SecHack365

競技向けデバッガ拡張の開発

学習駆動コース 坂井ゼミ 黒瀬海耀

動機と背景

デバッガをもっと万能に

初めてデバッガを触ったときに感じた、

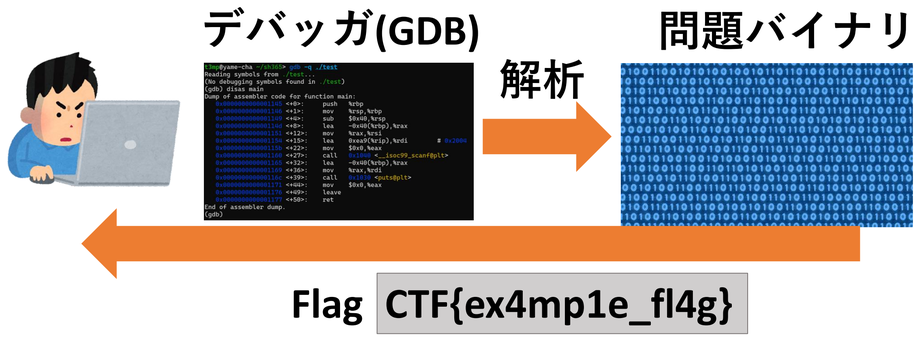
- **見えにくいメモリの内部を覗ける楽しさ**
- **動作の根幹を知れる嬉しさ**
- **知らなかった世界が見えてくる高揚感**
- **他のツールにはない圧倒的な万能感**

これらに大変感銘を受けたため、デバッガの機能性について開発を行う事で
デバッガで見える世界をもっと広くしたいと考えた。
また、実際にデバッガを使用していく中で
「**どのような仕組みで動いているのか**」という疑問を感じていた。
今回の開発を通してデバッガの仕組みについての知見を深め、
この疑問を解消したいと考えた。

競技に特化したデバッガ

今回はCTFと呼ばれる競技に特化したデバッガの機能を開発した。
CTF(Capture The Flag)と呼ばれる様々な専門知識や技術を駆使して
Flagと呼ばれる特定のフォーマットの文字列を探しだす競技の事であり、
問題のジャンルによってはデバッガを使用してプログラムの動作を
解析する必要がある。

デバッガを使用してCTFの問題を解く例



しかし、CTFは時間制限が存在する競技であるため、
このような解析は素早く行えることが望ましい。
そこで、
デバッガの機能性を拡張する事で解析を素早く行えるのではないか
と考えた。

デバッガの機能によって省略できる作業

- **シンボル情報が削除されたバイナリの解析**

→ シンボル情報が削除されているため、
解析の糸口となるmain関数を探す必要がある

- **特定のバージョンの共有ライブラリの読み込み**

→ バージョンによる動作の違いがあるため、
問題の環境と同じ環境を再現する必要がある

これらを実現するために、Pythonを使用してデバッガの拡張を開発した。

実装した機能と今後

main関数の探索機能

```
(gdb) whereis
[*] searching main function...
[*] saving program counter
[*] set break point at __libc_start_main
Breakpoint 1 at 0x7ffff7e27c20: file ../csu/libc-start.c, line 141.
[*] restart program

Breakpoint 1, __libc_start_main (main=0x555555555145 <main>, argc=1,
    argv=0x7fffffffe508, init=0x555555555180 <__libc_csu_init>,
    fini=0x5555555551e0 <__libc_csu_fini>, rtld_fini=0x7ffff7fe21b0 <_dl_fini>,
    stack_end=0x7fffffffe4f8) at ../csu/libc-start.c:141
141     in ../csu/libc-start.c
[*] main function found !
$1 = (int *) 0x555555555145 <main>
```

シンボル情報が削除されたバイナリから
main関数を自動で検出し、変数に保存する。
これによってプログラムの動作の起点を素早く解析できる。

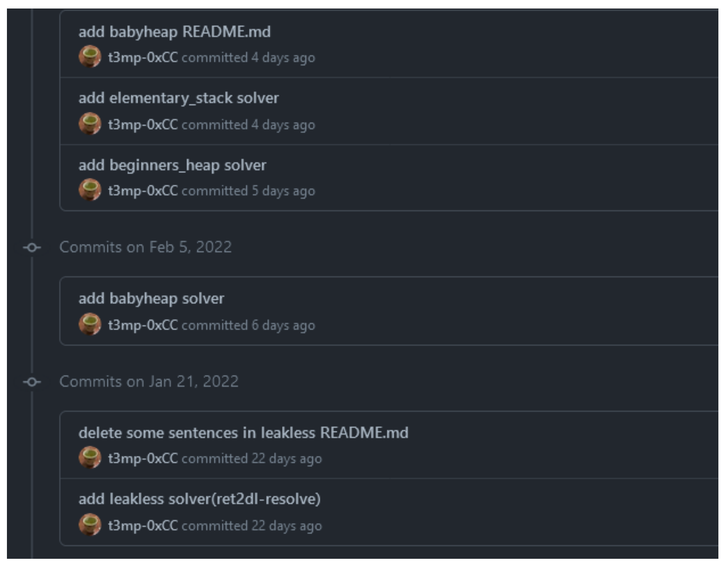
任意のglibcを読み込み

glibcとはC言語で使用される標準ライブラリであり、バージョンによって

- メモリのアロケーションの挙動
- セキュリティ機構の有無
- 関数および変数のアドレス

などが異なる。
これらはCTFで問題を解く上で重要な要素となりうるので、**問題で使用されるglibcのバージョンと解析環境のglibcのバージョンを一致させる作業**が必要となる。
この作業を上記のmain関数の探索機能と同様にコマンドによって実行できる機能を今後開発したいと考えている。

CTFへの取り組み



SecHack365参加中はデバッガの機能の実装や改善のために、参加前に比べて一層CTFに取り組むようになった。
修了後も取り組みを続け、新しい機能の実装について構想を練っていきたい。
また、最終的には自身で作成したツールによる解法の掲載を行いたいと考えている。

国立研究開発法人 情報通信研究機構