Flexible ROM Corruptor

Corruption Guide

Version [A.1.5.8]

Table of Contents

Note: It is strongly recommended that the user is familiar with the command line before using.

1. How to Compile

Assuming you've already downloaded the package from SourceForge.net or ArchLinux.org, didn't download from any other website and have already extracted the contents using 7zip, do the following:

1.1. Windows

Since Windows OS only comes bundled with an assembler, it's best to just use the precompiled binary that comes with the package. If you wish to manually compile files in the future, I'd strongly recommend using a Windows variant of GCC and following the instructions for Unix-like systems, replacing "corrupt" with "corrupt.exe".

1.2. Unix-like

First, ensure that you have GCC installed on your system. In a terminal or equivalent, run

```
gcc main.cpp -o corrupt -std=c++11
```

Ensure that the file paths are correct.

1.3. ArchLinux

ArchLinux users should download the snapshot at *https://aur.archlinux.org/packages/corrupt/* and follow the installation process.

2. How to Corrupt

2.1. Summary

```
943597 = 0xE65ED = x3462755
```

-v
-l *cfile ifile ofile*
-e *“script"*
-c *ifile ofile startnum stopnum pernum op addnum*
-s s*file*

Op	What it does
add	Add
sub	Subtract
mul	Multiply
div	Divide
mod	Modular
rep	Replace
not	Not
xor	Xor
or	Or
and	And
bsl	Bit-Shift Left
bsr	Bit-Shift Right
ran	Randomize

2.2. Number Formatting

This program is capable of using number formats other than base 10 (regular numbers). Unless you are an advanced user or want to load a corruption created by the Vinesauce ROM Corruptor, it is okay to skip this section. Decimal numbers do not require any formatting. Hexadecimal numbers are prefixed with a `0x`, octal numbers are prefixed with an `x`. The Vinesauce ROM Corruptor uses hexadecimal as its inputs for memory addresses, so startnum and stopnum should be formatted as such. To make things faster, you can write `EOF` for the stopnum value to indicate the end of the file.

2.3. -v (Check the Program Version)

This switch displays the corruptor version, this is so you can check to see if the corruptor is outdated.

```
corrupt -v
```

2.4. -l (Load a Corruption)

This is the load switch, this allows you to apply corruptions that other people have made in the form of a file. The syntax of this switch is:

`-l` *`cfile ifile ofile`*

cfile represents the file path which points to the corruption file. ifile represents the ROM that you want to corrupt and ofile is the file path that the ROM should be saved to.

```
corrupt -l glitch.frc ~/docs/SMB3.nes
o.nes
```

2.5. -e (Execute System Command)

This is the execute switch, which was added for convenience. It is common for this switch to me used to automatically run the emulator and load the game. This program takes the contents of the argument and passes it to the system command interpreter. For Windows, the scripting language is batch. For Unix-like systems, it's bash. Example command:

```
-e "fceux o.nes"
```

2.6. -c (Create a New Corruption)

This is the corrupt switch, the most complicated one. The syntax of this switch is:

```
-c ifile ofile startnum stopnum pernum op addnum
```

ifile and ofile represent the file paths to the ROM you want to corrupt and where you want to save the file to. startnum and stopnum are numbers which indicate where the corruptor should start and stop corrupting, measured in bytes. pernum is the period of corruption, measured in bytes per corruption. It represents how many bytes should be skipped. For example, a corruption with a range of six bytes at two bytes per corruption will have three corrupted bytes. addnum represents the secondary input of an operation. If op is set to add, the current byte is equal to ten and addnum is set to five, then the resulting byte will be equal to fifteen. op represents the operations performed on the currently read byte.

Op	What it does
add	Add
sub	Subtract
mul	Multiply
div	Divide
mod	Modular
rep	Replace
not	Not
xor	Xor
or	Or
and	And
bsl	Bit-Shift Left
bsr	Bit-Shift Right
ran	Randomize

```
corrupt -c SMB3.nes o.nes 41002 384000 3026 or 2
```

2.7. -s (Save the Created Corruption)

This is the save switch, this allows you to save a corruption to a file, so it can be applied again without having to remember the numbers and so it's easier to share your corruptions with others. The only argument is a file path to the place you want the data to be saved. This produces the files that can be loaded with -l.

```
corrupt -c SMB3.nes o.nes 41002 384000 3026 or 2 -s glitch.frc
```

3. Corruption Guide

3.1. NES

When corrupting NES files, there are three major factors. The first major factor is the range of corruption. The start byte should be set to a value large enough so essential components of the game are left intact, but should be small enough that the corruption is visible. It is recommended that the start byte should be between `37000` and `61000`. Trial and error may be necessary to find a good spot. It is generally recommended to set the stop byte to the end of the file, but some recommend setting the value to a little less than that. The next major part is the period. This is known for requiring the most trial and error, a recommended spot to start is between `1000` and `3000`. The third major part is the operation. I personally prefer the randomizer, but users of the Vinesauce ROM Corruptor suggest using the Bit-Shift Right operation. Because of the nature of this operation, the value should never be set to a multiple of eight or zero, because nothing will happen. You should set the value somewhere between `1` and `7`, the higher the value, the more distorted the file. Once you find a good spot, simply increment the value of the period up or down, causing a small change in the total bytes corrupted, but a drastic change in how the game behaves. If you see a black screen, it means that an essential byte has been corrupted. You can fix this by either setting your start byte to a higher value or setting the period to a higher value. Usually, you should adjust the period before adjusting the start value, as this is the more likely cause.

4. Contribution

It would be awesome if you could help me and others make this program better. Surprisingly, it's usually the small things that make a big difference.

4.1. Suggest Features/Rate Me/Report Bug

To suggest features, report a bug or ask a question, go to *https://sourceforge.net/projects/corrupter-for-linux/* and, under the discussion tab, you can message me or a developer. If you want to rate this program, click on the reviews tab.

4.2. Programming/Translating

If you wish to take on the brave task of helping me write this program/guide or you want to help with the even more brave task of translating this program/guide into a different language, don't hesitate to make a post on the forum. At the current moment, the goal is to release this program and guide in thirteen different languages.

- English
- Russian
- German
- Japanese
- Spanish
- French
- Portuguese
- Italian
- Chinese
- Polish
- Turkish
- Dutch
- Persian