PROJECT OVERVIEW

You will exploit a program using buffer overflow attacks to deliver unintended results.

This type of exploit/cyber attack is one of the oldest and most dangerous, effective and persistent type of attacks because it relies on how computers fundamentally work.
ROADMAP

- `cs0330_install buffer`
- Get your unique cookie according to handout directions. This is crucial to your project having a unique solution.
- `./makecookie <cslogin>`
- Read and understand every section of the handout and the x86_64 guide before you start.
- Go through the levels one at a time (they increase in order of difficulty).
THE STACK

- High Memory Address
- Buffer Start
- Low Memory Address
- Stack Grows

- Bottom (origin) of stack
- Buffer grows toward the bottom (higher address)
- Top of stack
ENDIANNESS

- The order of bytes in a word
- Varies by architecture
- Big Endian: most significant byte first
- Little Endian: least significant byte first
  - Department machines are little endian
- Different from bit order
DEMO

Generating Machine Code
DEMO

$ cat exploit | ./hex2raw > exploit-raw.txt

$ ./buffer -u <cslogin> < exploit-raw.txt

$ gdb buffer

$ run -u <cslogin> < exploit-raw.txt
TIPS

● Understand how the stack works.
● Understand what a buffer overflow is
● GDB disassemble command: disas or layout asm
● GDB provides info about registers:
  (gdb) info registers
  (gdb) i r
● si is a GDB command which you can use to step over a single x86-64 instruction
MORE TIPS

● Remember that dept. machines are *little-endian*
  ○ Hex addresses will be in reverse order (addresses increase as you go down the exploit file)

● **hex2raw** expects two-digit hex values separated by whitespace
  ○ To create a byte with a value of 0, you need to specify **00**.

● Use newlines to separate functionality and to make your code readable

● Comment each section with what you are doing (important)
  ○ You can use /* this */ to comment inline
GOOD REFERENCES

(Can also be found on CS33 website)

- [http://cs.brown.edu/courses/csci0330/docs/guides/x64_cheatsheet.pdf](http://cs.brown.edu/courses/csci0330/docs/guides/x64_cheatsheet.pdf)
- [http://cs.brown.edu/courses/csci0330/docs/guides/gdb.pdf](http://cs.brown.edu/courses/csci0330/docs/guides/gdb.pdf)
WHAT YOU’LL LEARN

- How to exploit a buffer overflow vulnerability
- How to avoid buffer overflow vulnerabilities (not trivial)!
- Reading and writing assembly code
- x86 function call conventions
- How buffer overflow attacks take advantage of said x86-64 function call conventions