An Architectural Approach to Preventing Code Injection Attacks

Ryan Riley, Xuxian Jiang, Dongyan Xu
Purdue University, George Mason University
To appear in the proceedings of DSN-DCCS 2007

Problem #1
Most computers are inherently vulnerable to code injection attacks
• Their von Neumann memory architecture allows data to be executed as code
• Attackers can inject malicious code using data sources (files, the network, etc.) and execute it

von Neumann Architecture
Physical Memory
Instructions
Data
...
Instruction/Data Fetch
Processor

Harvard Architecture
Instruction Memory
Processor

Data Memory
Instructions
Instructions
...

Instruction Fetch
Data Fetch

Problem #2
No one will follow Solution #1
• Intel/AMD/IBM all use von Neumann architectures
• Grad students don’t have much clout with those three
• This fundamental change would take a long time

Solution #1
Use a Harvard memory architecture
• Completely separates instructions and data
• The processor cannot even load data as instructions
• Code injected as data can never be run

Solution #2
Create a virtual Harvard architecture
• Exploit tricks in the Intel processor related to memory management
• Have the operating system build the architecture on a per process basis
• Implement it in Linux 2.6.13
• Test it against a suite of attacks (All thwarted)

Performance Graph

Virtual Harvard Architecture

Memory

Instructions
Instructions
Data
Data

Instruction Fetch
Data Fetch

Processor