IS 2620

Developing Secure Systems

Introduction

Aug 30, 2018

James Joshi

Professor

School of Computing and Information

Contact

- James Joshi
- 706A, IS Building
- Phone: 412-624-9982
- E-mail: jjoshi@pitt.edu
- Web: http://www.sis.pitt.edu/~jjoshi/courses/IS2620/Fall18/
  Office Hours: By appointment
- GSA: Runhua Xu and team

Course Objectives

- Understand the principles and methodologies for designing and implementing secure systems, and establishing software assurance
  - Life cycle models/ security engineering principles, …
  - Architectural risk analysis; threat modeling, …
- Understand and analyze code for vulnerabilities and learn secure programming practices
  - Secure programming & vulnerability analysis (e.g., C, C++ /Java); Web application security,
- To learn about the tools/techniques towards assurance (validation/verification/testing)
  - Use of tools/techniques to detect coding/design flaws; formal verification issues,
- Apply secure design principles to build a real system (projects)
- Understand emerging technologies and secure design challenges (time permitting)

Course Coverage

- Secure programming
  - Coding practices, issues and guidelines
    - Code analysis;
      - Buffer overflows
      - Input validation
      - Cross-site scripting
      - Mobile Code
      - Race conditions
      - SQL injection
      - Safe Languages
- Secure software development & Assurance process
  - Security Engineering/Lifecycle models
    - E.g. Capability Maturity Models and Extensions, Building security In
  - Secure Design, Testing, Implementation Principles
    - Systems / software &Formal methods and testing
      - UMLSec, Model Checking (code, protocols)
- Secure environments -- Supply Chain, Healthcare, etc.
- Verification / model checking, Threat modeling, reverse engineering
- Trusted computing modules/environments

Several sources: Books, Research papers / article / Standard documents, etc.

Mostly available online

Pre-requisite

- IS 2150/TEL 2810 Information Security & Privacy
  - OR background in security
- Following courses are preferred but not required:
  - IS 2170/TEL 2820 Cryptography; TEL 2821 Network Security
- Talk to me if you are not sure of the background
- Course Reference: Check website

Grading (Tentative)

- Assignments/Presentation: 40%
  - Read/Review and/or present research papers or articles
  - Assignments/quizzes
  - Lab exercises
- Two Exams: 30%
- Project : 30%

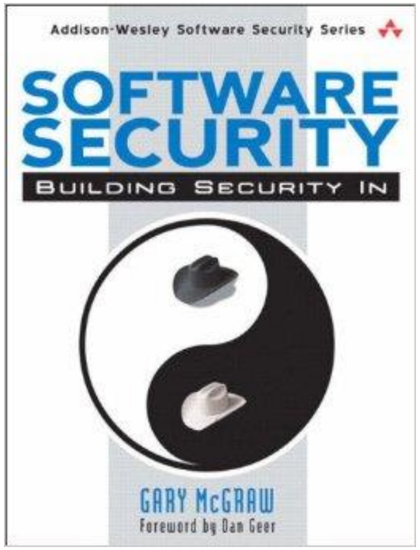
Course Policy

- Your work MUST be your own
  - Zero tolerance for cheating/plagiarism
  - You get an F for the course if you cheat in anything however small – NO DISCUSSION
  - Discussing the problem is encouraged
- Homework
  - Penalty for late assignments (15% each day)
  - Ensure clarity in your answers – no credit will be given for vague answers
- Check webpage for everything!
  - You are responsible for checking the webpage for updates

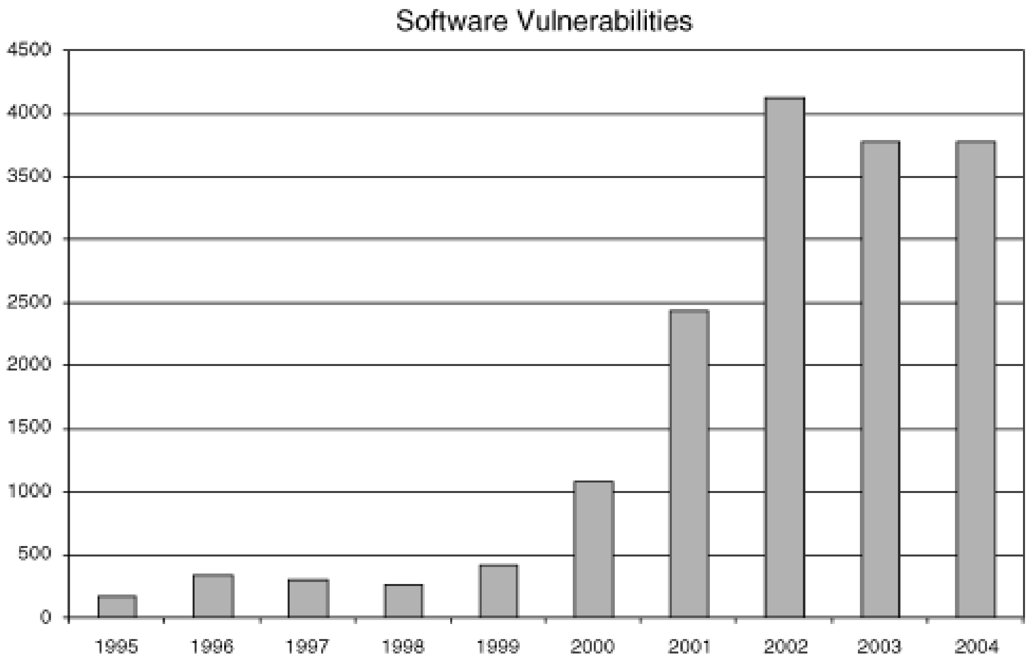
Why Secure Software/System Development?

Software/Systems Security

- Renewed ---- interest & importance
  - “*idea of engineering software so that it continues to function correctly under malicious attack*”
  - Existing software is riddled with design flaws and implementation bugs
    - ~70% related to design flaws*
  - “any program, no matter how innocuous it seems, can harbor security holes” [Cheswick & Bellovin, 1994]



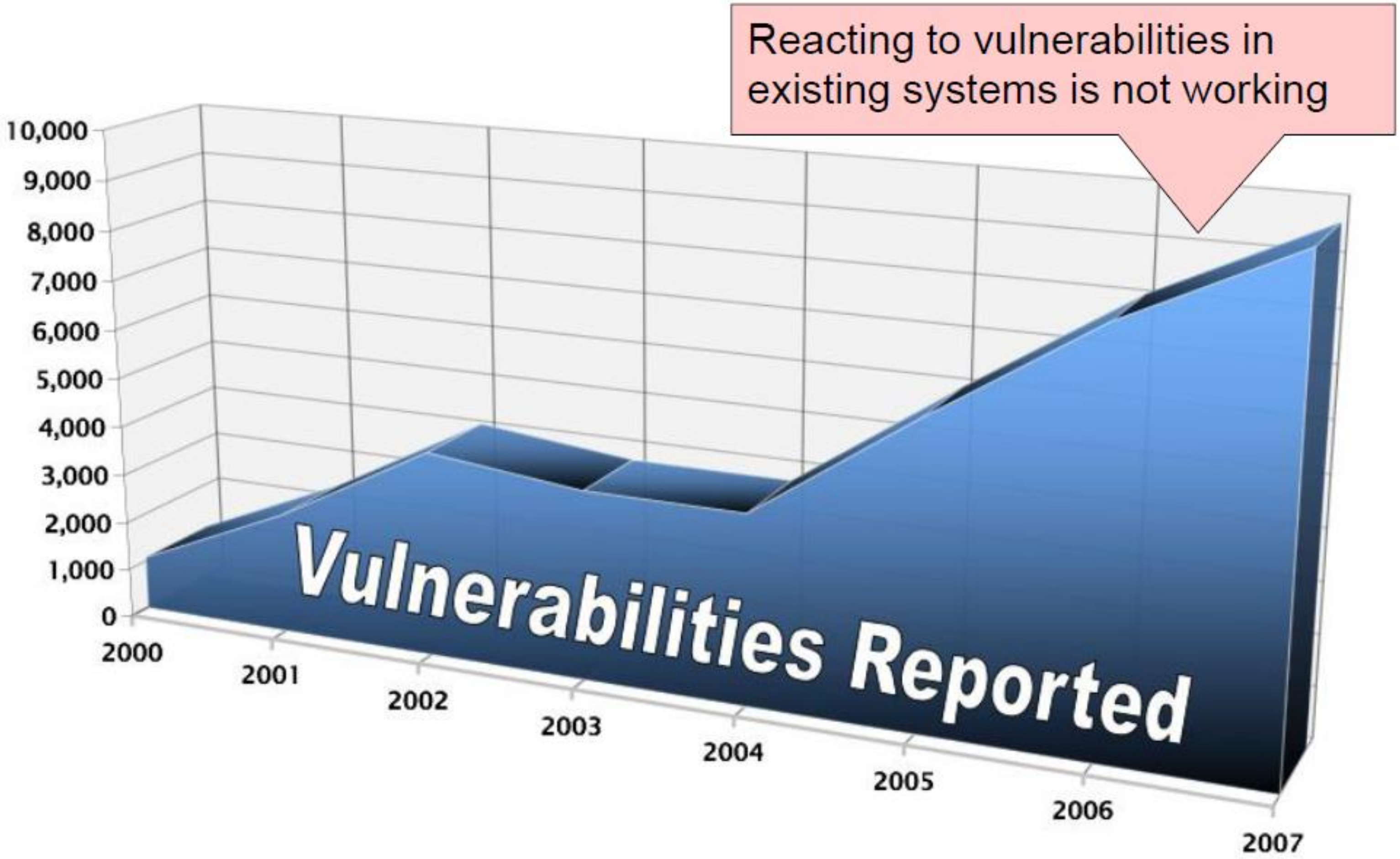
Software Problem



# vulnerabilities
Reported by CERT/CC

- More than half of the vulnerabilities are due to buffer overruns
- Others such as race conditions, design flaws are equally prevalent

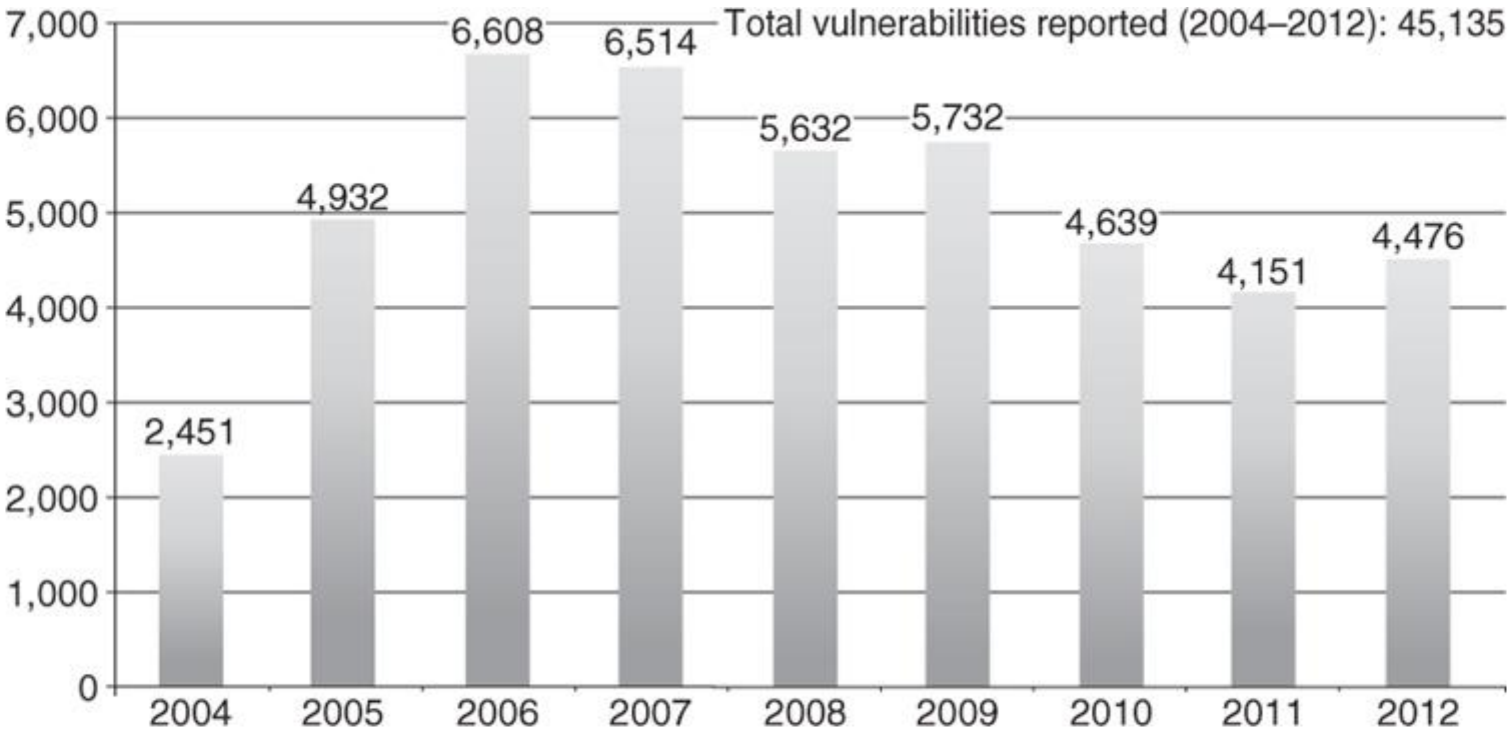
CERT Vulnerability



Reacting to vulnerabilities in existing systems is not working

Source: Seacord's Webinar on Secure Coding on C and C++

NVD statistics (NIST)



SourceFire report:

25 years of vulnerabilities (1988 – 2012)

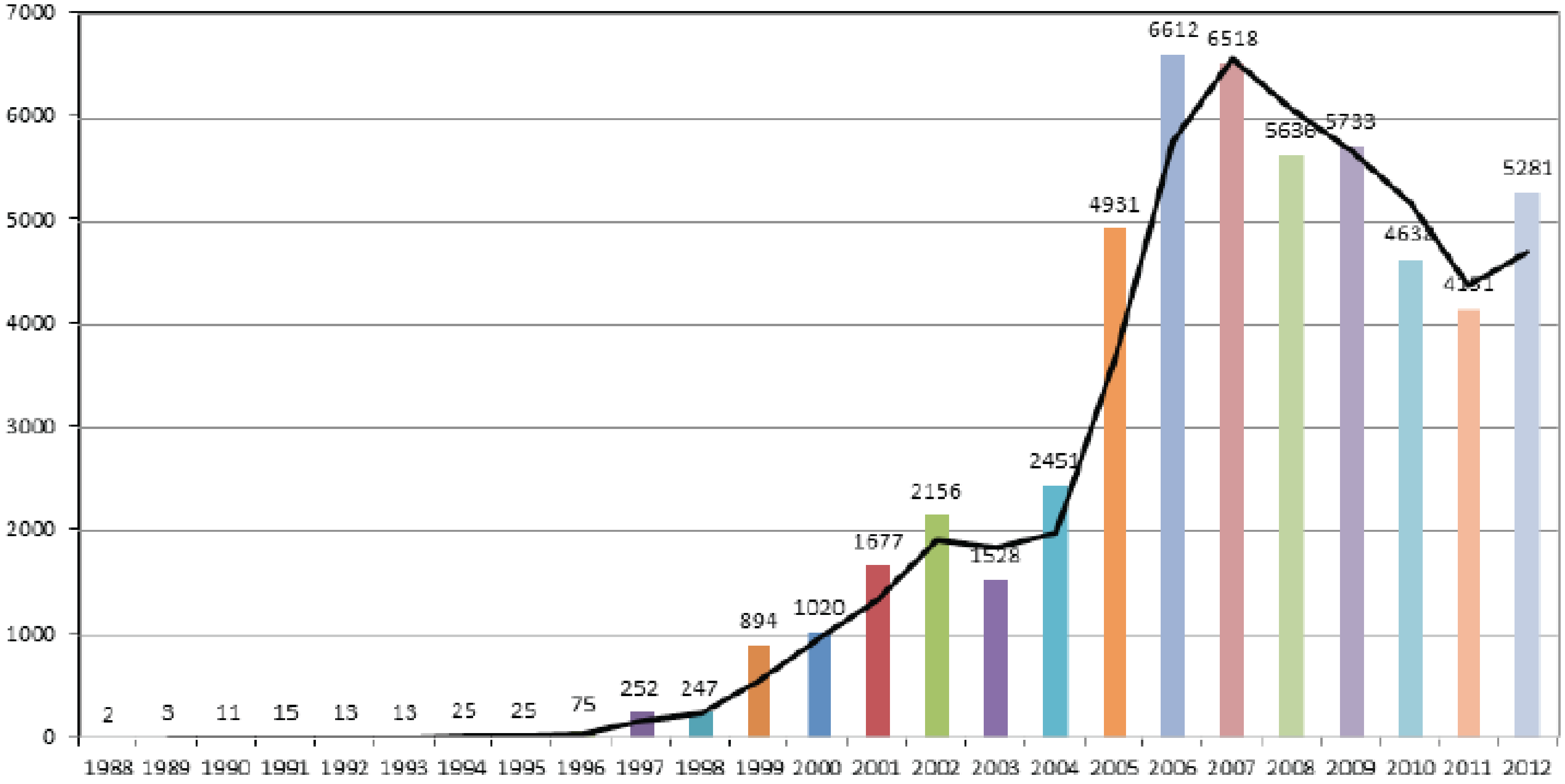


Figure 1. Vulnerabilities by year

- Based on CVE database classification & NVD

Source: https://courses.cs.washington.edu/courses/cse484/14au/reading/25-years-vulnerabilities.pdf

Severity of 7 or higher (SourceFire)

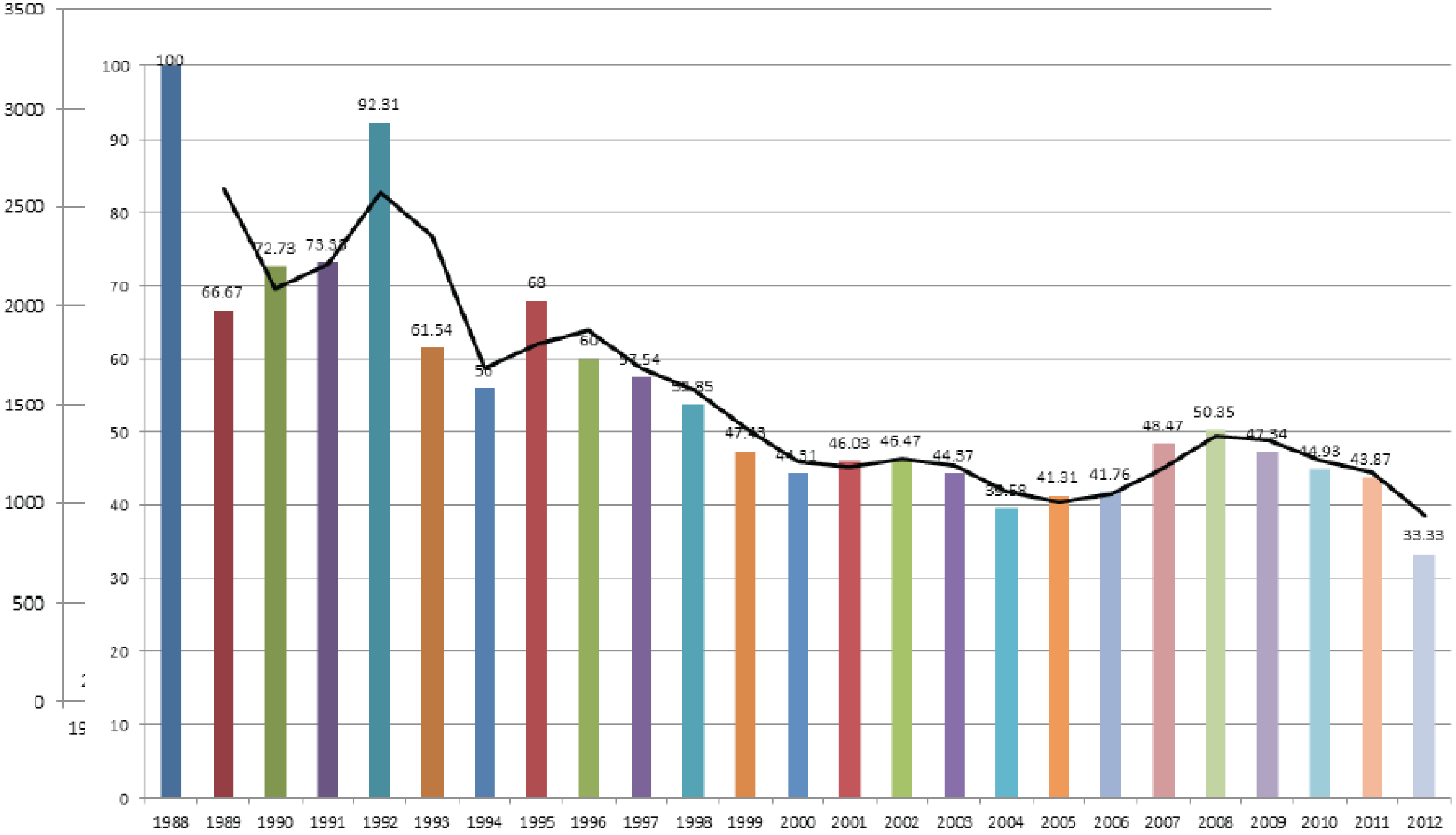


Figure 3. High severity vulnerabilities by year as a percentage of total vulnerabilities

SourceFire (over 25 years)

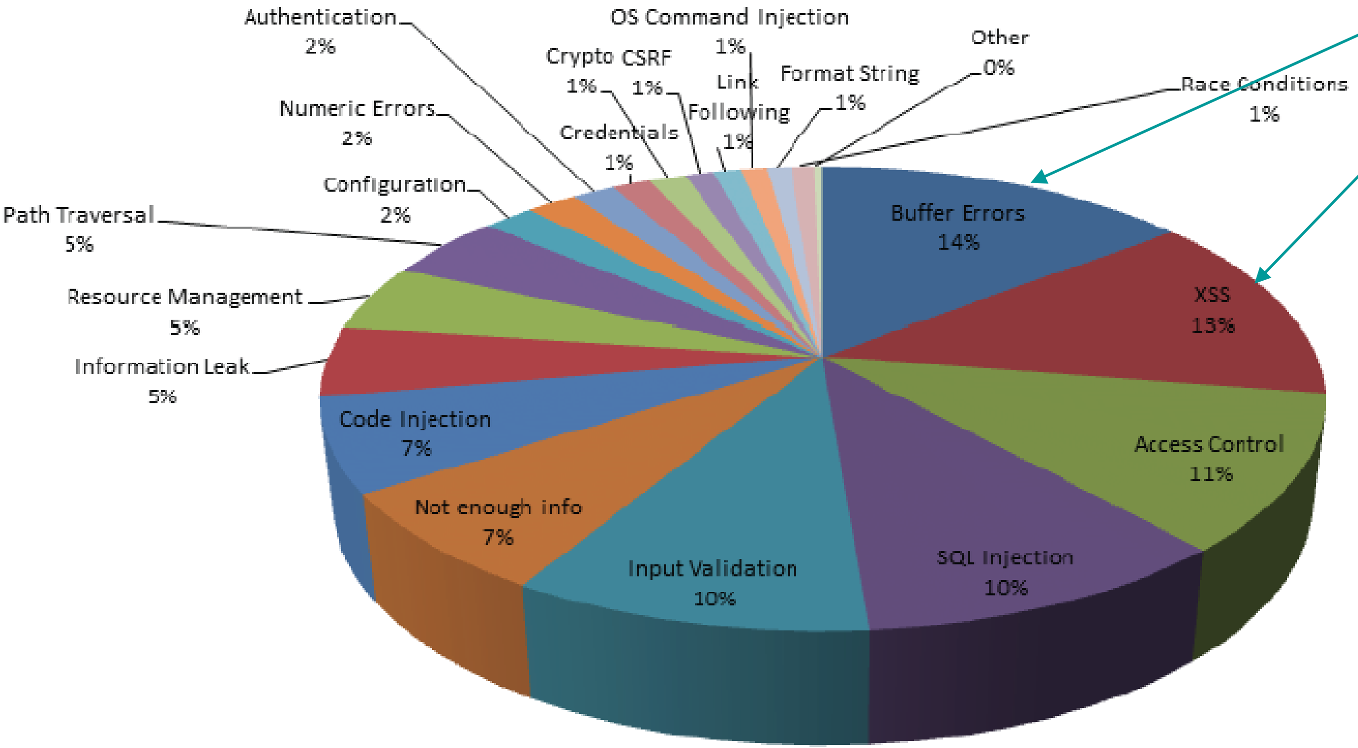
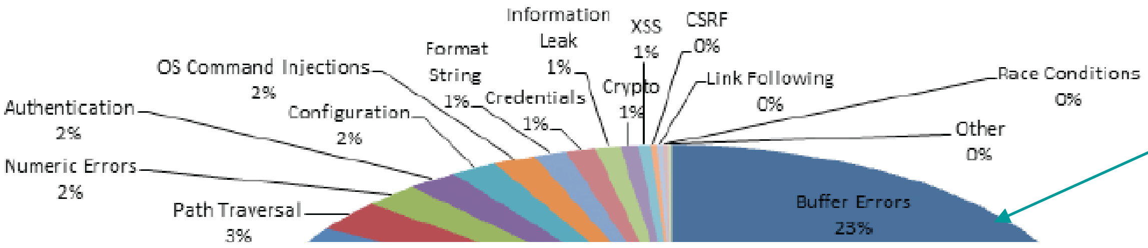


Figure 6. Top vulnerability types

Buffer Overflow

XSS Scripting

SourceFire (over 25 years): High & Critical



Buffer Overflow

SQL Injection

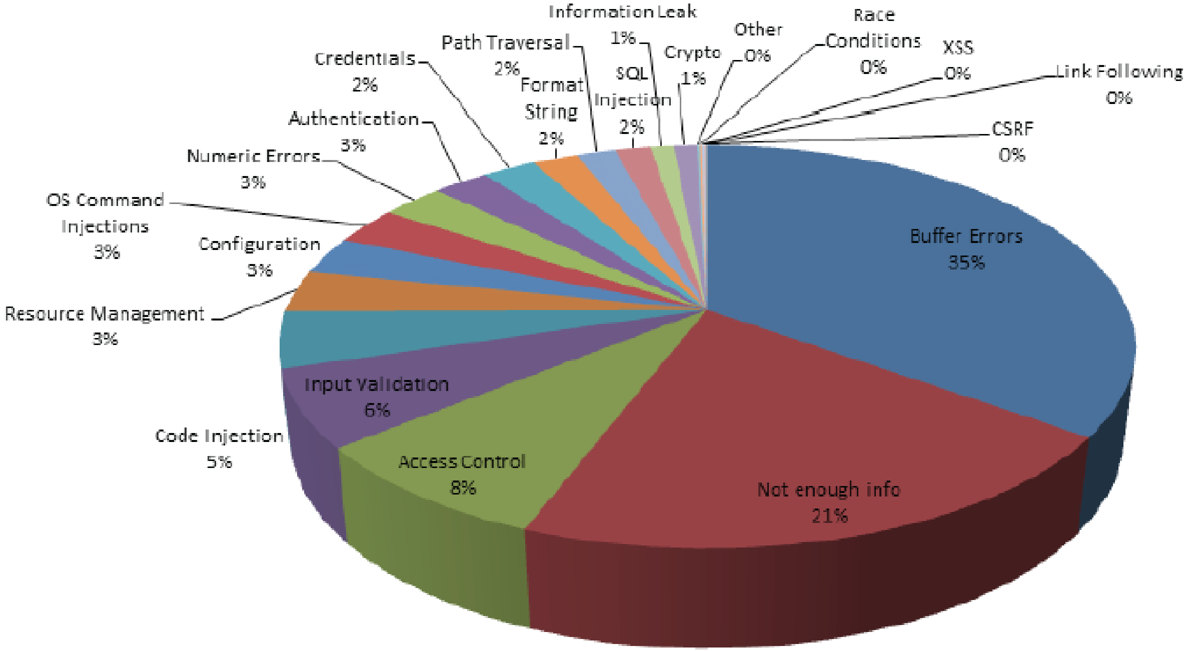


Figure 8. Top vulnerability types with a critical severity

By product ..

- Note different versions o

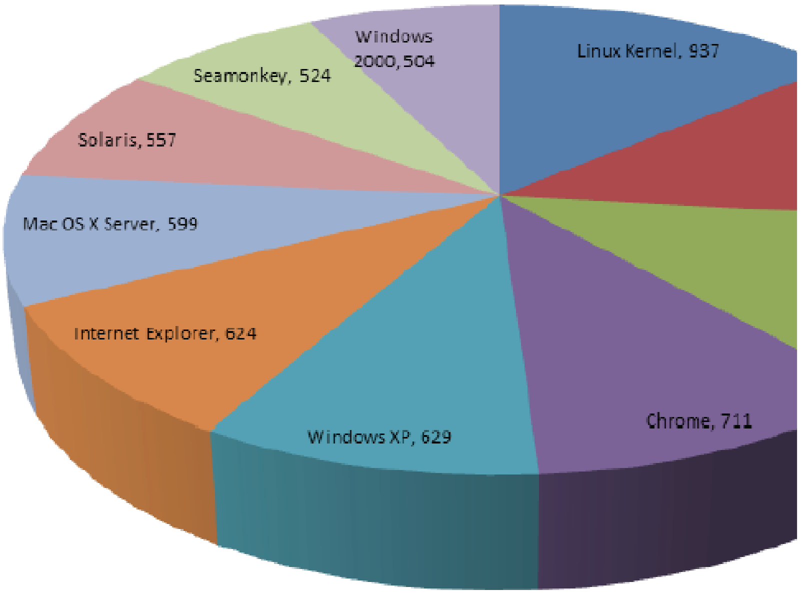


Figure 14. Top 10 products with the most reported vulnerabil

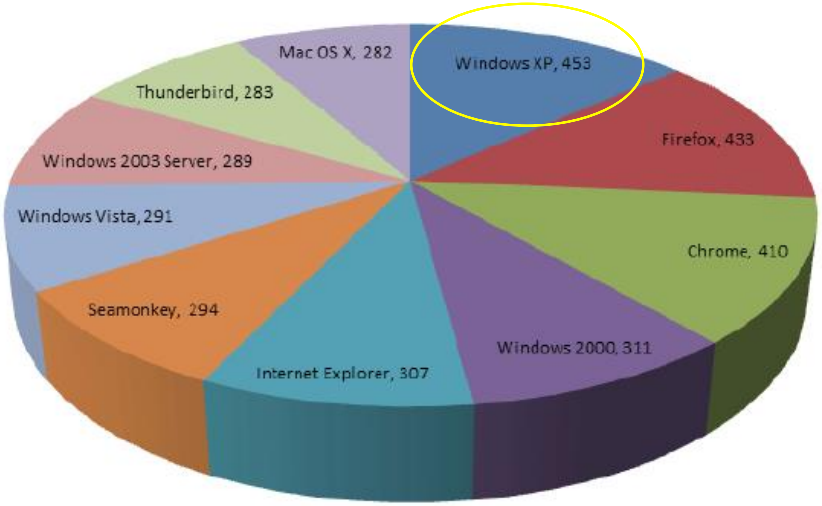


Figure 15. Top 10 products with high severity vulnerabilities

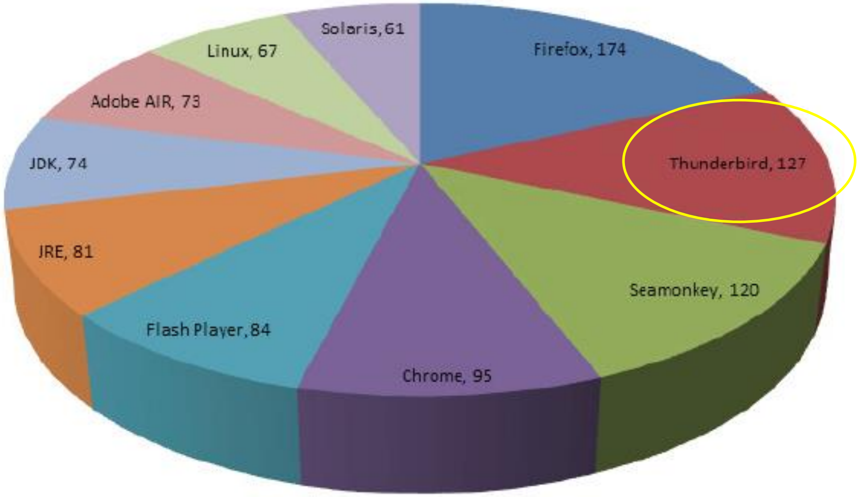


Figure 16. Top 10 products with critical severity vulnerabilities

Mobile …

- .. Although iPhone has the most – now they are market leaders in mitigations

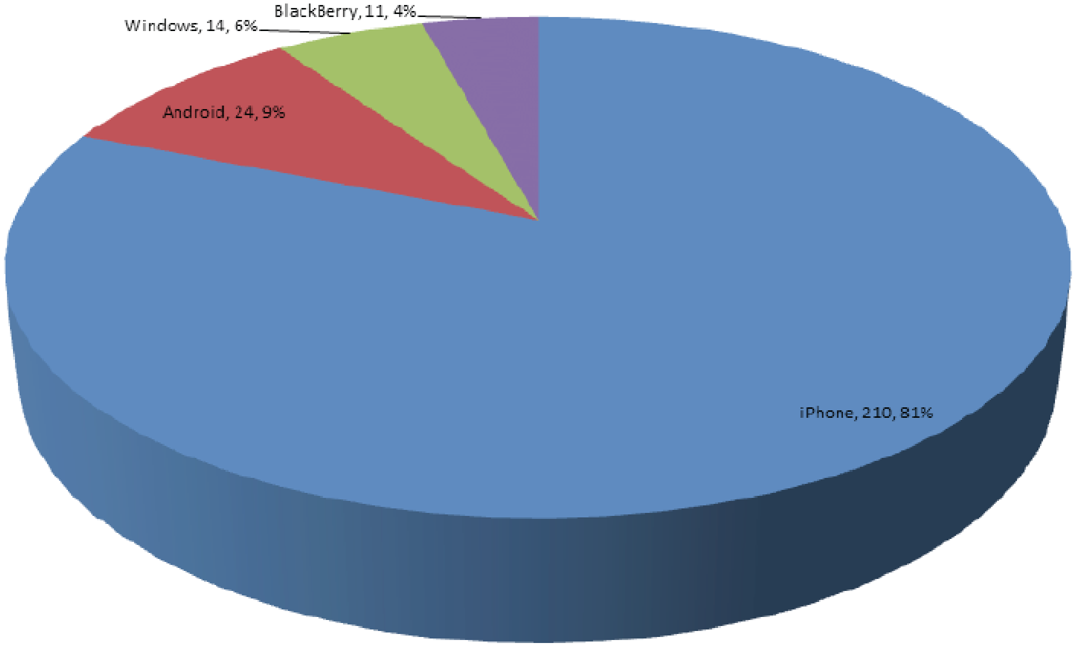


Figure 19. Mobile phone vulnerability market share

Windows M-OS: W-CE, W-Mobile, W-RT, W-Phone

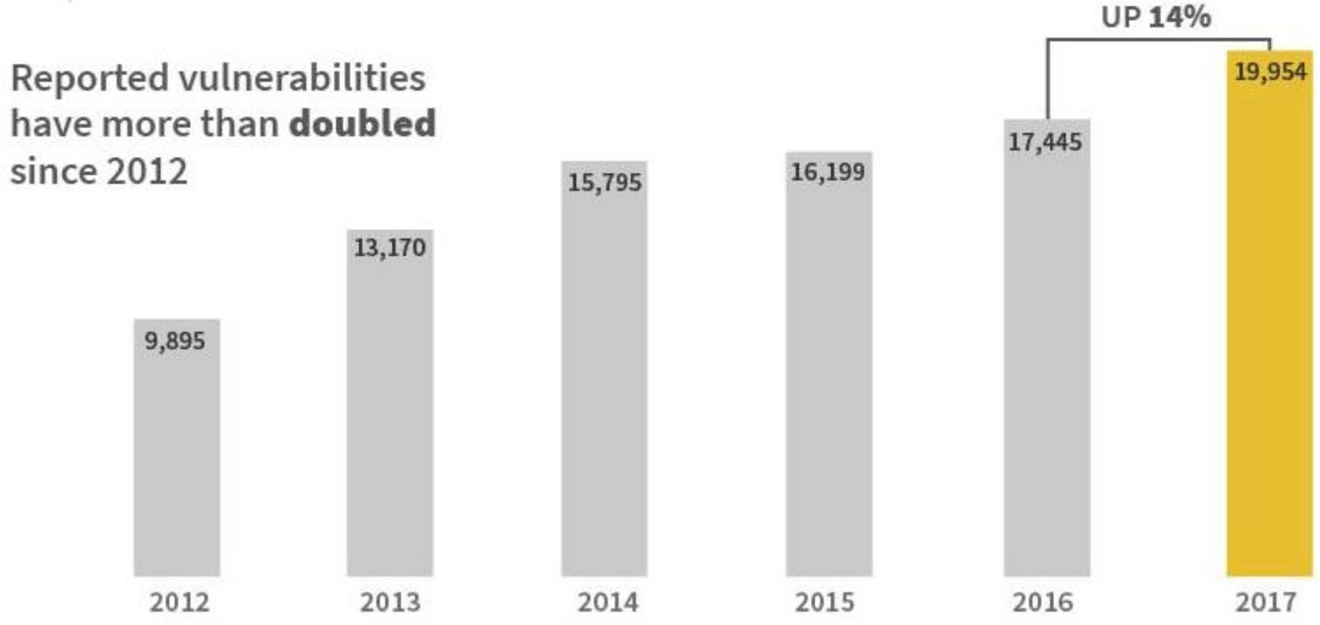
SourceFire ..

- Buffer overflow is one of the top ..
- While fewer vulnerabilities were reported % of more critical vulnerabilities has increased
- Microsoft has significantly improved
- Chrome is quite high in terms of # vulnerabilities
- iPhone leads in the group

From Flexera.com ..

Figure 2

GLOBAL VULNERABILITIES REPORTED FOR ALL PRODUCTS OF ALL VENDORS



Copyright © 2018 Secunia Research at Flexera | *Source: Vulnerability Review 2018*

Source:
https://resources.flexera.com/web/pdf/Research-SVM-Vulnerability-Review-2018.pdf

2018 Vulnerability Statistics Report

APPLICATION VULNERABILITY TAXONOMY

29% INSECURE CONFIGURATION/ INSECURE DEPLOYMENT

Directory Listing
Development Files
Default Documents
Default/Weak Server/Framework Security Settings
Debugging Enabled
Insecure Protocols Enabled
Insecure HTTP Methods
Unsupported Frameworks
Insecure Libraries

3% EXPOSED INTERFACE

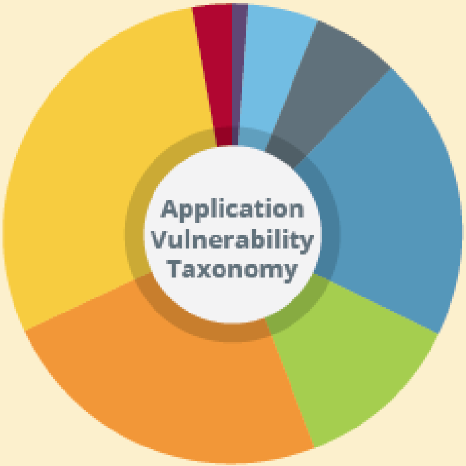
Web Admin consoles
Malicious file upload
Exposed S3 buckets
API's

1% DENIAL OF SERVICE

Application Layer DoS

5% AUTHORISATION WEAKNESSES

File Path Traversal
Vertical Authorisation
Horizontal Authorisation
Bypass Client-side Controls
Privilege Escalation



24% CLIENT-SIDE SECURITY

Cross-Site-Scripting (XSS)
Clickjacking
CORS
Cross-Domain Leakage
Form Hijacking
HTML Injection
Open Redirection
DOM Security

6% AUTHENTICATION WEAKNESSES

Bruteforce
Default Credentials
Weak Logic
Weak Password Policy
Username Enumeration
Credential transmission without encryption
Session Management
Weak Protocol
No encryption
CSRF

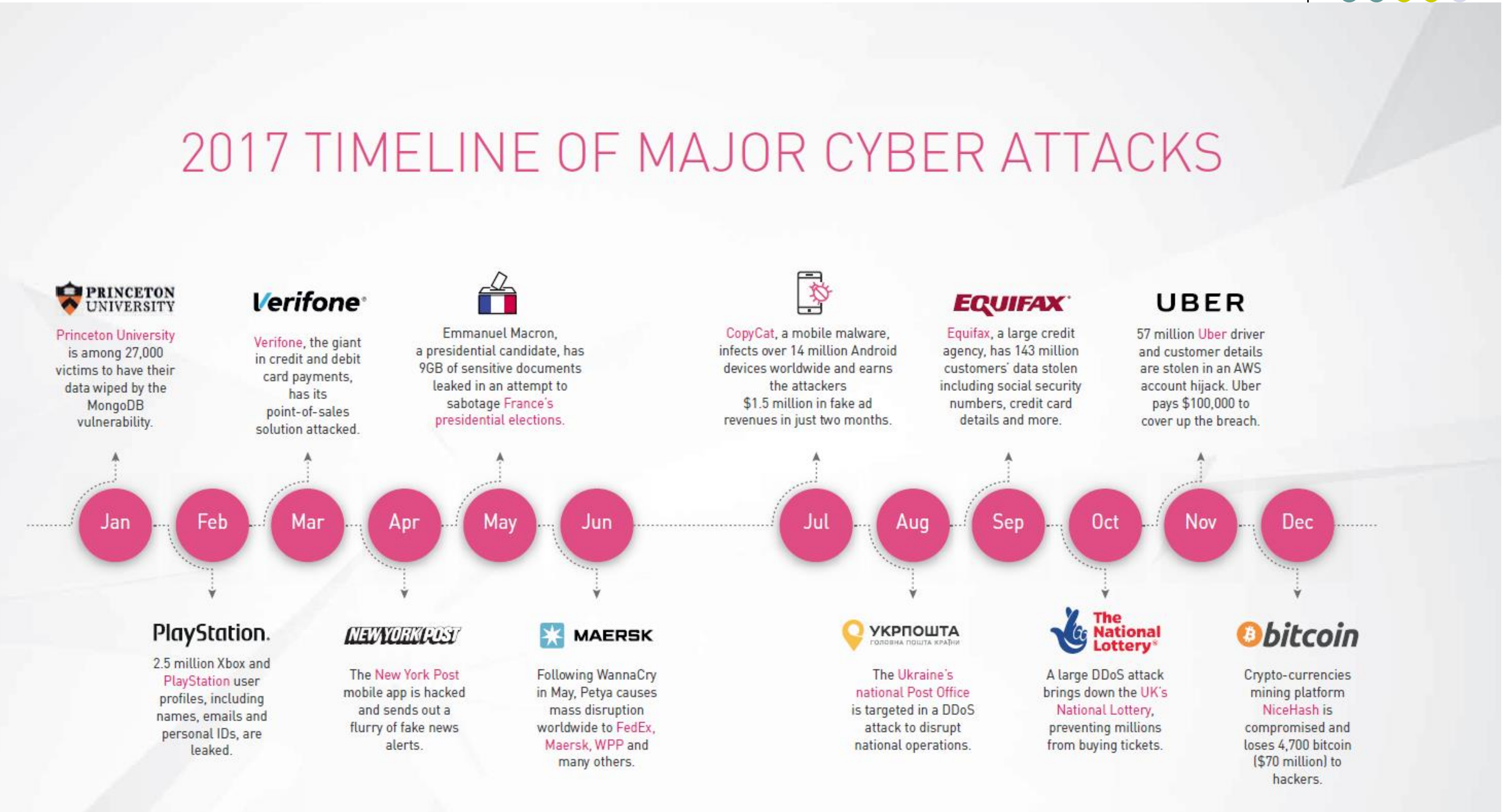
12% INJECTION ATTACKS

SQL Injection
CRLF Injection
XXE
External Service Interaction
File Path
Header Injection
OS Command Injection

20% INFORMATION LEAKAGE

Default Error Pages
System Information Leakage
Caching
Sensitive Information Disclosure Weaknesses
Metadata Disclosure
Exposed Business Intel & Documents
Private IP Address Leakage
Source Code Disclosure

From CheckPoint



2017 TIMELINE OF MAJOR CYBER ATTACKS

Jan — PRINCETON UNIVERSITY

Princeton University is among 27,000 victims to have their data wiped by the MongoDB vulnerability.

Feb — PlayStation.

2.5 million Xbox and PlayStation user profiles, including names, emails and personal IDs, are leaked.

Mar — Verifone

Verifone, the giant in credit and debit card payments, has its point-of-sales solution attacked.

Apr — NEW YORK POST

The New York Post mobile app is hacked and sends out a flurry of fake news alerts.

May

Emmanuel Macron, a presidential candidate, has 9GB of sensitive documents leaked in an attempt to sabotage France's presidential elections.

Jun — MAERSK

Following WannaCry in May, Petya causes mass disruption worldwide to FedEx, Maersk, WPP and many others.

Jul

CopyCat, a mobile malware, infects over 14 million Android devices worldwide and earns the attackers $1.5 million in fake ad revenues in just two months.

Aug — УКРПОШТА ГОЛОВНА ПОШТА КРАЇНИ

The Ukraine's national Post Office is targeted in a DDoS attack to disrupt national operations.

Sep — EQUIFAX

Equifax, a large credit agency, has 143 million customers' data stolen including social security numbers, credit card details and more.

Oct — The National Lottery

A large DDoS attack brings down the UK's National Lottery, preventing millions from buying tickets.

Nov — UBER

57 million Uber driver and customer details are stolen in an AWS account hijack. Uber pays $100,000 to cover up the breach.

Dec — bitcoin

Crypto-currencies mining platform NiceHash is compromised and loses 4,700 bitcoin ($70 million) to hackers.

And we have Russian cyber attack … increasing attack on CI ….

https://www.checkpoint.com/downloads/product-related/report/2018-security-report.pdf

Insider vs Outsider;

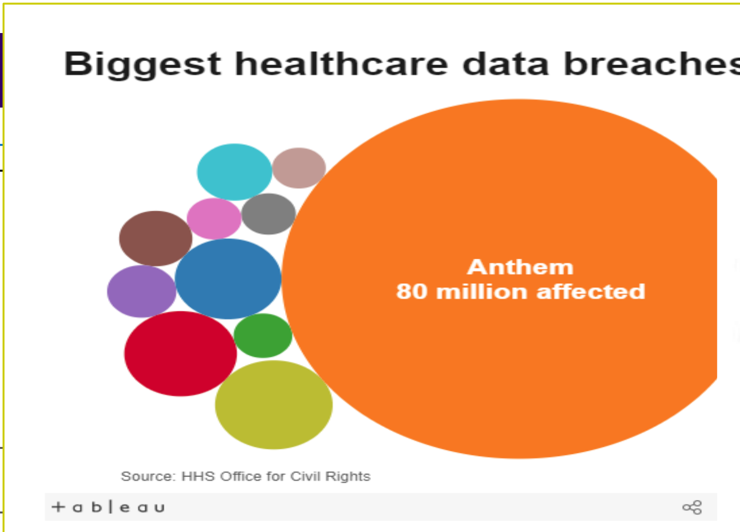
1

2

3

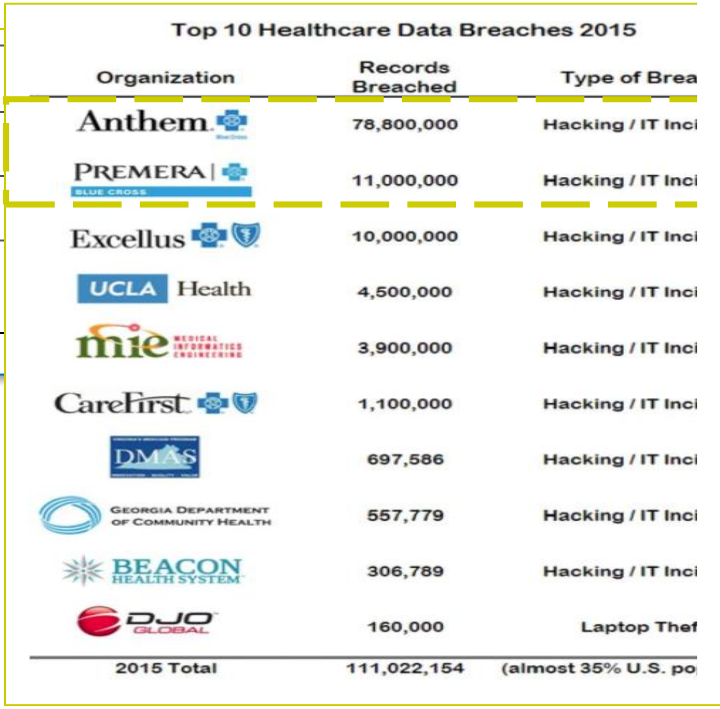
4

5



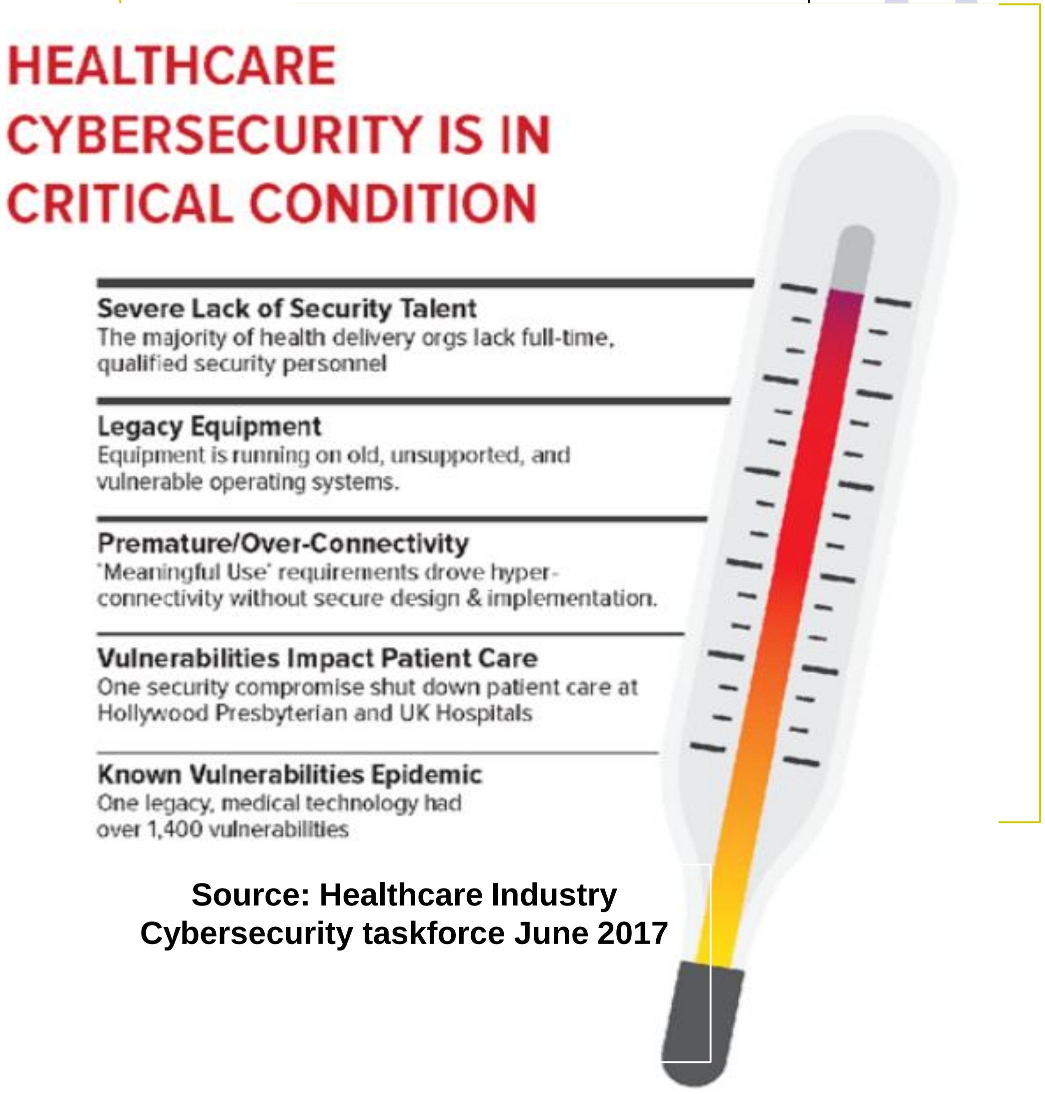
Biggest healthcare data breaches

Source: HHS Office for Civil Rights



Top 10 Healthcare Data Breaches 2015

Organization	Records Breached	Type of Brea
Anthem	78,800,000	Hacking / IT Inci
PREMERA BLUE CROSS	11,000,000	Hacking / IT Inci
Excellus	10,000,000	Hacking / IT Inci
UCLA Health	4,500,000	Hacking / IT Inci
mie MEDICAL INFORMATICS ENGINEERING	3,900,000	Hacking / IT Inci
CareFirst	1,100,000	Hacking / IT Inci
DMAS	697,586	Hacking / IT Inci
Georgia Department of Community Health	557,779	Hacking / IT Inci
BEACON HEALTH SYSTEM	306,789	Hacking / IT Inci
DJO GLOBAL	160,000	Laptop Thef
2015 Total	111,022,154	(almost 35% U.S. po



HEALTHCARE CYBERSECURITY IS IN CRITICAL CONDITION

Severe Lack of Security Talent
The majority of health delivery orgs lack full-time, qualified security personnel

Legacy Equipment
Equipment is running on old, unsupported, and vulnerable operating systems.

Premature/Over-Connectivity
'Meaningful Use' requirements drove hyper-connectivity without secure design & implementation.

Vulnerabilities Impact Patient Care
One security compromise shut down patient care at Hollywood Presbyterian and UK Hospitals

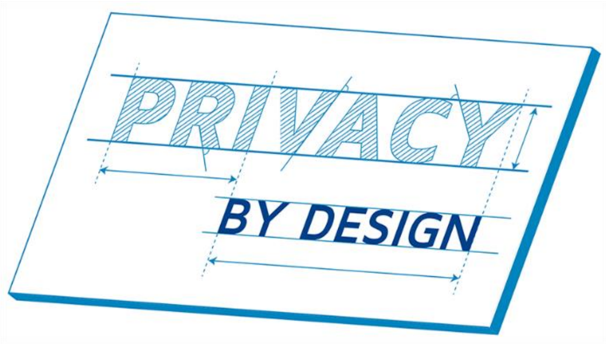
Known Vulnerabilities Epidemic
One legacy, medical technology had over 1,400 vulnerabilities

Source: Healthcare Industry Cybersecurity taskforce June 2017

Increasing Impact on Individual and society!

Critical to address security of systems/environments:

Secure-by-design
Privacy-by-design



Software security

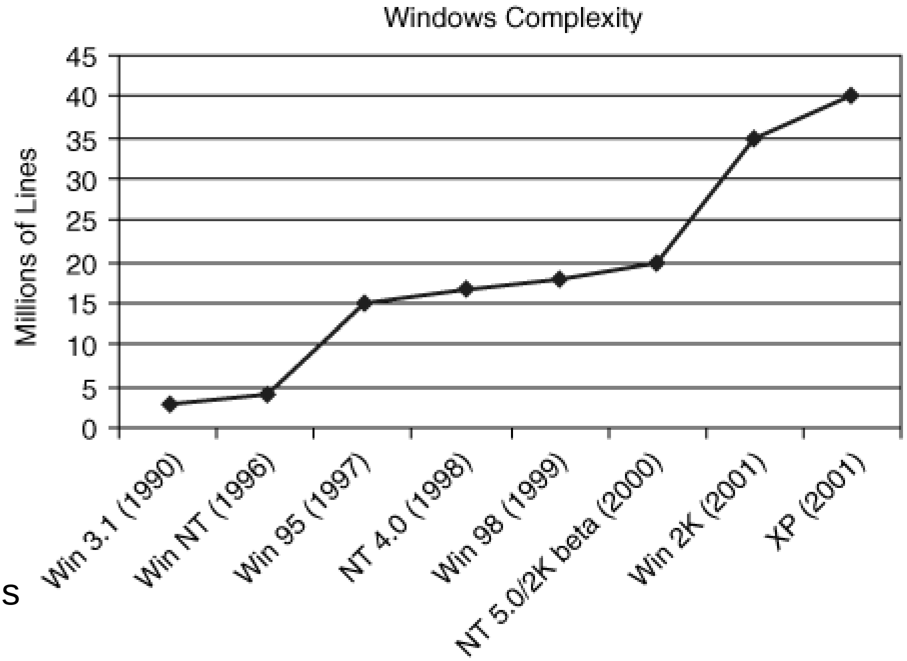
- It is about
  - Understanding software-induced security risks and how to manage them
  - Leveraging software engineering practice,
  - thinking security early in the software lifecyle
  - Knowing and understanding common problems
  - Designing for security
  - Subjecting all software artifacts to thorough objective risk analyses and testing
- It is a knowledge intensive field

Trinity of trouble

- Three trends
  - Connectivity
    - Inter networked, IoT/devices
    - Include SCADA (supervisory control and data acquisition systems)
    - Automated attacks, botnets
    - Multiple paths – attack vectors
  - Extensibility
    - Mobile code – functionality *evolves* incrementally
    - Web/OS Extensibility
  - Complexity
    - XP is at least 40 M lines of code
    - Add to that use of unsafe languages (C/C++)
    - Current estimate: Google Internet services total around 2B LoC & Windows ~50M (https://www.wired.com/2015/09/google-2-billion-lines-codeand-one-place/)

Bigger problem today

.. And growing



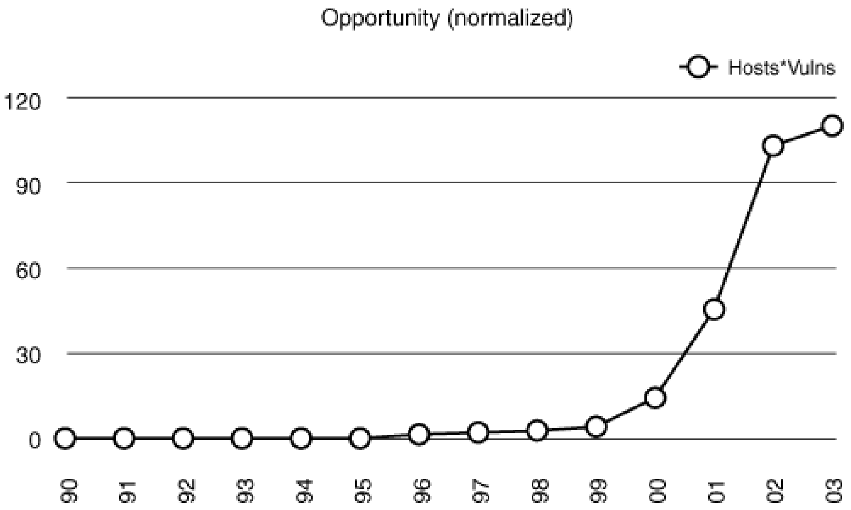
INFOGRAPHICS Link:
http://h.fastcompany.net/multisite_files/fastcompany/imagecache/inline-large/inline/2013/11/3021256-inline-800linesofcode5.jpg
(Click to see)

It boils down to …

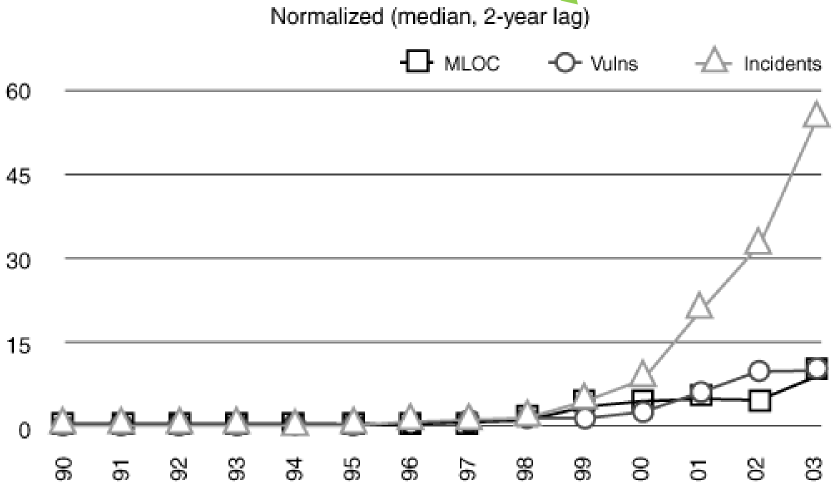
more code,

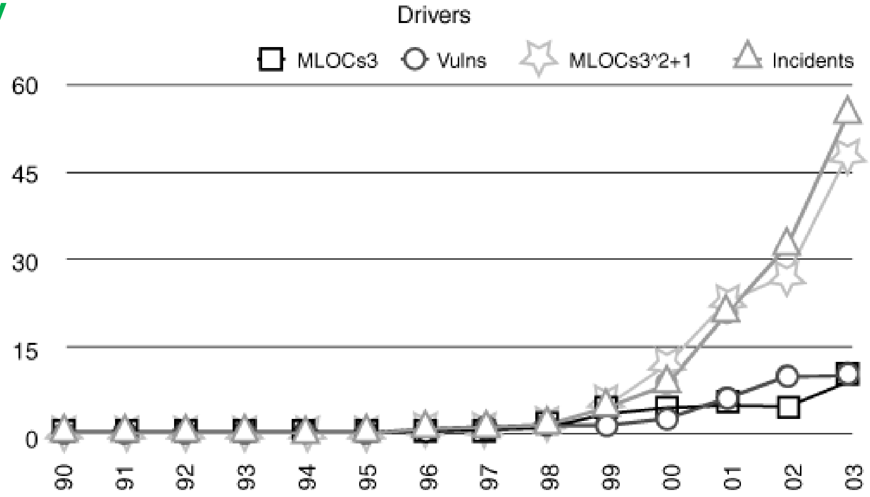
more bugs,

more security problems



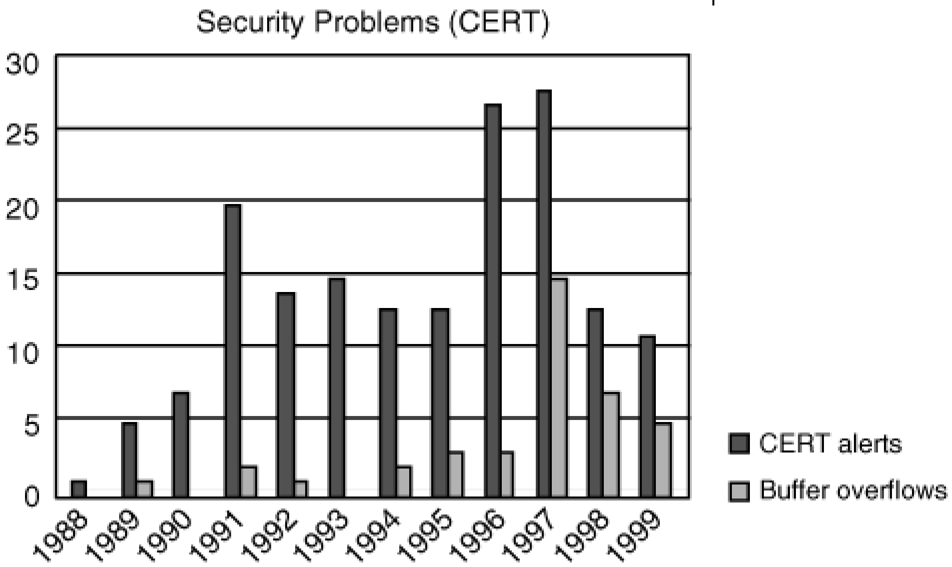
Diffusion delay





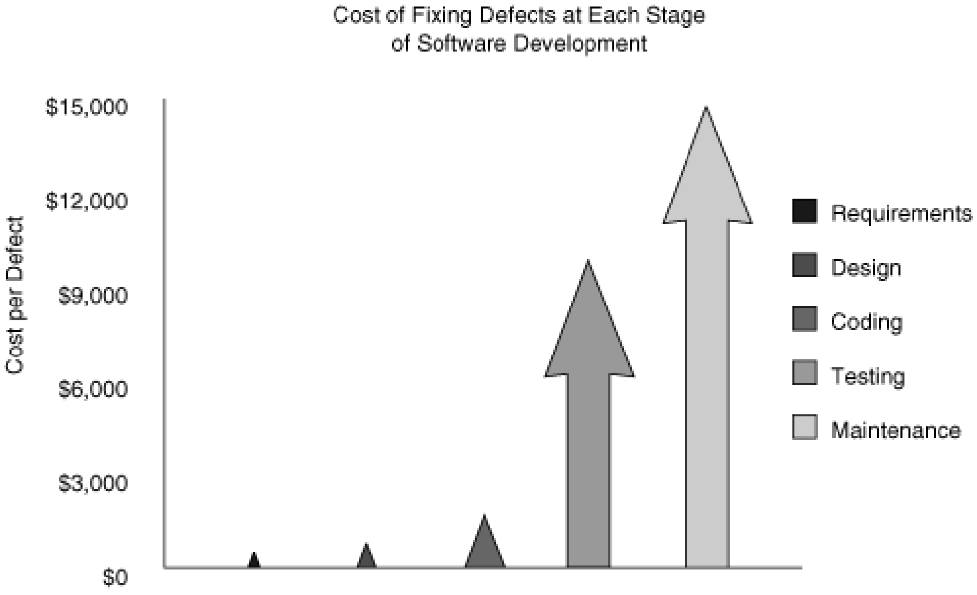
Security problems in software

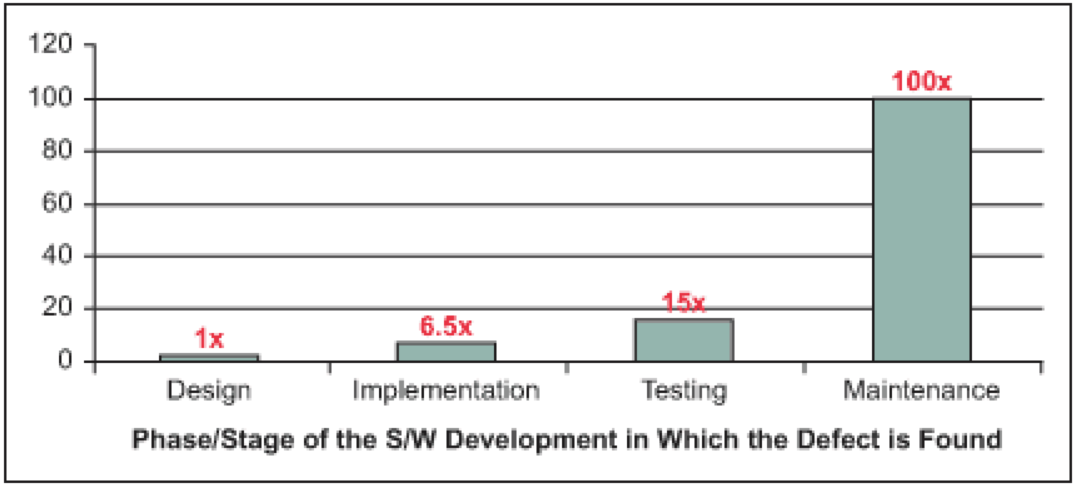
- Defect
  - implementation and design vulnerabilities
  - Can remain dormant
- Bug
  - An implementation level software problem
- Flaw
  - A problem at a deeper level
- Bugs + Flaws
  - leads to Risk



Bug	Flaw
Buffer overflow: stack smashing Buffer overflow: one-stage attacks Buffer overflow: string format attacks Race conditions: TOCTOU Unsafe environment variables Unsafe system calls (fork(), exec(), system()) Incorrect input validation (black list vs. white list	Method over-riding problems (subclass issues) Compartmentalization problems in design Privileged block protection failure (DoPrivilege()) Error-handling problems (fails open) Type safety confusion error Insecure audit log design Broken or illogical access control (role-based access control [RBAC] over tiers) Signing too much code

Cost of fixing





Relative Costs to Fix Software Defects (Source: IBM Systems Sciences Institute)

OWASP Top Ten Vulnerabilities (for 2013)

- A1-Injection
  - SQL, OS, LDAP – input validation problem
- A2-Broken Authentication and Session Management
  - Incorrect implementation (compromise passwords, keys, implementation flaws
- A3-Cross-Site Scripting (XSS)
  - Improper validation
- A4-Insecure Direct Object References
  - Improper exposure of internal implementation
- A5-Security Misconfiguration
- A6-Sensitive Data Exposure

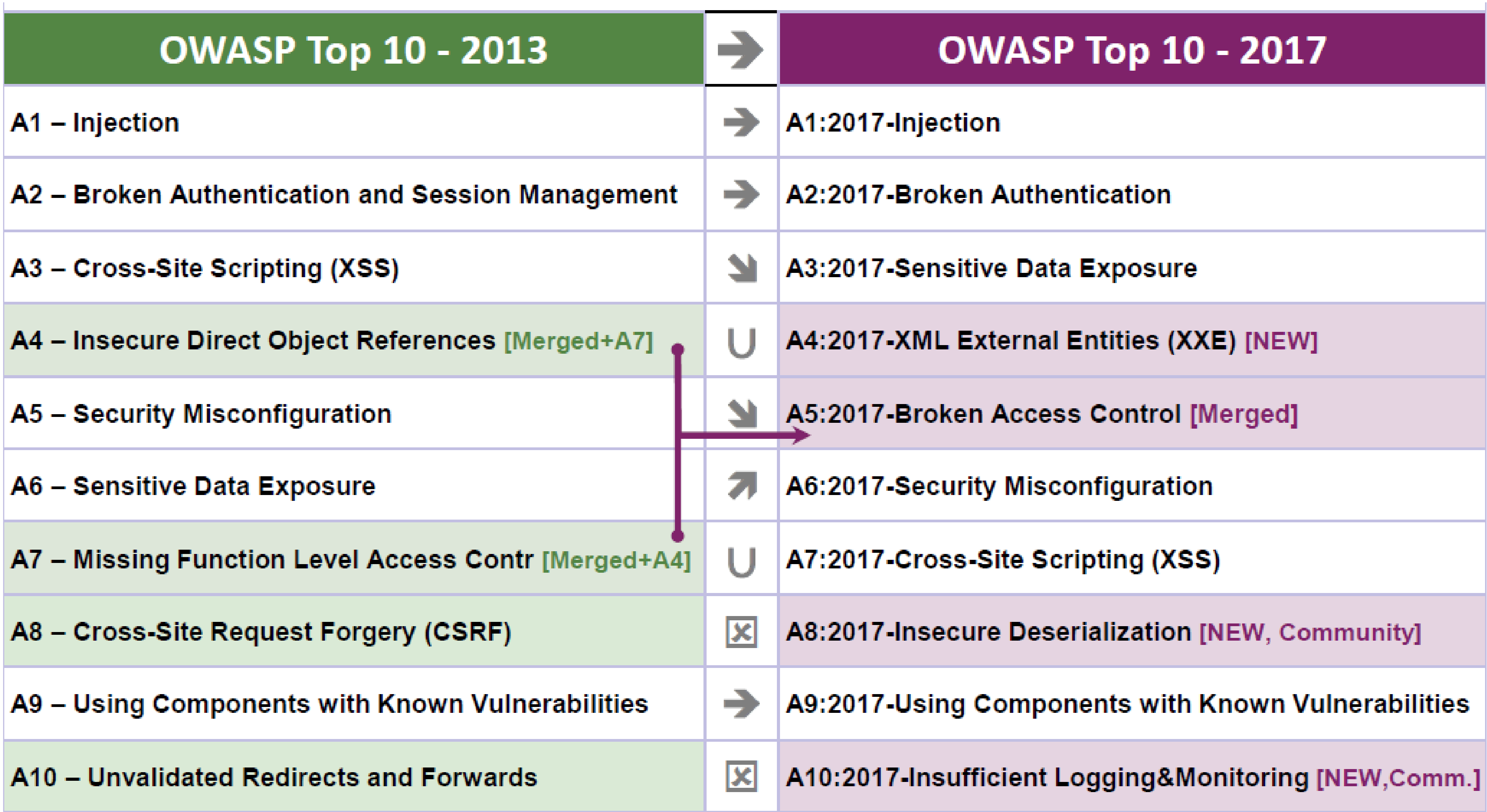
OWASP
The Open Web Application Security Project

OWASP Top Ten Vulnerabilities (for 2013)

- A7-Missing Function Level Access Control
  - Web applications UI and server need to enforce consistent access control enforcement
- A8-Cross-Site Request Forgery (CSRF)
  - Forged HTTP requests and compromise of victim’s session cookie
  - Victim’s browser is forced to generate requests to the vulnerable application
- A9-Using Components with Known Vulnerabilities
  - Components could run with full privileges – vulnerable program could be exploited
  - Components could be libraries or software modules and frameworks
- A10-Unvalidated Redirects and Forwards
  - Improper validation issue
  - Web apps can redirect victims to phishing or malware sites.

Comparison: http://www.port80software.com/support/articles/2013-owasp-top-10

2013 -> 2017 OWASP top 10



OWASP Top 10 - 2013	→	OWASP Top 10 - 2017
A1 – Injection	→	A1:2017-Injection
A2 – Broken Authentication and Session Management	→	A2:2017-Broken Authentication
A3 – Cross-Site Scripting (XSS)	↘	A3:2017-Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	∪	A4:2017-XML External Entities (XXE) [NEW]
A5 – Security Misconfiguration	↘	A5:2017-Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017-Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	∪	A7:2017-Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	☒	A8:2017-Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017-Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	☒	A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

Recent incidents ..

- HeartBleed (CVE-2014-0160)
  - A serious threat in OpenSSL
  - Estimated to have made 2/3 of Internet vulnerable
  - Essentially a buffer overflow issue (overreads)
  - Improper input validation – allows access to more data
    - Automated software testing did not catch !!
    - Static analysis did not catch it ! And dynamic/hybrid not designed for such vulnerability
  - Some approaches that would have helped
    - Negative testing/Fuzzing with special checks
    - Better Source code analysis; safer language (it was in C)
    - Formal methods

Source: "Preventing Heartbleed" by David Wheeler, IEEE Computer
Also Check out: http://www.kb.cert.org/vuls/id/720951

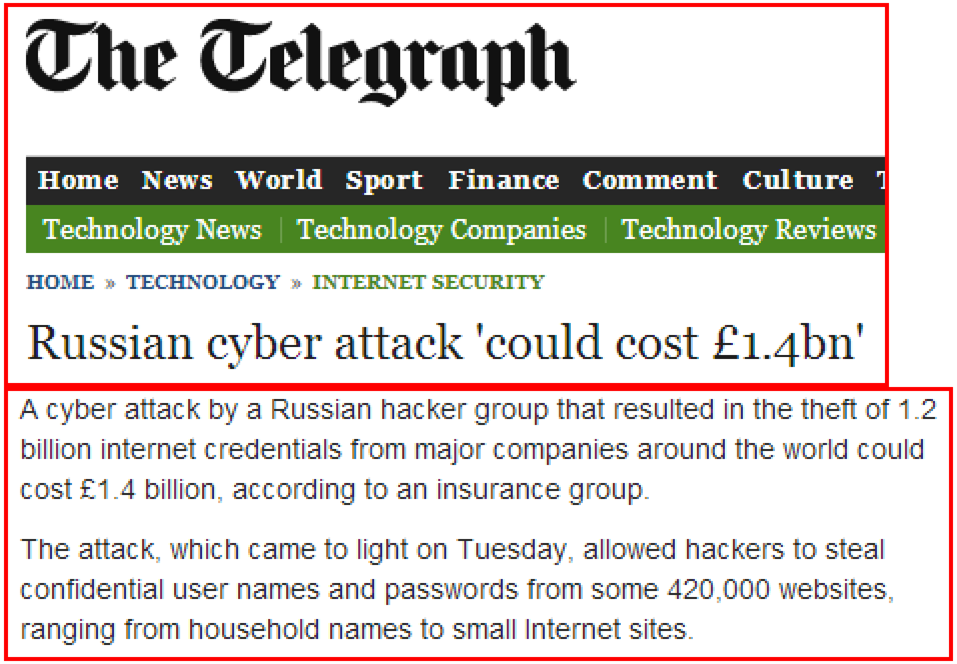
Recent incidents ..

- Stuxnet
  - Affected several ICSs; Includes
    - exploit of the LNK files – shortcut file in windows as a start (other exploits possible)
    - exploit some unpatched version of Win XP
- Target data breach*
    - Financial and personal info of ~110M customers
    - Payment card system flaw – malware installed in POS terminals (RAM Scraping attack)
    - Network access from third party (PA HVAC) which was weak in security – allowed to gain foothold in Target's network

***http://docs.ismgcorp.com/files/external/Target_Kill_Chain_Analysis_FINAL.pdf**

Recent incidents ..

- Russian hackers
  - Targets: Oil, Gas, Energy security – industrial espionage
  - Also target seizing control of ICS



The Telegraph

Home News World Sport Finance Comment Culture T

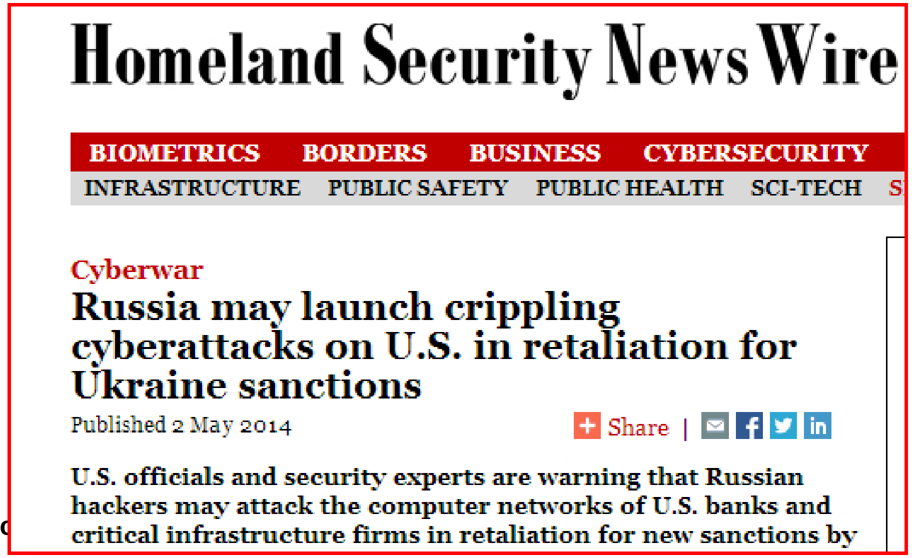
Technology News | Technology Companies | Technology Reviews

HOME » TECHNOLOGY » INTERNET SECURITY

Russian cyber attack 'could cost £1.4bn'

A cyber attack by a Russian hacker group that resulted in the theft of 1.2 billion internet credentials from major companies around the world could cost £1.4 billion, according to an insurance group.

The attack, which came to light on Tuesday, allowed hackers to steal confidential user names and passwords from some 420,000 websites, ranging from household names to small Internet sites.



Homeland Security News Wire

BIOMETRICS BORDERS BUSINESS CYBERSECURITY

INFRASTRUCTURE PUBLIC SAFETY PUBLIC HEALTH SCI-TECH S

Cyberwar

Russia may launch crippling cyberattacks on U.S. in retaliation for Ukraine sanctions

Published 2 May 2014

U.S. officials and security experts are warning that Russian hackers may attack the computer networks of U.S. banks and critical infrastructure firms in retaliation for new sanctions by

http://www.nytimes.com/2014/07/01/technology/energy-sector-fac

Hence we need …

- Robust and Secure Software Design and Secure Systems Engineering practice
  - Secure development life-cycle/methodologies
  - Secure process models to support large scale team management
  - Fix flaw early in the life-cycle – LOW COST !!
- Secure Design principles & Secure coding practices/standards
- Proper Testing and Verification/Validation
- Effective Tools and Techniques
- Security Engineering education
- Etc..

Let’s get started with basics

- **Secure design principles**
  1. Least Privilege
  2. Fail-Safe Defaults
  3. Economy of Mechanism (KISS)
  4. Complete Mediation
  5. Open Design
  6. Separation Privilege
  7. Least Common Mechanism
  8. Psychological Acceptability
  9. Defense in Depth

  (http://www.cs.virginia.edu/~evans/cs551/saltzer/)

McGraw’s Update

1. Secure the weakest link
2. Defend in depth
3. Fail securely
4. Grant least privilege
5. Separate privileges
6. Economize mechanism
7. Do not share mechanism
8. Be reluctant to trust
9. Assume your secrets are not safe
10. Mediate completely
11. Make security usable
12. Promote privacy (PII)
13. Use your resources – ask for help

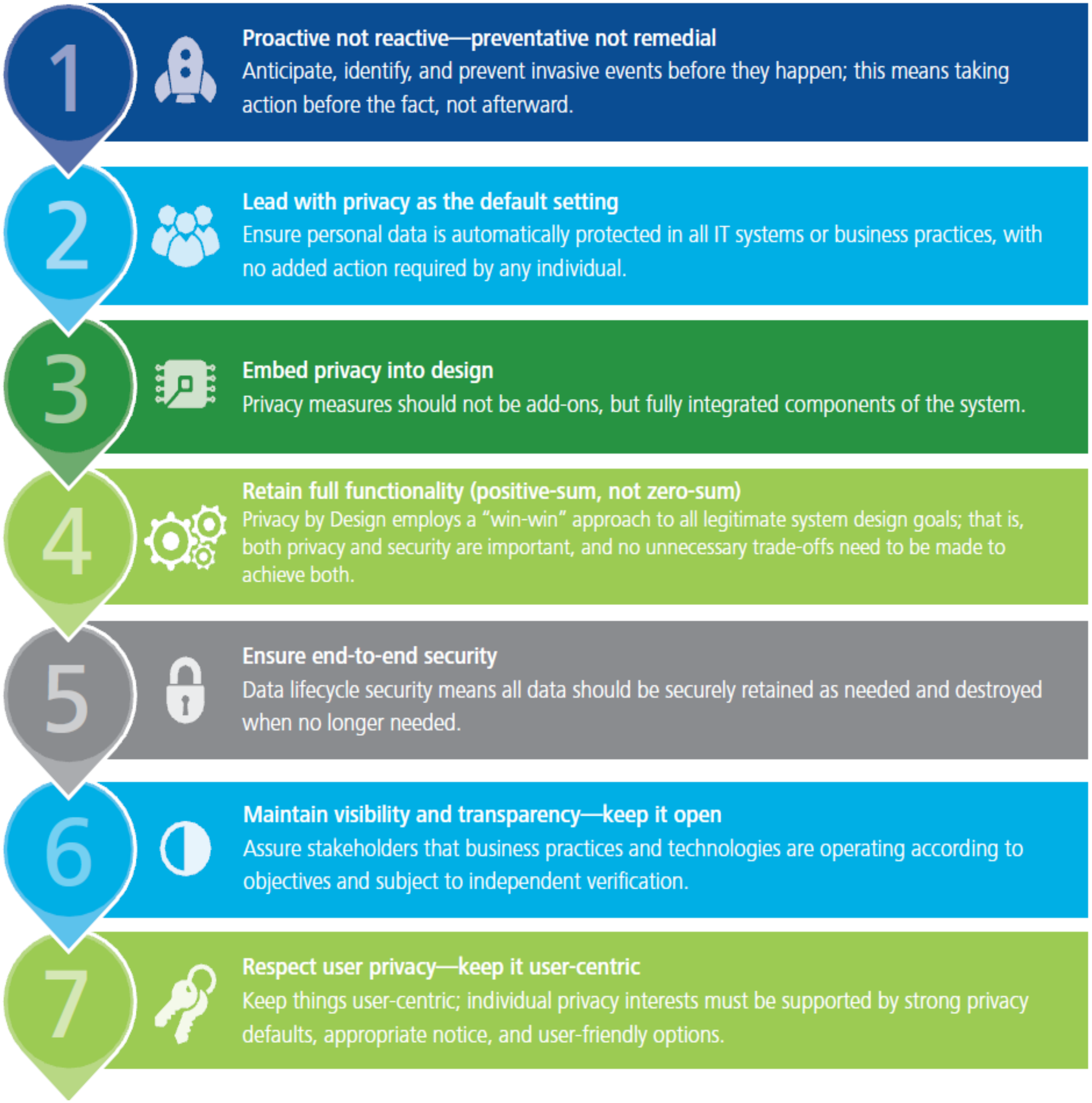
(http://searchsecurity.techtarget.com/opinion/Thirteen-principles-to-ensure-enterprise-system-security)

Mead et al.’s 7 principles

- To address challenges of acquiring, building, deploying, and sustaining systems to achieve a desired level of confidence for Software assurance:
    1. Risk shall be properly understood in order to drive appropriate assurance decisions
    2. Risk concerns shall be aligned across all stakeholders and all interconnected technology elements
    3. Dependencies shall not be trusted until proven trustworthy
    4. Attacks shall be expected
    5. Assurance requires effective coordination among all technology participants
    6. Assurance shall; be well planned and dynamic
    7. A means to measure and audit overall assurance shall be built in\

Book: “Cybersecurity Engineering: …”

Privacy by design



1 **Proactive not reactive—preventative not remedial**
Anticipate, identify, and prevent invasive events before they happen; this means taking action before the fact, not afterward.

2 **Lead with privacy as the default setting**
Ensure personal data is automatically protected in all IT systems or business practices, with no added action required by any individual.

3 **Embed privacy into design**
Privacy measures should not be add-ons, but fully integrated components of the system.

4 **Retain full functionality (positive-sum, not zero-sum)**
Privacy by Design employs a "win-win" approach to all legitimate system design goals; that is, both privacy and security are important, and no unnecessary trade-offs need to be made to achieve both.

5 **Ensure end-to-end security**
Data lifecycle security means all data should be securely retained as needed and destroyed when no longer needed.

6 **Maintain visibility and transparency—keep it open**
Assure stakeholders that business practices and technologies are operating according to objectives and subject to independent verification.

7 **Respect user privacy—keep it user-centric**
Keep things user-centric; individual privacy interests must be supported by strong privacy defaults, appropriate notice, and user-friendly options.

By Ann Covoukian

https://www.ryerson.ca/pbdce/certification/seven-foundational-principles-of-privacy-by-design/

https://iab.org/wp-content/IAB-uploads/2011/03/fred_carter.pdf

Deloitte

https://www2.deloitte.com/content/dam/Deloitte/ca/Documents/risk/ca-en-ers-privacy-by-design-brochure.PDF

Summary

- Highly complex systems on which increasing dependence
- Secure-by-design & privacy-by-design
  - Increasingly crucial for trustworthy Computing and Information infrastructures