# Hacker's Invitational

What happens when you invite 30 Universities to hack you?

Presenters

Jonathan Bender Luke Wentz

8/21/08



### .Overview

#### Recap

- CTF
- Tools

#### The Monster (packet capture)

- Tools
- Deciphering
- WTF just happened?

#### Exploits

- Misconfiguration
- Injection



### .Recap

#### Capture the Flag (CTF)

- UCSB iCTF
- CIPHER4

#### Tools

- Network Sniffing (Wireshark)
- Proxies (Paros, Burp)
- Disassembly (IDA)
- Packet Injection (Nemesis)
- Firewall (IPTables, etc...)
  - TCP Wrappers



### .The\_Monster

Capture from UCSB iCTF 1.2 GB in volume, 5.0MB chunks

Contains:

- HTTP
- SSH
- Media Streaming
- Scans
- etc... (nearly every dirty trick in

# .More\_Tools

#### Ngrep

- Use grep against pcap captures or live traffic
- Pattern matching against packet contents and header data

#### Pcapmerge

Stitch multiple pcap files into a single file



### .Exploits

# Focused on UCSB iCTF and CIPHER4

#### **Application Level Attacks**

- 1 Misconfiguration
- 2. Injection
- 3. Buffer Overflow
- 4. Disassembly



•

### .Configuration

Web Servers - Admin Sites **File Servers** - Permissions **Custom Applications**  Respect all of the above Least Privelage



### .Injection

#### What is vulnerable?

- Any web script that accepts input
- Scripts that pass input into other programs
  - SQL
  - Shell
- Why?
- What can it do?



### .Injection/Protection

Know your tools and systems
Determine special characters
Determine meaningful strings
Develop input sanitization
Shell

- Semicolons, slashes, double dots, etc...
- SQL
  - Quotes, semicolons, dashes, etc...



## .Buffer\_Overflow

What is vulnerable?Anything that accepts inputHow it works:

- Input is larger than buffer
- Input typically contains code
- Stack return pointer overwritten
- New pointer points to arbitrary code



### Buffer Overflow/Protecti Know your languages Know your program Validate input size before moving Use safe methods, arbitrary size Use safe data types Compile with stack protections A safeguard, not a fix



### .Disassembly

What is vulnerable? - Almost any compiled code Why?

- Determine makeup, resources
- Determine code flow
- Find potential exploits

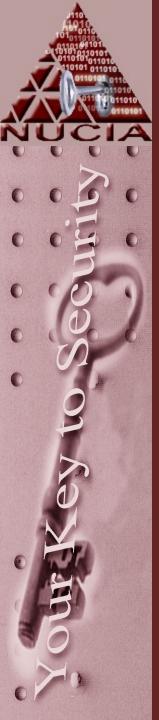
Also known as reverse engineering



### .Disassembly/Protection

What can be done?

- Very little
- Obfuscation (adds complexity)
- Why?
  - It is not a direct exploit
  - Disassembly does not break anything
  - Looking at code that is there



## .More\_Tools -part 2

IDA Pro (Disassembly & Debugging) GDB, MSDB (Debugging) DTrace (Tracing library) STrace (Syscall tracing library) LTrace (Library call tracing)

Other



### .Contacts & .Bookmarks

#### Jonathan Bender

- jbender@nucia.unomaha.edu

#### Luke Wentz

- lwentz@nucia.unomaha.edu

#### NUCIA Website

- http://nucia.unomaha.edu

#### UCSB iCTF

- http://cs.ucsb.edu/~vigna/CTF