# Reversing Lab 02

Carlo Ramponi <carlo.ramponi@unitn.it>

# Approaching Reverse Challenges

Carlo Ramponi

## Approaching Reverse Challenges

When approaching a Reverse Engineering Challenge, before jumping straight into Ghidra, you firstly need to understand what kind of file you are dealing with.

Useful tools:

- `file` - determine file type
- `strings` - print the sequences of printable characters in files
- `hexdump` - display file contents in hexadecimal, decimal, octal, or ascii
- The binary itself! If you can execute it (through an emulator, perhaps), do it!
- `ltrace` - A library call tracer
- `strace` - trace system calls and signals

## Approaching Reverse Challenges

When you need to **start reversing**, import the file in `Ghidra`, depending on the file format, you might need to **instruct** `Ghidra` on how to open the file, by **installing an extension** perhaps.

Common binary formats, such as `ELF` or `PE`, are straightforward, but you might encounter some strange files, trust me!

## Approaching Reverse Challenges

When in **`Ghidra`**, you need to **identify the interesting code**

- If there are symbols, look at the function names 😅
- Look for the **entry address**
  - This is where the program starts executing
  - For ELF or PE binaries this is straightforward, other formats might require a little bit of googling.
- Look where interesting **library functions** (e.g. **`system`**) are used (i.e. **`XREFS`**)
- Look for **interesting strings** and their **`XREFS`**

## Approaching Reverse Challenges

When you’ve identified the functions you need to reverse:

- Don’t just look at the **decompiler**:
  - The decompiler is **not perfect**, it could have missed something
  - The disassembly should be your ground truth of what the program does
- Reverse engineering requires **manual work**:
  - **Rename variables** and functions
  - **Retype variables** and function arguments
  - **Create complex types** in Ghidra (structures, classes, …)
- Google is your friend!

## Challenges

## Chall 00 - Reversing 106

**Description**

*This ain't no reversing challenge, it's a web challenge, right?*

**Points**: *300*
**Author**: *carlo*
**Hints**:
1. <REDACTED>
2. <REDACTED>
3. <REDACTED>
4. <REDACTED>
5. <REDACTED>

# Chall 00 - Reversing 106

**Description**

*This ain't no reversing challenge, it's a web challenge, right?*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. The flag is checked **client side**, by whom?
2. <REDACTED>
3. <REDACTED>
4. <REDACTED>
5. <REDACTED>

## Chall 00 - Reversing 106

**Description**

*This ain't no reversing challenge, it's a web challenge, right?*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. The flag is checked **client side**, by whom?
2. The javascript code is calling a Module function `check_flag`, where is it?
3. <REDACTED>
4. <REDACTED>
5. <REDACTED>

## Chall 00 - Reversing 106

**Description**

*This ain't no reversing challenge, it's a web challenge, right?*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. The flag is checked **client side**, by whom?
2. The javascript code is calling a Module function `check_flag`, where is it?
3. The function is defined in a **WebAssembly** (`wasm`) module, what's this?
4. <REDACTED>
5. <REDACTED>

## Chall 00 - Reversing 106

**Description**

*This ain't no reversing challenge, it's a web challenge, right?*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. The flag is checked **client side**, by whom?
2. The javascript code is calling a Module function `check_flag`, where is it?
3. The function is defined in a **WebAssembly** (`wasm`) module, what's this?
4. There is a **mapping** between functions defined in wasm and `check_flag`
5. <REDACTED>

## Chall 00 - Reversing 106

**Description**

*This ain't no reversing challenge, it's a web challenge, right?*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. The flag is checked **client side**, by whom?
2. The javascript code is calling a Module function `check_flag`, where is it?
3. The function is defined in a **WebAssembly** (`wasm`) module, what's this?
4. There is a **mapping** between functions defined in wasm and `check_flag`
5. You need to reverse the function `$c`, how does the `wasm` VM work?

# Solution:

## Chall 00 - Reversing 106

## Chall 01 - Don't get Rusty

**Description**

*Be careful, reverse engineering is not an easy job. It is very easy to get* ***rusty****.*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. <REDACTED>
2. <REDACTED>
3. <REDACTED>
4. <REDACTED>

## Chall 01 - Don't get Rusty

**Description**

*Be careful, reverse engineering is not an easy job. It is very easy to get* ***rusty****.*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. The challenge was written in `rust`, how to **reverse** it?
2. <REDACTED>
3. <REDACTED>
4. <REDACTED>

## Chall 01 - Don't get Rusty

**Description**

*Be careful, reverse engineering is not an easy job. It is very easy to get* ***rusty****.*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. The challenge was written in `rust`, how to **reverse** it?
2. The main function defined by the user is not `main`, it's `chall::main`
3. <REDACTED>
4. <REDACTED>

## Chall 01 - Don't get Rusty

**Description**

*Be careful, reverse engineering is not an easy job. It is very easy to get* ***rusty****.*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. The challenge was written in `rust`, how to **reverse** it?
2. The main function defined by the user is not `main`, it's `chall::main`
3. Looks like we are dealing again with **some kind of encryption**, what kind?
4. <REDACTED>

## Chall 01 - Don't get Rusty

**Description**

*Be careful, reverse engineering is not an easy job. It is very easy to get **rusty**.*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. The challenge was written in `rust`, how to **reverse** it?
2. The main function defined by the user is not `main`, it's `chall::main`
3. Looks like we are dealing again with **some kind of encryption**, what kind?
4. Find the **key**, **iv**, **ciphertext** and **mode of operation** to decrypt the flag

# Solution:
## Chall 01 - Don't get Rusty

## Chall 02 - Reversing 104

**Description**

*Alright then, encryption is not enough, let's try with something else.*
*I wouldn't try to* ***solve*** *this by hand, but you can try if you want.*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. <REDACTED>
2. <REDACTED>
3. <REDACTED>
4. <REDACTED>

## Chall 02 - Reversing 104

**Description**

*Alright then, encryption is not enough, let's try with something else.*
*I wouldn't try to* ***solve*** *this by hand, but you can try if you want.*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. Identify the function that **checks the input**
2. <REDACTED>
3. <REDACTED>
4. <REDACTED>

## Chall 02 - Reversing 104

**Description**

*Alright then, encryption is not enough, let's try with something else.*
*I wouldn't try to* ***solve*** *this by hand, but you can try if you want.*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. Identify the function that **checks the input**
2. It performs **a lot of checks** on the input, **solving them by hand** will take long
3. <REDACTED>
4. <REDACTED>

## Chall 02 - Reversing 104

**Description**

*Alright then, encryption is not enough, let's try with something else.*
*I wouldn't try to* ***solve*** *this by hand, but you can try if you want.*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. Identify the function that **checks the input**
2. It performs **a lot of checks** on the input, **solving them by hand** will take long
3. You can try to rewrite the checks in **Z3** or solve it with **symbolic execution**
4. <REDACTED>

## Chall 02 - Reversing 104

**Description**

*Alright then, encryption is not enough, let's try with something else.*
*I wouldn't try to* ***solve*** *this by hand, but you can try if you want.*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. Identify the function that **checks the input**
2. It performs **a lot of checks** on the input, **solving them by hand** will take long
3. You can try to rewrite the checks in **Z3** or solve it with **symbolic execution**
4. If you get solutions that are not the flag, **add more constraints**!

# Solution:
# Chall 02 - Reversing 104

## Chall 03 - RandomPasswordGenerator (RPG)

**Description**

*Can you guess the password?*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. <REDACTED>
2. <REDACTED>
3. <REDACTED>
4. <REDACTED>

## Chall 03 - RandomPasswordGenerator (RPG)

**Description**

*Can you guess the password?*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. The challenge generates a password **rAnDomLy**, right?
2. <REDACTED>
3. <REDACTED>
4. <REDACTED>

## Chall 03 - RandomPasswordGenerator (RPG)

**Description**

*Can you guess the password?*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. The challenge generates a password **rAnDomLy**, right?
2. What is the seed used in `srand`?
3. <REDACTED>
4. <REDACTED>

## Chall 03 - RandomPasswordGenerator (RPG)

**Description**

*Can you guess the password?*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. The challenge generates a password **rAnDomLy**, right?
2. What is the seed used in `srand`?
3. You can **emulate** libc's **random functions** using python bindings or directly in C
4. <REDACTED>

## Chall 03 - RandomPasswordGenerator (RPG)

**Description**

*Can you guess the password?*

**Points**: *300*
**Author**: *carlo*
**Hints**:

1. The challenge generates a password **rAnDomLy**, right?
2. What is the seed used in `srand`?
3. You can **emulate** libc's **random functions** using python bindings or directly in C
4. This is a remote challenge, you can interact with it manually using `nc`, or in python using `pwntools`

# Solution:
## Chall 03
## RandomPasswordGenerator (RPG)

## Chall 04 - GiveBackAssembly (GBA)

### Description

*Welcome to a mysterious journey into the depths of retro technology!*
*Prepare to unleash your inner detective as you dive into the enigmatic world of* ***GiveBackAssembly****. This mind-boggling Reverse Engineering challenge will transport you back in time to a handheld device that holds secrets waiting to be unraveled. Sharpen your skills, grab your virtual magnifying glass, and embark on a quest to uncover hidden treasures buried within the cryptic assembly code.*
*Can you crack the code and emerge victorious?*
*The challenge awaits, brave explorer!*

**Points**: *300*

**Author**: *carlo*

**Hints**:

1. <REDACTED>
2. <REDACTED>
3. <REDACTED>
4. <REDACTED>

## Chall 04 - GiveBackAssembly (GBA)

### Description

*Welcome to a mysterious journey into the depths of retro technology!*
*Prepare to unleash your inner detective as you dive into the enigmatic world of* ***GiveBackAssembly****. This mind-boggling Reverse Engineering challenge will transport you back in time to a handheld device that holds secrets waiting to be unraveled. Sharpen your skills, grab your virtual magnifying glass, and embark on a quest to uncover hidden treasures buried within the cryptic assembly code.*
*Can you crack the code and emerge victorious?*
*The challenge awaits, brave explorer!*

**Points**: *300*

**Author**: *carlo*

**Hints**:

1. Use the `file` command to understand what the challenge is about
2. <REDACTED>
3. <REDACTED>
4. <REDACTED>

## Chall 04 - GiveBackAssembly (GBA)

### Description

*Welcome to a mysterious journey into the depths of retro technology!*
*Prepare to unleash your inner detective as you dive into the enigmatic world of* ***GiveBackAssembly****. This mind-boggling Reverse Engineering challenge will transport you back in time to a handheld device that holds secrets waiting to be unraveled. Sharpen your skills, grab your virtual magnifying glass, and embark on a quest to uncover hidden treasures buried within the cryptic assembly code.*
*Can you crack the code and emerge victorious?*
*The challenge awaits, brave explorer!*

**Points**: *300* **Author**: *carlo*

**Hints**:

1. Use the `file` command to understand what the challenge is about
2. You can run the challenge using a **GBA emulator**
3. <REDACTED>
4. <REDACTED>

## Chall 04 - GiveBackAssembly (GBA)

### Description

*Welcome to a mysterious journey into the depths of retro technology!*
*Prepare to unleash your inner detective as you dive into the enigmatic world of* ***GiveBackAssembly****. This mind-boggling Reverse Engineering challenge will transport you back in time to a handheld device that holds secrets waiting to be unraveled. Sharpen your skills, grab your virtual magnifying glass, and embark on a quest to uncover hidden treasures buried within the cryptic assembly code.*
*Can you crack the code and emerge victorious?*
*The challenge awaits, brave explorer!*

**Points**: *300* **Author**: *carlo*

**Hints**:

1. Use the `file` command to understand what the challenge is about
2. You can run the challenge using a **GBA emulator**
3. To correctly load the file in Ghidra, look for **gba-ghidra-loader**
4. <REDACTED>

## Chall 04 - GiveBackAssembly (GBA)

### Description

*Welcome to a mysterious journey into the depths of retro technology!*
*Prepare to unleash your inner detective as you dive into the enigmatic world of* ***GiveBackAssembly****. This mind-boggling Reverse Engineering challenge will transport you back in time to a handheld device that holds secrets waiting to be unraveled. Sharpen your skills, grab your virtual magnifying glass, and embark on a quest to uncover hidden treasures buried within the cryptic assembly code.*
*Can you crack the code and emerge victorious?*
*The challenge awaits, brave explorer!*

**Points**: *300*

**Author**: *carlo*

**Hints**:

1. Use the `file` command to understand what the challenge is about
2. You can run the challenge using a **GBA emulator**
3. To correctly load the file in Ghidra, look for **gba-ghidra-loader**
4. You might want to run an **aggressive instruction finder** analysis

# Solution:

## Chall 04

## GiveBackAssembly (GBA)

## Rated Challenge RandomPasswordGenerator 2.0 (RPG2)

**Description**

*Can you guess the password?*

**Points**: *300*
**Author**: *carlo*

**Deadline**: May 2nd, 2024 at 23:59

***GL HF!***

That's All
Folks