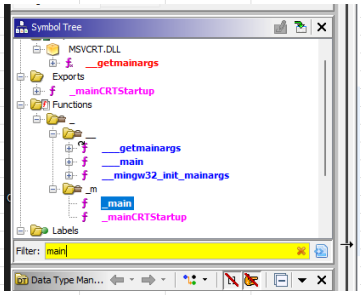
Autor:	RedXen
Tytuł:	Learning C CrackMe
Język:	C/C++
Platforma:	Windows
Trudność:	1.9
Jakość:	4.0
Arch:	x86
Link:	https://crackmes.one/crackme/62c5da5d33c5d44a934e9684
Opis:	A CrackMe written in C, as I'm learning it. Let me know how you find it! (It's easy).
Plik:	https://crackmes.one/static/crackme/62c5da5d33c5d44a934e9684.zip

Krok	Rozwiązanie	Zrzuty Ekranu
1	Sprawdzenie jak działa plik .exe Działanie polega na przyjmowaniu hasła. Zwraca czy wartość jest poprawna czy nie.	
2	Odnalezienie głównej funkcji w programie za pomocą Symbol Tree.	

```
C:\Users\judasz\Desktop\projekt so> C:\Users\
What's your username? (MAX 10 CHAR): username
What's the password? haslo
Incorrect! Try again.
C:\Users\judasz\Desktop\projekt so>
```




```
Decompile: _main - (learningcrackme.exe)

int __cdecl _main(int _Argc,char **_Argv,char **_Env)

{
  __main();
  _o_d0924b2757e74284f8020705e3c3e036();
  return 0;
}
```

```
Decompile: main2 - (learningcrackme.exe)

void __fastcall main2(void)

{
  _printf("What\'s your username? (MAX 10 CHAR): ");
  _scanf("%10s",&_o_f65bce5421998a005cdf766a147a9a20);
  _o_2be86c2e902c113b7a94b2c1b18bfe9c();
  return;
}
```

Tak naprawdę w funkcji main nic się nie dzieje oprócz tego że jest wywoływana jakaś dodatkowa funkcja, zmieniłem jej nazwę na **main2()**.

W **main2()** użytkownik podaje hasło do jakiejś pamięci w programie.
Zmieniłem te nazwę na **username.**

Kolejna funkcja jest wywoływana nazwę ją **main3().**

```
Decompile: main3 - (learningcrackme.exe)

void __fastcall main3(void)

{
  size_t sVar1;
  size_t sVar2;
  uint local_10;

  local_10 = 0;
  while( true ) {
    sVar1 = _strlen(&username);
    sVar2 = _strlen(&username);
    if (sVar2 <= local_10 || sVar1 <= local_10) break;
    (&_o_7f8edf6d5330c5a7ef8ae28d51b6c753)[local_10] = (char)local_10 + (&username)[local_10] + 'e';
    local_10 = local_10 + 1;
  }
  _o_2dc2bfa9c8b92a2af3a058b8f16fb387();
  return;
}
```

W funkcji **main3()** na bazie naszego podanego "username" jest tworzona jakaś nowa zmienna.
Nazwę ją **variable_base_on_username.**
Można zauważyć że jest ona dokładnie tej samej długości co nasz podany **username.**
Dodatkowo widzimy schemat jej budowy.
local_10 to **iterator**, i przechodząc po każdym wyrazie tablicy,
dodawany jest **char(iterator)+username[iterator] + 'e'**

Zaprezentuje działanie na "tekst", **nowy_tekst** zostanie stworzony w taki sposób:

tekst

t -> char(0) + 't' + 'e'
e -> char(1) + 'e' + 'e'
k -> char(2) + 'k' + 'e'
s -> char(3) + 's' + 'e'
t -> char(4) + 't' + 'e'

```
cout << char(char(0) + 't' + 'e' );
cout << char(char(1) + 'e' + 'e' );
cout << char(char(2) + 'k' + 'e' );
cout << char(char(3) + 's' + 'e' );
cout << char(char(4) + 't' + 'e' );
```

```
Konso
╝╦Ď╠T
C:\Users
Aby auto
znie zam
Naciśnij
```

Napisałem prosty program, żeby zobaczyć jak bedzie wyglądać **nowy_tekst**
Jego wartość widać w konsoli, dla **"tekst"** jest ona równa **"╝╦Ď╠T"**

Po stworzeniu **nowy_tekst** kolejna wywoływana funkcja nazwę ją **main4().**

Przyjmuje ona hasło i zapisuje go do zmiennej **local_16**, nazwe ją **password.**

Następnie jest wywoływana funkcja **strcmp(password,variable_base_on_username)**

Jeżeli ta funkcja zwórci **1** to printf pokaże nam że dobrze rozwiązaliśmy.

Teraz trzeba zrozumieć kiedy **strcmp** zwraca **1**

```
cout << strcmp("tekst","teksty")<< endl;   -1
cout << strcmp("tekst", "tekst") << endl;  0
cout << strcmp("teksty", "tekst") << endl; 1
cout << strcmp("a", "z") << endl;          -1
cout << strcmp("z", "a") << endl;          1
cout << strcmp("c", "!") << endl;          1
```

```
Decompile: main4 - (learningcrackme.exe)

void __fastcall main4(void)

{
  int iVar1;
  char local_16 [18];

  _printf("What\'s the password? ");
  _scanf("%s",local_16);
  iVar1 = _strcmp(local_16,&variable_base_on_username);
  if (iVar1 == 1) {
    _printf("Nice! Submit your solution on crackmes.one!");
  }
  else {
    _printf("Incorrect! Try again.");
  }
  return;
}
```

Po przeczytaniu dokumentacji i własnym testowaniu funkcji można wysnuć parę wniosków.
Funcka zwórci **1** gdy pierwszy arguemnt będzie **dłuższy** od drugiego.
Niestety ten przypadek nas nie interesuje gdyż nasze argumenty będą tej samej długości.
Gdy są tej samej długości wartość **1** otrzymamy gdy argument pierwszy bedzie mial *większą wartość* od drugiego.

```
C:\Users\judasz\Desktop\projekt so> C:\Users\ju
What's your username? (MAX 10 CHAR): tekst
What's the password? ╝╦Ď╠T
Incorrect! Try again.
C:\Users\judasz\Desktop\projekt so>
```

Dlatego nasze hasło stworzone na bazie "tekst" nie zadziała, ponieważ wartości będą takie same i funkcja zwórci **0.**

Czyli możemy zamiast dodawać **'e' = 101,** wartość większą. np **102**

Prowizorycznie napisany kod do generowania takiego hasła, dla username = "aaaaaaaaaa" oraz kod żeby pokazać wartość wygenerowaną przez program.

ä╚╩╦╠╬¤Ď (nasza wartość)

```
for (int i = 0; i < 10; i++) {
    cout << char(char(i) + 'a' + 102);
```

Ää╚╩╦╠╬¤ (wartość stworzona przez program)

```
for (int i = 0; i < 10; i++) {
    cout << char(char(i) + 'a' + 'e');
```

```
}
Konsola debugowania programu Microsoft Visual Studio
```

```
}
Konsola debugowania programu Microsoft Visual Studio
C:\Users\judasz\source\repos\ConsoleApplicatio
```

Testujemy teraz co by zwórciła funkcja strcmp dla tych argumentów w tej kolejności

```
cout << strcmp("ä╚╝╠╣¤d","Ää╚╝╠╣¤") << endl;
/**
cout
Konsola debugowania programu Microsoft Visual Studio
1
```

Jest to **1** więc pora sprawdzić to na programie

```
C:\Users\judasz\Desktop\projekt so>"C:\Users\ju
What's your username? (MAX 10 CHAR): aaaaaaaaaa
What's the password? ä╚╝╠╣¤d
Nice! Submit your solution on crackmes.one!
C:\Users\judasz\Desktop\projekt so>
```

Działa! Podsumuwując trzeba było stworzyć hasło większe od tego wygenerowanego przez program na bazie naszego username.