【WIP】

x86_64 アセンブリでのライフゲームの実装

Arch B2
tatsu
親:

背景

- OSやセキュリティに興味があった。
- CTFに挑戦してみた結果、バイナリ及びアセンブリを解析する問題の意味が何度読んでもわからなかった。
- 中には、ツールを使うことで、解けるような問題もあったが、それは本質的な理解ではないと感じた。

背景

- その他有名な脆弱性、例えばバッファオーバーフローに関する説明を読んだりもしたが、ぼんやりとわかっただけで、自分で説明できる状態と言い難かった。

- 色々と勉強して行く過程で、バイナリ及びアセンブリを理解することは必須だということがわかった。

- そこで、アセンブリを自分で書くことで、理解を深めたかった。

手法

- x86_64アーキテクチャのアセンブリで、ライフゲームを実装する。

- セキュリティやOSを今後やって行く上で、その第一歩として、ライフゲームというものに挑戦する。

手法

- ただし、コンパイラから生成されたアセンブリからのコピペはしない。

- 全部自分で書く。

x86_64とは

- 32bitだった、x86アーキテクチャを64bitに拡張した命令セットアーキテクチャ
- CISCという可変長命令のアーキテクチャ
- MacBookや、Windowsなど、市販のコンピュータで使われていることが多い。

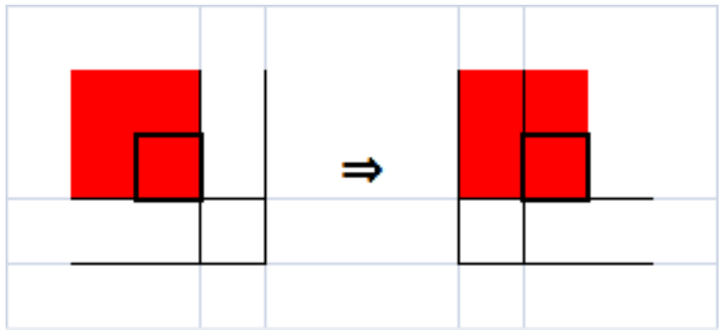
ライフゲームとは

- 基盤のような格子があり、その格子一つ一つをセルと呼ぶ。
- そのセルが生きていた場合
    - そのセルの回りに、2個あるいは3個生きたセルがあれば生存。
    - その他の場合は死。
- そのセルが死んでいた場合
    - そのセルの回りに、3個の生きたセルがあれば、誕生。
    - その他の場合は変化なし。

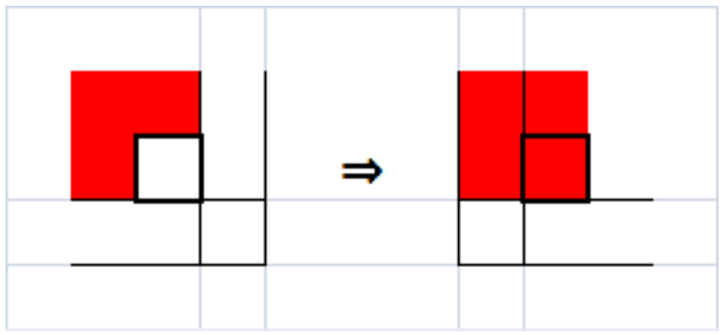
ライフゲームとは

- 過密により死ぬ

- 生存



- 誕生

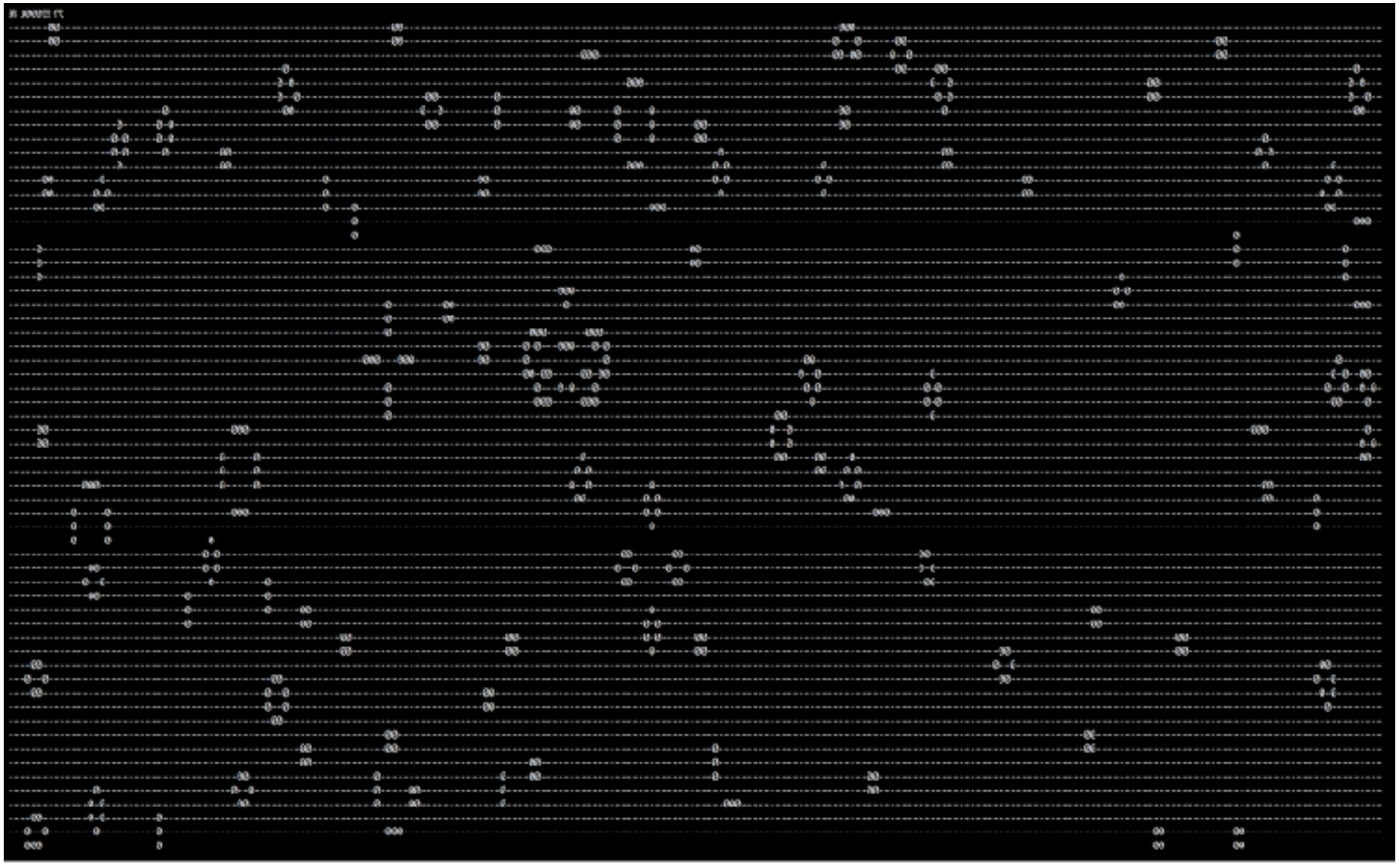


実装の工程

- Cでライフゲームを書き、ロジックを確認する。
  https://github.com/doooooooooingggggg/lifegame/blob/master/lifegame.c
- このコードを参考に、フルスクラッチでアセンブリを書いていく。
- 書いたコードを、nasmを用いてアセンブルし、オブジェクトファイルを生成
- ld で、データをリロケートし、シンボルの参照をまとめる。
- 実行

実装

- Cでの実装を実行した結果



アセンブリでの実装

- 必要そうな処理を考えてみたところ、
  - printf
  - 分岐
  - ループ
  - 配列の操作
  - 関数ジャンプ
  - sleep
- 以上を組み合わせて、頑張る。

アセンブリでの実装

- 必要そうな処理を考えてみたところ、
  - printf -> **システムコールのwriteを使うことで解決**
  - 分岐 -> **cmpで解決**
    - ループ -> **カウンタとcmpで解決**
    - 配列の操作 -> **バイト長に気をつける**
    - 関数ジャンプ -> **jmp**
    - sleep -> **ループし、1秒おきくらいになるように調節**
- 以上を組み合わせて、頑張る。

実装

- https://github.com/doooooooooingggggg/lifegame/blob/master/lifegame.s

今後の展望

- 乱数の生成がうまくいかなかったため、この部分を訂正する。

- スタックポインタをうまく活用できなかった。

- 春休みは、この分野の勉強を進め、組み込みOSに挑戦してみたい。

参考文献

- 「Jun's Homepage」http://www.mztn.org/index.html
- 「原書で学ぶ64bitアセンブラ入門」
  http://warabanshi.hatenablog.com/search?q=%E5%8E%9F%E6%9B%B8%E3%81%A7%E5%AD%A6%E3%81%B6
- 「プログラミング講座2-1・ライフゲーム-まずはライフゲームって何？って話」
  https://fujori.com/access-excel-vba/enjoy-vba/programming-course-2-1/