Mark Shtern

INJECTION

Example

- Code from old IRIX login screen:

```
char buf[1024];
snprintf(buf, "system lpr -P %s", user_input,
sizeof(buf)-1);
system(buf);
```

- Attack:

FRED; xterm&

Injections

- Injection problems occur when untrusted data is
  - Placed into trusted data
  - Passed to some sort of compiler or interpreter
  - Treated as something other than data

Impact

- Privilege escalation
- Information Leakage
- Control of a victim machine
- Denial of Service
- Access to victim resources
- Software unlocking

Attack ideas

- Code injection into script (eval, execv)
- Module Substitution (DLL injection, Java class injection)
- System call, external command, environment variable substitution
- Debug mode

DLL injection

- Linker dynamically load modules
- Linux

  LD_PRELOAD="./test.so" prog
- Windows

  HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows\AppInit_DLLs

Substitute external command

- Example

  system("ls -l");

- Execution depends on environment variable

Debug mode

- Application allows to execute diagnostic command

Java class injection

- Java VM loads class dynamically
- Attacker may substitute any Java classes
  - Develop new class with the same signature
  - Modify existing class

Modify existing class

- Decompile Java class
- Modify source code
- Compile modified source code
- Overwrite original class

Countermeasures

- Validate user input
- Do not store confidential information in the source files
- Deploy obfuscated code
- Use least privileges design concept
- Do not pass user input to an interpreter
- Digitally sign production code

Tools

- Debuggers (gdb, WinDgb)
- Java decompilers (Jode, JD)
- Java code obfuscators (Jode, ProGuard)
- IDE (eclipse, netbeans, Visual Studio 2008)