“79% of breached records are web application attacks" as stated from the 2009 Data Breach Investigations Report conducted by Verizon Business Risk Team.

"30% of the 57 attacks were carried out by SQL injection" from the 2008 Web Hacking Incidents Database Annual Report conducted by Breach Security.

Web Application Security & Vulnerability Management Tools

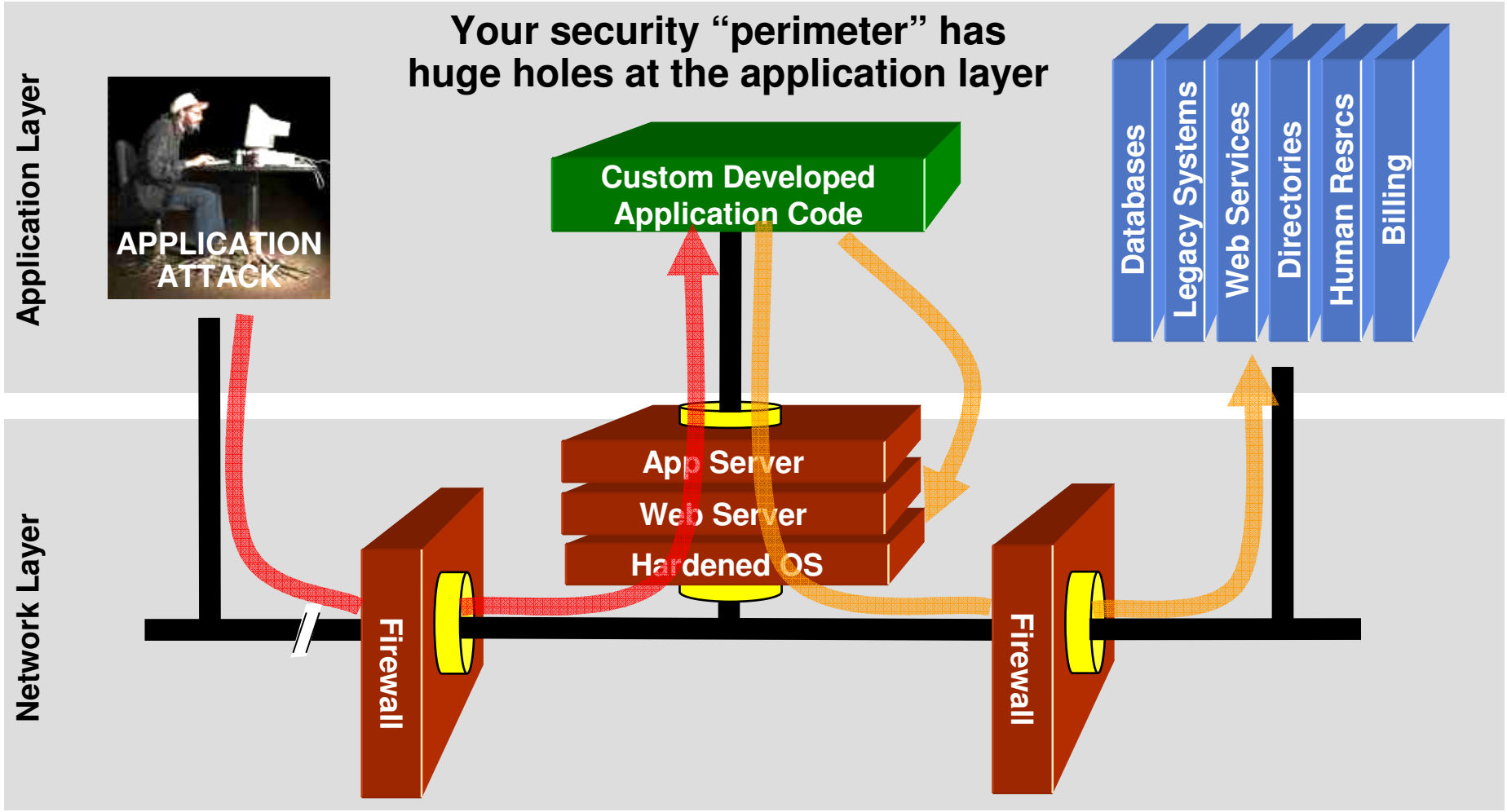
Boston University Security Camp
08/20/2009

Roy Wattanasin
websecr at gmail dot com

What is Web Application Security?

- Securing the “custom code” that drives any web application
  - Securing libraries
  - Securing backend systems
  - Securing web and application servers

Your Code is Part of Your Security Perimeter



You can't use network layer protection (firewall, SSL, IDS, hardening) to stop or detect application layer attacks - OWASP

What is OWASP?

- Open Web Application Security Project
  - http://www.owasp.org
  - Open group focused on understanding and improving the security of web applications and web services!
  - Hundreds of volunteer experts from around the world

OWASP
The Open Web Application Security Project
http://www.owasp.org

  - Top Ten Project
    - Raise awareness with a simple message
    - Lead by Aspect Security

What is Web Application Security?

- Securing the “custom code” that drives any web application
  - Securing libraries
  - Securing backend systems
  - Securing web and application servers

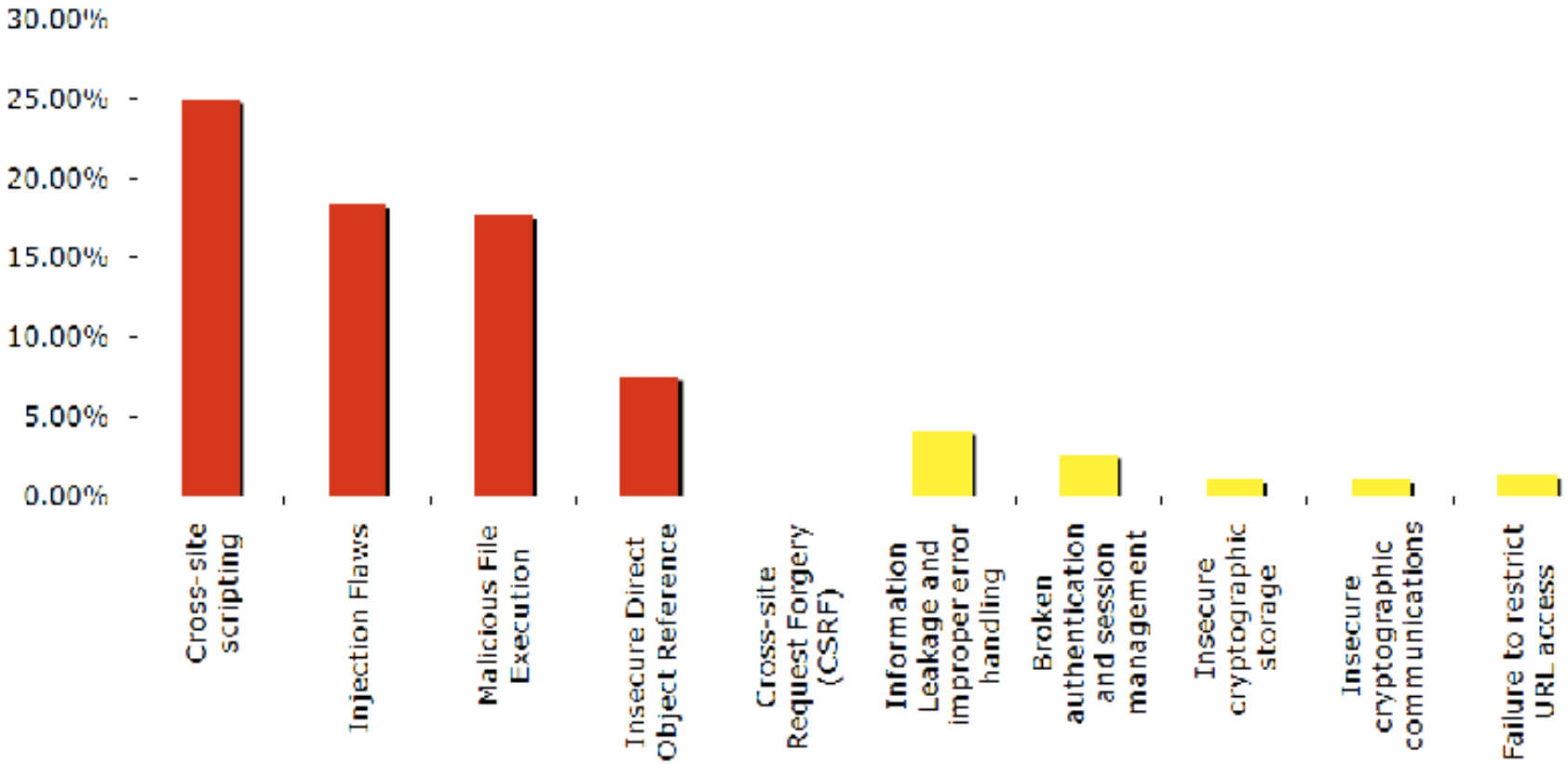
Why Care?

- **How likely is a successful web application attack?**
  - Stunningly prevalent
  - Easy to exploit without special tools or knowledge
  - Little chance of being detected
  - Hundreds of thousands of developers, tiny fraction with security

- **Consequences?**
  - Corruption or disclosure of database contents
  - Root access to web and application servers
  - Loss of authentication and access control for users
  - Defacement
  - Secondary attacks from your site

- **Web Application Security is <u>just</u> as important as Network Security**

OWASP Top 10



OWASP Top 10, 2007

Most Common Application Security Flaws

- Cross Site Scripting (XSS or CSS)
  - Inject code in to the target web server / deliver users to a malicious link
    - Javascript, VBScript, ActiveX, HTML, Flash in applications
  - Example : Script may hijack user's account information, steal cookies or session information, change user privileges

- Input Validation Failures
  - Validate input before accepting request & resuming process
  - Validate both trusted and untrusted resources prior to application processing
    - Data types (string, integer), format, length, range, null value handling, character set, locale, patterns, context, legal values, validity.
  - Example : Web application failed to encode square brackets [ ] and this causes a malicious code injection to execute. (ex. Client side data validation – flaw may be exploited)

Most Common Application Security Flaws

- Output Sanitation
  - Redisplaying and echoing the data values entered by users allows for malicious data inputs
  - Comments & identifiers in the output response must be removed.
  - It provides a mean for malicious users to match the given input / output
  - Example : Web not properly sanitized allows for malicious HTML tags to create pop up banners or may allow to change the content originally displayed.

- Buffer Overflow
  - Application / process tries to store more data in a data storage or memory buffer than its capacity or its fixed length can handle leads to corruption or buffer overwrite
  - Typically carried out using application weaknesses
  - Example : Passed malicious input by manipulating or tampering the input parameters to force an application buffer overflow which could lead to a DOS attack.

Most Common Application Security Flaws

- Data Injection Flaw
  - Inject malicious code with user data to exploit a weakness in the user data environment.
  - Comments & identifiers in the output response must be removed.
  - Found in browsers with pop up windows or SQL statements.
  - Example : Hijack a named web browser window after a user opens both the malicious website and a trusted website.

- Insecure Use of Cryptography
  - Don't invent a new algorithm
  - Be extremely careful storing keys, certs, and passwords
  - Rethink whether you need to store the information
  - Don't store user passwords – use a hash like SHA-256
  - The "master secret" can be split into two locations and assembled

Most Common Application Security Flaws

- Improper Error Handling
  - Displaying internal error messages and implementation details about the application
    - Out of memory, null pointer exceptions, system call failures, database access failure, network timeout
  - Requires proper error handling mechanisms to user input but no internal details about the application environment or its components.
  - Example : Allows to crash the application.

- Weak Session Identifiers
  - Issuing or using session identifiers before authentication or over unencrypted channels allows others to steal session information.
  - Requires encrypted session identifiers after initiating a secure communication channel using SSL.
  - Example : Clear session information allows to spoof the user identity information.

Most Common Application Security Flaws

- Weak Security Tokens
  - Use of password security tokens that allow others to guess passwords by a dictionary or token decrypting tools to impersonate the user.
    - Out of memory, null pointer exceptions, system call failures, database access failure, network timeout
  - Adopt strong authentication or multifactor authentication mechanisms using certificates, biometrics or smart cards.
  - Example : Allows a user to guess passwords.

- Weak Password Exploits
  - Passwords are the weakest mechanisms for user authentication because they can be compromised or guessed using keystroke tools.
  - Password protect files with encryption.

Most Common Application Security Flaws

- Insecure Data Transit or Storage
  - Confidentiality of data in transit or storage is very important because security is compromised when data is in plain text.
  - Adopt cryptographic mechanisms and data encryption techniques to ensure the integrity and confidentiality of data in transit or storage.

- Session Theft
  - Session hijacking to create or reuse an existing session.
  - Always invalidate a session after a logout.

- Insecure Configuration Data
  - Misconfigured SSL certificates and web server plug-ins.
  - Always test and verify the environment for configuration related weaknesses.

Most Common Application Security Flaws

- Broken Authentication
  - Improper configuration of authentication mechanisms and flawed credential management through a password change, forgotten password and certificate issues.
  - Verify its authentication mechanisms and enforce authentication by verifying user's credentials before granted access to the application.
  - Example : Manipulate credentials to impersonate them.

- Broken Access Control
  - Determines an authenticated user's rights and privileges for access to an application or data and any loss leads to loss of confidential information & unauthorized disclosure.
  - Verify the application specific access control lists for all know risks

- Audit and Logging Issues
  - Provide evidence about all key application events.
  - Logs files must have restricted access and monitoring of auditing & logging processes of applications of high availability is crucial.

Most Common Application Security Flaws

- Denial of Service (DOS) and Distributed DOS (DDOS)
  - Network attacks
    - CPU utilization, memory, network bandwidth
  - Web based applications are most vulnerable
  - Routers to drop connections from attacks from un-trusted hosts.

- Multiple Sign On Issues
  - Requires user to log on multiple times because the integrated application does not share a common sign on mechanism within the environment.
  - Adopt a single sign on (SSO) mechanism to resolve these problems by eliminating the need to remember usernames and passwords other than initial application login.

- Man in the Middle (MITM)
  - Ability to read or modify business transactions between two parties without either party knowing about it.

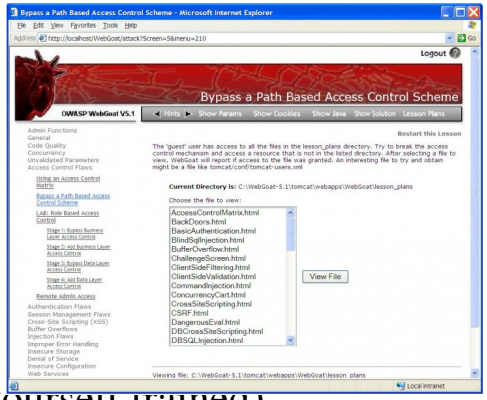
Most Common Application Security Flaws

- Deployment Problems
  - Inconsistencies within and conflicts between application configuration data and the deployment infrastructure
    - Hosts, network environment
  - Review and test all infrastructure security policies to make sure that both infrastructure and application security policies reflect each other.

- Coding Problems
  - Cause flaws and erroneous conditions in programming and application program flow.
  - Adopt a coding review methodology.

Source: Core Security Patterns, 2006 & OWASP 2007

What Can You Do?



- Training
  - Read the Top Ten paper!
  - Read the news!
  - Try OWASP WebGoat to learn how flaws work (Get yourself trained)
  - Get developers trained in web application security (Get others trained)
    - Help them understand web application security for the developer's mindset
    - When should security be talked about?
    - Integrate security in to the SDLC
    - Uses security tools and examples to help you!

- Policy
  - Write down the security rules for your application
    - Create policies for web application security reviews
    - Integrate them in to the SDLC

- Reviews
  - Get expert code reviews and have penetration tests periodically

What Can Others Do?

- Customers
  - Demand web applications that don't have these ten simple problems
- Developers
  - Take responsibility for securing your code
- Software Development Organizations
  - Guarantee that your web applications don't have the top ten flaws
- Educators
  - Stop teaching insecure coding
- Project Managers
  - Split your security budget between network and application
  - Make security part of developer performance reviews

What Web Application Tools Can Be Used?

- A "hybrid" variety should be used among opensource, commercial tools and your skills

Commercial Tools:	OpenSource Tools:
WebInspect	Nikto
Acunetix	Paros Proxy
Hailstorm	WebScarab
N-stalker	Whisker/libwhisker
NTOSpider	Burpsuite
AppScan	Wikto
N-Stealth	WebGoat
QualysGuard	

- Videos of web application attacks

What Web Vulnerability Management Tools Can Be Used?

- A “hybrid” variety should be used among opensource, commercial tools and your skills

Commercial Tools:

- Nessus
- Nexpose
- GFI LANGuard
- Retina
- QualysGuard
- SAINT
- Microsoft Baseline Security Analyzer (MBSA)
- ISS
- Core Impact

OpenSource Tools:

- Google search
- Etc.

Conclusion

- Questions/Comments
- Thank you!