CSC 6580 Spring 2020

Instructor: Stacy Prowell

Basic Information

Phone / Text: +1 (724) 307-8229

Email: sprowell@tntech.edu

Tuesday and Thursday
8:00am - 9:20am Central

Assumptions

I am going to assume you know some basic computer architecture (processor, cache, memory, files, etc.). I will assume you know C.

I will *not* assume you know:

- **Assembly**, at least not well. It will help if you do, but this is not really a course to teach you assembly.
- **Python**. We're going to use Python, and I'll go over that as we encounter it, but this is not really a course to teach you Python.

Most of the work will be programming projects.

Grades

Homework	60
Midterm	20
Final	20
Total	100

Academic integrity

Trust the value of your intellect and ask for help when you need it.

Take responsibility and credit for your own work and ideas; give credit for the work or ideas of others, share credit for collaborative work. Work to treat others ethically and give then the same considerations you would want. Be as honest as you can be.

You are surrounded by people who went through this before you.

Automated Malware Analysis

Automated ~~Malware~~ Binary Analysis

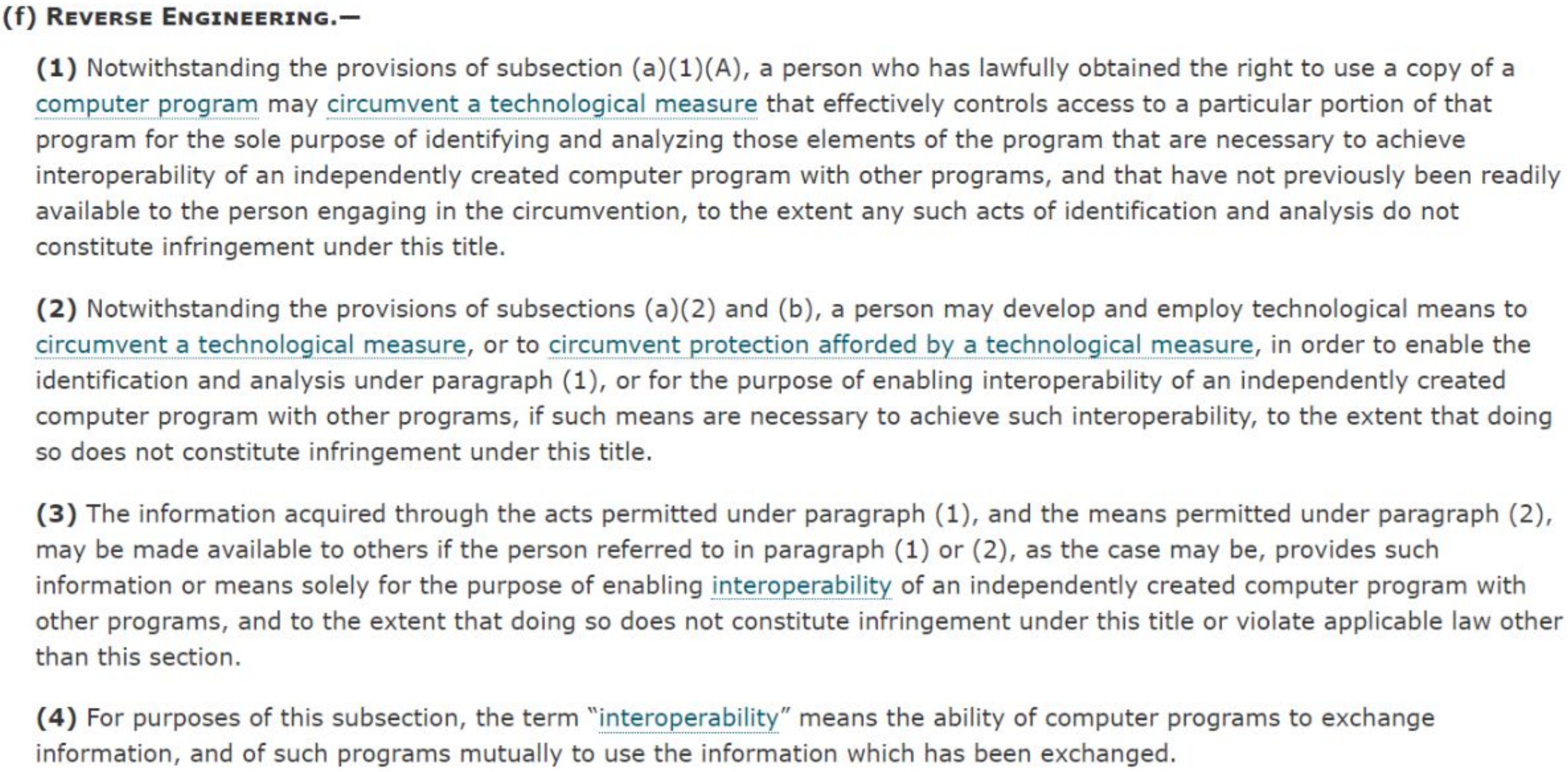
Binary Analysis

- Reverse Engineering
- Vulnerability Discovery
- Malware Discovery
- Code Protection

- Optimization
- Program Transformation
- Code Recovery

Binary Analysis

- **Reverse Engineering**
- Vulnerability Discovery
- Malware Discovery
- Code Protection
- Optimization
- Program Transformation
- Code Recovery



(f) Reverse Engineering.—

(1) Notwithstanding the provisions of subsection (a)(1)(A), a person who has lawfully obtained the right to use a copy of a computer program may circumvent a technological measure that effectively controls access to a particular portion of that program for the sole purpose of identifying and analyzing those elements of the program that are necessary to achieve interoperability of an independently created computer program with other programs, and that have not previously been readily available to the person engaging in the circumvention, to the extent any such acts of identification and analysis do not constitute infringement under this title.

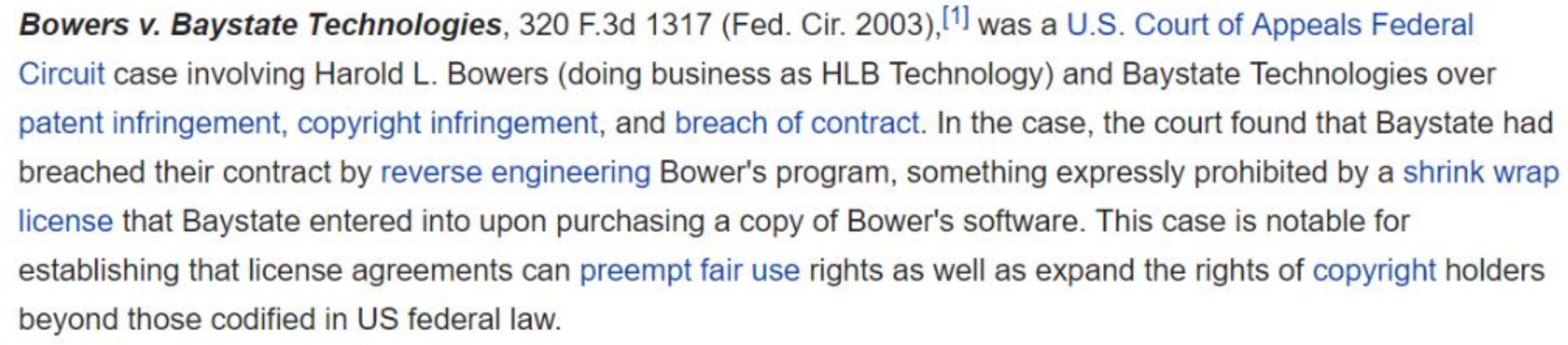
(2) Notwithstanding the provisions of subsections (a)(2) and (b), a person may develop and employ technological means to circumvent a technological measure, or to circumvent protection afforded by a technological measure, in order to enable the identification and analysis under paragraph (1), or for the purpose of enabling interoperability of an independently created computer program with other programs, if such means are necessary to achieve such interoperability, to the extent that doing so does not constitute infringement under this title.

(3) The information acquired through the acts permitted under paragraph (1), and the means permitted under paragraph (2), may be made available to others if the person referred to in paragraph (1) or (2), as the case may be, provides such information or means solely for the purpose of enabling interoperability of an independently created computer program with other programs, and to the extent that doing so does not constitute infringement under this title or violate applicable law other than this section.

(4) For purposes of this subsection, the term "interoperability" means the ability of computer programs to exchange information, and of such programs mutually to use the information which has been exchanged.

Source: 17 U.S. Code § 1201 - Circumvention of copyright protection systems (https://www.law.cornell.edu/uscode/text/17/1201)

End User License Agreements



Bowers v. Baystate Technologies, 320 F.3d 1317 (Fed. Cir. 2003),[1] was a U.S. Court of Appeals Federal Circuit case involving Harold L. Bowers (doing business as HLB Technology) and Baystate Technologies over patent infringement, copyright infringement, and breach of contract. In the case, the court found that Baystate had breached their contract by reverse engineering Bower's program, something expressly prohibited by a shrink wrap license that Baystate entered into upon purchasing a copy of Bower's software. This case is notable for establishing that license agreements can preempt fair use rights as well as expand the rights of copyright holders beyond those codified in US federal law.

Source: Wikipedia (https://en.wikipedia.org/wiki/Bowers_v._Baystate_Technologies,_Inc.)

Questions?

Recommend the **Coders' Rights Project Reverse Engineering FAQ**:
https://www.eff.org/issues/coders/reverse-engineering-faq

ELECTRONIC
FRONTIER
FOUNDATION

Binary Analysis

- Reverse Engineering
- **Vulnerability Discovery**
- Malware Discovery
- Code Protection

- Optimization
- Program Transformation
- Code Recovery
- Instrumentation

Disclose?

“If you see something… should you say something?”

- Many companies have **vulnerability disclosure policies**… but not all!
- Look for the **safe harbor** policy… if there is one.

Example: Dropbox
https://blogs.dropbox.com/tech/2018/03/protecting-security-researchers/

Questions?

Recommend you visit HackerOne:
https://www.hackerone.com/

hackerone

Binary Analysis

- Reverse Engineering
- Vulnerability Discovery
- Malware Discovery
- Code Protection

- Optimization
- **Program Transformation**
- Code Recovery
- Instrumentation

Binary to binary

- Dynamic recompilation
- Instrumentation

DynamoRIO: http://dynamorio.org/

Static Analysis

Rice’s Theorem

A string *s* is accepted by Turing machine *M* iff *M* halts in an accepting state. The set of all such strings is the language *L* of *M*.

A property of the Turing machine languages is *trivial* iff it is either true for all languages or false for all languages.

Rice’s Theorem: Any nontrivial property of Turing machine languages is undecidable.

Discuss!

Undecidable

There is no effective mechanical process that can decide whether or not:

- A program halts on all inputs.
- A program copies itself (a virus).
- A program prints a string.
- Etc.

Undecidable

Does this program halt?

```
int main( void ) {
    puts( "Hello, World!" );
    return 0;
}
```

Undecidable

```c
int main( void ) {
    puts( "Hello, World!" );
    return 0;
}
```

Can we write a program to decide if this program halts?

Undecidable

How about this?

- If a program contains no while loops or for loops, contains no branches to prior locations, and all subprograms halt, then the program halts.

Undecidable

What about this?

```
for (
    int64_t index = START;
    index != END;
    index += STRIDE) {
    f(index);
}
```

Tools

Lots of tools...

- Ida Pro
  https://www.hex-rays.com/products/ida/support/download_freeware.shtml
- Binary Ninja
  https://binary.ninja/
- Binary Analysis Platform (BAP)
  https://github.com/BinaryAnalysisPlatform/bap
- BitBlaze (early BAP)
  http://bitblaze.cs.berkeley.edu/

- The Online Disassembler
  https://onlinedisassembler.com/odaweb/
- Capstone
  https://onlinedisassembler.com/odaweb/
- Triton
  https://triton.quarkslab.com/
- Pin
  https://software.intel.com/en-us/articles/pin-a-dynamic-binary-instrumentation-tool
- Pharos
  https://insights.sei.cmu.edu/sei_blog/2015/08/the-pharos-framework-binary-static-analysis-of-object-oriented-code.html

Lots of tools...

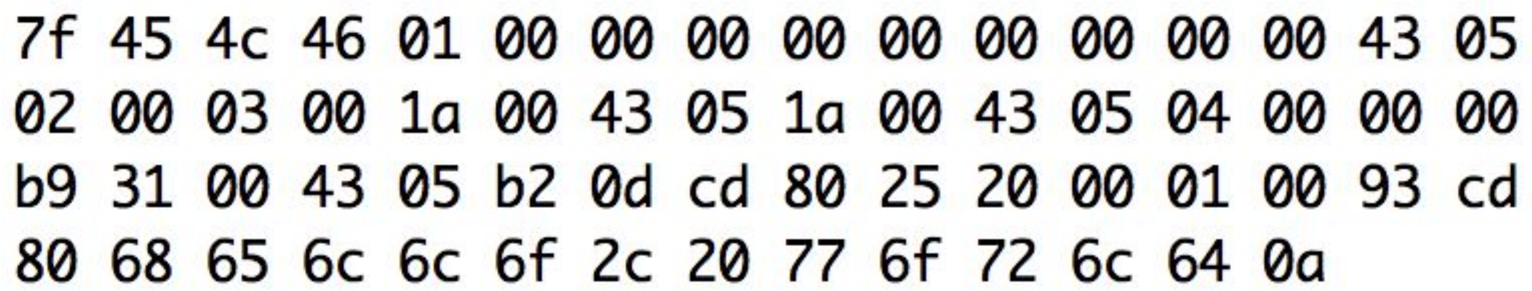
- Hexedit
- Hexdump
- Objdump

Environment for This Class

- **Linux**
  Unlike Windows or OS X, everybody has access to Linux. If you don't have a Linux machine, get VirtualBox. I will be using Ubuntu 19.10. https://www.virtualbox.org/wiki/Linux_Downloads
- **Capstone**
  This is a Python library to perform disassembly. http://www.capstone-engine.org/
- **Ghidra**
  This is a C / C++ / Java library to perform disassembly. https://ghidra-sre.org/

Some Bytes

?



```
7f 45 4c 46 01 00 00 00 00 00 00 00 00 00 43 05
02 00 03 00 1a 00 43 05 1a 00 43 05 04 00 00 00
b9 31 00 43 05 b2 0d cd 80 25 20 00 01 00 93 cd
80 68 65 6c 6c 6f 2c 20 77 6f 72 6c 64 0a
```

?

```
7f 45 4c 46 01 01 01 00 00 00 00 00 00 00 00 00
01 00 03 00 01 00 00 00 00 00 00 00 00 00 00 00
6c 00 00 00 00 00 00 00 34 00 00 00 00 00 28 00
03 00 02 00 00 00 00 00 00 00 00 00 00 00 00 00
31 c0 50 68 2f 2f 73 68 68 2f 62 69 6e 89 e3 50
89 e2 53 89 e1 b0 0b cd 80 00 2e 73 68 73 74 72
74 61 62 00 2e 74 65 78 74 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 0b 00 00 00 01 00 00 00 06 00 00 00
00 00 00 00 40 00 00 00 19 00 00 00 00 00 00 00
00 00 00 00 10 00 00 00 00 00 00 00 01 00 00 00
03 00 00 00 00 00 00 00 00 00 00 00 59 00 00 00
11 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00
00 00 00 00
```

Homework

Homework Due in One Week

1. Get a Linux environment set up for development with:
   a. NASM
   b. C compiler
   c. Python 3
   d. And then pip install capstone

Next Time:
When disassembly fails