LunarROMCorruptor Help/Guide:

**About**

This project started around 2019 and was originally made in VB.NET. I was fascinated by other corruptors and wanted to make my own. The project was finished in the same year, but since the code was very messy, I decided not to release it.

A year later, I decided to remake it in C#, mainly because I want to teach myself more C#, but also to do some code refactoring to make it more optimized.

LunarROMCorruptor uses engines to corrupt files, each engine has its usage and its own settings that can be changed. I've included the Nightmare Engine, which is an engine from the Windows Glitch Harvester, into the project so that people that are most familiar with that engine can use it.

**Requirements**

.NET Framework 4 is required to run the program. For development, Microsoft Visual Studio Community 2019 is recommended.

**Features**

**Corrupt Every Nth Byte and Intensity Mode**

Corrupt Every Nth Byte allows for a more "regular" corruption. (e.g. corrupt every 1st byte corrupts the entire file) Intensity allows for a more randomized corruption by selecting random addresses to corrupt.

**File Saves and Stash Saves**

When you have found the perfect corruption, you can use both the File Save and the Stash Save features. File save copies the corrupted file and stores it in the "Saves" directory.

Stash saves are different. Instead of saving the whole file, instead, it saves the address of the changed byte and what the byte value is now. Which takes up less file size. Also using the stash editor, if you find that a certain corruption changes the color of Mario, you can remove bytes one at a time until you find the corrupted byte that causes it, then you can go nuts with that byte. (Similar to how the Stash Saves and Editor works on the Windows Glitch Harvester)

**Engines**

Here are the following engines that come with LunarROMCorruptor

* Nightmare Engine:

A simple, yet effective way of corrupting files, with three corruption modes: RANDOM, RANDOMTILT and TILT.

* Merge Engine

An engine that copies bytes from one file to the other to create corruption (For example, 'merging' Super Mario Bros. with Donkey Kong).

* Logic Engine

An engine that uses bitwise/logical operations on two selected bytes.

* Lerp Engine

An engine that takes the 2 neighbouring bytes where the selected byte is and corrupts the selected byte by using linear interpolation (e.g. 0 21 100 would become 0 50 100 if the Lerp Setting was set to 0.5)

* Hell Engine (Not really an engine, just picks a random engine to corrupt the selected byte)
* Vector2 Engine (Not finished)
* Manual (User manually enables and sets what types of corruption takes place in the file.)

**Misc**

* With more engines and settings to play around with, the more interesting results you can get with the engines.
* Allows drag and drop. No need to search for the file you want to corrupt.
* Can run an emulator after corruption.
* Audible feedback when corruption is done.

**LunarROMCorruptor Help:**

Controls: Intensity: The number of bits changed.

Starting Address and Ending Address: Where the corruption engine starts and ends, changing this can make the ROM run again.

Corruption Engines: Set what engine is going to corrupt the file.

Auto Saves and Current Save Files: Allows you to save your favorite corruption so you can load it again. Start Emulator Program: Override Arguments: Overrides the normal arguments of the selected emulator, use if the selected program/emulator doesn’t open the file.

Stash List: Allows you to save your favorite corruption so you can load it again, but instead of saving the whole file, it saves the instructions on how to corrupt it again, making it smaller than using the file save method.

Allow significant intensity and start/end byte. Allow for more highly intensive corruptions.

Corruption settings:

Increment byte. Adds or removes a custom number from the byte.

Repeat byte. Goes to the next byte that is closest to the byte selection. And copies and pastes itself.

EG: Viewable example

Before: 123123123

After: 121233123 (Selection length: 3)

Do math operations on the byte:

You can multiply divide or do doubles with the byte, using a custom number that you input.

Replace: If a random byte has your specified value in it, it will replace it with the specified value that you want to replace it with.

Shift right X amount of byte:

Each byte selected is shifted right by the number of bytes specified.

Paste byte to a random place:

Self-Explanatory.

Make bit equal:

When a byte is selected, the byte will be set to the number that you specify.

Corruption Engines:

Manual:

Where the user manually enables and sets what types of corruption takes place in the file.

Nightmare Engine:

Corrupt on the raw byte level.

Corruption type:

RANDOM: Will replace the Byte with a random Byte

RANDOMTILT: Will replace the Byte with a random Byte or Increments it or Decrements it.

TILT: Will Increment or Decrement a random Byte.

Merge Engine:

The easiest way of explaining is this:

String 1 & 2

“5F8AF8” “51D4F7”

Say bit 2 was chosen on both

F and 1

1 would copy itself to F

“518AF8” “51D4F7”

However, the other file (Merge file) will stay the same.

Corruption Type:

NONE:

No corruption type

RANGE

Gets the file byte and the merge byte and takes both away, then sets the file byte to that result.

Logic Engine:

Performs logical/bitwise operations to the bytes.

Here are the following operations:

AND

OR

XOR

NOT

NAND

NOR

XNOR

SWAP

SHIFT

Lerp Engine:

Corrupt on the raw byte level.

Chooses a random byte and gets changed via linear interpolation of the byte neighbors.

E.g.

If the middle one was chosen

0 0 100

The result if the split value is 0.5 is

0 50 100

Since it is between 0 and 100, thus interpolation.

Hell Engine:

Chooses a random engine for each byte and corrupts that byte with the selected engine. And Repeat.

Mess around with each corruption parameter/option to make random, unique corruptions.

ByteEdit:

ByteEdit is the new feature where you can look at and modify a value in a specific address that the user specifies.

Stash Editor:

Stash Editor is a new feature where you can edit the selected corruption stash file and modify the corruption stash. Like the Windows Glitch Harvester.