LetterBomb

WhoAteMyButter

Feb 09, 2022

CONTENTS

CHAPTER

ONE

MAC ADDRESSES

Each Wii has it's own MAC address. These addresses are core to identifying different Wii's, both to the other Wii's and to LetterBomb.

For LetterBomb to work, the Wii's MAC address must be embedded in the exploit file for the Wii to to recognize it.

MAC addresses are 12 bytes in length, and look like this:

- `AA:BB:CC:DD:EE:FF`
- `AABBCCDDEEFF`

LetterBomb will accept any valid Wii MAC address formats that match this regular expression:

`^([0-9a-fA-F]{12})$`

In other words, it must contain only A-F or 0-9, and be 12 characters in length.

Note: All valid Wii MAC addresses must also start with one of the sequences in an OUI list registered to Nintendo.

Run the included update_oui.sh to retrieve the newest copy of this list.

Warning: If your MAC address is `"0017AB999999"`, you are using an emulator such as Dolphin. The **[INFO]** log entry should tell you how to resolve this.

For more information regarding the MAC address in other contexts, read:

- Wikipedia
- IEEE - EUI
- IEEE - MCGRP

CHAPTER

TWO

HOW IT WORKS

LetterBomb works by exploiting a buffer overflow exploit in the Wii Messaging Board. By addressing a specific letter to the Wii (by using it's *MAC Addresses*), you can cause the Wii to read such a large message that it overflows and accidentally reads and executes the embedded Homebrew Channel inside of the letter's file.

To know more about how it works, watch the 23C5 video where the developers explain how they did this.

CHAPTER

THREE

LIMITATIONS

LetterBomb cannot run on any System Menu below version **4.3**. To exploit a **4.2** or lower Wii, look into BannerBomb or other related exploits.

LetterBomb requires:

- Any computer that has a Python installation
- Any computer with an SD card slot
- An SD/SDHC card formatted as either FAT16 or FAT32
- Your Wii's MAC address
- A Wii manufactured as RVL-001
- System Menu 4.3 *(not 4.2 or lower)*

CHAPTER

FOUR

TUTORIAL

To fully install LetterBomb start to finish, ensure your Wii meets the requirements set out in *Limitations*.

This guide also assumes you are using WhoAteMyButter's implementation of LetterBomb, and not the original website.

1. Get your Wii's MAC address and region
   - MAC: https://www.nintendo.com/consumer/systems/wii/en_na/includes/rvl-ht-int-find-mac-address.jsp
   - Region: This is where your Wii was manufactured. The first letter ID is what we want.

U	E	J	K
United States	Europe	Japan	Korea

2. Prepare SD/SDHC card
   1. Make sure **it is not** an SDXC card; it must be SD or SDHC
   2. If any data is on it, backup that data
   3. Format the SD card to either FAT16 or FAT32

> **Warning:** SDXC cards will not be recognized by the Wii. Only SD and SDHC cards are. Verify this before using the card.

3. Install LetterBomb
   - The easiest way to do this is by running this in a shell/terminal:

```
python3 -m pip install -U letterbomb
```

4. Ensure it's installed
   1. Open a Python shell:

```
python3
```

   2. Import LetterBomb and get its' package version

```
import letterbomb
print(letterbomb.__version__)
```

   3. Verify the version matches the version you downloaded
3. Run it through CLI

- Exit out of the previous shell with `exit()`
- Run:

```
python3 -m letterbomb <MAC address from step 1.1> <region from step 1.2> <filename␣
↪of zip> -b
```

Note: The ZIP archive patch can be either relative or absolute. `..`, ~, and other shortcuts are allowed.

4. Extract resulting ZIP archive to the SD card
   - Open the archive with any compatible program (7zip, WinRAR, Ark, etc.)
   - Extract the entire archive contents directly to the SD card
5. Verify extraction is completed
   - Ensure you extracted the exploit correctly, the card's contents should have the files & folders shown below:

     – **Note:** *The above images' specific dates, IDs, and timestamps do not need to match; only the general structure*
   - Eject the SD card safely
6. Insert SD card
   - Turn off your Wii first
   - Remove the card from the computer and insert it into the Wii's SD slot until you hear a click
     – See below image on where this slot is:
7. Boot Wii and run exploit
   - Start your Wii normally
   - Navigate towards the message board
   - Scroll through the dates until you find a red mail icon with a bomb inside of it

- Select the letter
- Install HomeBrew and/or BootMii

CHAPTER

FIVE

RUNNING THE WEB-SERVICE

While LetterBomb does not **require** a network connection, you can run your own local version of please.hackmii.com.

The web-service is written in Flask. To install it:

```
python3 -m pip install -U flask
```

Once complete, put the following two lines in your code or a Python console *(either works)*:

```
import letterbomb.web
letterbomb.web.app.run("0.0.0.0", 8080)
```

This will bind the service to port *8080* on all IPs assigned to the machine.

Note: Although `letterbomb.web` is part of `letterbomb`, `letterbomb.web` is not importable **through** `letterbomb`. Instead, *it is an isolated module*.

Note: To bind to a port below 1000, run the file/console with superuser priviledges.

Warning: Don't use the above method for deployment servers, see Flask docs.

Once bound, there are two ways of contacting the service:

5.1 POST (Normal)

Go to http://localhost:<port> (http://localhost:8080 by default).

Here is an image of it in action:

1. Check a region, and enter in your MAC address letter by letter.
2. If you would like to include BootMii in the LetterBomb, check the option.
3. If you configured a Captcha, verify it.
4. Cut a wire.

Note: It does not matter what wire you cut.

Warning: The MAC input fields only accept characters A-F, 0-9. Other characters are ignored, and arrows keys, etc. will still function normally.

For technical use, this is a summary of the POST request fields:

Field	Description	Expected Type	Example
a	1st 2 letters of MAC	str	`ec`
b	2nd 2 letters of MAC	str	`cd`
c	3rd 2 letters of MAC	str	`40`
d	4th 2 letters of MAC	str	`df`
e	5th 2 letters of MAC	str	`b9`
f	Last 2 letters of MAC	str	`a0`
region	Character of region	str, char	`U`, `J`
bootmii	Include BootMii in archive	int, bool	`1`, `0`

GET (JSON API) ^^^

There is also an endpoint `/get`, meant for plain-text HTTP GET methods. It is very basic, and accepts *nearly* the same parameters in the above POST request, only as URL parameters.

Examples:

- `/get?mac=000000000000&region=U&bootmii=1`
- `/get?mac=000000000000&region=E&bootmii=0`
- `/get?mac=000000000000&region=K&bootmii=1`
- `/get?mac=000000000000&region=J&bootmii=0`

Guide:

Field	Description	Expected Type	Example
mac	MAC address of Wii	str	`eccd40dfb9a0`
region	Character of region	str, char	`U`, `J`
bootmii	Include BootMii in archive	int, bool	`1`, `0`

Note: URL parameters look like this `?first=value&second=value&third=value`

The below default-port URL should error out, telling you the MAC was invalid: http://localhost:8080/get?mac=000000000000&region=U&bootmii=1

JSON will be sent following this inclusive format:

```
{
   "success": "boolean",
   "response": {
       "error": "str, description of error",
       "shorthand": "str, one/two words describing error"
   }
}
```

Note: On error, the *"success"* field will always be **false**. It will *never* be **true**.

On success, a file is sent. JSON will not be sent.

CHAPTER

SIX

API DOCS

Modules

LetterBomb: A fork of the classic Wii hacking tool from fail0verflow.

For most usage, you should be using `write_zip()`.

For additional usage, either:

- view documentation on ReadTheDocs
- build and view the documentation located in the *docs* folder.

If you downloaded this package from PyPi, the *docs* folder is not included.

Obtain the latest copy of LetterBomb here: https://gitlab.com/whoatemybutter/letterbomb

Note: *This exploit only works for System Menu 4.3. 4.2 and below will not work.*

LetterBomb is licensed under the GPLv3+ license. You can grab a copy here: https://www.gnu.org/licenses/gpl-3.0.txt.

exception `letterbomb.`**BadLengthMACError**(*message='bad mac, length should be 12 characters only'*)

Bases: `ValueError`

Raised when a MAC is not 12 characters in length.

exception `letterbomb.`**EmulatedMACError**(*message='bad mac, you cannot use a mac address from an emulator'*)

Bases: `ValueError`

Raised when a MAC is of an emulator.

exception `letterbomb.`**InvalidMACError**(*message='bad mac, does not belong to a Wii'*)

Bases: `ValueError`

Raised when a MAC does not belong to a Wii.

exception `letterbomb.`**InvalidRegionError**(*message='region must be one of U, E, K, J'*)

Bases: `ValueError`

Raised when region is a valid region character.

letterbomb.**mac_digest**(*mac: str*) → bytes
Process *mac* through a SHA1 encoding with '\x75\x79\x79' added.

Parameters **mac** (*str*) – MAC address to digest

Returns SHA-1 hash of MAC, plus \x75\x79\x79, then digested

Return type bytes

letterbomb.**serialize_mac**(*mac: str*) → str
Return *mac* as a string, each field split by a ":".

Padded with zeros to two-lengths.

Parameters **mac** (*str*) – MAC address

Returns ":" split string

Return type str

letterbomb.**validate_mac**(*mac: str*, *oui_list: list*) → int
Ensure *mac* is a valid Wii MAC address.

If MAC is valid, returns **:int:`0`**

Parameters

- **oui_list** (*list*) – OUI list, not a path to a OUI file
- **mac** (*str*) – MAC address to validate

Raises

- **BadLengthMACError** – if MAC is not the proper length
- **EmulatedMACError** – if MAC is from an emulator
- **InvalidMACError** – if MAC does not belong to a Wii

Returns 0 if MAC is valid

Return type int

letterbomb.**pack_blob**(*digest: bytes*, *time_stamp: int*, *blob: bytearray*) → bytearray
Pack *blob* with corresponding timestamps and the MAC *digest*.

Parameters

- **digest** (*bytes*) – MAC digest
- **time_stamp** (*int*) – Unix epoch time
- **blob** (*bytearray*) – Blob content

Returns Resulting blob content

Return type bytearray

letterbomb.**sd_path**(*digest: bytes*, *deltatime: datetime.datetime*, *time_stamp: int*) → str
Return the path of the LetterBomb, relative to the root of the SD card.

Parameters

- **digest** (*bytes*) – MAC digest, see `mac_digest()`
- **deltatime** (*datetime*) – Time of letter receival
- **time_stamp** (*int*) – Unix epoch time

Returns String of resulting path, relative

Return type str

letterbomb.**timestamp**() → list
Return a list of timestamps.

Returns List of [deltatime, delta, timestamp]

Return type list

letterbomb.**write_zip**()
Write LetterBomb archive to *output_file*, return nothing.

Depending upon the *region*, different LetterBomb templates will be used. If *pack_bundle* is True, the BootMii installer will be included with the archive.

Parameters

- **mac** (*str*) – Full string of the Wii's MAC address
- **region** (*str*) – Region of Wii, must be single letter of U,J,K,E
- **pack_bundle** (*bool*) – Pack the BootMii installer with archive
- **output_file** (*str,pathlib.Path*) – File to write archive to, can be relative or absolute

Raises

- **BadLengthMACError** – if MAC is not the proper length
- **EmulatedMACError** – if MAC is from an emulator
- **InvalidMACError** – if MAC does not belong to a Wii

letterbomb.**write_stream**(*mac: str*, *region: str*, *pack_bundle: bool*) → _io.BytesIO
Write LetterBomb archive as a raw bytes stream.

This ZIP will be streamed directly to an io.BytesIO object. This means the output will be in bytes.

Depending upon the *region*, different LetterBomb templates will be used.

If *pack_bundle* is True, the BootMii installer will be included with the archive.

Parameters

- **mac** (*str*) – Full string of the Wii's MAC address
- **region** (*str*) – Region of Wii, must be single letter of U,J,K,E
- **pack_bundle** (*bool*) – Pack the BootMii installer with archive

Raises

- **BadLengthMACError** – if MAC is not the proper length
- **EmulatedMACError** – if MAC is from an emulator
- **InvalidMACError** – if MAC does not belong to a Wii

Returns io.BytesIO object

Return type io.BytesIO

—