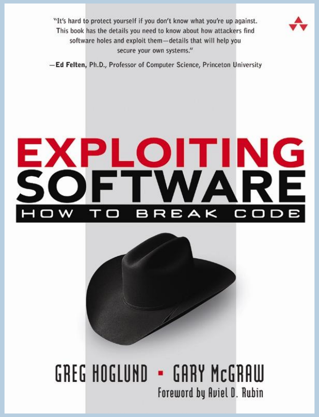
Exploiting Software: How to Break Code

Gary McGraw, Ph.D.
CTO, Cigital
http://www.cigital.com


"It's hard to protect yourself if you don't know what you're up against. This book has the details you need to know about how attackers find software holes and exploit them—details that will help you secure your own systems."

—**Ed Felten,** Ph.D., Professor of Computer Science, Princeton University

EXPLOITING SOFTWARE
HOW TO BREAK CODE

GREG HOGLUND ▪ GARY McGRAW
Foreword by Aviel D. Rubin

Pop quiz

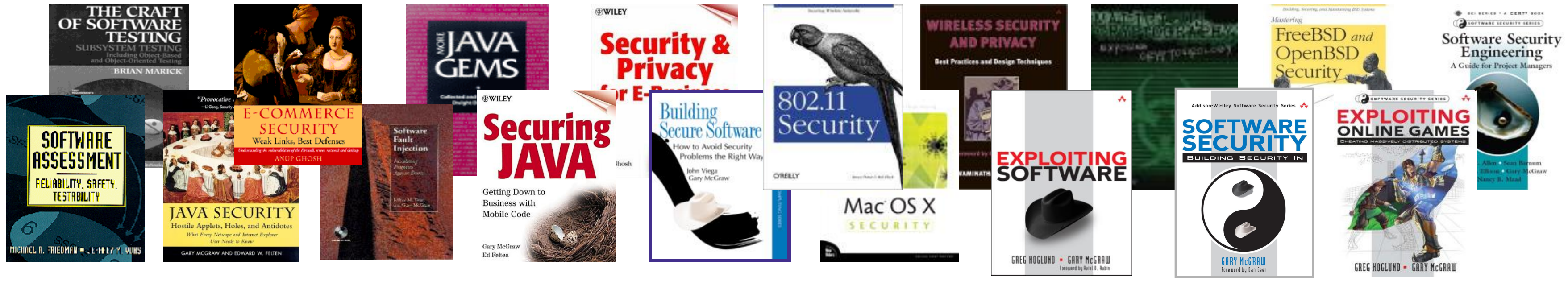
- What do wireless devices, cell phones, PDAs, browsers, operating systems, servers, routers, personal computers, public key infrastructure systems, and firewalls have in common?

badness-ometer

Software

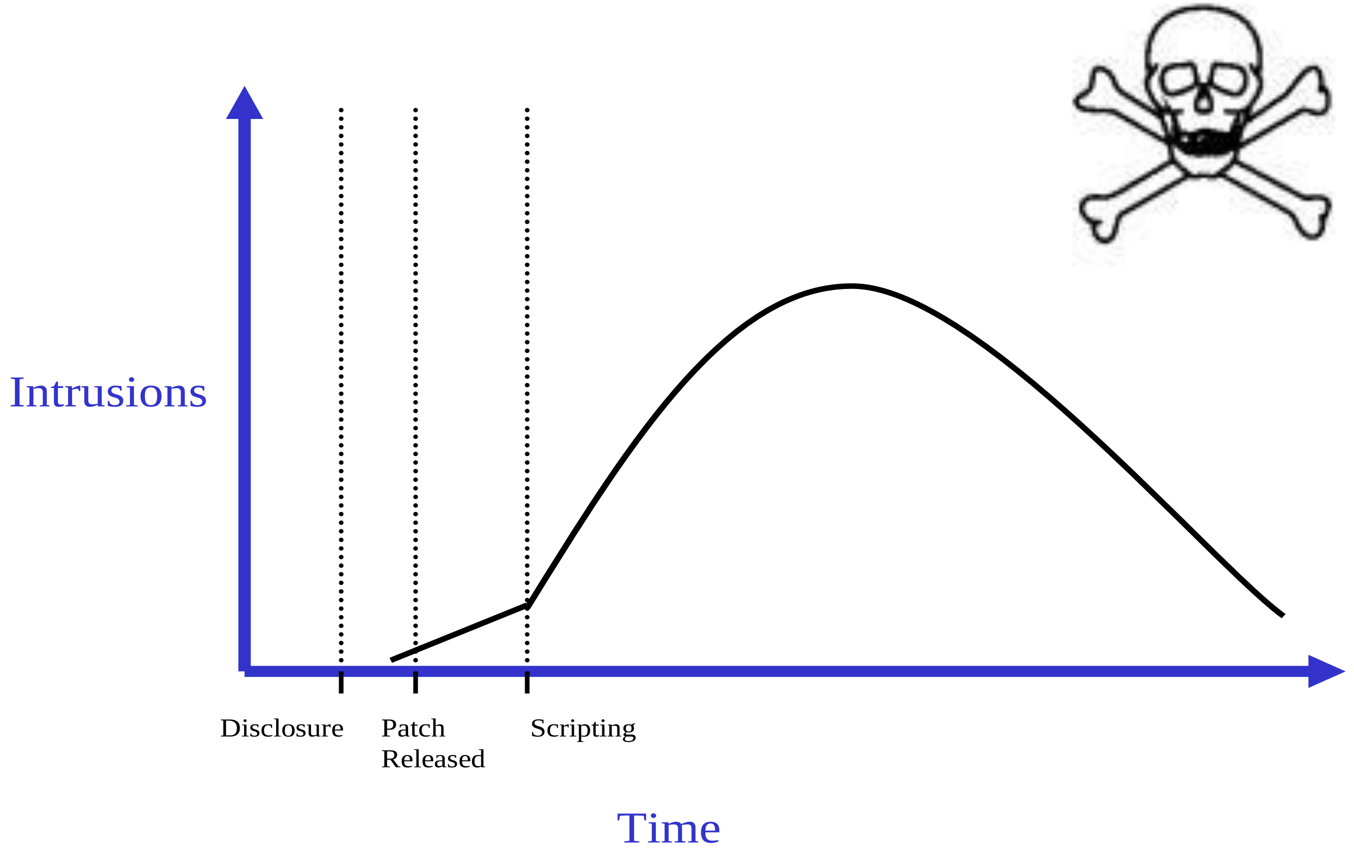
Cigital

- Founded in 1992 to provide software security and software quality professional services
- Recognized experts in software security and software quality
  - Widely published in books, white papers, and articles
  - Industry thought leaders



So what’s the problem?

Patches are attack maps



Builders versus operators

- Most security people are operations people
  - Network administrators
  - Firewall rules manipulators
  - COTS products glommers
  - These people need training

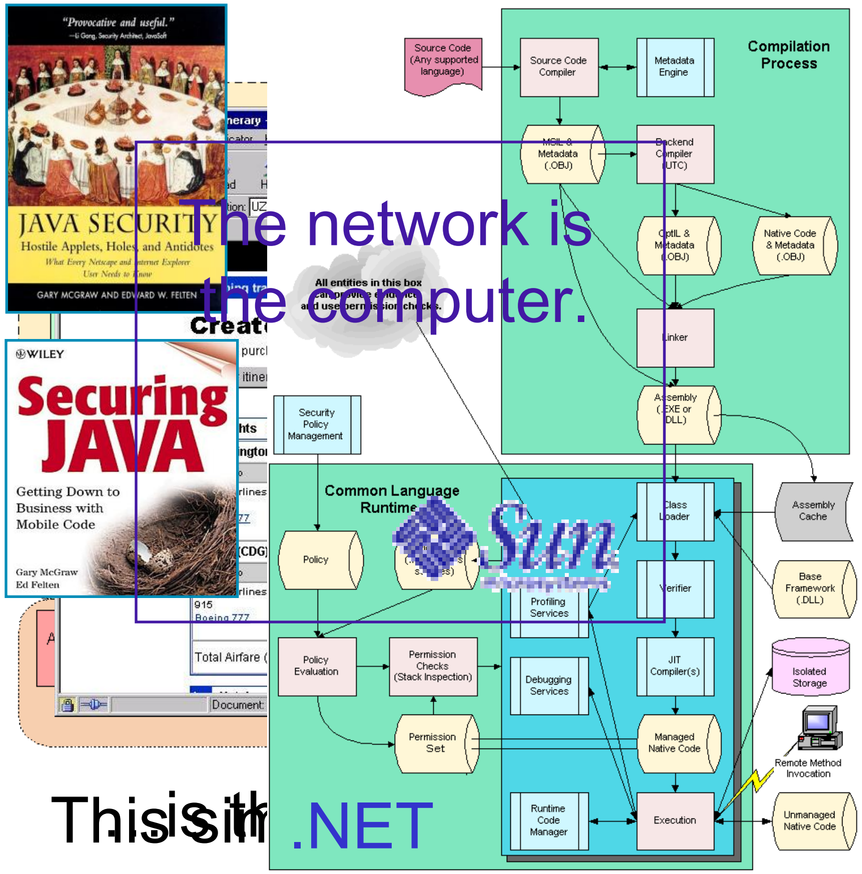
Security means different things to different people

- Most builders are not security people
  - Software development remains a black art
  - How well are we doing teaching students to engineer code?
  - Emergent properties like security are hard for builders to grok
  - These people need academic education

Attaining software security gets harder

The Trinity of Trouble

- **Connectivity**
  - The Internet is everywhere and most software is on it
- **Complexity**
  - Networked, distributed, mobile code is hard
- **Extensibility**
  - Systems evolve in unexpected ways and are changed on the fly



Who is the bad guy?

- Hackers
  - "Full disclosure" zealots
- "Script kiddies"
- Cyber criminals
  - Lone guns or organized
- Malicious insiders
  - Compiler wielders
- Business competition
- Police, press, terrorists, intelligence agencies

Above: Deputy Attorney General Christopher Bubb, center, N.J. Gov. Christie Whitman (left) and Attorney General Peter Verniero conferencing on alleged virus writer David L. Smith (right).

History is quirky

1995

- Dan Farmer fired from Silicon Graphics for releasing SATAN with Wietse Venema
- FUD: possible attack tool!

Fall 2004

- John Aycock at University of Calgary publicly criticized for malware course
- FUD: possible bad guy factory

2009

- Any system administrator not using a port scanner to check security posture runs the risk of being fired

Should we talk about attacking systems?

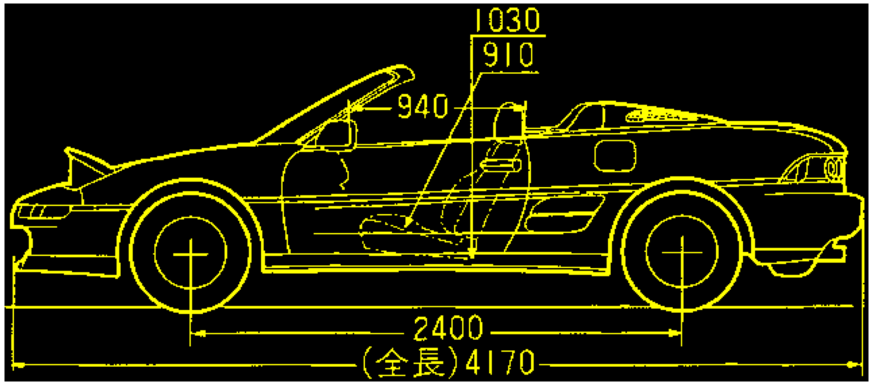
The good news and the bad news

Good news

- The world loves to talk about how stuff breaks
- This kind of work sparks lots of interest in computer security

Bad news

- The world would rather not focus on how to build stuff that does not break
- It's harder to build good stuff than to break junky stuff



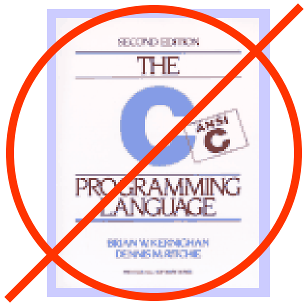
Know your enemy: How stuff breaks

Security problems are complicated

IMPLEMENTATION BUGS

- Buffer overflow
  - String format
  - One-stage attacks
- Race conditions
  - TOCTOU (time of check to time of use)
- Unsafe environment variables
- Unsafe system calls
  - System()
- Untrusted input problems

50%



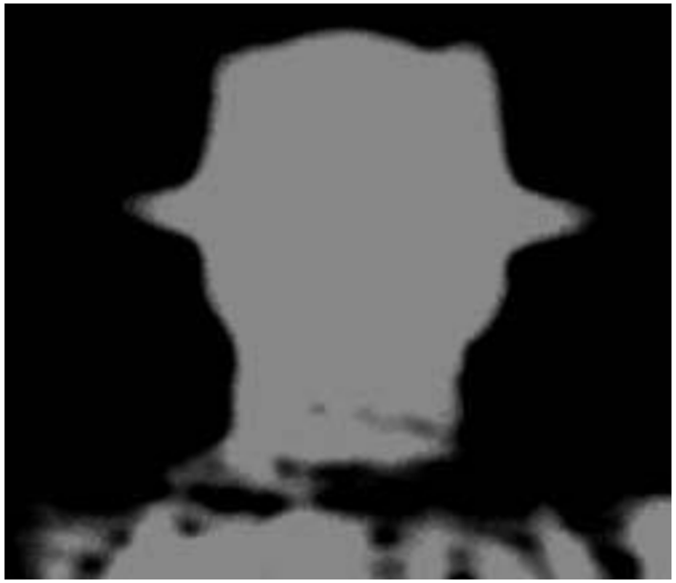
ARCHITECTURAL FLAWS

- Misuse of cryptography
- Compartmentalization problems in design
- Privileged block protection failure (DoPrivilege())
- Catastrophic security failure (fragility)
- Type safety confusion error
- Insecure auditing
- Broken or illogical access control (RBAC over tiers)
- Method over-riding problems (subclass issues)
- Signing too much code

50%

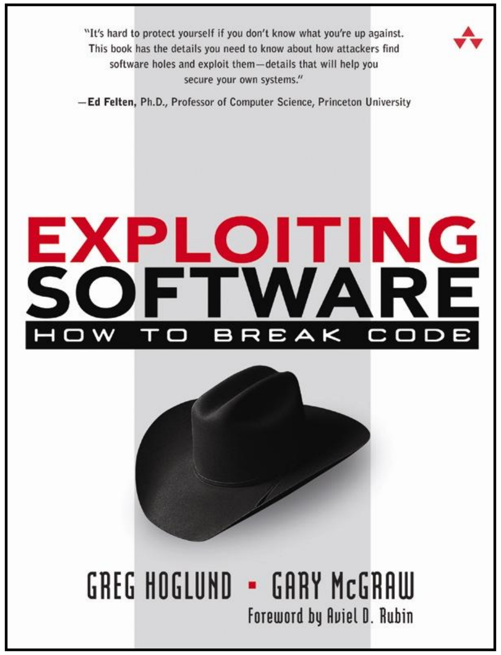
Attackers do not distinguish bugs and flaws

- Both bugs and flaws lead to vulnerabilities that can be exploited
- Attackers write code to break code
- Defenders are network operations people
  - Code?! What code?



The attacker’s toolkit

- The standard attacker’s toolkit has lots of (software analysis) stuff
  - Disassemblers and decompilers
  - Control flow and coverage tools
  - APISPY32
  - Breakpoint setters and monitors
  - Buffer overflow
  - Shell code
  - Rootkits



"It's hard to protect yourself if you don't know what you're up against. This book has the details you need to know about how attackers find software holes and exploit them—details that will help you secure your own systems."

—Ed Felten, Ph.D., Professor of Computer Science, Princeton University

EXPLOITING SOFTWARE

HOW TO BREAK CODE

GREG HOGLUND ▪ GARY McGRAW

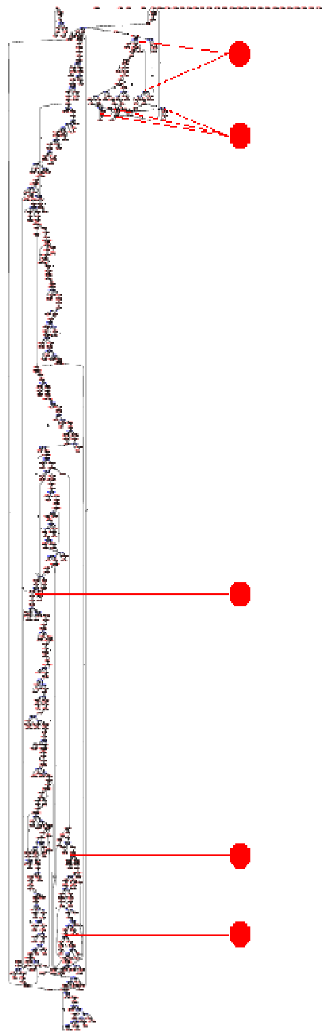
Foreword by Aviel D. Rubin

Attacker’s toolkit: dissasemblers and decompilers

- Source code is not a necessity for software exploit
- Binary is just as easy to understand as source code
- Disassemblers and decompilers are essential tools
- Reverse engineering is common and must be understood (not outlawed)
- IDA allows plugins to be created
- Use bulk auditing

Attacker's toolkit: control flow and coverage

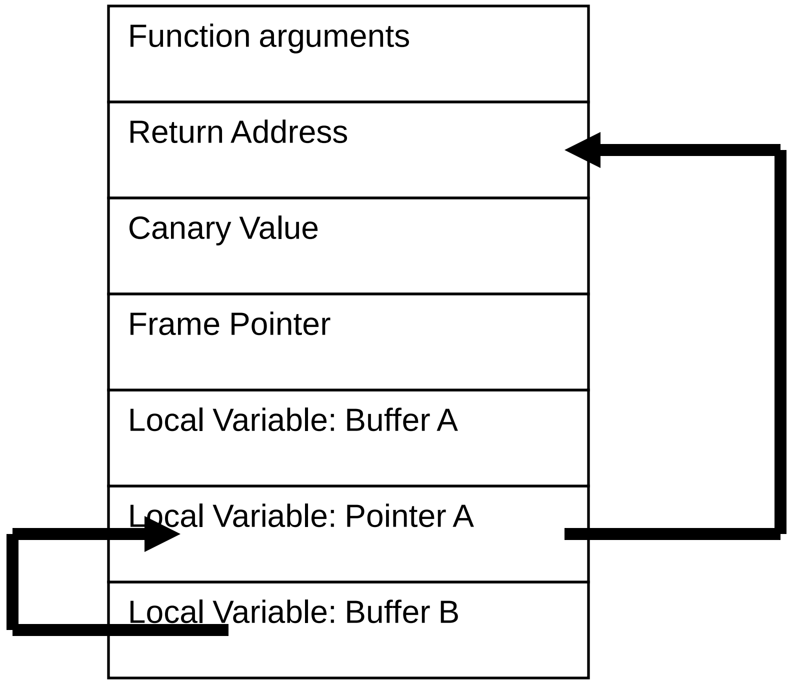
- Tracing input as it flows through software is an excellent method
- Exploiting differences between versions is also common
- Code coverage tools help you know where you have gotten in a program
  - dyninstAPI (Maryland)
  - Figure out how to get to particular system calls
  - Look for data in shared buffers



Attacker’s toolkit: buffer overflow foo

- Find targets with static analysis
- Change program control flow
  - Heap attacks
  - Stack smashing
  - Trampolining
  - Arc injection
- Particular examples
  - Overflow binary resource files (used against Netscape)
  - Overflow variables and tags (Yamaha MidiPlug)
  - MIME conversion fun (Sendmail)
  - HTTP cookies (apache)

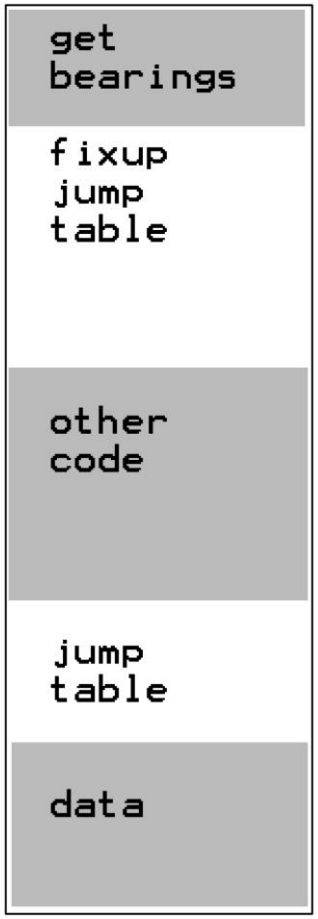
- Trampolining past a canary



Attacker's toolkit: shell code and other payloads

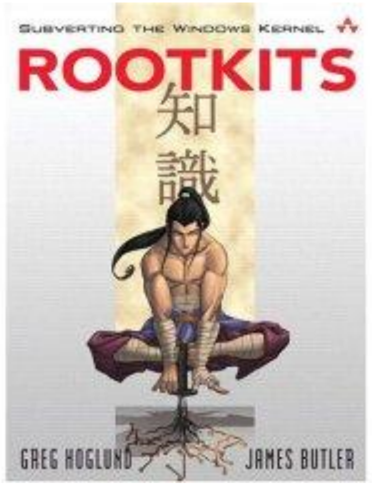
- Common payloads in buffer overflow attacks
- Size matters (small is critical)
- Avoid zeros
- XOR protection (also simple crypto)

- Payloads exist for
  - X86 (win32)
  - RISC (MIPS and sparc)
  - Multiplatform payloads



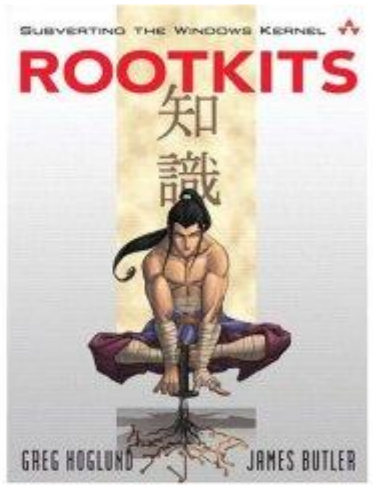
Attacker's toolkit: rootkits

- The apex of software exploit…complete control of the machine
- Live in the kernel
  - XP kernel rootkit in the book
  - See http://www.rootkit.com
- Hide files and directories by controlling access to process tables
- Provide control and access over the network
- Get into the EEPROM (hardware viruses)



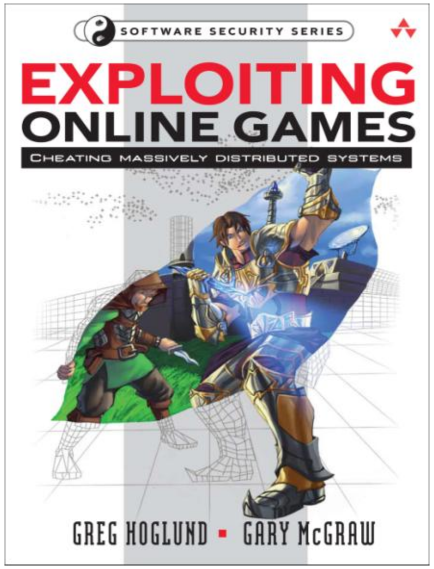
Example: Advanced game hacking fu

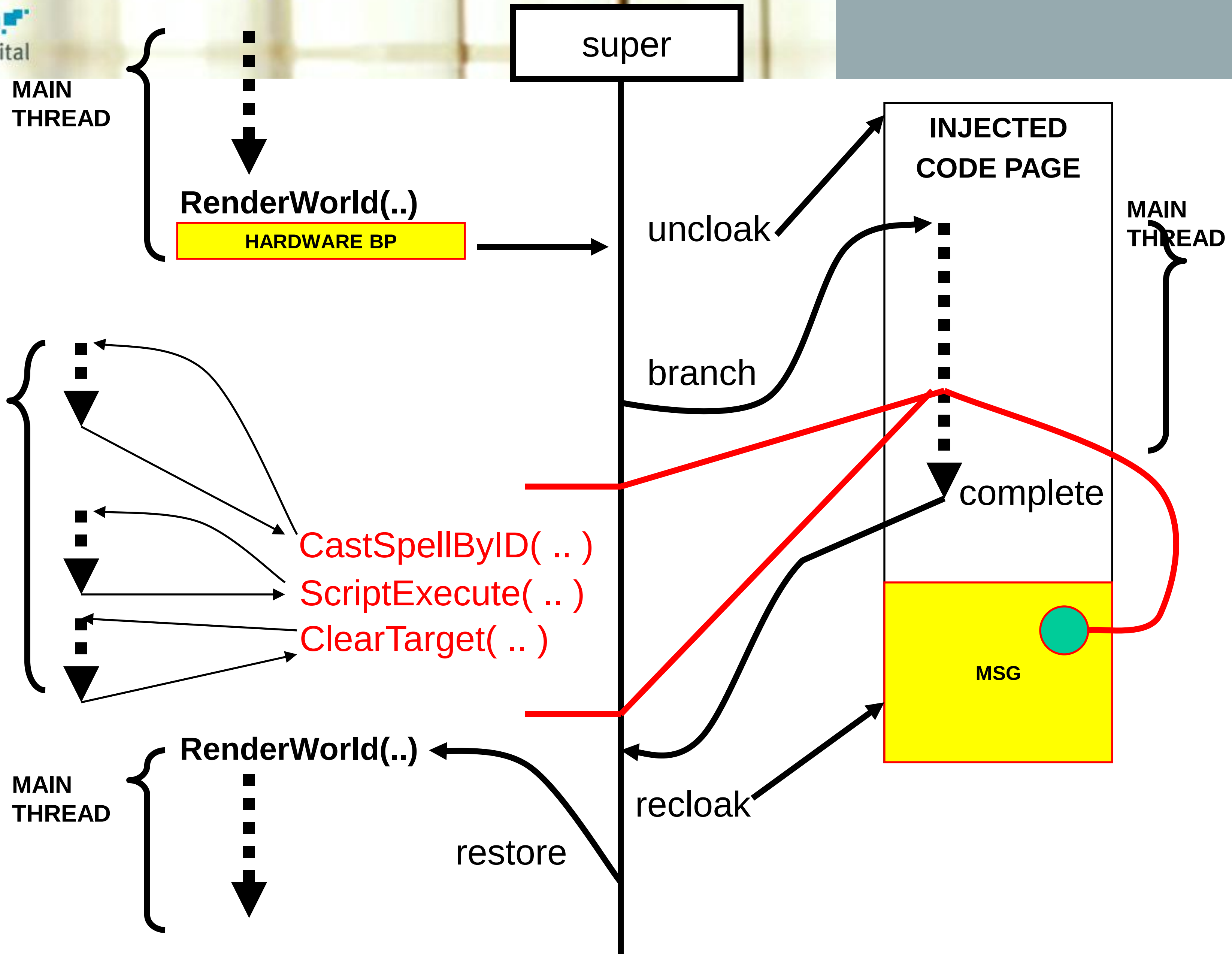
- See *Hacking World of Warcraft: An exercise in advanced rootkit development*
  - Greg Hoglund's presentation from Black Hat 2006
  - http://www.rootkit.com/vault/hoglund/GregSlidesWoWHack.rar



State of the art

- Combine injected payload with cloaking and thread hijacking to FORCE in-game events
  - Spell casting
  - Movement
  - Chat
  - Acquire and clear targets
  - Loot inventory




super
MAIN THREAD
RenderWorld(..)
HARDWARE BP
uncloak
INJECTED CODE PAGE
MAIN THREAD
branch
complete
CastSpellByID(..)
ScriptExecute(..)
ClearTarget(..)
MSG
RenderWorld(..)
MAIN THREAD
recloak
restore

Attacker's toolkit: other miscellaneous tools

- Debuggers (user-mode)
- Kernel debuggers
  - SoftIce
- Fault injection tools
  - FUZZ
  - Failure simulation tool
  - Hailstorm
  - Holodeck
- Boron tagging
- The "depends" tool
- Grammar rewriters

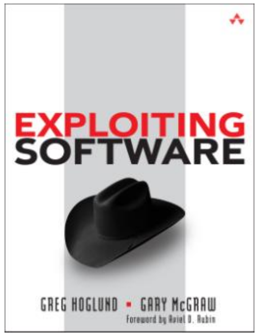
How attacks unfold

- The standard process
  - Scan network
  - Build a network map
  - Pick target system
  - Identify OS stack
  - Port scan
  - Determine target components
  - Choose attack patterns
  - Break software
  - Plant backdoor

- Attacking a software system is a process of discovery and exploration
  - Qualify target (focus on input points)
  - Determine what transactions the input points allow
  - Apply relevant attack patterns
  - Cycle through observation loop
  - Find vulnerability
  - Build an exploit

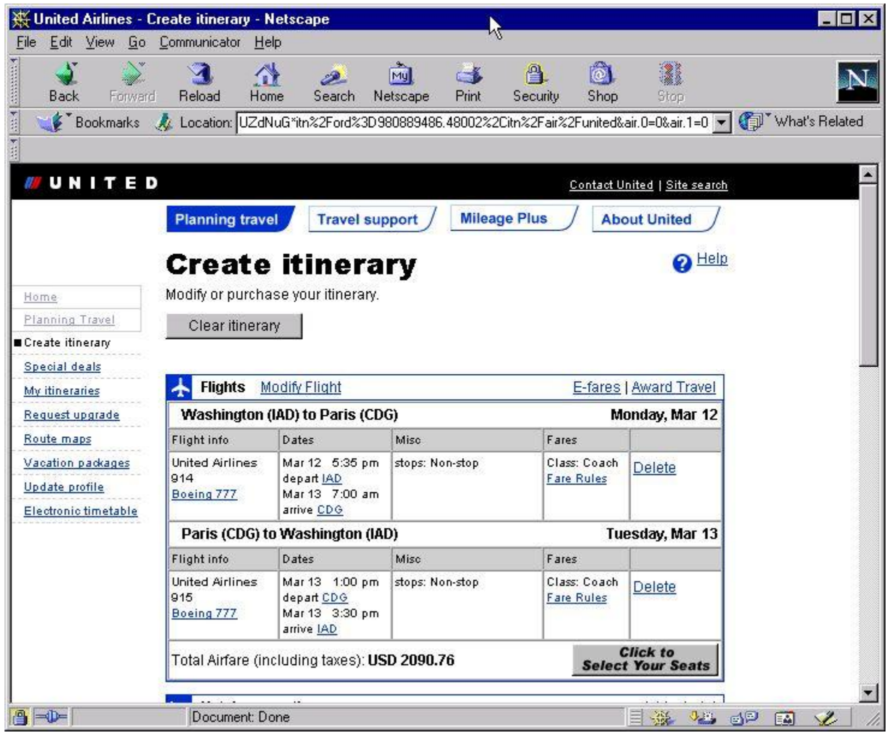
Knowledge: 48 Attack Patterns

- Make the Client Invisible
- Target Programs That Write to Privileged OS Resources
- Use a User-Supplied Configuration File to Run Commands That Elevate Privilege
- Make Use of Configuration File Search Paths
- Direct Access to Executable Files
- Embedding Scripts within Scripts
- Leverage Executable Code in Nonexecutable Files
- Argument Injection
- Command Delimiters
- Multiple Parsers and Double Escapes
- User-Supplied Variable Passed to File System Calls
- Postfix NULL Terminator
- Postfix, Null Terminate, and Backslash
- Relative Path Traversal
- Client-Controlled Environment Variables
- User-Supplied Global Variables (DEBUG=1, PHP Globals, and So Forth)
- Session ID, Resource ID, and Blind Trust
- Analog In-Band Switching Signals (aka “Blue Boxing”)
- Attack Pattern Fragment: Manipulating Terminal Devices
- Simple Script Injection
- Embedding Script in Nonscript Elements
- XSS in HTTP Headers
- HTTP Query Strings
- User-Controlled Filename
- Passing Local Filenames to Functions That Expect a URL
- Meta-characters in E-mail Header
- File System Function Injection, Content Based
- Client-side Injection, Buffer Overflow
- Cause Web Server Misclassification
- Alternate Encoding the Leading Ghost Characters
- Using Slashes in Alternate Encoding
- Using Escaped Slashes in Alternate Encoding
- Unicode Encoding
- UTF-8 Encoding
- URL Encoding
- Alternative IP Addresses
- Slashes and URL Encoding Combined
- Web Logs
- Overflow Binary Resource File
- Overflow Variables and Tags
- Overflow Symbolic Links
- MIME Conversion
- HTTP Cookies
- Filter Failure through Buffer Overflow
- Buffer Overflow with Environment Variables
- Buffer Overflow in an API Call
- Buffer Overflow in Local Command-Line Utilities
- Parameter Expansion
- String Format Overflow in syslog()



Attack pattern 1: Make the client invisible

- Remove the client from the communications loop and talk directly to the server
- Leverage incorrect trust model (never trust the client)
- Example: hacking browsers that lie (opera cookie foo)


United Airlines - Create itinerary - Netscape

File Edit View Go Communicator Help

Back Forward Reload Home Search Netscape Print Security Shop Stop

Bookmarks Location: UZdNuG*itn%2Ford%3D980889486.48002%2Citn%2Fair%2Funited&air.0=0&air.1=0 What's Related

UNITED — Contact United | Site search

Planning travel | Travel support | Mileage Plus | About United

Home
Planning Travel
Create itinerary
Special deals
My itineraries
Request upgrade
Route maps
Vacation packages
Update profile
Electronic timetable

Create itinerary — Help

Modify or purchase your itinerary.

Clear itinerary

Flights Modify Flight — E-fares | Award Travel

Washington (IAD) to Paris (CDG) — **Monday, Mar 12**

Flight info	Dates	Misc	Fares	
United Airlines 914 Boeing 777	Mar 12 5:35 pm depart IAD Mar 13 7:00 am arrive CDG	stops: Non-stop	Class: Coach Fare Rules	Delete

Paris (CDG) to Washington (IAD) — **Tuesday, Mar 13**

Flight info	Dates	Misc	Fares	
United Airlines 915 Boeing 777	Mar 13 1:00 pm depart CDG Mar 13 3:30 pm arrive IAD	stops: Non-stop	Class: Coach Fare Rules	Delete

Total Airfare (including taxes): **USD 2090.76**

Click to Select Your Seats

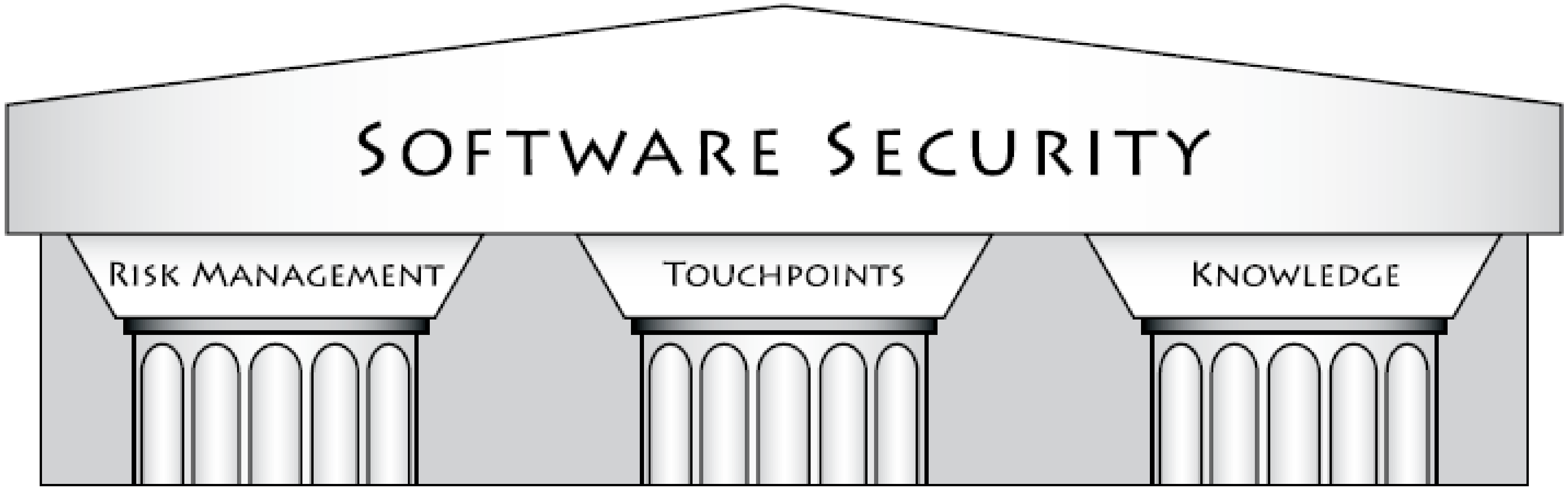
Document: Done

Breaking stuff is important

The Shellcoder's Handbook
Discovering and Exploiting Security Holes
Jack Koziol\David Litchfield\
Sinan Eren\Neel Mehta\Riley H
"It's hard to protect yourself if you don't know what you're up against. This book has the details you need to know about how attackers find software holes and exploit them—details that will help you secure your own systems."
—Ed Felten, Ph.D., Professor of Computer Science, Princeton University
EXPLOITING SOFTWARE
HOW TO BREAK CODE
GREG HOGLUND · GARY McGRAW
Foreword by Aviel D. Rubin

- Learning how to think like an attacker is essential
- Do not shy away from discussing attacks
  - Engineers learn from stories of failure
- Attacking projects is useful

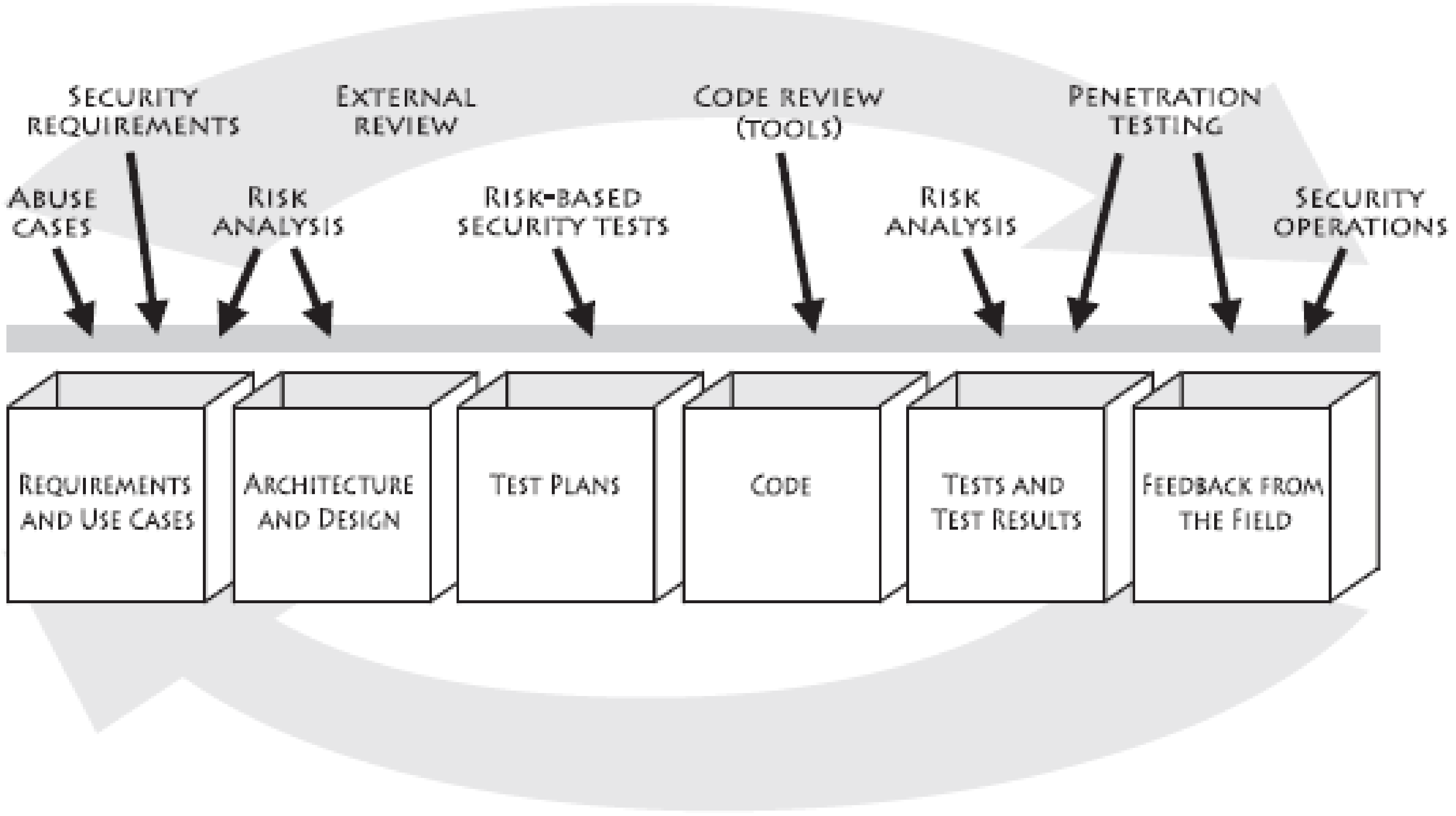
Great, now what do we do about this?



Three pillars of software security

1. Risk management framework
2. Touchpoints
3. Knowledge

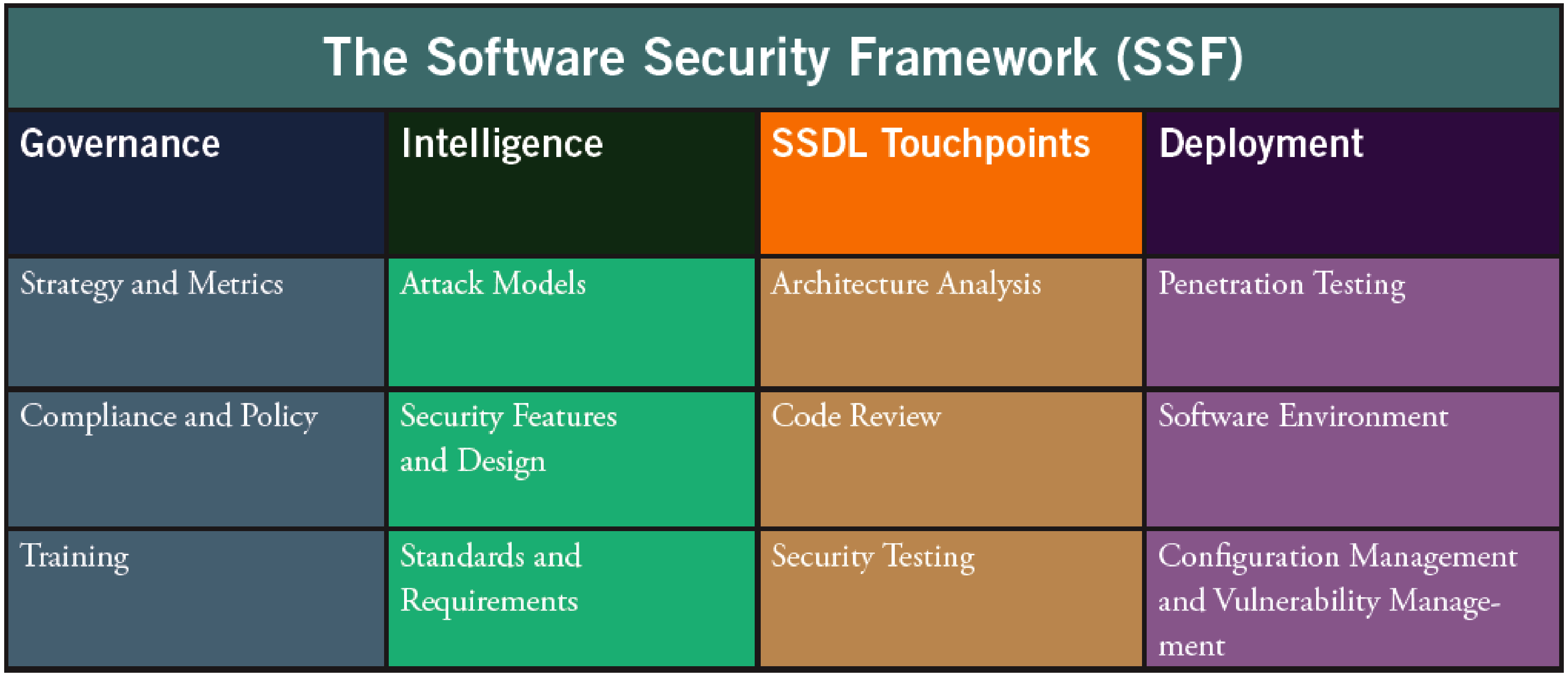
Software security touchpoints



What works: BSIMM

- Building Security In Maturity Model
- Real data from real initiatives

A Software Security Framework



The Software Security Framework (SSF)			
Governance	**Intelligence**	**SSDL Touchpoints**	**Deployment**
Strategy and Metrics	Attack Models	Architecture Analysis	Penetration Testing
Compliance and Policy	Security Features and Design	Code Review	Software Environment
Training	Standards and Requirements	Security Testing	Configuration Management and Vulnerability Management

- Twelve practices
- See informIT article at http://www.informit.com/articles/article.aspx?p=1271382

Ten surprising things

1. Bad metrics hurt
2. Secure-by default frameworks
3. Nobody uses WAFs
4. QA can’t do software security
5. Evangelize over audit
6. ARA is hard
7. Practitioners don’t talk attacks
8. Training is advanced
9. Pen testing is diminishing
10. Fuzz testing

- http://www.informit.com/articles/article.aspx?p=1315431

Where to Learn More

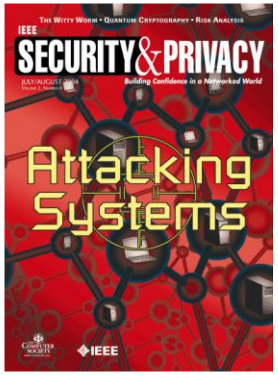
informIT & Justice League

- www.cigital.com/justiceleague
- In-depth thought leadership blog from the Cigital Principals
  - Scott Matsumoto
  - Gary McGraw
  - Sammy Migues
  - Craig Miller
  - John Steven

- www.informIT.com
- No-nonsense monthly security column by Gary McGraw

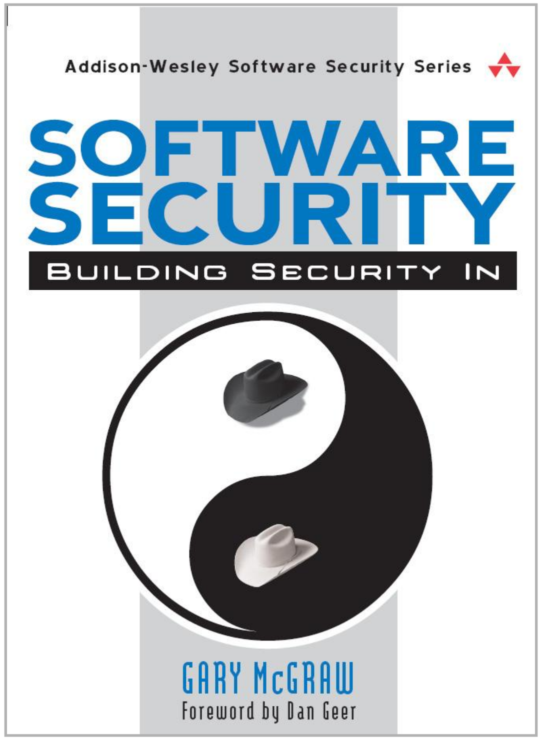
IEEE Security & Privacy Magazine + 2 Podcasts

- Building Security In
- Software Security Best Practices column edited by John Steven
- www.computer.org/security/bsisub/

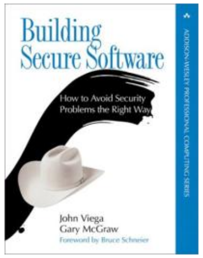


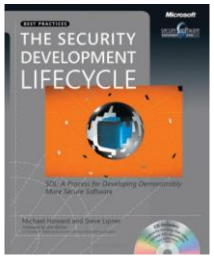
- www.cigital.com/silverbullet
- www.cigital.com/realitycheck

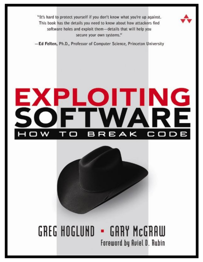
Software Security: the book



- How to DO software security
  - Best practices
  - Tools
  - Knowledge
- Cornerstone of the Addison-Wesley Software Security Series
- www.swsec.com


Building Secure Software
How to Avoid Security Problems the Right Way
John Viega
Gary McGraw
Foreword by Bruce Schneier

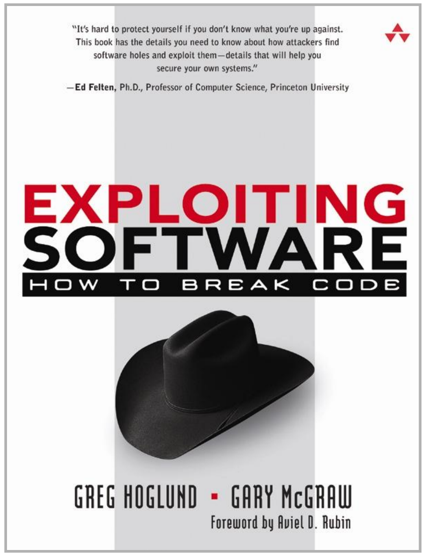




For more

- Cigital's Software Security Group invents and delivers Software Quality Management
- See the Addison-Wesley Software Security series
- WE NEED GREAT PEOPLE
- Send e-mail: gem@cigital.com

"So now, when we face a choice between adding features and resolving security issues, we need to choose security."
-Bill Gates


"It's hard to protect yourself if you don't know what you're up against. This book has the details you need to know about how attackers find software holes and exploit them—details that will help you secure your own systems."
—Ed Felten, Ph.D., Professor of Computer Science, Princeton University

EXPLOITING SOFTWARE
HOW TO BREAK CODE

GREG HOGLUND ▪ GARY McGRAW
Foreword by Aviel D. Rubin