Stack Overflow

Faculty Workshop on Cyber Security

May 23, 2012

Goals

- Learn to hack into computer systems using buffer overflow
- Steal sensitive data
- Crash computer programs
- Lay waste to systems throughout the world

Real Goals

- Learn how “black hat” hackers can corrupt computer programs
- Understand how programs should be written to protect against attacks

Educational Goals of Exercise

- Students will be able to explain how a program can do something that is not in the written program (j) An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices
- Students will be able to create exploit data to force a program to execute code that was not in the original program
- Students will be able to explain the memory layout of a program stack (i) An ability to use current techniques, skills, and tools necessary for computing practice

National Cyber Alert System

Technical Cyber Security Alert TA09-161A

Adobe Acrobat and Reader Vulnerabilities

Original release date: June 10, 2009

Last revised: --

Source: US-CERT

Systems Affected

* Adobe Reader versions 9.1.1 and earlier
* Adobe Acrobat versions 9.1.1 and earlier

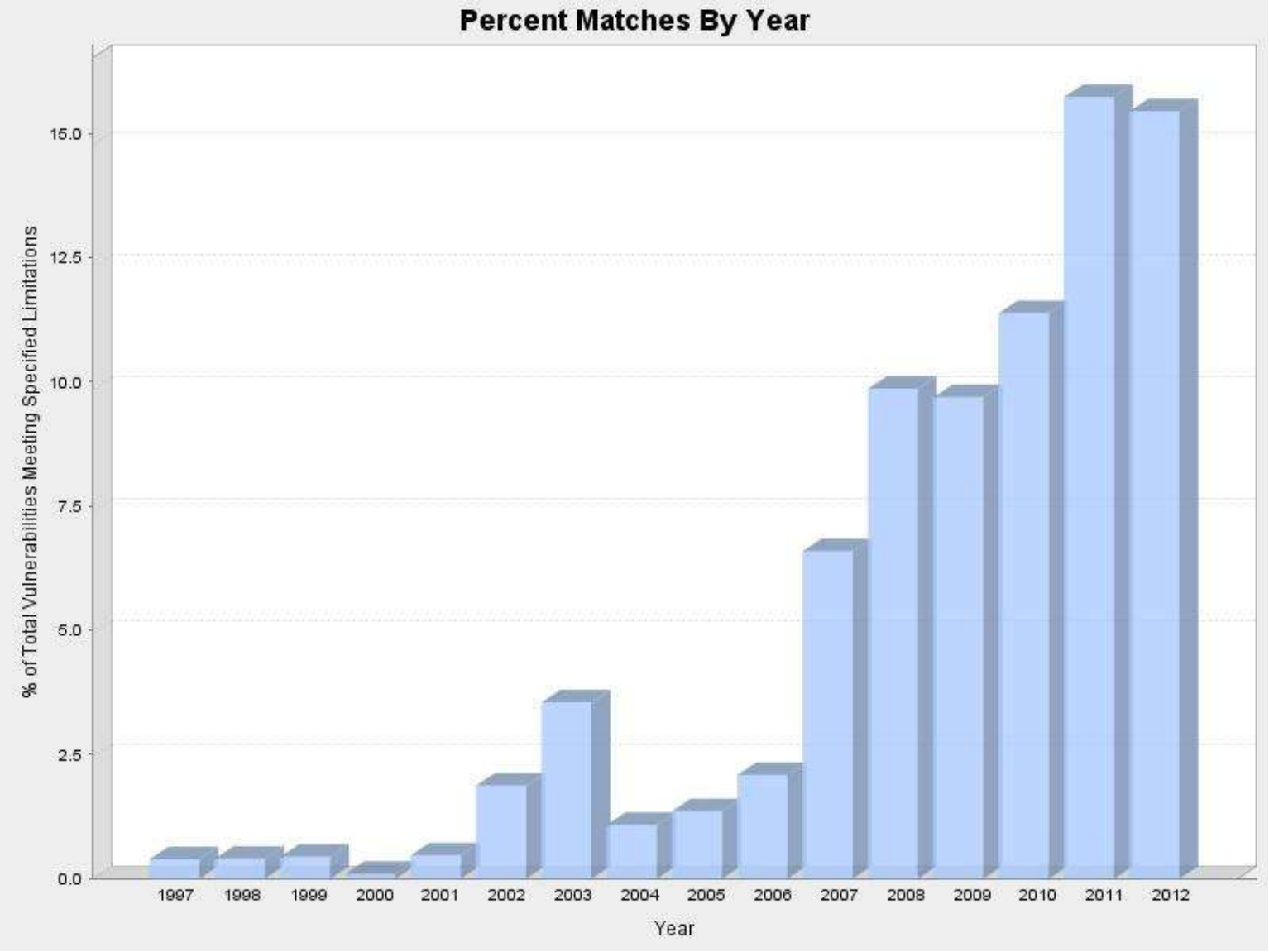
Overview

Adobe has released Security Bulletin APSB09-07, which describes several **buffer overflow** vulnerabilities that could allow a remote attacker to execute arbitrary code.

Buffer Overflows Are A Major Threat

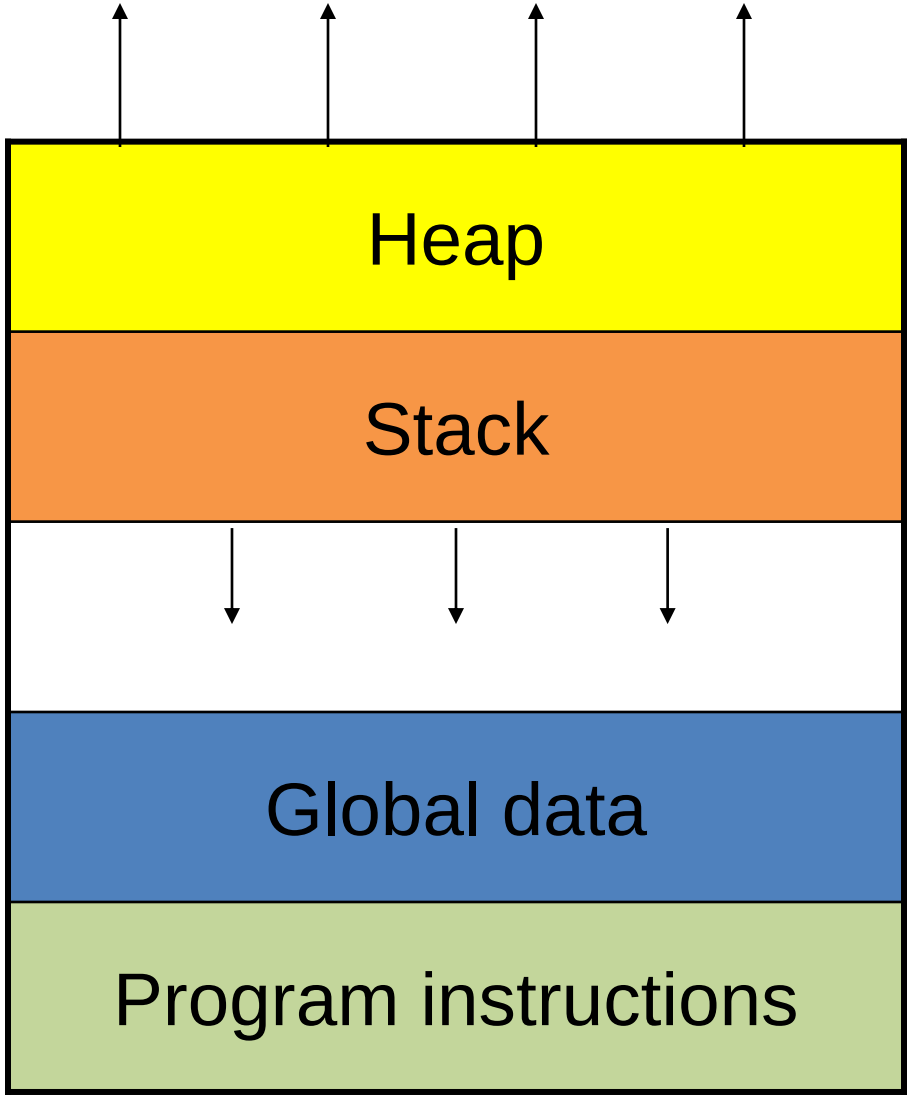
- Buffer overflows are a major security vulnerability.
- When a security alert contains the phrase **"The most severe of these vulnerabilities allows a remote attacker to execute arbitrary code."**, the underlying problem is probably a buffer overflow.
- The Morris worm *(the first Internet worm)* spread in part by exploiting a stack buffer overflow in the Unix finger server.
- Many students do not know what they are.

Buffer Overflows as Percent of Total Vulnerabilities



source: DHS National Cyber Security Division/US-CERT National Vulnerability Database

Program Memory Organization



Intel method

Stack Format

High addresses

Purpose	Values
Parameters	The address of an array or a value if simple type
Return address	The address in the calling program where the function should return
Frame pointer	The address on the stack of the previous frame
Local variables	Local variable data

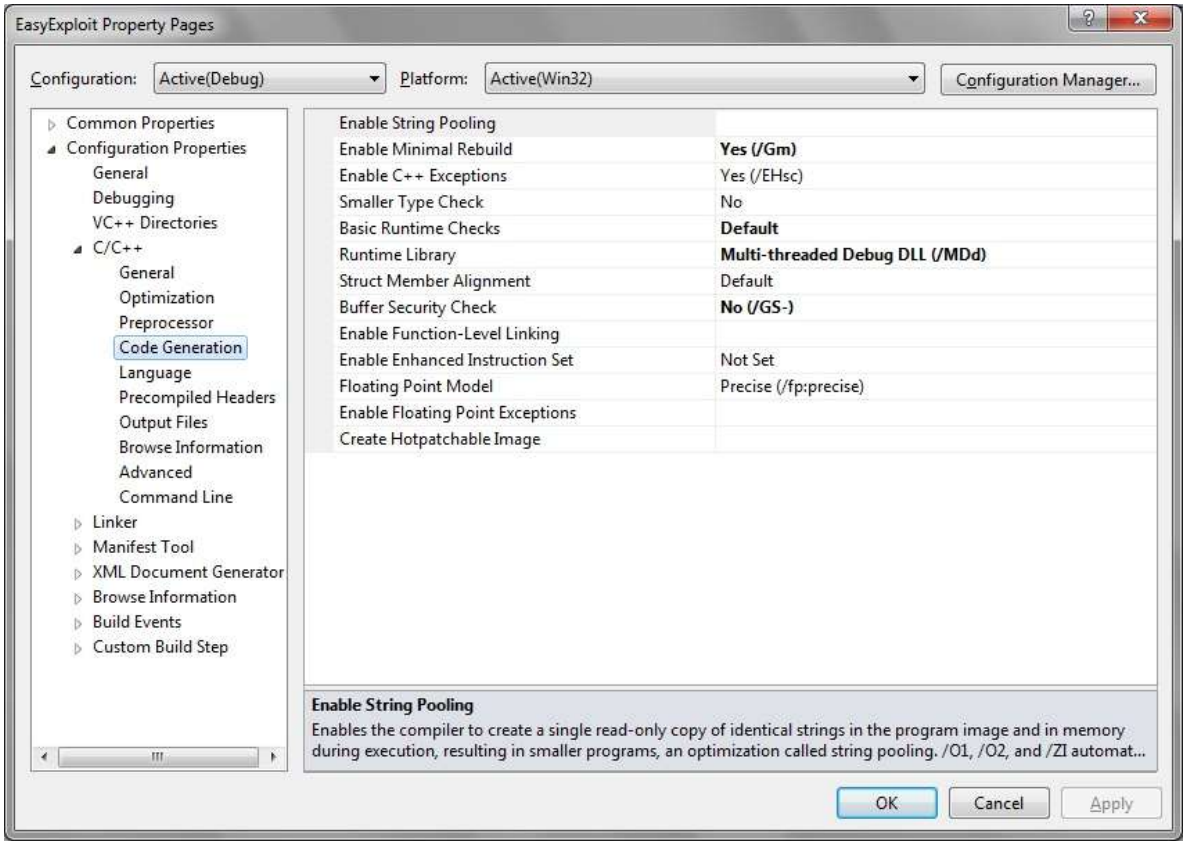
Low addresses

System Defenses

- The Microsoft Visual C++ Express compiler takes several steps to avoid stack overflow
  - It uses stack canaries to detect overflow
- Microsoft Vista, Windows 7 and Linux start the stack at a random address that is different for each execution of the same program

Turning Off the Defenses

- To more easily attack the stack, we can turn off the Visual C++ security features.
- Select Project → Properties → Configuration Properties → C/C++ → Code Generation



Turning Off the Defenses

- To more easily attack the stack, we can turn off the Visual C++ security features.
- Set **`Basic Runtime Checks`** to **`default`**
- Set **`Buffer Security Check`** to **`No`**
- When using Visual C++ Express version 2010, select Tools → Settings → Expert Settings

- With a little more effort, you can do the exercise without turning off the overflow checks

Overflow Exercise

- We created an assignment that steps the student through the creation of a file to cause an arbitrary code overflow exploit of a simple vulnerable program
- The students use the Microsoft C++ debugger to inspect the memory and program addresses

Using the Debugger

- If you hover the mouse over a method name, it will give you the start address of the method
- You can also find the address of variables
- Once you know the start address of a method, the return address of a function called in that method will be a little bit higher.

Endian

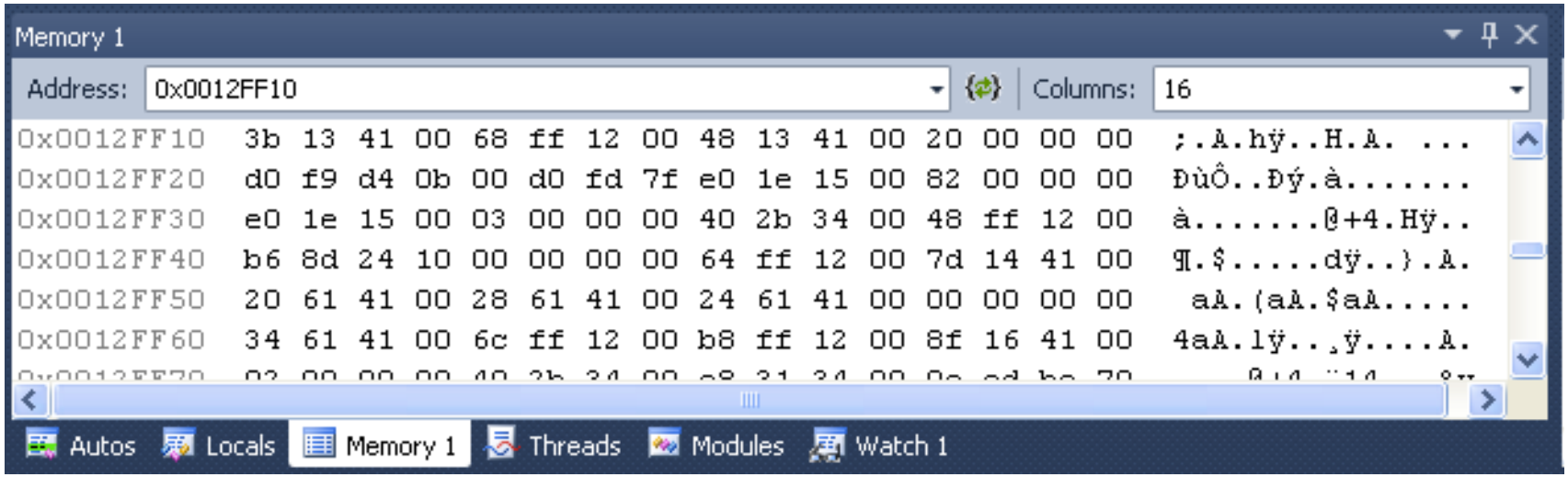
- The Intel Pentium is a Little Endian machine
- The bytes of numerical data, including addresses, is stored in reverse order
- The hexadecimal value 0X1234ABCD will appear in a memory dump as

CD AB 34 12

- When entering a hexadecimal number that will be stored in RAM, you have to reverse the bytes

Memory View

- Once you know the address of a variable on the stack, you can look at that variable and the memory around it with the debugger.
- Look for a likely return address or other data values.



Creating Attack Text

- It is helpful to have a text editor that allows you to input hexadecimal values so you can easily enter machine language
- A free hex editor is Frhed, available from http://frhed.sourceforge.net

Virtual Machine

- The exercise runs in a virtual machine with Windows XP and Visual C++ express

  Start **OverflowVM** virtual machine

- The project has already been created in Visual C++
- Students need to collect machine addresses and then edit a data file to perform the exploit