Defending Computer Networks
Lecture 3: More On Vulnerabilities

Stuart Staniford

Adjunct Professor of Computer Science

Logistics

- Website now up
  - http://www.cs.cornell.edu/courses/cs5434/2014fa/
- CMS requested
- Piazza set up (will email invites once CMS up)
- First Homework Today
  - Find on class website later today
  - Due next Friday 9/12 at midnight
    - Via CMS
- T.A. Intros

More Logistics

CS 5434 - Defending Computer Networks - Fall 2014

Summary | Lectures | **Readings** | Homework | Projects

#	Author	Title/Link
1	Aleph1	*Smashing the Stack for Fun and Profit*
2	Matt Connover	*w00w00 on heap overflows*
3	Scut	*Exploiting Format String Vulnerabilities*
4	Blexim	*Basic Integer Overflows*
5	Mitre	*Common Weakness Enumeration*
6	Steve Christey	*Unforgivable Vulnerabilities*
7	Christey et al	*Structured CWE Descriptions*
8	Cowan et al	*StackGuard: Automatic Adaptive Detection and Prevention of Buffer-Overflow Attacks*
9	Shacham et al	*On the Effectiveness of Address-Space Randomization*
10	Hovav Shacham	*The Geometry of Innocent Flesh on the Bone*

Main Goals for Today

- Understand format string vulnerabilities
- Introduction to CVE/CWE - enumerating and classifying vulnerabilities
- The economic/social big picture: why is the Internet riddled with vulnerabilities?
- Understand stack canary defenses against buffer overflows

Interesting News This Time

NATO Set to Ratify Pledge on Joint Defense in Case of Major Cyberattack

By **DAVID E. SANGER** AUG. 31, 2014

BRUSSELS — When President Obama meets with other NATO leaders later this week, they are expected to ratify what seems, at first glance, a far-reaching change in the organization's mission of collective defense: For the first time, a cyberattack on any of the 28 NATO nations could be declared an attack on all of them, much like a ground invasion or an airborne bombing.

The most obvious target of the new policy is Russia, which was believed behind computer attacks that disrupted financial and telecommunications systems in Estonia in 2007 and Georgia in 2008, and is believed to have used them in the early days of the Ukraine crisis as well.

But in interviews, NATO officials concede that so far their cyberskills are limited at best.

While NATO has built a gleaming new computer security center, and now routinely runs computer exercises, it possesses no cyberweapons of its own — and, apparently, no strategy for how it might use the weapons of member states to strike back in a computer conflict. In fact, its most powerful members, led by the United States and Britain, have spent billions of dollars on secret computer offensive programs — but they have declined so far to tell NATO leaders what kind of weapons they might contribute in a NATO-led computer conflict.

Refresh – System vulnerabilties

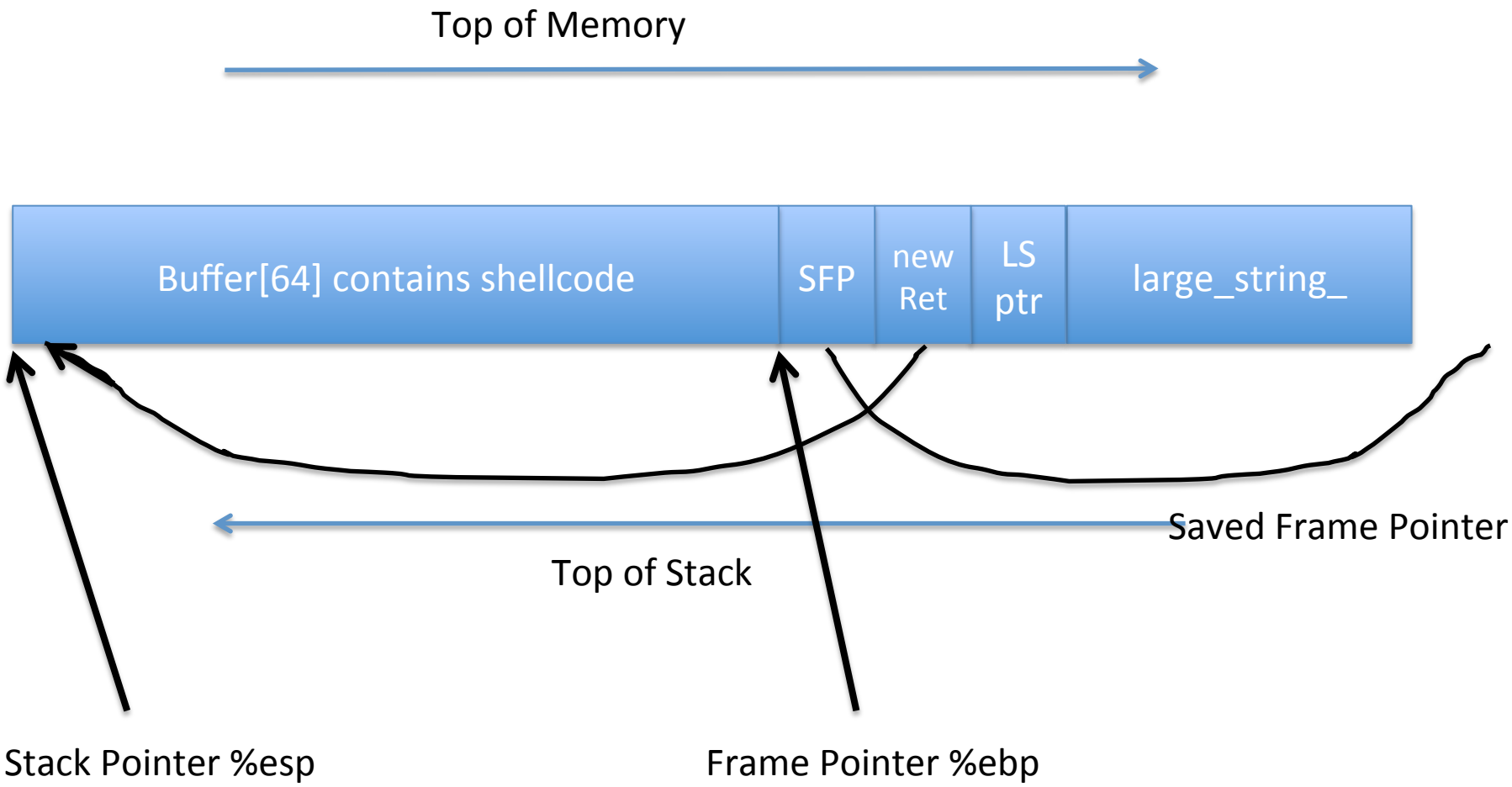
- sprintf(command, “blah %s blah”, userInput);
- system(command);

Refresh: Example 2

```
void myFunc(char *str)
{
  char buffer[64];
  strcpy(buffer, str);
}

int main(int argc, char* argv[])
{
  char large_string[256];
  int i;
  for( i = 0; i < 255; i++)
    large_string[i] = 'A';
  myFunc (large_string);
}
```

Refresh: Stack On Exploit



Format String Vulnerabilities

- Class of vulnerabilities in C printf/sprintf/etc
  - Discovered at end of 1990s
  - Almost thirty years after C invented
- Also can affect syslog(), warn(), err()
- Core issue is when the format string is (partially) user supplied, not static
  - printf(userData,…) is bad
  - printf("%s", userData,…) doesn't have the issue
- Note that '…' arguments are on the stack
- And format string controls powerful functionality to do stuff with the printf arguments (ie the stack)
- Bypass any compile time checks when user supplied

printfVulnerability.c

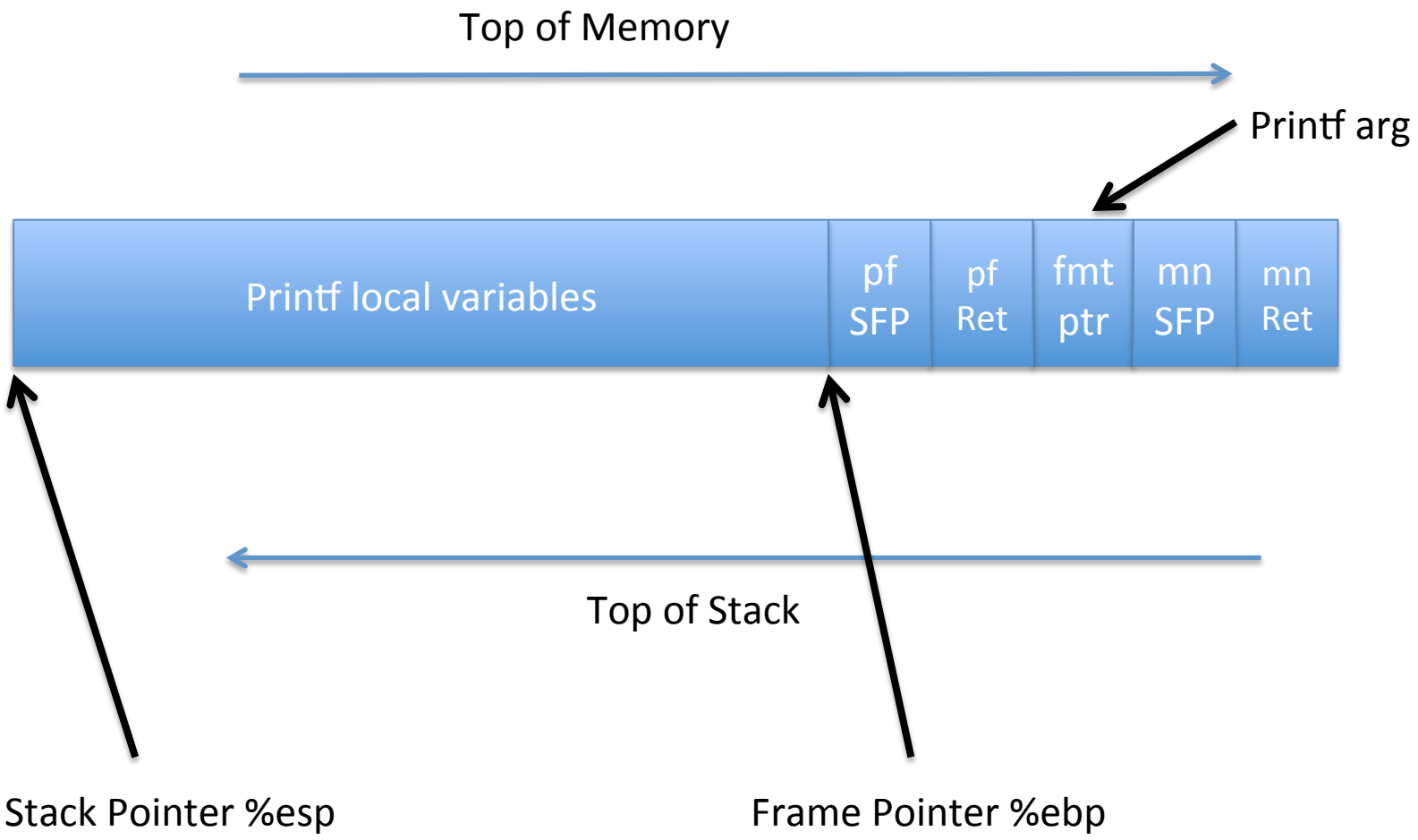
```
#include <stdio.h>

int main(int argc, char* argv[])
{
  printf(argv[1]);
  return 0;
}
```

So let’s try…

- ./printfVulnerability foo
- ./printfVulnerability “foo \n”
- ./printfVulnerability “%08x.%08x.%08x.%08x ”

Stack in printfVulnerability



Still in the no-defense, old-style, slightly fictionalized view of the world

The Really Big Problem

The ‘%n’ format directive. From ‘man 3 printf’:

> **n** The number of characters written so far is stored into the integer indicated by the int * (or variant) pointer argument. No argument is converted.

This allows us to write on the stack at a location we can control! by doing ./printfVulnerability “%08x.%08x… %n” we can move around the stack position where the %n will write to.

By doing “aaaaa…%08x.%08x… %n” we can control what number is written into that stack address.

By doing this repeatedly with small width fields, we can write one byte at a time, and overwrite an address (eg a saved IP)

Common Vulnerabilities and Exposures List (CVE)

- Initiative by Mitre Corp to create a common dictionary of known vulnerabilities
  - US govt funded
- Now an industry standard
- Guided by an editorial board from across industry/academia/government
- Excellent Place to Start Looking for Publicly Known Vulnerabilities in something
  - http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=excel

‘Excel’ Vulnerabilities

Search Results

There are **201** CVE entries that match your search.

Name	Description
CVE-2012-5672	Microsoft Excel Viewer (aka Xlview.exe) and Excel in Microsoft Office 2007 (aka Office 12) allow remote attackers to cause a denial of service (read access violation and application crash) via a crafted spreadsheet file, as demonstrated by a .xls file with battery voltage data.
CVE-2012-4233	LibreOffice 3.5.x before 3.5.7.2 and 3.6.x before 3.6.1, and OpenOffice.org (OOo), allows remote attackers to cause a denial of service (NULL pointer dereference) via a crafted (1) odt file to vcllo.dll, (2) ODG (Drawing document) file to svxcorelo.dll, (3) PolyPolygon record in a .wmf (Window Meta File) file embedded in a ppt (PowerPoint) file to tllo.dll, or (4) xls (Excel) file to scfiltlo.dll.
CVE-2012-2543	Stack-based buffer overflow in Microsoft Excel 2007 SP2 and SP3 and 2010 SP1; Office 2011 for Mac; Excel Viewer; and Office Compatibility Pack SP2 and SP3 allows remote attackers to execute arbitrary code via a crafted spreadsheet, aka "Excel Stack Overflow Vulnerability."
CVE-2012-1887	Use-after-free vulnerability in Microsoft Excel 2003 SP3, 2007 SP2 and SP3, and 2010 SP1, and Office 2008 and 2011 for Mac, allows remote attackers to execute arbitrary code via a crafted spreadsheet, aka "Excel SST Invalid Length Use After Free Vulnerability."
CVE-2012-1886	Microsoft Excel 2003 SP3, 2007 SP2 and SP3, and 2010 SP1; Excel Viewer; and Office Compatibility Pack SP2 and SP3 allow remote attackers to execute arbitrary code or cause a denial of service (memory corruption) via a crafted spreadsheet, aka "Excel Memory Corruption Vulnerability."
CVE-2012-1885	Heap-based buffer overflow in Microsoft Excel 2003 SP3, 2007 SP2 and SP3, and 2010 SP1; Office 2008 and 2011 for Mac; and Office Compatibility Pack SP2 and SP3 allows remote attackers to execute arbitrary code via a crafted spreadsheet, aka "Excel SerAuxErrBar Heap Overflow Vulnerability."
CVE-2012-1847	Microsoft Excel 2003 SP3, 2007 SP2 and SP3, and 2010 Gold and SP1; Office 2008 and 2011 for Mac; Excel Viewer; and Office Compatibility Pack SP2 and SP3 do not properly handle memory during the opening of files, which allows remote attackers to execute arbitrary code via a crafted spreadsheet, aka "Excel Series Record Parsing Type Mismatch Could Result in Remote Code Execution Vulnerability."
CVE-2012-0185	Heap-based buffer overflow in Microsoft Excel 2007 SP2 and SP3 and 2010 Gold and SP1, Excel Viewer, and Office Compatibility Pack SP2 and SP3 allows remote attackers to execute arbitrary code via a crafted spreadsheet that triggers incorrect handling of memory during opening, aka "Excel MergeCells Record Heap Overflow Vulnerability."
CVE-2012-0184	Microsoft Excel 2003 SP3, 2007 SP2 and SP3, and 2010 Gold and SP1; Office 2008 and 2011 for Mac; Excel Viewer; and Office Compatibility Pack SP2 and SP3 do not properly handle memory during the opening of files, which allows remote attackers to execute arbitrary code via a crafted spreadsheet, aka "Excel SXLI Record Memory Corruption Vulnerability."
CVE-2012-0143	Microsoft Excel 2003 SP3 and Office 2008 for Mac do not properly handle memory during the opening of files, which allows remote attackers to execute arbitrary code via a crafted spreadsheet, aka "Excel Memory Corruption Using Various Modified Bytes Vulnerability."
CVE-2012-0142	Microsoft Excel 2003 SP3, 2007 SP2 and SP3, and 2010 Gold and SP1; Office 2008 for Mac; Excel Viewer; and Office Compatibility Pack SP2 and SP3 do not properly handle memory during the opening of files, which allows remote attackers to execute arbitrary code via a crafted spreadsheet, aka "Excel File Format Memory Corruption in OBJECTLINK Record Vulnerability."
CVE-2012-0141	Microsoft Excel 2003 SP3, 2007 SP2 and SP3, and 2010 Gold and SP1; Office 2011 for Mac; Excel Viewer; and Office Compatibility Pack SP2 and SP3 do not properly handle memory during the opening of files, which allows remote attackers to execute arbitrary code via a crafted spreadsheet, aka "Excel File Format Memory Corruption Vulnerability."

Example Entry

CVE-ID	
CVE-2012-2543	Learn more at National Vulnerability Database (NVD) • Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings

Description

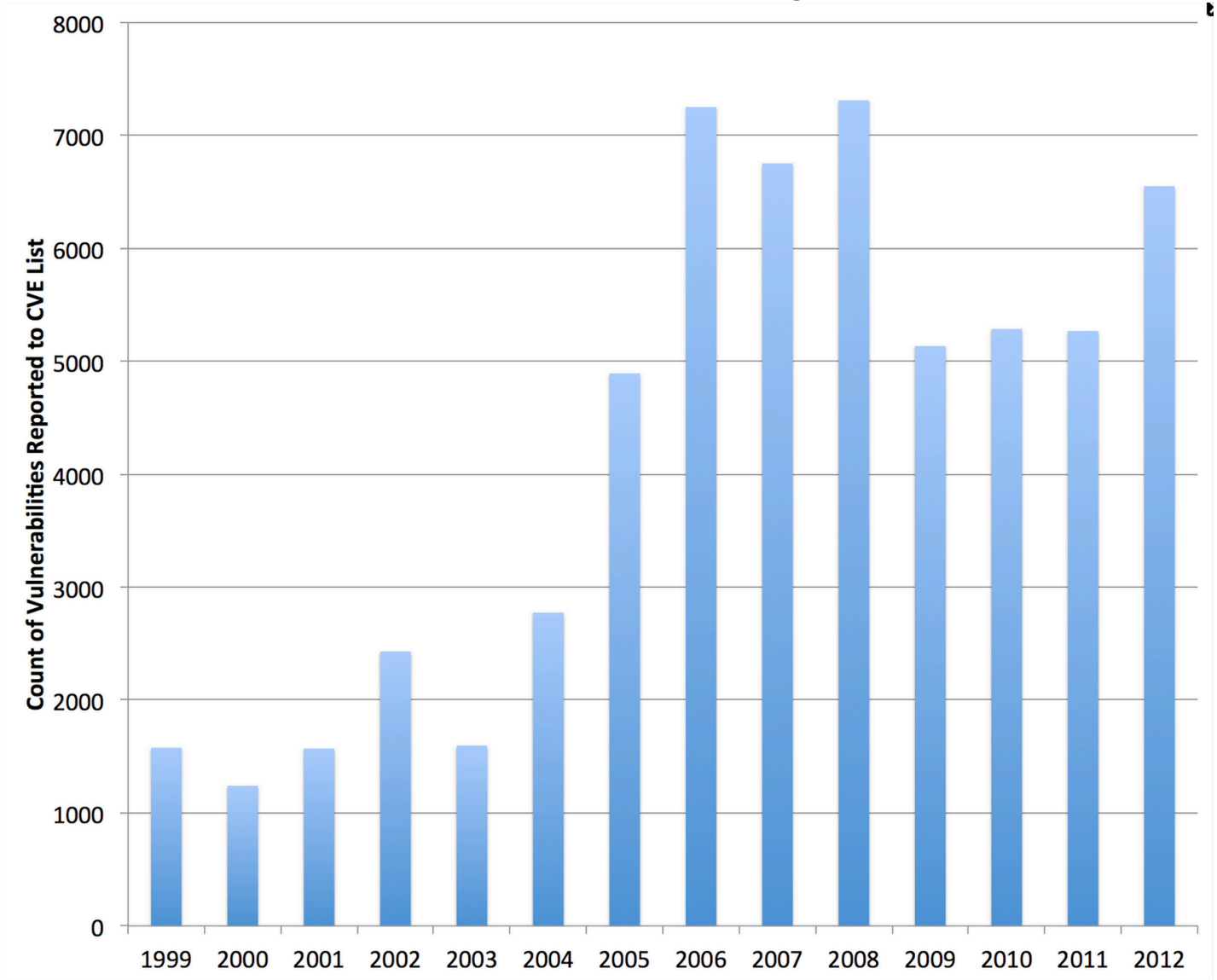
Stack-based buffer overflow in Microsoft Excel 2007 SP2 and SP3 and 2010 SP1; Office 2011 for Mac; Excel Viewer; and Office Compatibility Pack SP2 and SP3 allows remote attackers to execute arbitrary code via a crafted spreadsheet, aka "Excel Stack Overflow Vulnerability."

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

- MS:MS12-076
- URL:http://technet.microsoft.com/security/bulletin/MS12-076
- CERT:TA12-318A
- URL:http://www.us-cert.gov/cas/techalerts/TA12-318A.html
- BID:56431
- URL:http://www.securityfocus.com/bid/56431
- SECTRACK:1027752
- URL:http://www.securitytracker.com/id?1027752

CVE Counts By Year



Is That All The Vulnerabilities?

- No!
  - Anecdote/single datapoint (8000 vs 122)
- Basically, we have no idea how many software vulnerabilities exist in total
  - Probably millions at least,
  - Probably not billions

Common Weakness Enumeration(CWE)

- More recent effort by Mitre
  - cwe.mitre.org
- Idea is to classify vulnerabilities into general types
  - See the recurring patterns
  - Prioritize what's most important
  - Learn not to generate more and more of these things
- Excellent place to look for general issues when you get into a new domain

CWE Top 25

Rank	Score	ID	Name
[1]	93.8	CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
[2]	83.3	CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
[3]	79.0	CWE-120	Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')
[4]	77.7	CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
[5]	76.9	CWE-306	Missing Authentication for Critical Function
[6]	76.8	CWE-862	Missing Authorization
[7]	75.0	CWE-798	Use of Hard-coded Credentials
[8]	75.0	CWE-311	Missing Encryption of Sensitive Data
[9]	74.0	CWE-434	Unrestricted Upload of File with Dangerous Type
[10]	73.8	CWE-807	Reliance on Untrusted Inputs in a Security Decision
[11]	73.1	CWE-250	Execution with Unnecessary Privileges
[12]	70.1	CWE-352	Cross-Site Request Forgery (CSRF)
[13]	69.3	CWE-22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
[14]	68.5	CWE-494	Download of Code Without Integrity Check
[15]	67.8	CWE-863	Incorrect Authorization
[16]	66.0	CWE-829	Inclusion of Functionality from Untrusted Control Sphere
[17]	65.5	CWE-732	Incorrect Permission Assignment for Critical Resource
[18]	64.6	CWE-676	Use of Potentially Dangerous Function
[19]	64.1	CWE-327	Use of a Broken or Risky Cryptographic Algorithm
[20]	62.4	CWE-131	Incorrect Calculation of Buffer Size
[21]	61.5	CWE-307	Improper Restriction of Excessive Authentication Attempts
[22]	61.1	CWE-601	URL Redirection to Untrusted Site ('Open Redirect')
[23]	61.0	CWE-134	Uncontrolled Format String
[24]	60.3	CWE-190	Integer Overflow or Wraparound
[25]	59.9	CWE-759	Use of a One-Way Hash without a Salt

940 CWE's in total – another 37 pages like this

Example From a Different Domain

J2EE Bad Practices: Direct Management of Connections

Weakness ID: 245 *(Weakness Variant)* **Status:** Draft

Description

Description Summary

The J2EE application directly manages connections, instead of using the container's connection management facilities.

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

Java

Common Consequences

Scope	Effect
Other	**Technical Impact:** *Quality degradation*

Demonstrative Examples

Example 1

In the following example, the class DatabaseConnection opens and manages a connection to a database for a J2EE application. The method openDatabaseConnection opens a connection to the database using a DriverManager to create the Connection object conn to the database specified in the string constant CONNECT_STRING.

Example Language: **Java** *(Bad Code)*

```
public class DatabaseConnection {
  private static final String CONNECT_STRING = "jdbc:mysql://localhost:3306/mysqldb";
  private Connection conn = null;

  public DatabaseConnection() {
  }

  public void openDatabaseConnection() {
    try {
      conn = DriverManager.getConnection(CONNECT_STRING);
    } catch (SQLException ex) {...}
  }

  // Member functions for retrieving database connection and accessing database
  ...
}
```

The use of the DriverManager class to directly manage the connection to the database violates the J2EE restriction against the direct management of connections. The J2EE application should use the web application container's resource management facilities to obtain a connection to the database as shown in the following example.

Why So Many Vulnerabilities?

1. Writing secure code is hard
2. Software engineers are poorly trained in security (but not you guys!)
3. Economic incentives at software companies do not prioritize doing things right

Writing Secure Code is Hard

- 940 CWEs (and counting)
- Large numbers of different categories of things to do wrong
  - Hard to know about all of them
- Humans are error prone at best
  - Most of us write a noticeable number of bugs/kloc
  - Significant fraction of bugs turn out to be exploitable
  - Testing all code-paths is both theoretically and practically impossible

Software Engineers Untrained in Security

You are here →

Rank	School name	Score
#1	Carnegie Mellon University Pittsburgh, PA	5.0
#1	Massachusetts Institute of Technology Cambridge, MA	5.0
#1	Stanford University Stanford, CA	5.0
#1	University of California–Berkeley Berkeley, CA	5.0
#5	Cornell University Ithaca, NY	4.6
#5	University of Illinois–Urbana-Champaign Urbana, IL	4.6
#7	University of Washington Seattle, WA	4.5
#8	Princeton University Princeton, NJ	4.4
#8	University of Texas–Austin Austin, TX	4.4
#10	Georgia Institute of Technology Atlanta, GA	4.3

Source: http://grad-schools.usnews.rankingsandreviews.com/best-graduate-schools/top-science-schools/computer-science-rankings

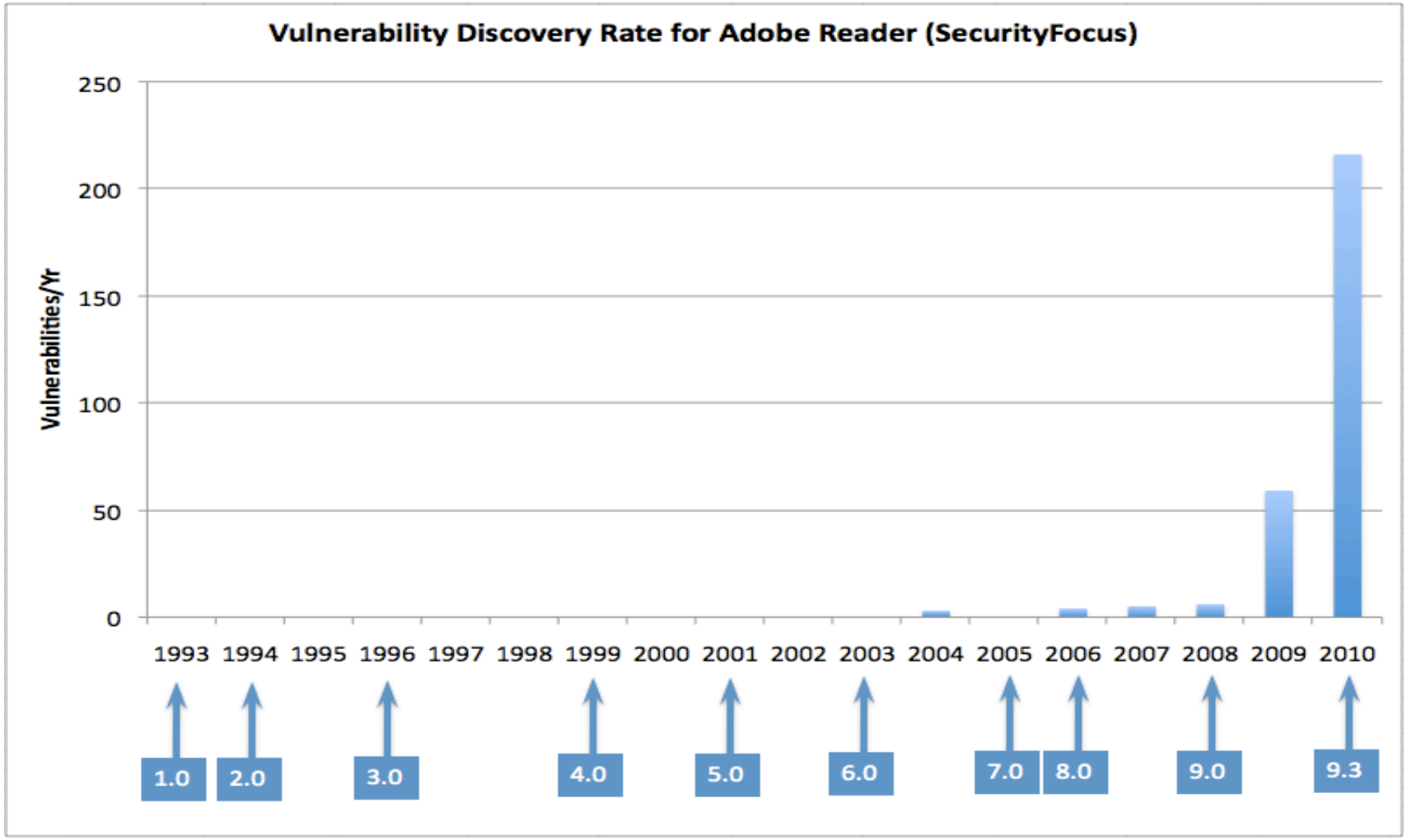
Undergraduate Security Course @Top 10

	Available	**Required**
Carnegie Mellon	Yes	No
MIT	Yes	No
Stanford	Yes	No
Berkeley	Yes	No
Cornell	Yes	No
Illinois/Urbana-Champaign	Yes	No
University of Washington	Yes	No
Princeton	Yes	No
UT Austin	Yes	No
Georgia Tech	Yes	No
Purdue*	Yes	No
UC Davis*	Yes	No

Economic Incentives

- Most tech market categories have a winner-take-all structure
  - Microsoft in operating systems and office suites
  - Google in search
  - Apple/Google in smartphones
  - Oracle in databases
  - Cisco in routers/switches
- Company executives/VCs know this
  - So very strong pressure to ship code fast
    - Dominate the market, then solve problems later

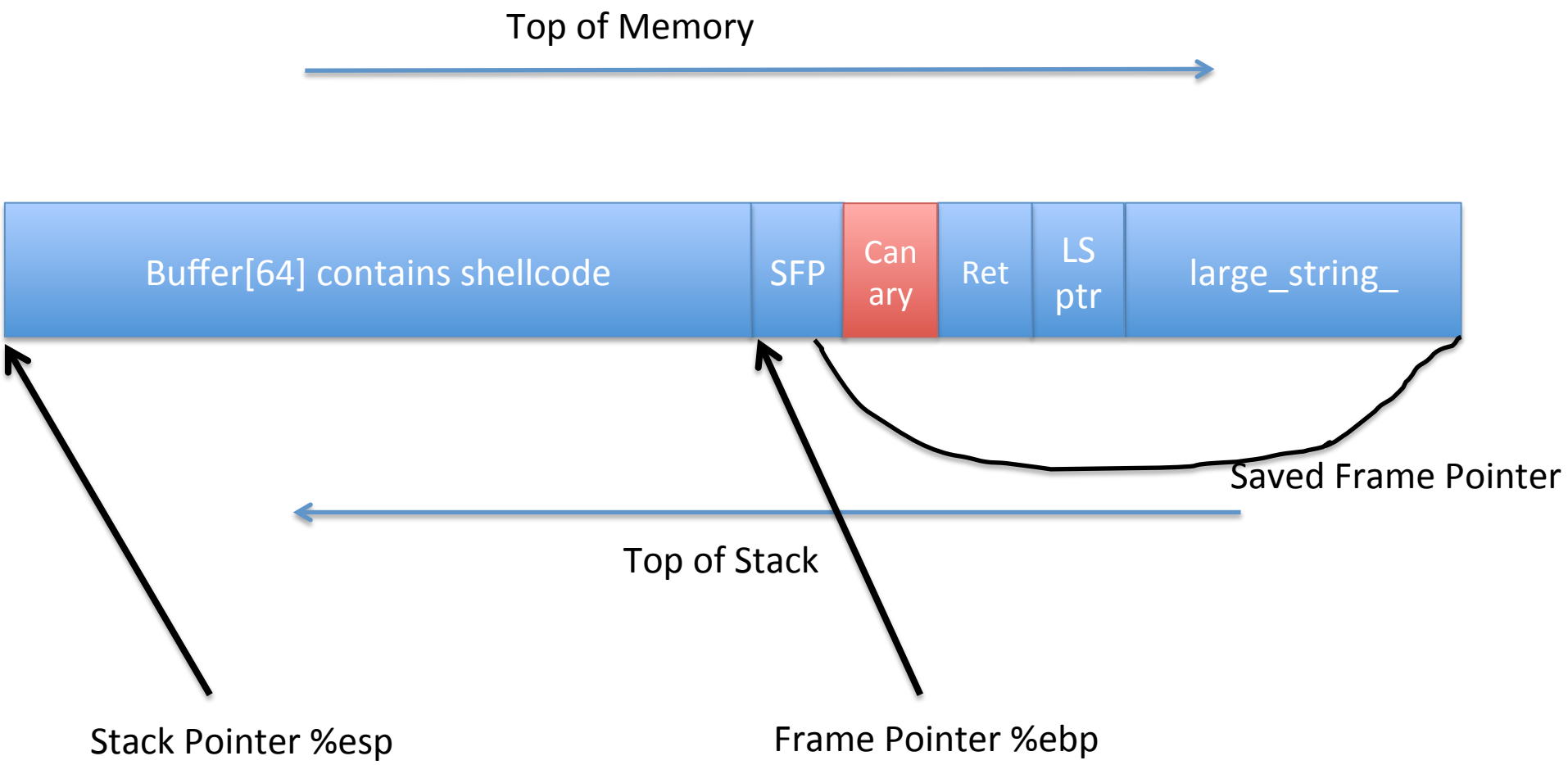
Typical Life of a Vulnerability



Economic Incentives

- Not writing vulnerabilities in the first place is
  - Hard
  - Therefore expensive and slow
- Finding vulnerabilities takes lots of QA/test
  - Also, expensive and slow
- So companies have a strong incentive to not worry about it **too** much
  - Fix it later, when it becomes a PR problem

Canary-based Overflow Defense



Not invincible. Eg http://phrack.org/issues.html?issue=56&id=5