AR, RG, AZ, JS, AC

Reversing Tips
Good Challenge?

- Yellow Wire – easy
- Green Wire – hard according to Noizeman, but you guys did it in ~1hr
- Blue Wire – you guys did it after class
- Red Wire - ???
We Still Made It Too Easy

- Full Symbols
  - Function names
  - Global variables
- Dynamically linked binary
- Realistic?
Reversing in the Dark

- Whittle away, instruction by instruction
- 100k program \( \approx \) 20k instructions
  - 1MB?
How do you reverse a large, symbol-less program?

- Invite these guys?
Start reversing from easy mode

- Use binary's data against it
  - Left over debug output
- Find and analyze interesting data references
Embedded System Trivia

- 80c32 Tattoo from a Removal Machine
- Serial console returns uppercase input
- Which function did we find first?
Identify the magic bytes we used to find this function:
Answer

- \( \sim 0x20 == 0xdf \)
  - Common bit twiddling trick
  - Ironically, it was the first we tried
- \( 0x6a/0x7b \) would have been good choices too
  - 'a'/'z'+1 \( \rightarrow \) range values
Dynamic Analysis

- Isolate code of interest
  - Flow-graph leading to point of crash
  - Easily Detect Attack Surface Entry Points
- Careful when analyzing malware
Work Backwards

- Set software/hardware breakpoints on interesting data
- Look at stack trace
  - Should reveal relevant functions
Be Creative

• Patch out uninteresting code
• Execute portions of your binary from testing harness
  ∗ Ctypes.CDLL
  ∗ Ruby DLL loading
• ???
TodoHT Demo

- Hit-tracing a static, stripped version of the bomb
Other Demos
Tips on finding super secret key algorithms

- Specialized math instructions
  - Extended instruction set SSE, MMX
- Magic bytes