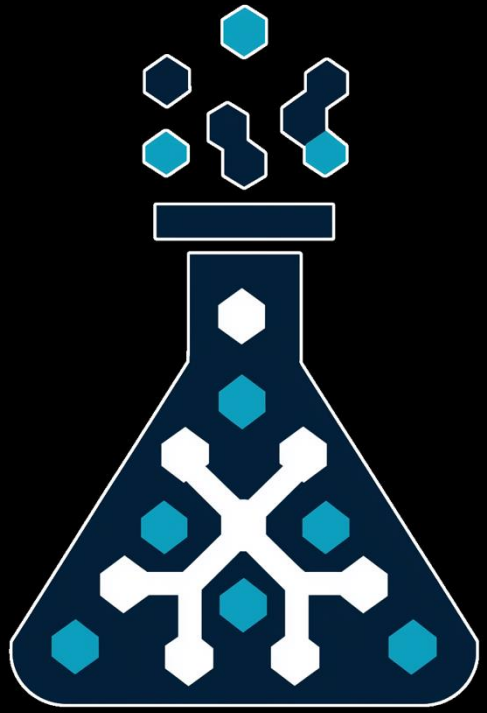


VirLabs

VirLabs

- /me
- Reverse-engineering
- IA x RE
- Demo
- Sous le capot

/me

/me

- Ex MinArm
- Ex Kaspersky
- Meta
- @JusticeRage

/me

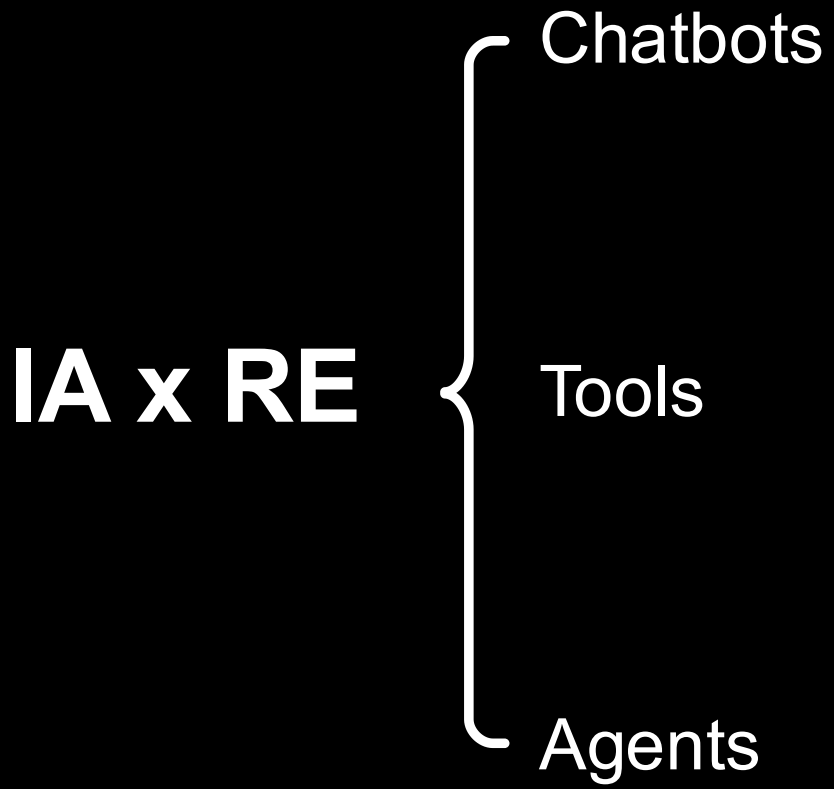
@JusticeRage

Reverse-engineering

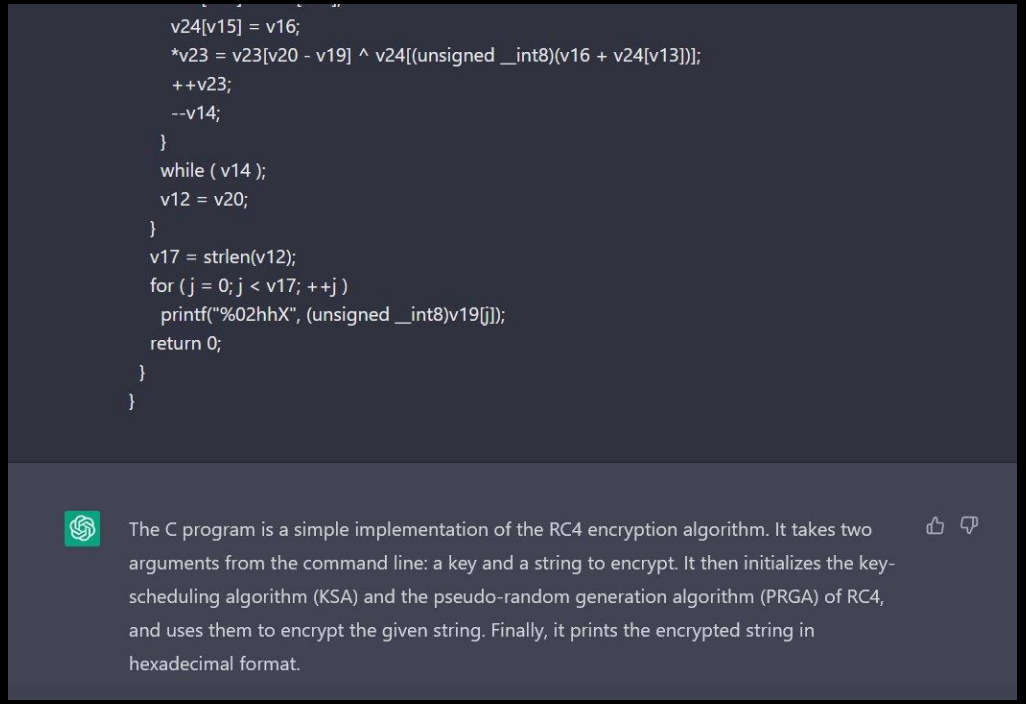
Reverse-engineering

- Long et coûteux
- 📈 Echantillons de malware
  📉 Reversers
- Problème de passage à l'échelle

IA x RE


IA x RE
Chatbots
Tools
Agents

Chatbots



```
      v24[v15] = v16;
      *v23 = v23[v20 - v19] ^ v24[(unsigned __int8)(v16 + v24[v13])];
      ++v23;
      --v14;
    }
    while ( v14 );
    v12 = v20;
  }
  v17 = strlen(v12);
  for ( j = 0; j < v17; ++j )
    printf("%02hhX", (unsigned __int8)v19[j]);
  return 0;
 }
}
```

The C program is a simple implementation of the RC4 encryption algorithm. It takes two arguments from the command line: a key and a string to encrypt. It then initializes the key-scheduling algorithm (KSA) and the pseudo-random generation algorithm (PRGA) of RC4, and uses them to encrypt the given string. Finally, it prints the encrypted string in hexadecimal format.

Tools


```
int __cdecl sub_10001000(int a1, int a2, char *Str)
{
  int result; // eax
  int v4; // esi
  int i; // edx

  result = strlen(Str);
  v4 = 0;
  for ( i = 0; v4 < a2; ++v4 )
  {
    *(_BYTE *)(v4 + a1) ^= Str[i];
    if ( ++i >= result )
      i -= result;
  }
  return result;
}
```

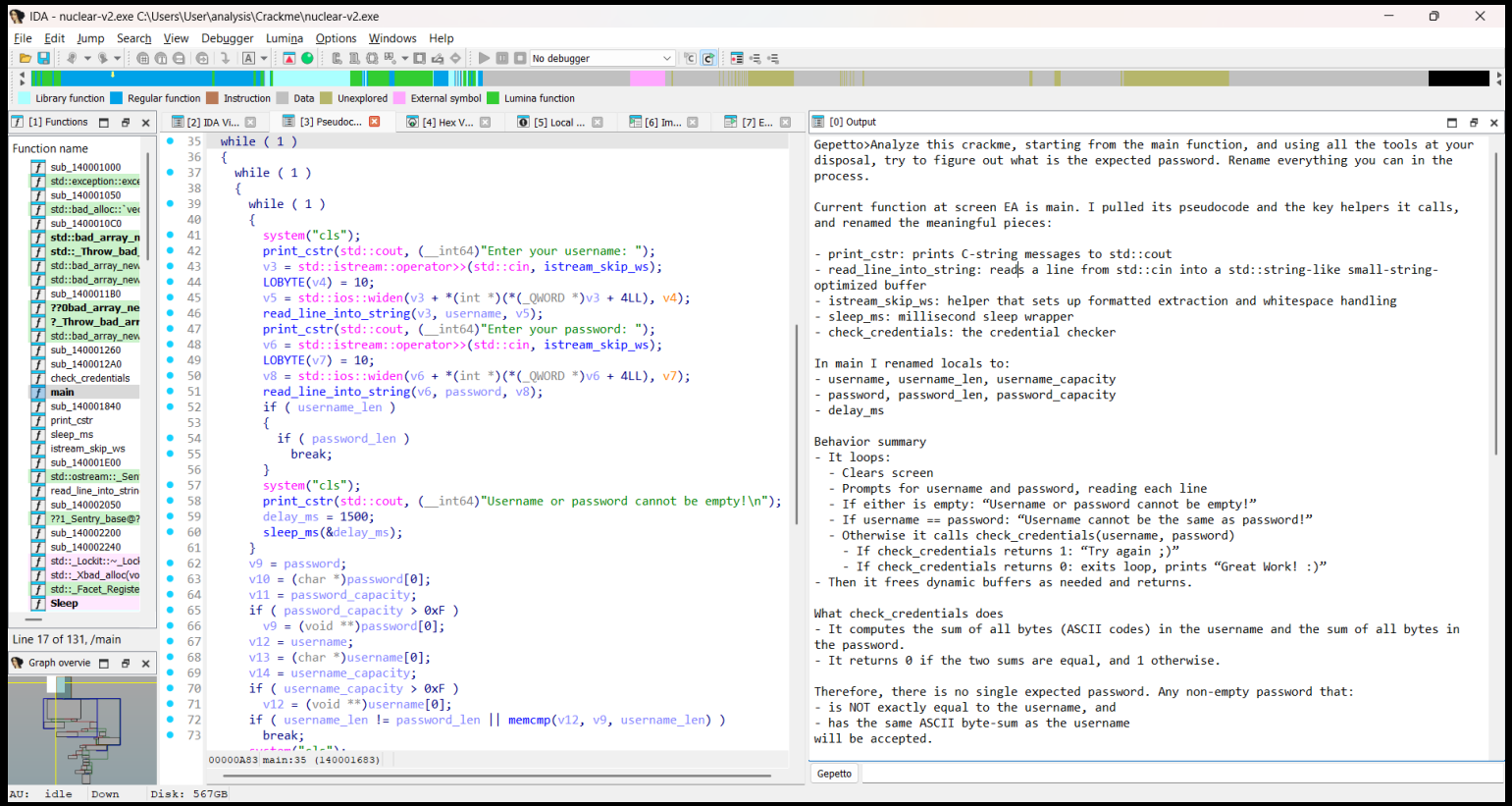
```
//
// This function is performing a bitwise XOR operation on the bytes of a character
// array starting at position "a1" and doing it for "a2" number of bytes. It is
// using the characters in Str to perform the XOR operations. A better name for
// this function could be 'xorCharacterArrayBytes()'.
int __cdecl sub_10001000(int startPos, int numBytes, char *stringToXorWith)
{
  int result; // eax
  int v4; // esi
  int i; // edx

  result = strlen(stringToXorWith);
  v4 = 0;
  for ( i = 0; v4 < numBytes; ++v4 )
  {
    *(_BYTE *)(v4 + startPos) ^= stringToXorWith[i];
    if ( ++i >= result )
      i -= result;
  }
  return result;
}
```

Agents

- 
- Février 2026: création de VirLabs
- Juin 2026: go-to-market

IA x RE


IDA - nuclear-v2.exe C:\Users\User\analysis\Crackme\nuclear-v2.exe
File Edit Jump Search View Debugger Lumina Options Windows Help
No debugger
Library function Regular function Instruction Data Unexplored External symbol Lumina function
[1] Functions
Function name
sub_140001000
std::exception::exce
sub_140001050
std::bad_alloc::`ve
sub_1400010C0
std::bad_array_n
std::_Throw_bad_
std::bad_array_new
std::bad_array_new
sub_140001180
??0bad_array_ne
?_Throw_bad_arr
std::bad_array_new
sub_140001260
sub_1400012A0
check_credentials
main
sub_140001840
print_cstr
sleep_ms
istream_skip_ws
sub_140001E00
std::ostream::_Sen
read_line_into_strin
sub_140002050
??1_Sentry_base@?
sub_140002200
sub_140002240
std::_Lockit::~_Locl
std::_Xbad_alloc(vo
std::_Facet_Registe
Sleep
Line 17 of 131, /main
Graph overvie
[2] IDA Vi...
[3] Pseudoc...
[4] Hex V...
[5] Local ...
[6] Im...
[7] E...
while (1)
{
while (1)
{
while (1)
{
system("cls");
print_cstr(std::cout, (__int64)"Enter your username: ");
v3 = std::istream::operator>>(std::cin, istream_skip_ws);
LOBYTE(v4) = 10;
v5 = std::ios::widen(v3 + *(int *)(*(_QWORD *)v3 + 4LL), v4);
read_line_into_string(v3, username, v5);
print_cstr(std::cout, (__int64)"Enter your password: ");
v6 = std::istream::operator>>(std::cin, istream_skip_ws);
LOBYTE(v7) = 10;
v8 = std::ios::widen(v6 + *(int *)(*(_QWORD *)v6 + 4LL), v7);
read_line_into_string(v6, password, v8);
if (username_len)
{
if (password_len)
break;
}
system("cls");
print_cstr(std::cout, (__int64)"Username or password cannot be empty!\n");
delay_ms = 1500;
sleep_ms(&delay_ms);
}
v9 = password;
v10 = (char *)password[0];
v11 = password_capacity;
if (password_capacity > 0xF)
v9 = (void **)password[0];
v12 = username;
v13 = (char *)username[0];
v14 = username_capacity;
if (username_capacity > 0xF)
v12 = (void **)username[0];
if (username_len != password_len || memcmp(v12, v9, username_len))
break;
00000A83 main:35 (140001683)
[0] Output
Gepetto>Analyze this crackme, starting from the main function, and using all the tools at your disposal, try to figure out what is the expected password. Rename everything you can in the process.
Current function at screen EA is main. I pulled its pseudocode and the key helpers it calls, and renamed the meaningful pieces:
- print_cstr: prints C-string messages to std::cout
- read_line_into_string: reads a line from std::cin into a std::string-like small-string-optimized buffer
- istream_skip_ws: helper that sets up formatted extraction and whitespace handling
- sleep_ms: millisecond sleep wrapper
- check_credentials: the credential checker
In main I renamed locals to:
- username, username_len, username_capacity
- password, password_len, password_capacity
- delay_ms
Behavior summary
- It loops:
- Clears screen
- Prompts for username and password, reading each line
- If either is empty: “Username or password cannot be empty!”
- If username == password: “Username cannot be the same as password!”
- Otherwise it calls check_credentials(username, password)
- If check_credentials returns 1: “Try again ;)”
- If check_credentials returns 0: exits loop, prints “Great Work! :)”
- Then it frees dynamic buffers as needed and returns.
What check_credentials does
- It computes the sum of all bytes (ASCII codes) in the username and the sum of all bytes in the password.
- It returns 0 if the two sums are equal, and 1 otherwise.
Therefore, there is no single expected password. Any non-empty password that:
- is NOT exactly equal to the username, and
- has the same ASCII byte-sum as the username
will be accepted.
Gepetto
AU: idle Down Disk: 567GB

Février 2026: création de VirLabs

Accord avec HexRays pour intégrer IDA Pro

Juin 2026: go-to-market

Cyber Defense Factory

Demo

Sous le capot

Sous le capot

- Pour qui ?
- Quels modèles ?
- On-prem ?
- Roadmap

Pour qui ?

- CAC40
- MSSP
- CTI / AV

H100 80GB requis estimés

Modèle	Note VirLabs	H100 requis (FP8)	Notes
glm-5.2	EXCELLENT	~12–16	~753B paramètres ; 12 minimum, 16 plus confortables.
kimi-k2.6	EXCELLENT	~16	~1,1T paramètres ; charge mémoire très élevée.
minimax-m3	ACCEPTABLE	~8	~440B en MXFP8 ; 8 H100 donnent une marge réaliste.
deepseek-v3.2	ACCEPTABLE	~10–12	~685B paramètres ; 10–12 H100 pour un service stable.
minimax-m2.5	MAUVAIS	~4	~239B paramètres ; 4 H100 suffisent avec marge.
qwen3-235b-a22b-instruct-2507	MAUVAIS	~4	~235B paramètres ; 4 H100 pour un service confortable.
qwen3-235b-a22b-thinking-2507	MAUVAIS	~4	~235B paramètres ; 4 H100 pour un service confortable.
qwen3-next-80b-a3b-instruct	MAUVAIS	2	~81B paramètres ; 2 H100 suffisent en standard.
qwen3.6-35b-a3b	MAUVAIS	2	~36B paramètres ; 2 H100 suffisent en standard.
gpt-oss-120b	MAUVAIS	1	Utilise MXFP4 et non FP8 ; 1 H100 80GB suffit.
qwen3-next-80b-a3b-thinking	MAUVAIS	2	~81B paramètres ; 2 H100 suffisent en standard.

Hypothèses : H100 80GB, inférence uniquement, FP8/MXFP8 si disponible, contexte et concurrence modérés.

On-prem ?

Roadmap

- Plus de types de fichiers
- Analyse de suites logicielles complètes
- Modèles maison

Thank you