Hi, thanks for inviting me.

Abstract: Everyone seems to agree that the software ecosystem isn't producing terribly secure code… even though we know quite a lot about finding vulnerabilities, security architecture, threat modeling, risk analysis, and secure development lifecycles. Jeff will argue that we know almost nothing about what actually changes software culture. He'll share his experiences with several brilliant but tragically flawed security efforts from ancient history. He'll also reflect on his successes and failures leading the Open Web Application Security Project (OWASP) to drive change in the software industry.

High Confidence Software and Systems (HCSS) Conference being held in Annapolis May 1-6. The HCSS is a highly technical conference that focuses on innovative research and technology to advance the cause of building highly secure software. It is organized by the NSA R&D community and typically has an attendance of a couple of hundred people (mostly researchers and technical practitioners) as opposed to program managers.

The Day! "practical technology that is available "today" to increase the assurance posture of software." The CAS has been given a "day" to focus on practical technology that is available "today" to increase the assurance posture of software. (Again, the work at OWASP continues to be at the forefront of "publically available" tools and techniques). We thought a talk on your involvement in the "Rugged Software" work (or your take on it) along with some practical "tidbits" of what's going on in OWASP (and Aspect) would make for a great intro to the day.

Hacking
My first teen crush with "Hacking" started in 1982. I had a Radio Shack Color Computer and I bought these videogames that came on cassettes. But after a while the tapes would get worn out and I couldn't back them up because they were copy-protected. My tried hooking together two tape recorders, but the audio quality was awful.

Then one day, "Hacking" walks in, leans up next to me and says "I know you can break that copy-protection." I couldn't resist her. I found that all these games would all load a small boot loader which would then load the actual program using a whole bunch of crazy tricks. We would trace through all the tricks and eventually we'd find a jump to the start of the program, and that's where I would hook and save off a copy of the actual program. Hacking was smart, she was sexy, she was fun. And she was a secret. My parents definitely did not find out we were dating.

But like most teen romances, it was over fast.

“Disclosure”
bea weblogic
YOU DON’T GET TO 500 MILLION FRIENDS WITHOUT MAKING A FEW ENEMIES

Disclosure
Years later, I was working with a dot-com company that wanted to run multiple customer programs in the same instance of WebLogic. So we were hoping to use the SecurityManager, a strong policy, code signing, and custom classloading to keep them from being able to attack each other. But it wasn’t working, so I started digging and I found a serious vulnerability in WebLogic.

So I called up their support number. Now, I was expecting a “thank you” for my research like Zuckerberg in The Social Network. But strangely, that’s not how they treated me on the phone. I started getting a little heated, and that’s when I met my next girlfriend “Disclosure” <CLICK>. She leaned in close and whispered in my ear. She told me “tell them you’re going to post the issue to the full-disclosure mailing list.” So that’s what did.

As it turns out, Disclosure and I had one great night together. But when my CEO walked into the lab the next morning, she was long gone. As it turns out, he wasn’t very happy. Apparently he had received an angry call from the CEO of WebLogic who didn’t want this vulnerability released.

Turns out that the grownups weren’t really happy that I was dating Disclosure at all. I’ve gone out with here a few times over the years, and it just never works out. <CLICK>

ANGELINA JOLIE
SALT
WHO IS SALT?
SUMMER 2010
WhoIsSalt.com
"Policy"
DEPARTMENT OF DEFENSE STANDARD
DEPARTMENT OF DEFENSE TRUSTED COMPUTER SYSTEM EVALUATION CRITERIA
DECEMBER 1985
digital
VT220

Policy
My first grownup relationship was with Policy. I was working on the user interface for a Navy application written in Pascal running on Digital VAX/VMS. The system targeted Orange Book B2 and I learned about Multi Level Security. Like some of you I found the MLS policy simple and beautiful and I fell in love for the first time.

I got the opportunity port a major Naval platform tracking system to the brand new Compartmented Mode Workstation (now called Trusted Solaris). MLS was beautiful, but I found out that she was a little high-maintenance. Ivana Trump is lower maintenance. The label encodings wouldn't work, processes failed silently, logging was complex, and on and on.

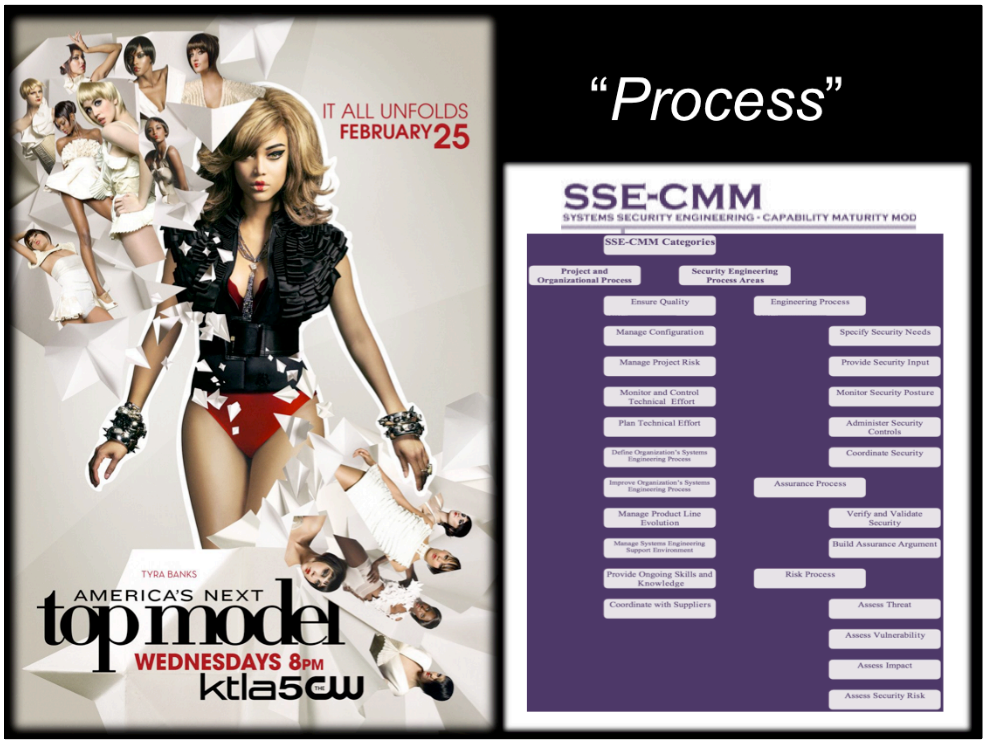
So I broke up with her. I actually never see her around. I've dated her sister, Same-Origin, a lot – and I see her everywhere. But those girls are just too complex for me.

"Assurance"
Common Criteria
EAL 1
Functionally tested
EAL 2
Structurally tested
EAL 3
Methodically tested and checked
EAL 4
Methodically designed, tested, and reviewed
EAL 5
Semi-formally designed and tested
EAL 6
Semi-formally verified, designed, and tested
EAL 7
Formally verified, designed, and tested
CC EAL 7+
THE HIGHEST EVALUATED PRODUCT IN THE WORLD
MICHAEL YORK
JENNIFER O'NEILL
ELIZABETH GRACEN
DEE WALLACE STONE
DISCRETION Assured
Every man's fantasy... One man's nightmare.

Assurance
Later, Marv Shaeffer (of Orange Book fame) introduced me to my next girlfriend, "Assurance". She was supposed to be very powerful in the US government, and she took me to all of these formal events. She had Test Plans, Traceability, Evidence! Minimal. Modular. Beautiful. I was in love again.

I brought her on a business trip with me to a copier company in Long Island. They wanted a CC Security Target for one of their copiers. Which isn't actually crazy because copiers store lots of sensitive information. But then I had to keep lowering and lowering the bar until the ST said that their copier did weak security logging and there was basically no assurance evidence.

When Assurance found out I was cheating on her, she went crazy and left me. I still love her, but I know that nobody will ever be good enough for her.

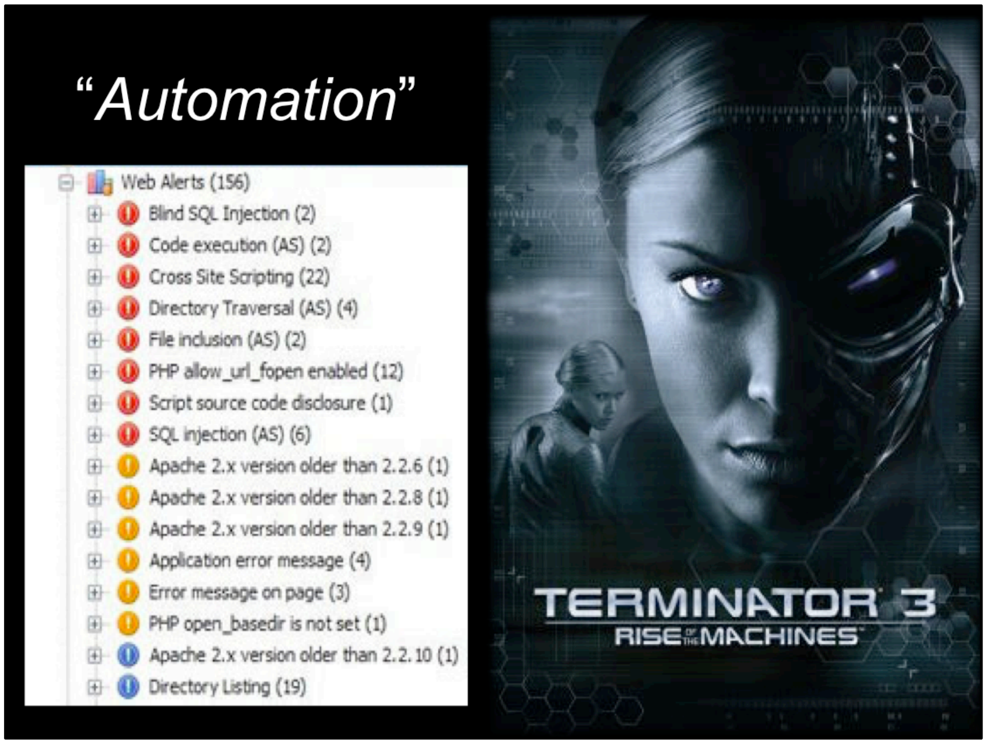

IT ALL UNFOLDS
FEBRUARY 25
TYRA BANKS
AMERICA'S NEXT
top model
WEDNESDAYS 8PM
ktla5 CW
“Process”
SSE-CMM
SYSTEMS SECURITY ENGINEERING - CAPABILITY MATURITY MOD
SSE-CMM Categories
Project and Organizational Process
Security Engineering Process Areas
Ensure Quality
Manage Configuration
Manage Project Risk
Monitor and Control Technical Effort
Plan Technical Effort
Define Organization's Systems Engineering Process
Improve Organization's Systems Engineering Process
Manage Product Line Evolution
Manage Systems Engineering Support Environment
Provide Ongoing Skills and Knowledge
Coordinate with Suppliers
Engineering Process
Specify Security Needs
Provide Security Input
Monitor Security Posture
Administer Security Controls
Coordinate Security
Assurance Process
Verify and Validate Security
Build Assurance Argument
Risk Process
Assess Threat
Assess Vulnerability
Assess Impact
Assess Security Risk

Process
In the late-90's I met “Process.” I led the team that wrote the SSE-CMM and our relationship changed me forever. But she was too demanding. She wanted to get married right away. She was always trying to “improve the relationship” and take our relationship to the “next level.”

Turns out that models make terrible girlfriends. There are always a bunch of fanboys hanging around that never do anything but talk about models. And models are really really shallow. They have no interest in what you're actually doing. They can't tell you how to actually do anything. They won't adapt, and they'll leave you if you don't follow them around.

I've built dozens of software assurance programs in all sorts of companies. I still bring “Process” along with me for everyone to look at, but she never actually does anything. I always have to create the a custom program for that particular organization. And all the while, she keeps reading off of her checklist about stuff I really *have* to do and why I need better documentation.

So I don't date models anymore. But she did change me, and I will always remember the lessons that she taught me about security culture, lifestyle, and discipline.



"Automation"

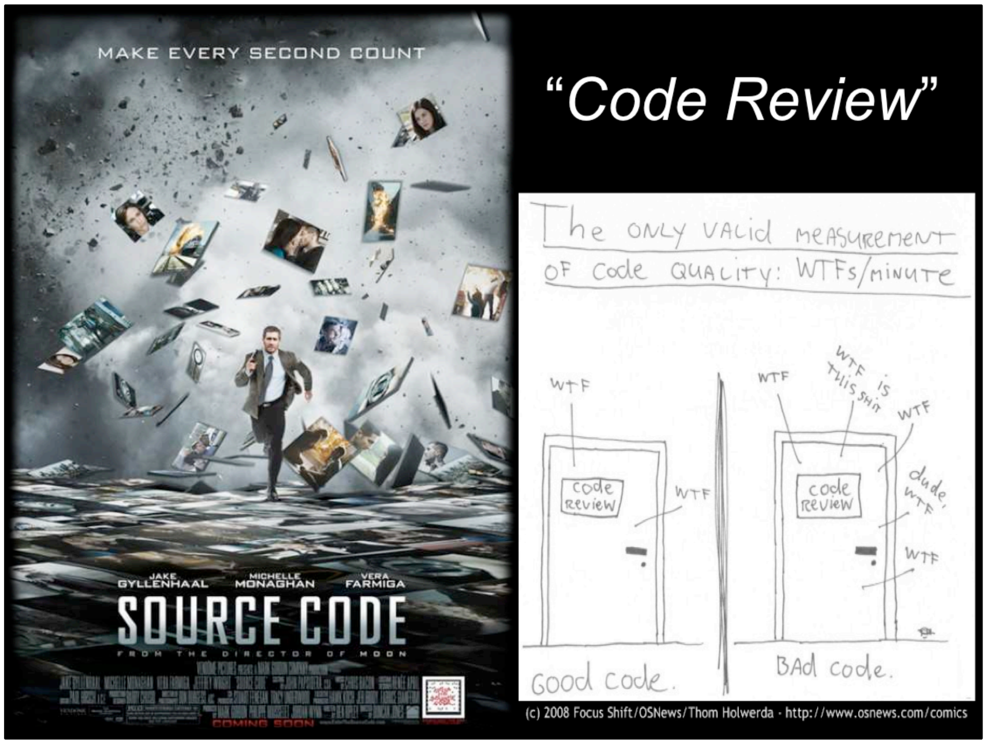
```
Web Alerts (156)
  Blind SQL Injection (2)
  Code execution (AS) (2)
  Cross Site Scripting (22)
  Directory Traversal (AS) (4)
  File inclusion (AS) (2)
  PHP allow_url_fopen enabled (12)
  Script source code disclosure (1)
  SQL injection (AS) (6)
  Apache 2.x version older than 2.2.6 (1)
  Apache 2.x version older than 2.2.8 (1)
  Apache 2.x version older than 2.2.9 (1)
  Application error message (4)
  Error message on page (3)
  PHP open_basedir is not set (1)
  Apache 2.x version older than 2.2.10 (1)
  Directory Listing (19)
```

Automation
Everyone seems to think that Automation would be a perfect match for me. I've tried to date her for years, but it always takes so much work to set up a date, and it never leads anywhere. First she takes forever to get ready, you have to teach her your rules, and she never seems to know where to enter or leave…

Also, I like to do lots of different things, but Automation only likes to do one thing. So I end up having to date a lot of her friends. And they don't get along with each other. I think they enjoy disagreeing with each other. They all love to talk, but the signal-to-noise ratio I so low that I frequently stop paying attention. It takes me forever to weed through all her emails, texts, and voicemails to find something useful.

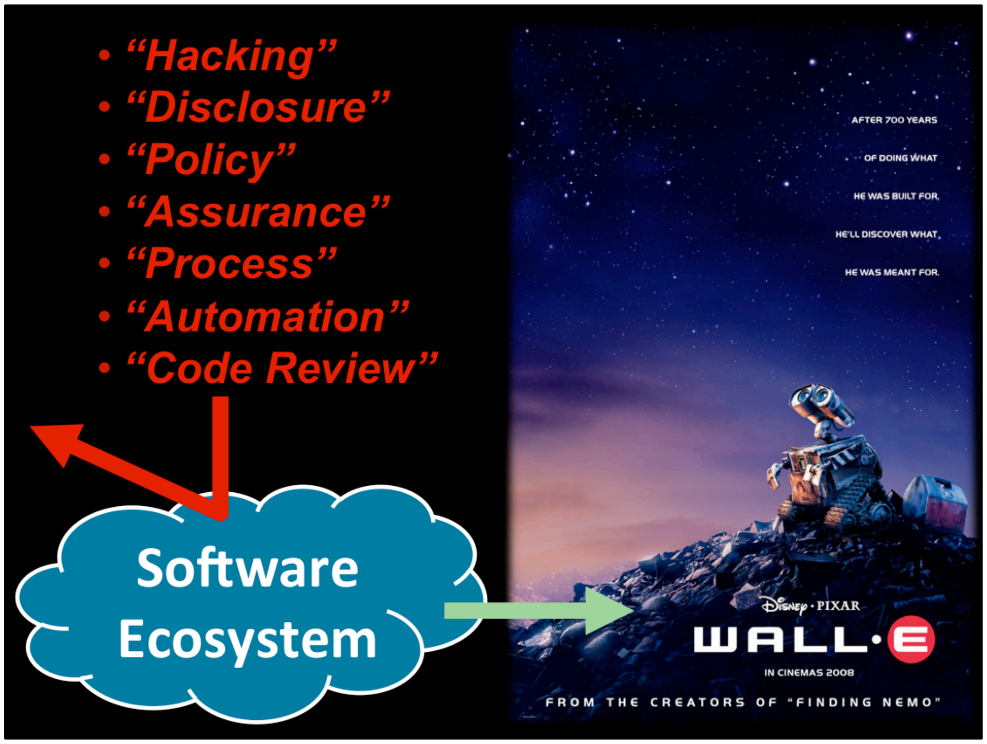
So Automation can be great, but it's just so exhausting.

Then when it doesn't work out, she tells me, "I'll never do that again. I can be better." But she never is.



Manual
For the last 10 years, I've had a great relationship with Code Review. I mean, I guess I shouldn't complain. She and I have reviewed the code for some of the world's most critical financial applications and discovered almost 5,000 vulnerabilities. These are huge applications, often several million lines of code but together we can break them down and verify security quickly. We work together really well.

Of all my relationships, I feel like Code Review makes the most sense. She's financially responsible and no-nonsense, and she's a very hard worker. But confidentially, our relationship isn't as exciting as it once was. So I don't think I ready to get married… It just seems like there must be a better way to do things.



The Future
Over the last 30 years, we have utterly failed to eliminate any of the categories of software vulnerabilities.

I don't think any of the girls I've dated over the years have any real chance of changing things – at least not by themselves. I think the software market is just way ahead of the security market. I'm not minimizing any of the unbelievable work that my girlfriends have done. But when you step back, you can see we're not being effective.

As each generation of technology comes out, we are still poking and prodding the last generation. Anyone here working on buffer overflow protection? We simply have to find ways of getting in front of the problem.

Problem #1 – We trust

There is something about software that people are willing to trust until it is proven to be insecure. Even in the face of massive evidence that new technologies are very likely to have flaws, we instinctively move to new technologies.

In fact, the Internet is the <u>most trusted</u> software environment of all time. Finances, healthcare, privacy, government, defense, energy…. Even if you don't want to, you are trusting your future to billions of lines of code that you know absolutely nothing about.

Which brings us to problem #2…

Problem #2: We blame

Have you noticed that developers are not exactly enamored of security people? It's not that developers don't want to write secure code. And it's not even that they don't want help.

No, it's because many "security" people aren't working towards improving security. It's the focus on finding holes and parading them around that undermines our credibility. The idea that continually poking holes in things is likely to lead to the creation of secure things doesn't pass the laugh test.

Right when we need to combine our skills with those of developers to create secure systems, some are calling to make developers liable for security mistakes. This is probably the most most divisive thing we could possibly do.

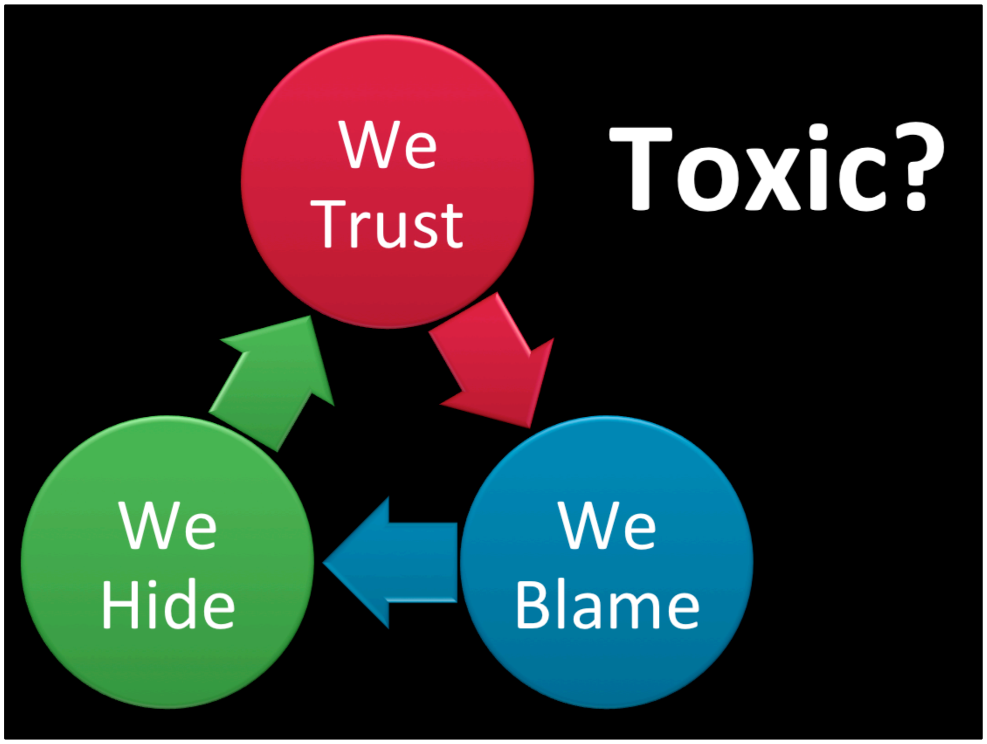
This blame is "acid rain" for security ecosystems. Which brings us to….

The first rule of security is…
…You do not talk about security

Problem #3: We hide security

The natural reaction of developers who are getting blamed for security problems is to hide details. They naturally do not want to participate in the security process or share the information that is needed to make security decisions.

As I mentioned there are virtually no applications in existence that provide a reasonable assurance case. That lack of information is a serious market failure that prevents buyers from using security as a buying criteria… and paradoxically makes it silly for vendors to sell secure code.



See how these three factors work together…

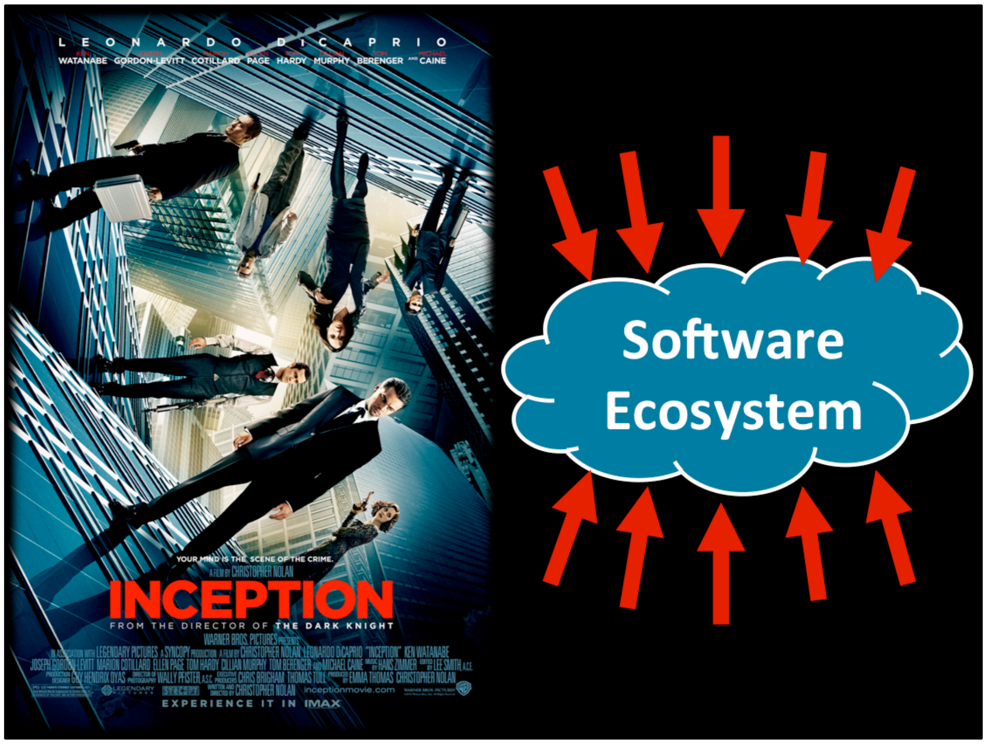
Our inherent trust for complex things with no obvious vulnerabilities leads to blame.

Blame leads to hiding.

And hiding leads to the inability to make informed decisions…. Which demands trust.

This culture is currently toxic to security ecosystems. It’s like trying to grow grass in my yard – totally impossible.

We need to create the conditions for security ecosystems to bloom and grow.

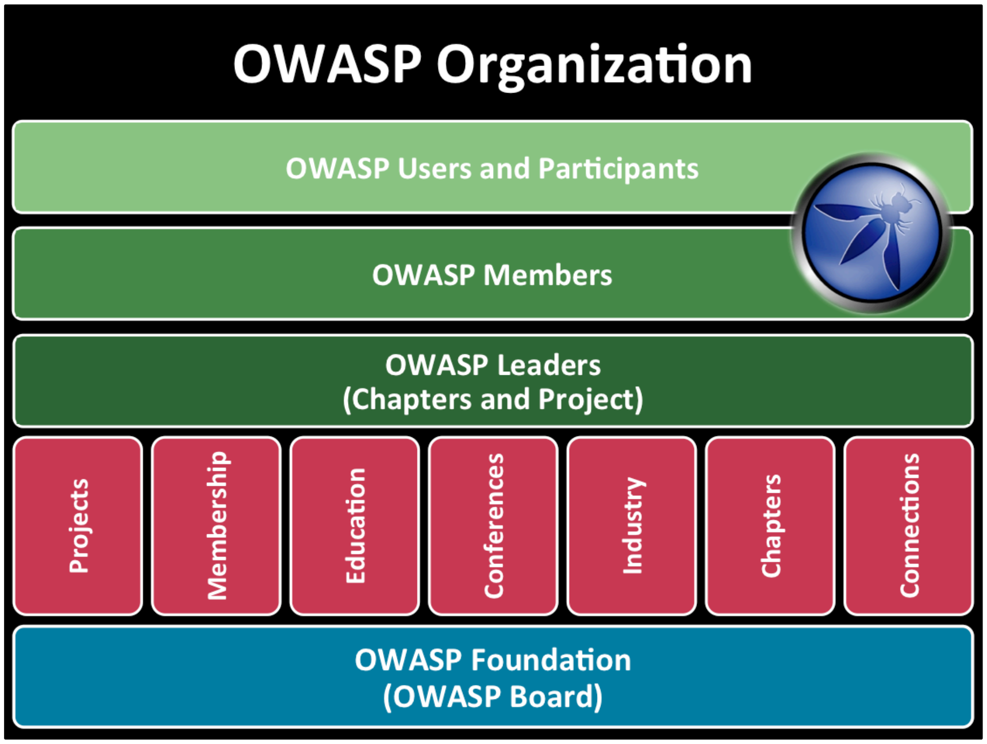

LEONARDO DICAPRIO
INCEPTION
YOUR MIND IS THE SCENE OF THE CRIME.
FROM THE DIRECTOR OF THE DARK KNIGHT
EXPERIENCE IT IN IMAX
Software Ecosystem

The Challenge
It makes me ask – how do we get security? What does it even mean? Schneier said security is a process, not a product. But if that was literally true, we could all just follow that process and be done. It's more than that to me.

So here's my challenge to you. Application security isn't about securing applications. It's about fixing the ecosystem that produces applications. You guys are security people, so focus on finding the vulnerabilities in that ecosystem. What can we exploit to cause a software ecosystem to change behavior?

Let's take a minute and envision a software ecosystem that is capable of producing secure applications. Once we imagine it, perhaps we can figure out how to get there. This ecosystem could be inside a company, could be an open source project, a government acquisition, or the whole planet.

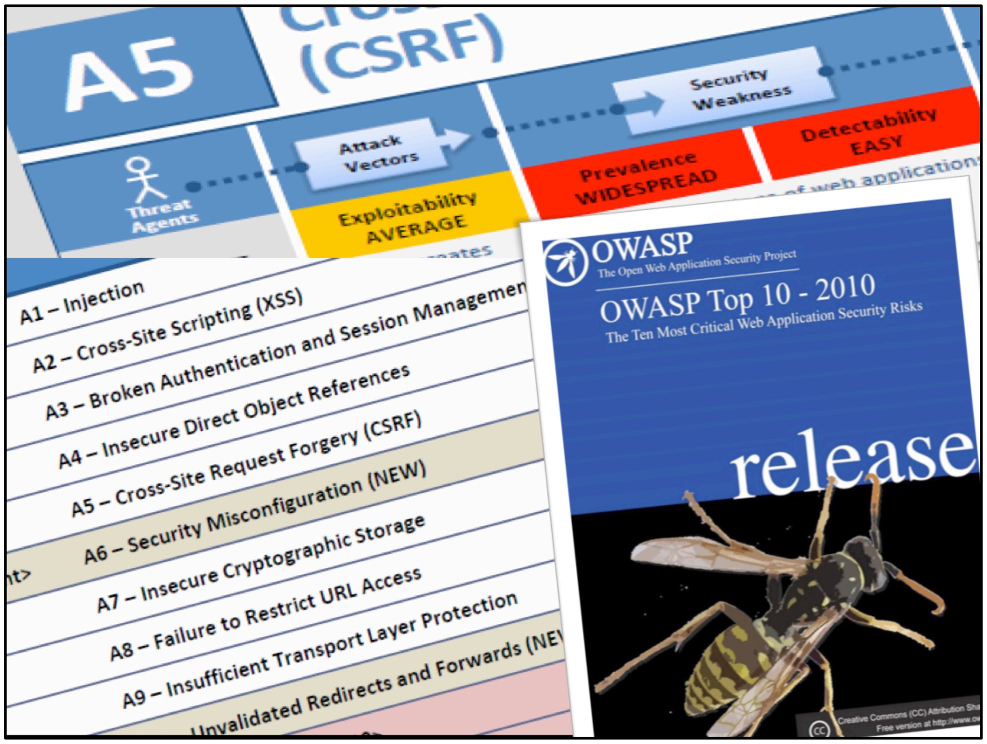
I don’t have any answers for you. That’s the beauty of a keynote. But I can share some of what we’ve been doing at OWASP to try to get ahead of the problem.



I also want to talk about how OWASP works, our ethics, our culture, and encourage membership.

OWASP is really simple. You decide your own level of participation. You want to build a tool, we will help you. You want to be a leader, we will help you with that too.

We have a board of directors and global committees: **Projects** **Membership** **Education** **Conferences** **Industry** **Chapters** **Connections**


A5

(CSRF)

Threat Agents

Attack Vectors

Security Weakness

Exploitability AVERAGE

Prevalence WIDESPREAD

Detectability EASY

A1 – Injection

A2 – Cross-Site Scripting (XSS)

A3 – Broken Authentication and Session Managemen

A4 – Insecure Direct Object References

A5 – Cross-Site Request Forgery (CSRF)

A6 – Security Misconfiguration (NEW)

A7 – Insecure Cryptographic Storage

A8 – Failure to Restrict URL Access

A9 – Insufficient Transport Layer Protection

Unvalidated Redirects and Forwards (NE

OWASP
The Open Web Application Security Project

OWASP Top 10 - 2010
The Ten Most Critical Web Application Security Risks

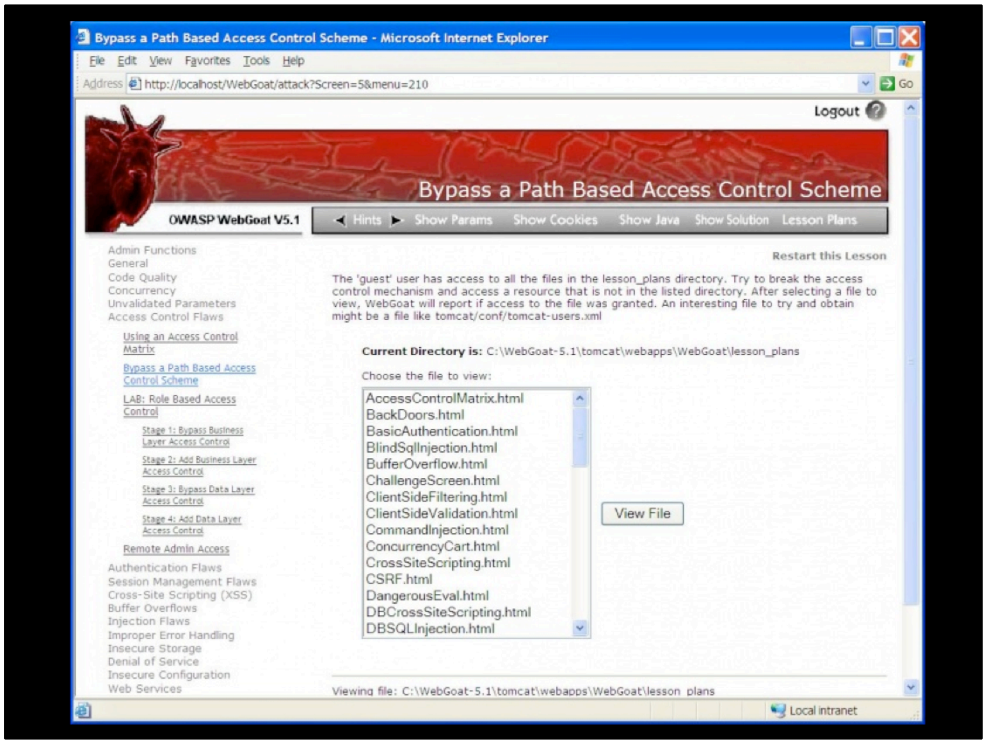
release

Creative Commons (CC) Attribution Sha

Top Ten has millions of users worldwide. Won't you help us get this important document into every developer's hands?

OWASP
The Open Web Application Security Project
OWASP Developers Guide v2.0
OWASP
Code Review Guide
rele
OWASP
The Open Web Application Security Project
OWASP Testing Guide v3.0
releas

OWASP has three free book-length PDF guides for building secure apps, app security testing, and security -code review.


Bypass a Path Based Access Control Scheme - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://localhost/WebGoat/attack?Screen=5&menu=210
Go
Logout
Bypass a Path Based Access Control Scheme
OWASP WebGoat V5.1
Hints
Show Params
Show Cookies
Show Java
Show Solution
Lesson Plans
Admin Functions
General
Code Quality
Concurrency
Unvalidated Parameters
Access Control Flaws
Using an Access Control Matrix
Bypass a Path Based Access Control Scheme
LAB: Role Based Access Control
Stage 1: Bypass Business Layer Access Control
Stage 2: Add Business Layer Access Control
Stage 3: Bypass Data Layer Access Control
Stage 4: Add Data Layer Access Control
Remote Admin Access
Authentication Flaws
Session Management Flaws
Cross-Site Scripting (XSS)
Buffer Overflows
Injection Flaws
Improper Error Handling
Insecure Storage
Denial of Service
Insecure Configuration
Web Services
Restart this Lesson
View File
Local intranet

The 'guest' user has access to all the files in the lesson_plans directory. Try to break the access control mechanism and access a resource that is not in the listed directory. After selecting a file to view, WebGoat will report if access to the file was granted. An interesting file to try and obtain might be a file like tomcat/conf/tomcat-users.xml

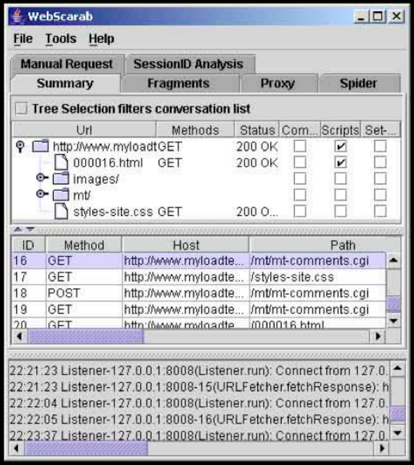
Current Directory is: C:\WebGoat-5.1\tomcat\webapps\WebGoat\lesson_plans

Choose the file to view:

AccessControlMatrix.html
BackDoors.html
BasicAuthentication.html
BlindSqlInjection.html
BufferOverflow.html
ChallengeScreen.html
ClientSideFiltering.html
ClientSideValidation.html
CommandInjection.html
ConcurrencyCart.html
CrossSiteScripting.html
CSRF.html
DangerousEval.html
DBCrossSiteScripting.html
DBSQLInjection.html

Viewing file: C:\WebGoat-5.1\tomcat\webapps\WebGoat\lesson_plans

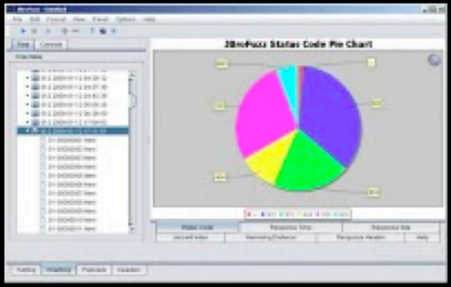
WebGoat is a full JavaEE learning environment for app security with roughly 50 different lessons on major vulnerabilities.

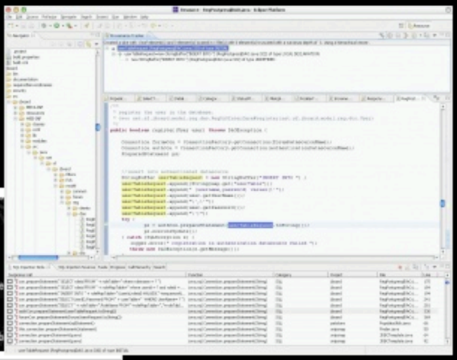


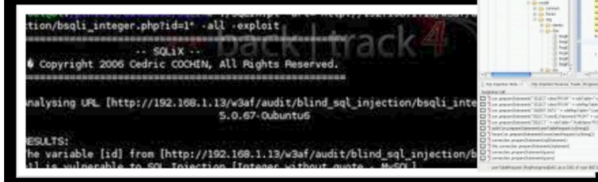
Url	Methods	Status	Com...	Scripts	Set-...
http://www.myloadtGET		200 OK	☐	☑	☐
000016.html	GET	200 OK	☐	☑	☐
images/			☐	☐	☐
mt/			☐	☐	☐
styles-site.css	GET	200 O...	☐	☐	☐

ID	Method	Host	Path
16	GET	http://www.myloadte...	/mt/mt-comments.cgi
17	GET	http://www.myloadte...	/styles-site.css
18	POST	http://www.myloadte...	/mt/mt-comments.cgi
19	GET	http://www.myloadte...	/mt/mt-comments.cgi
20	GET	http://www.myloadte...	/000016.html

```
22:21:23 Listener-127.0.0.1:8008(Listener.run): Connect from 127.0.
22:21:23 Listener-127.0.0.1:8008-15(URLFetcher.fetchResponse): h
22:22:04 Listener-127.0.0.1:8008(Listener.run): Connect from 127.0.
22:22:05 Listener-127.0.0.1:8008-16(URLFetcher.fetchResponse): h
22:23:37 Listener-127.0.0.1:8008(Listener.run): Connect from 127.0.
```



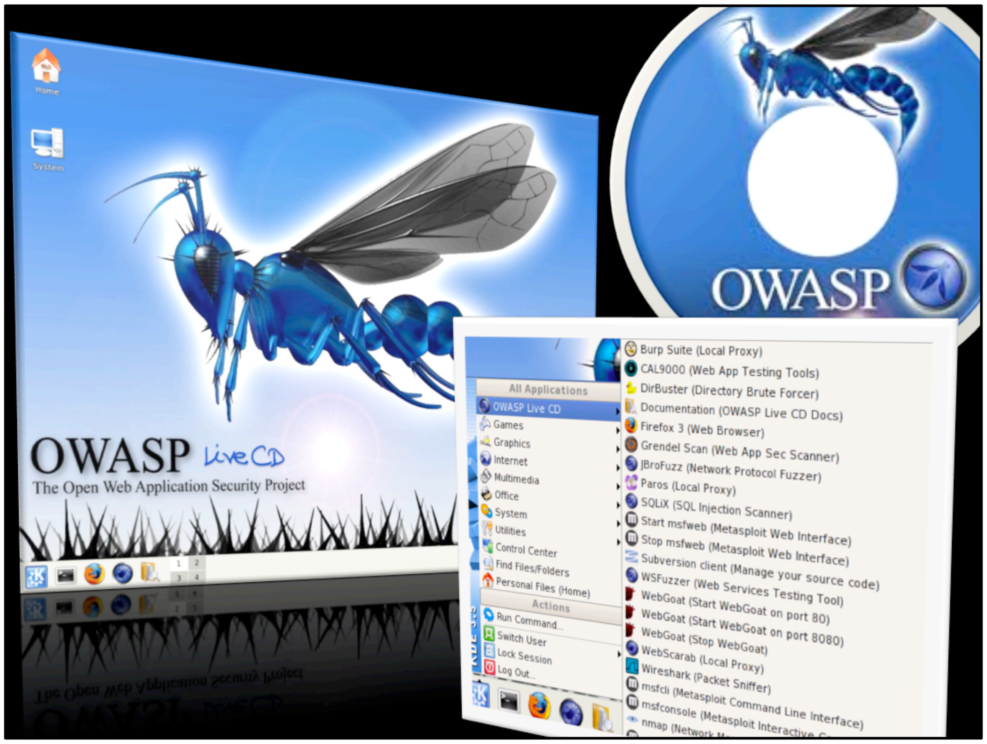




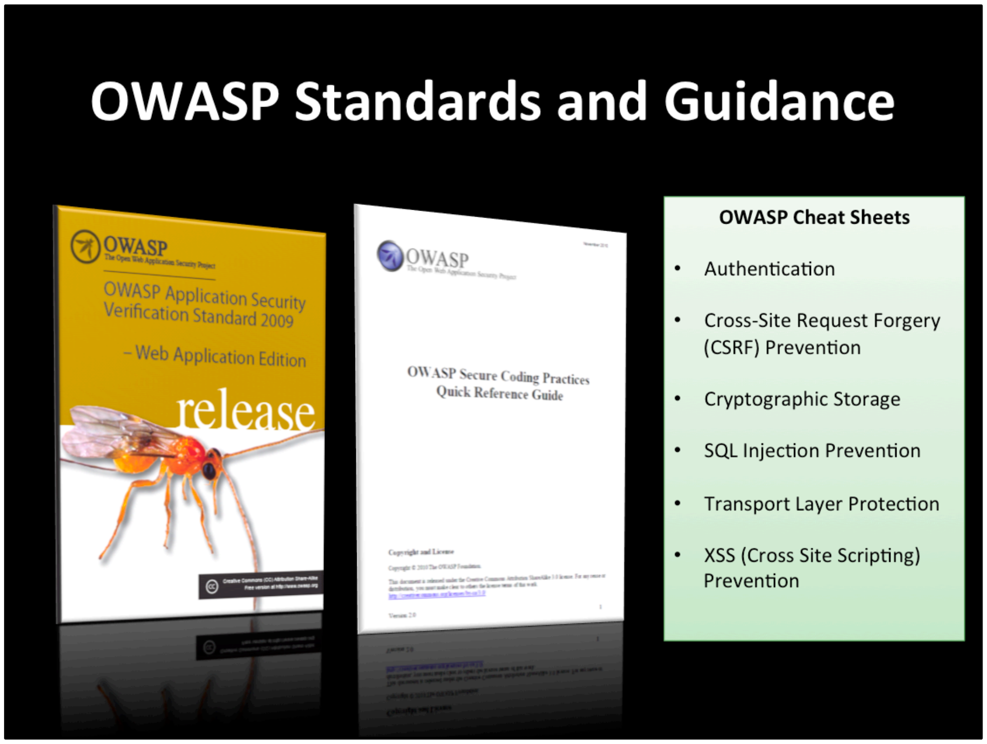
```
tion/bsqli_integer.php?id=1" -all -exploit
=====================================================
                  -- SQLiX --
© Copyright 2006 Cedric COCHIN, All Rights Reserved.
=====================================================

nalysing URL [http://192.168.1.13/w3af/audit/blind_sql_injection/bsqli_inte
                                    5.0.67-0ubuntu6

ESULTS:
he variable [id] from [http://192.168.1.13/w3af/audit/blind_sql_injection/b
l] is vulnerable to SQL Injection [Integer without quote - MySQL]
```


Home
System
OWASP Live CD
The Open Web Application Security Project
OWASP
All Applications
OWASP Live CD
Games
Graphics
Internet
Multimedia
Office
System
Utilities
Control Center
Find Files/Folders
Personal Files (Home)
Actions
Run Command...
Switch User
Lock Session
Log Out...
Burp Suite (Local Proxy)
CAL9000 (Web App Testing Tools)
DirBuster (Directory Brute Forcer)
Documentation (OWASP Live CD Docs)
Firefox 3 (Web Browser)
Grendel Scan (Web App Sec Scanner)
JBroFuzz (Network Protocol Fuzzer)
Paros (Local Proxy)
SQLiX (SQL Injection Scanner)
Start msfweb (Metasploit Web Interface)
Stop msfweb (Metasploit Web Interface)
Subversion client (Manage your source code)
WSFuzzer (Web Services Testing Tool)
WebGoat (Start WebGoat on port 80)
WebGoat (Start WebGoat on port 8080)
WebGoat (Stop WebGoat)
WebScarab (Local Proxy)
Wireshark (Packet Sniffer)
msfcli (Metasploit Command Line Interface)
msfconsole (Metasploit Interactive C
nmap (Network M

The OWASP Live CD is a bootable environment full of tools and other materials for appsec testing, learning, and more.


OWASP
The Open Web Application Security Project
OWASP Application Security Verification Standard 2009
– Web Application Edition
release
OWASP Secure Coding Practices Quick Reference Guide

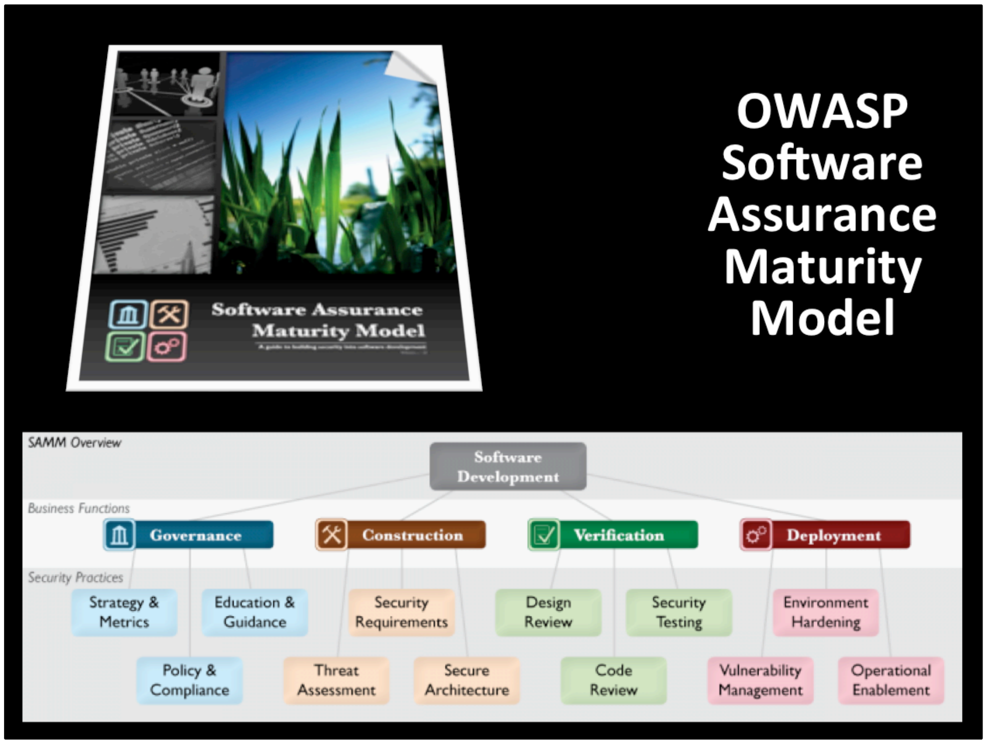
OWASP Standards and Guidance

OWASP Cheat Sheets

- Authentication
- Cross-Site Request Forgery (CSRF) Prevention
- Cryptographic Storage
- SQL Injection Prevention
- Transport Layer Protection
- XSS (Cross Site Scripting) Prevention

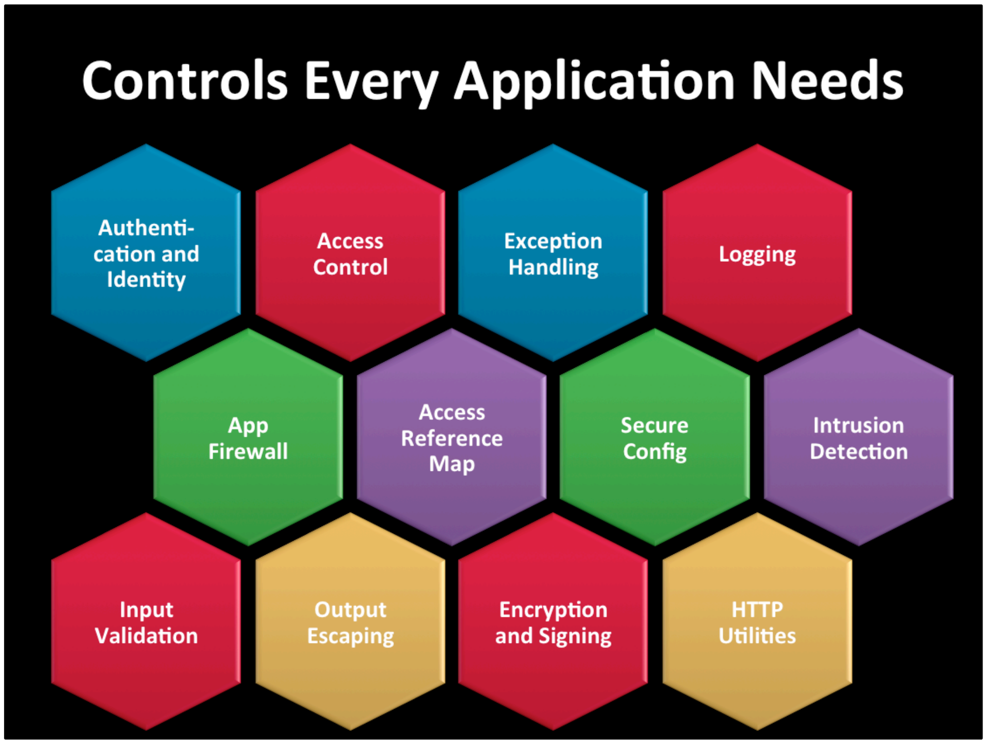
OWASP has the Application Security Verification Standard (ASVS) and more to help you build secure code.

Don’t forget the *prevention* cheat sheets!



OWASP Software Assurance Maturity Model

The OWASP OpenSAMM is a fantastic project to provide organizations a roadmap for improving their ability to build secure code.



Controls Every Application Needs

Authenti-cation and Identity

Access Control

Exception Handling

Logging

App Firewall

Access Reference Map

Secure Config

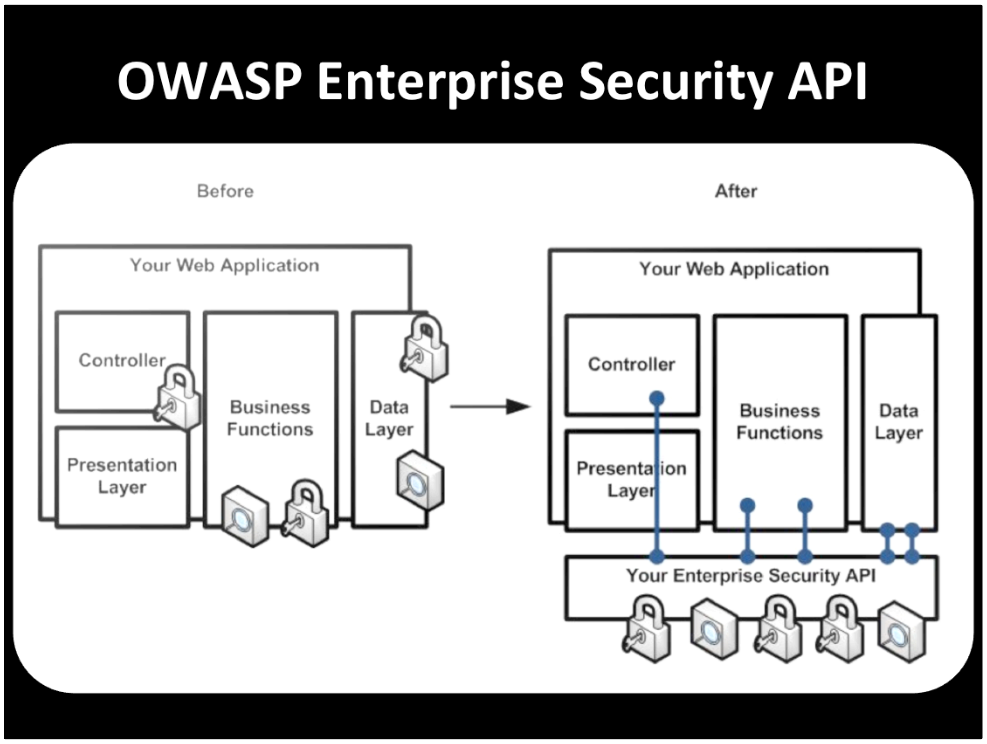
Intrusion Detection

Input Validation

Output Escaping

Encryption and Signing

HTTP Utilities



OWASP Enterprise Security API

The first thing that you can focus on today is to make the “secure” path as absolutely easy as possible.

One of the most difficult things for developers is to get strong security controls built. The vast majority of problems we find are because an appropriate security control is either missing or broken.

This is a really critical point. If you provide your developers with strong controls, all they have to do is use them in the right places.

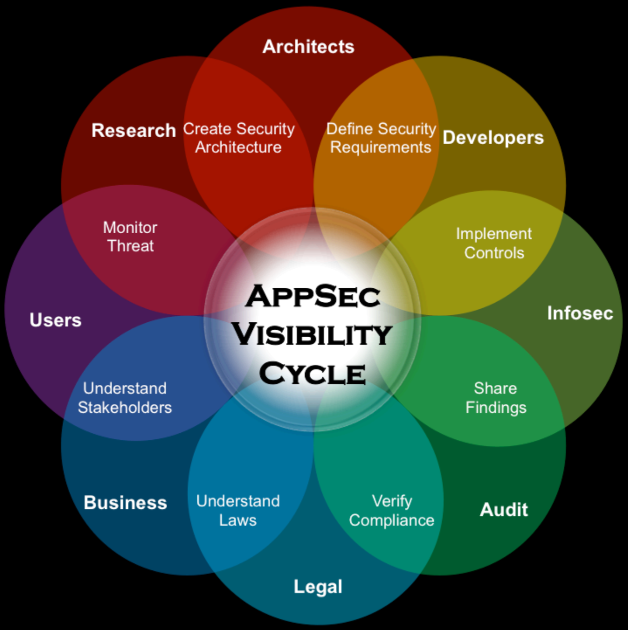
Many organizations have a few standard libraries they use for security, such as encryption and logging. But they’re usually not integrated or very well thought through. They’re almost always missing good support for validation, encoding, and access control. Which is a disaster because most of the biggest risks, like SQL injection, XSS, and forced browsing are left up to developers to write custom code to protect against.

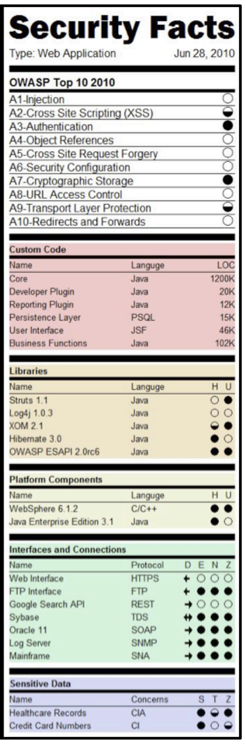
So I started a project at OWASP to define a standard API that contains all the methods a developer would need in order to secure a typical web application. That project is called the Enterprise Security API – and it’s catching on like crazy. We ended up not only defining an API, but we also built a reference implementation. And now there are implementations in Java, .NET, PHP, VB, Python, Cold Fusion, JavaScript, and force.com!

By standardizing the security controls, you eliminate a big percentage of where the errors get made.

This is just one of the things you can do to make application security as simple as possible.

Visibility





Security Facts

Type: Web Application Jun 28, 2010

OWASP Top 10 2010

A1-Injection	○
A2-Cross Site Scripting (XSS)	◒
A3-Authentication	●
A4-Object References	○
A5-Cross Site Request Forgery	○
A6-Security Configuration	○
A7-Cryptographic Storage	●
A8-URL Access Control	○
A9-Transport Layer Protection	◒
A10-Redirects and Forwards	○

Custom Code

Name	Languge	LOC
Core	Java	1200K
Developer Plugin	Java	20K
Reporting Plugin	Java	12K
Persistence Layer	PSQL	15K
User Interface	JSF	46K
Business Functions	Java	102K

Libraries

Name	Languge	H	U
Struts 1.1	Java	○	●
Log4j 1.0.3	Java	○	○
XOM 2.1	Java	◒	●
Hibernate 3.0	Java	●	○
OWASP ESAPI 2.0rc6	Java	●	●

Platform Components

Name	Languge	H	U
WebSphere 6.1.2	C/C++	●	●
Java Enterprise Edition 3.1	Java	●	○

Interfaces and Connections

Name	Protocol	D	E	N	Z
Web Interface	HTTPS	←	○	○	○
FTP Interface	FTP	←	●	●	●
Google Search API	REST	→	○	○	○
Sybase	TDS	↔	●	●	●
Oracle 11	SOAP	→	●	●	●
Log Server	SNMP	→	●	●	●
Mainframe	SNA	→	●	●	●

Sensitive Data

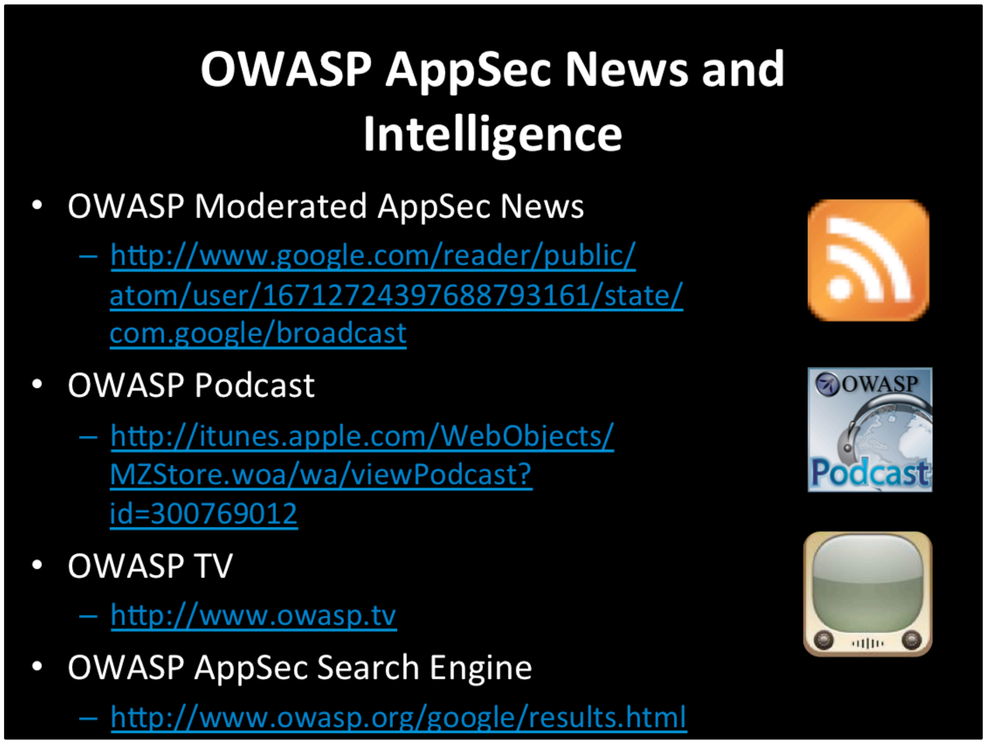
Name	Concerns	S	T	Z
Healthcare Records	CIA	●	◒	●
Credit Card Numbers	CI	●	○	◒

The first thing we have to do is create visibility. Once we've captured what we know, people can start to poke holes in it. Which will encourage others to think up solutions.

If you are working on a software project, what can you do to make security visible? Do you have a clear threat model and security architecture? Do you have coding patterns for using security controls?

Unless we make security concrete and visible, we won't be able to make improvements and everything we do will be forgotten when the next new thing comes along.

Security ecosystems can't evolve without sunshine.

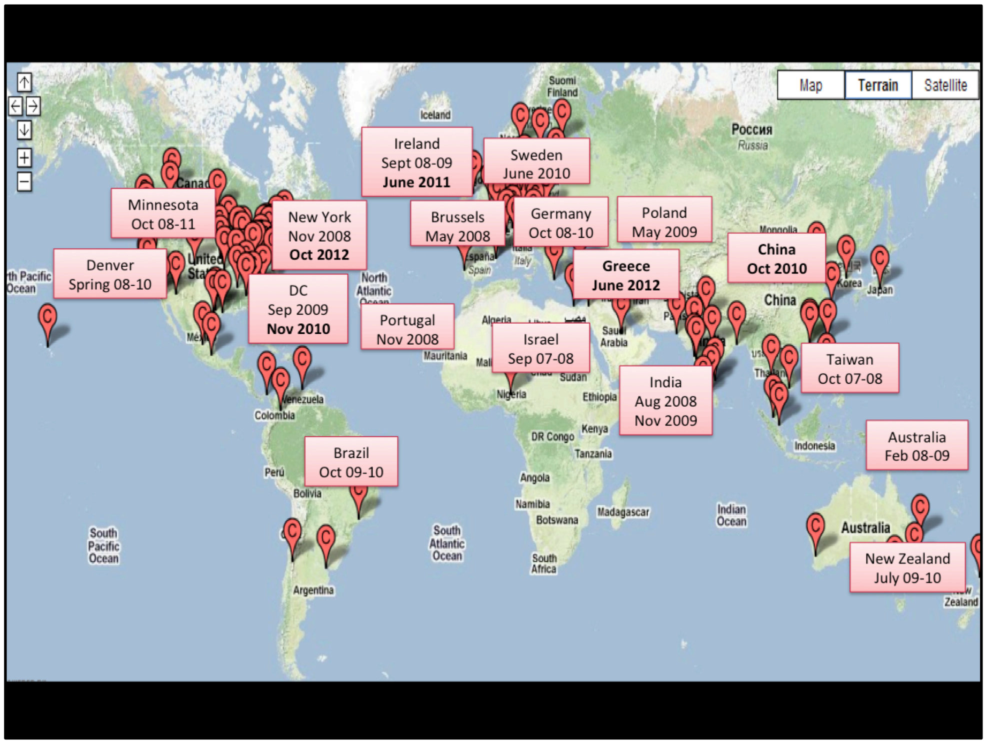


OWASP AppSec News and Intelligence

- OWASP Moderated AppSec News
  - http://www.google.com/reader/public/atom/user/16712724397688793161/state/com.google/broadcast
- OWASP Podcast
  - http://itunes.apple.com/WebObjects/MZStore.woa/wa/viewPodcast?id=300769012
- OWASP TV
  - http://www.owasp.tv
- OWASP AppSec Search Engine
  - http://www.owasp.org/google/results.html

OWASP
Podcast

You can follow what's going on in the appsec world through our various news channels!


Map
Terrain
Satellite
Minnesota
Oct 08-11
Denver
Spring 08-10
New York
Nov 2008
Oct 2012
DC
Sep 2009
Nov 2010
Ireland
Sept 08-09
June 2011
Sweden
June 2010
Brussels
May 2008
Germany
Oct 08-10
Poland
May 2009
Portugal
Nov 2008
Greece
June 2012
Israel
Sep 07-08
China
Oct 2010
India
Aug 2008
Nov 2009
Taiwan
Oct 07-08
Brazil
Oct 09-10
Australia
Feb 08-09
New Zealand
July 09-10

http://www.owasp.org/index.php/Summit_2011

I invite you to get involved. You can come to Lisbon and really make a difference in the state of application security.

jeff.williams @ aspectsecurity.com | 410-707-1487 | @planetlevel

OWASP is trying to change the world's software ecosystem so that it produces more secure software. If you share that goal, we can help you turn your idea into a project, a community, a movement, and a revolution. Join us.