Reverse engineering

ОБРАТНАЯ РАЗРАБОТКА И ВЗЛОМ ПО

Основные задачи

- Установить логику программы с закрытым исходным кодом
- Воссоздать программу, аналогичную проприетарной
- Избавить проприетарную программу от ненужного функционала (проверка производителя/лицензии)

Что такое программа?

- Исполняемый файл популярных ОС и архитектур
- Байткод виртуальной машины (Java/.NET)
- Интерпретируемый код (PHP/Python/Perl)

В рамках сегодняшней лекции это только пункт 1

Исполняемый файл

Это, собственно, набор инструкций процессора, смешанный с данными, необходимыми для работы программы. На разных ОС приняты разные форматы исполняемых файлов: для Windows это PE (Portable Executable), для Linux ELF (Executable and Linkable Format). Расширения файлов .exe и <ничего> для Windows и Linux соответственно.

Важно помнить, что разделяемые библиотеки (.dll/.so) имеют схожий формат, хотя их обратная разработка затруднена не столь простой отладкой.

Как… запустить исполняемый файл?

Windows

Ммм, двойным щелчком.

Серьезно, больше ничего не нужно, можно еще запустить файл по имени из командной строки (cmd.exe)

Linux

Любой файл можно исполнить только когда у него есть права на исполнение. Запускать с файловой системы FAT без перемонтирования с -o exec невозможно.

Типичный сценарий запуска программы:

```
chmod +x file
```

```
./file #вы только посмотрите на этот путь
```

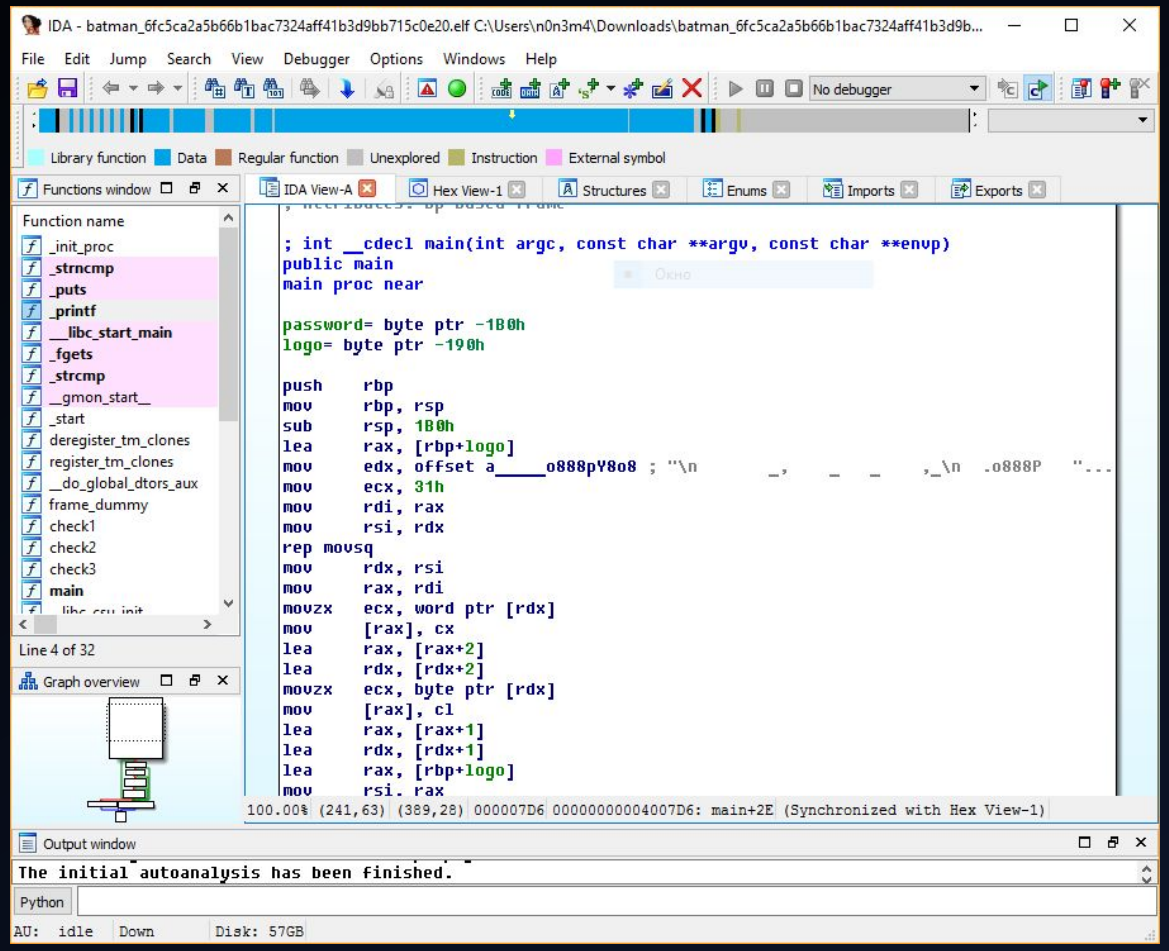
Структура исполняемого файла

Исполняемый файл состоит из сегментов, секций и всего такого. Вкратце они позволяют понять, где код, где данные, где константы и все такое.

В Linux это всё можно посмотреть командой readelf, в Windows – а черт его знает, это не очень нужно там.

IDA (о ней далее) покрасит все секции разными цветами сам.

Как понять что происходит?

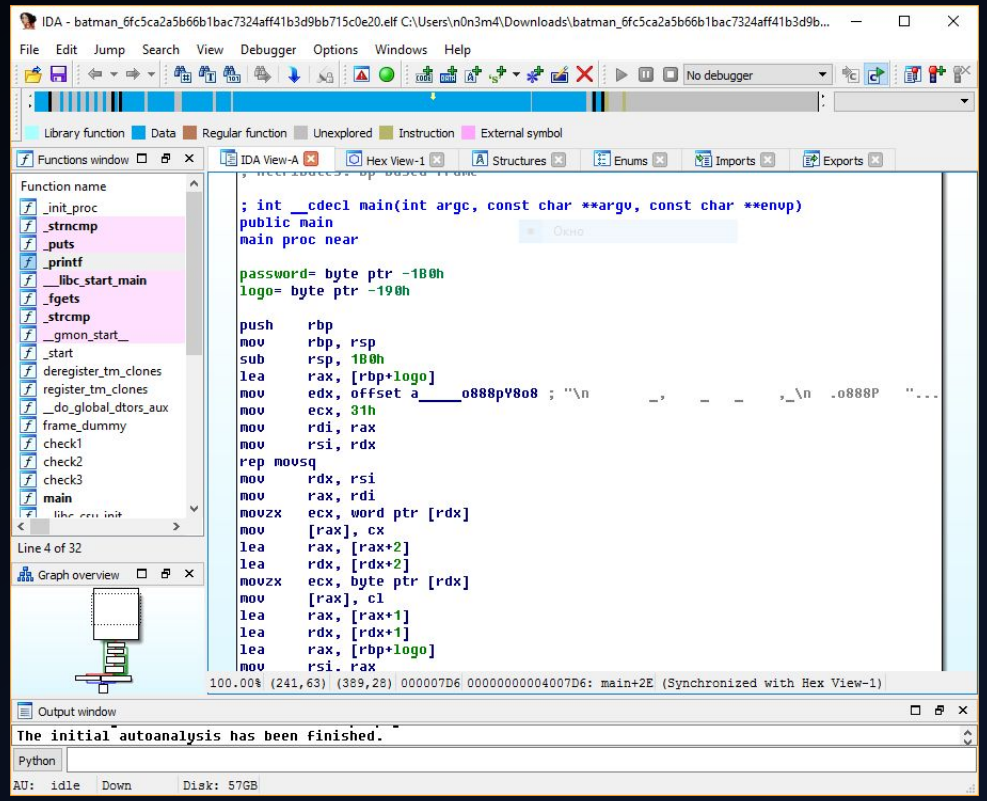


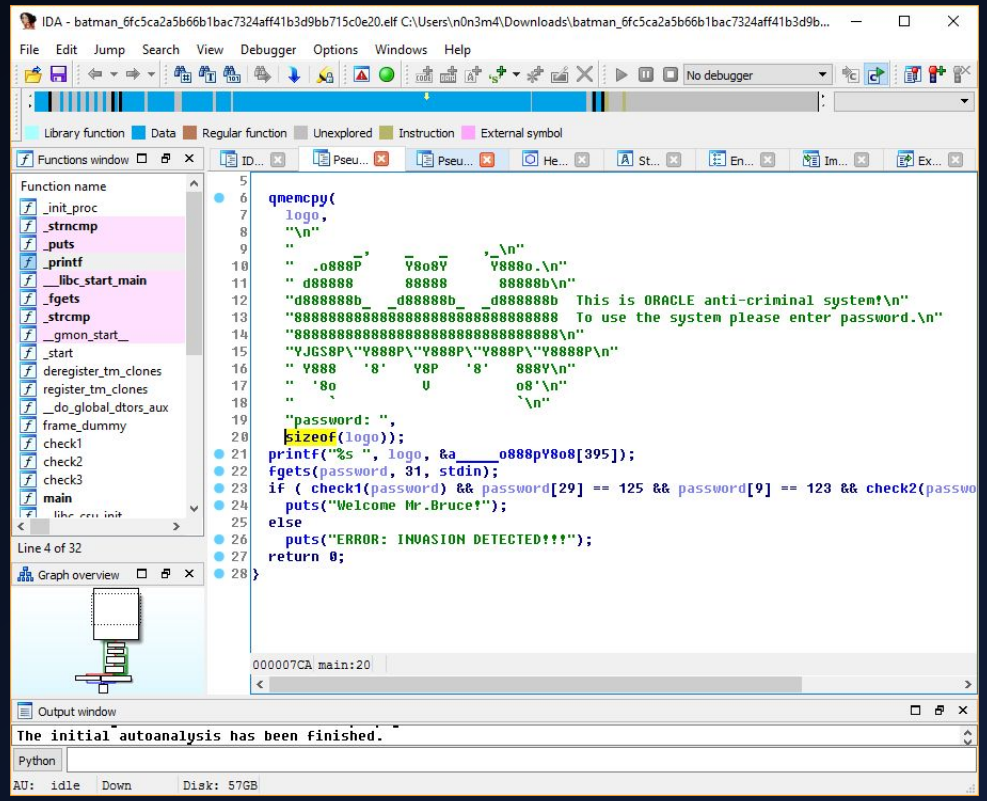
Interactive DisAssembler (IDA)

- Стоит всего лишь от $1200. И это без декомпиляторов.
- Умеет, тем не менее, дизассемблировать почти всё
- Стандарт индустрии
- Обладает декомпилятором HexRays
- Его автор параноик и думает, все его покупатели – пираты
- А они и правда пираты… ну или неудачники.
  В интернете доступна версия 6.8, которую украли у HackingTeam (если кто помнит новости)

Hexrays

- Стоит еще больше
- Умеет ДЕКОМПИЛИРОВАТЬ





Что же теперь делать?

- Заходите в подозрительную функцию (обычно это main)
- Жмете F5
- Готово, теперь вы можете читать “код”

- Двойным щелчком можно переходить между функциями, X выводит список ссылок на объект под курсором в программе.

Как, тем не менее, понять что происходит?

- Просто прочитать. Это же легко, правда?
- Загуглить встреченные константы
- Загуглить названия функций. Макрос assert() выдает имена оригинальных файлов, их можно гуглить
- Отладить
- Ничего не помогло? Есть система доказательства теорем Z3
  Это ее научное название, на самом деле это решалка всего подряд

Отладчик?

- Позволяет выполнять программу пошагово, смотреть регистры, инструкции и всё такое. Только ассемблер.
- Под Windows самыми известными являются OllyDbg и x64dbg (слышали о Denuvo? Его официальный “спонсор”)
- GNU Debugger (gdb). Вообще не только для Linux, но в винде не очень хорош. А вообще крут, еще и плагины есть (PEDA)
- IDA. Ходят слухи, она умеет отлаживать даже линукс через удаленный gdb, но это неточно. Под виндой отладчик даже и неплох

Гугл?

- Позволяет искать (кто бы мог подумать)
- Ищет весьма неплохо, даже по исходным кодам
- И по константам

- Еще позволяет скачать пиратскую IDA :3

Z3/Z3py?

- Если вы обратились к Z3 вы или очень круты
- Или совсем отчаялись, причем скорее всего второе
- Позволяет обращать хеш-функции, туповатые LFSR, решать уравнения
- Беда только в том, что почти никто не знает, как это делать правильно. Разработчики Z3 тоже, поэтому у них есть множество разных решателей систем условий (между которыми можно выбирать в самом Z3)

Пример

```
import string
from z3 import *
s = Solver()
x=BitVec('x',32)
y=BitVec('y',32)
s.add(x*y==2016)
s.add(x^y==0xDEADBEEF)
print s.check()
m=s.model()
print (m[x].as_long()*m[y].as_long())&0xFFFFFFFF,hex(m[x].as_long()^m[y].as_long())
```

Вывод:

```
sat
2016 0xdeadbeef
```

Сервер с задачами

http://dmz.n0n3m4.ru/tasks

Вопросы? :)