Week 8 – Dynamic Linking

Introduction to Offensive Security

What is Dynamic Linking?

- "Linking together" functions exported from one program to others
- Examples:
  - libc (provides fopen, gets, printf, puts, and *many* more)
  - openssl
- Opposite is "static" linking
  - Each binary has every function it needs

Why Do We Use It?

- Imagine if every program had to include every function it used…
- Demo

How Does it Work?

- Libraries "export" functions
- Program has a "relocation table", a Global Offset Table (GOT), and function "stubs"
- When a function is hit for the first time, the stub invokes the linker
- The linker resolves the desired function, and replaces the address in the GOT
- The address is now considered "resolved"

Demo

Why is this Useful?

- We now have (essentially) an array of function pointers
- ... at a known, constant address
- ... which we may be able to overwrite
  - GOT can't be read-only otherwise we couldn't link!

GOT Overwrites

- If we have a controlled write and one-shot shell function, we can just overwrite a GOT pointer to that function
  - GOT entry can be resolved or unresolved – doesn't matter

Mitigations

- RELRO
  - RELocations Read-Only
- 2 variants
  - Partial
  - Full
- Partial RELRO doesn't actually do anything against this exploit ☺
- Full RELRO pre-resolves all symbols and marks GOT RO
  - ☹