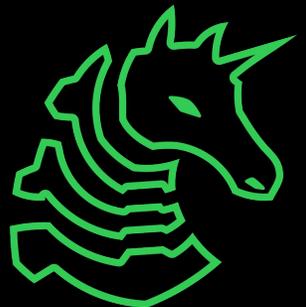
Other Stego

Audio stego

- 90 % of the time it is a spectrogram
- The other 10% is either
  - SSTV
  - Some frequency modulation
  - Some other guessy home-brewed bullshit

- Most audio stego can be solved with tools

Starting Point

- What kind of file is it?
  - Use a command like `file` or `binwalk` to identify what a file is using magic bytes
 - If it's an image/video/audio, it likely contains hidden information in the form of steganography
  - If it's a document, there might be hidden information in the file format (did you know that a .docx file is just a ZIP archive?)
  - If it's plain text ASCII, what does it contain? Are there any patterns?
  - If it's a binary file, what readable strings are there? Are there patterns in the hexadecimal representation?
- What metadata does it have?
  - Images: location, camera model, encoding
  - Documents: username of creator, directory where saved
  - Everything else
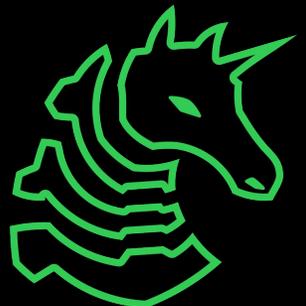
Filesystem Forensics

Log analysis, filesystem/disk

Logs

- Logs provide valuable information about what happened on the system
- You can construct a list of events that occur to determine entry point of an attack and what an attacker did

Linux:

- /var/log/http/access.log
- /var/log/syslog
- other various service logs

Windows:

- Event Viewer

Starting Point

What we are given:

- An archive/zip of the full/partial filesystem
  - We can inspect logs, user files, command history, etc.
- A disk image file (.dd, .iso, .vhd)
  - Follows a filesystem format (e.g. NTFS, ext4)
  - Deleted files can be present and ignored by the filesystem format
  - We can mount it to a live system to examine the contents
- Live access to a system
  - Extremely difficult to perform forensics on a live system while maintaining integrity of evidence since it is volatile
  - Obtain a disk image after attempting to preserve live system evidence
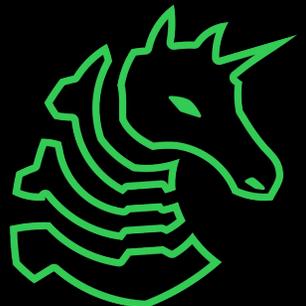
Memory Forensics

Memory Forensics

Instead of being given a traditional filesystem, you are given the contents of memory or RAM in the form of a dump file

- Dump files are usually created when a program crashes or your OS crashes for debugging purposes
- They can contain sensitive information that was located in memory at the time of crashing
    - Passwords in your password manager
    - Clipboard content at time of crash
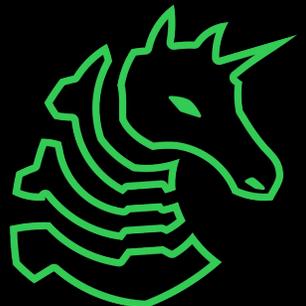
Volatility

- Analyze memory dump files
- There are two versions of Volatility:
  - Volatility 2.6
    - Is older, but works great when it works
    - Is limited to older OS versions (e.g. doesn't have latest Windows 10 symbols)
    - Intended to be used standalone
  - Volatility 3
    - Complete framework rewrite, under development, and missing some features
    - Is way faster and has better OS version profiling and support
    - Intended to be used as a library with standalone support
    - Annoyingly enough, also has versions such as 2.4.1

Setting up Volatility

git clone https://github.com/volatilityfoundation/volatility3.git

cd volatility3

pip install -r requirements.txt

python3 vol.py -f <FILE> windows.pslist

```
$ python3 vol.py -f MEMORY.DMP windows.pslist
Volatility 3 Framework 2.4.1
Progress:  100.00          PDB scanning finished
PID     PPID    ImageFileName    Offset(V)        Threads Handles SessionId    Wow64    CreateTime                   ExitTime      File output

4       0       System  0xb50a3627b040  137      -       N/A     False   2022-11-17 23:06:49.000000        N/A      Disabled88        4
     Disabled
260     4       smss.exe         0xb50a39093040  2       -       N/A     False   2022-11-17 23:06:49.000000        N/A      Disabled
388     376     csrss.exe        0xb50a36fb8140  12      -       0       False   2022-11-17 23:06:54.000000        N/A      Disabled
460     376     wininit.exe      0xb50a39790080  1       -       0       False   2022-11-17 23:06:54.000000        N/A      Disabled
468     452     csrss.exe        0xb50a397f7140  14      -       1       False   2022-11-17 23:06:54.000000        N/A      Disabled
556     452     winlogon.exe     0xb50a39e4b080  6       -       1       False   2022-11-17 23:06:55.000000        N/A      Disabled
596     460     services.exe     0xb50a39e4a140  8       -       0       False   2022-11-17 23:06:55.000000        N/A      Disabled
612     460     lsass.exe        0xb50a39e5d0c0  11      -       0       False   2022-11-17 23:06:55.000000        N/A      Disabled
708     596     svchost.exe      0xb50a39ecb280  1       -       0       False   2022-11-17 23:06:56.000000        N/A      Disabled
732     556     fontdrvhost.ex   0xb50a39f07180  5       -       1       False   2022-11-17 23:06:56.000000        N/A      Disabled
740     460     fontdrvhost.ex   0xb50a39f09180  5       -       0       False   2022-11-17 23:06:56.000000        N/A      Disabled
748     596     svchost.exe      0xb50a39f0b280  24      -       0       False   2022-11-17 23:06:56.000000        N/A      Disabled
852     596     svchost.exe      0xb50a39fc4300  15      -       0       False   2022-11-17 23:06:56.000000        N/A      Disabled
```

Or... just use basic commands

A lot of memory is simply stored as strings!

```
strings <FILE> | grep "text"
```

```
strings --encoding=l <FILE> | grep "text"
```

Next Meetings

2022-11-19 - **This Saturday**

- Sunshine CTF 2022
- Join us virtually to compete in UCF's CTF!

2022-12-01 - **Thursday After Break**

- PWN II with Kevin
- Learn how to exploit format strings for PWN

2022-12-04 - **Sunday After Break**

- Graduate talk with Jaron Mink
- Security Unleashed Part 4