

Integrating Web Application Penetration Testing into Your Vulnerability Management Program

Rich Mogull
Securosis, L.L.C.

Top Threats



Web Applications



Clientside

Why Web Applications Are Such a Problem

- Rapid development with limited QA
- Eternal beta cycles
- Un(security)trained developers
- New vulnerability classes
- Insecure browsers
- Inherent insecurity of web model

Major Webapp Attacks

Breaking Trust Relationships



Browser

Cross Site Scripting

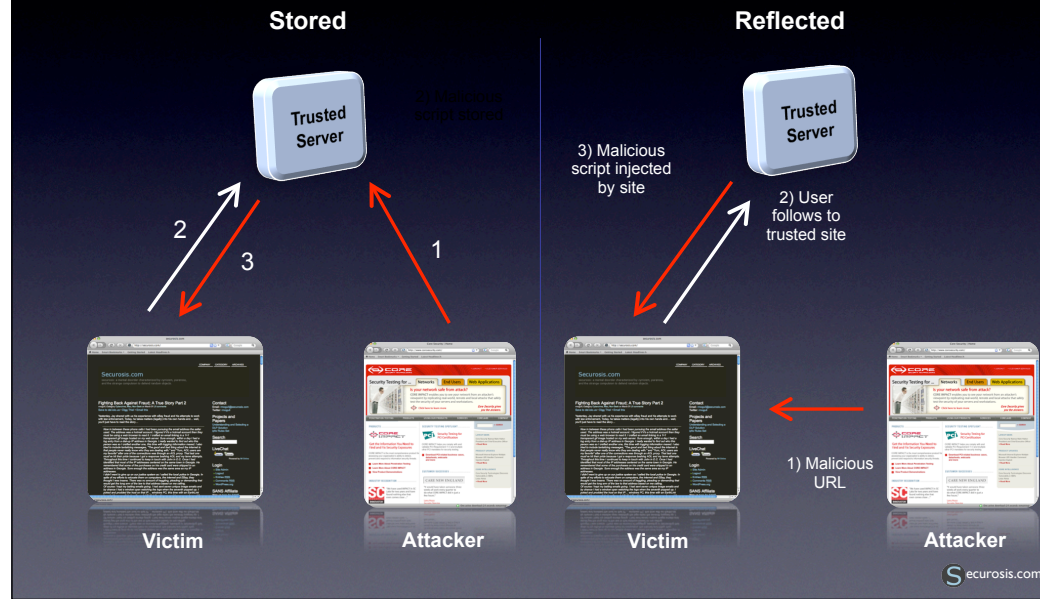
Cross Site Request Forgery

SQL Injection

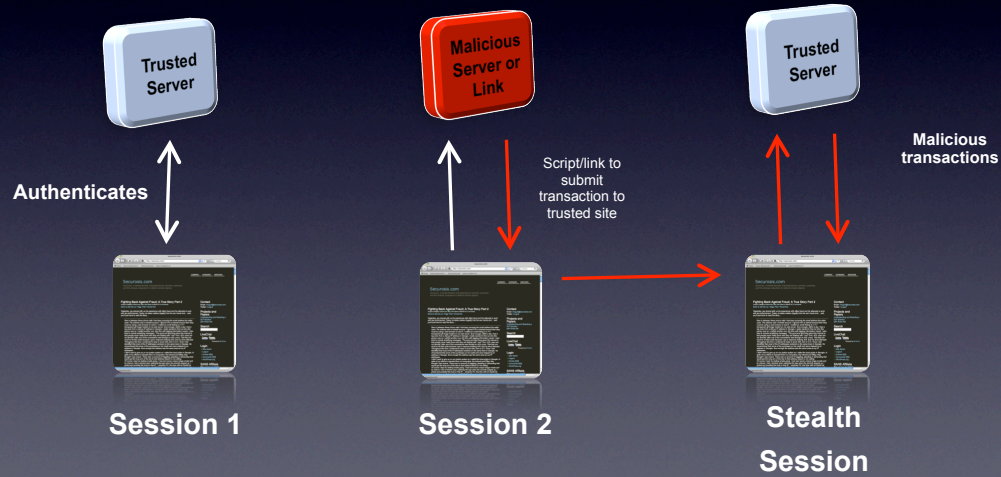


Server

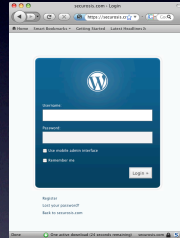
Cross Site Scripting



Cross Site Request Forgery



SQL Injection



admin'--

Attack Input

```
Statement: "SELECT * FROM users WHERE  
name = '" + uName + "' AND password =  
'" + upass + "';"
```

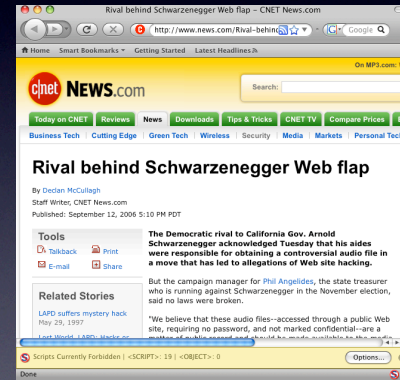
SQL Statement

```
SELECT * FROM users WHERE name =  
'admin'-- "' AND password = '" + upass  
+ "';"
```

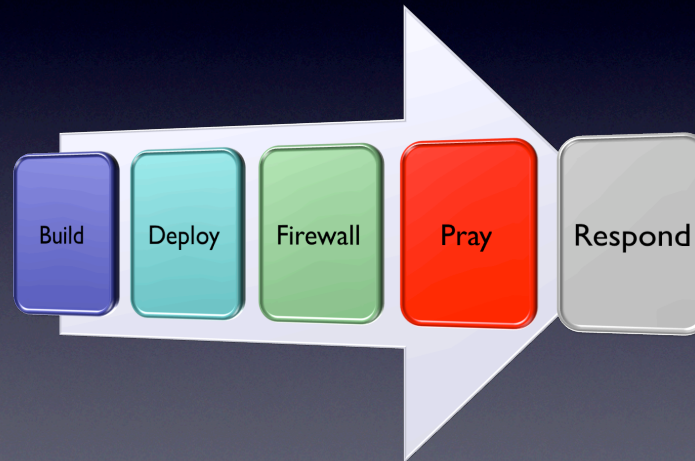
Executed Statement

Accidental/Directory Traversal

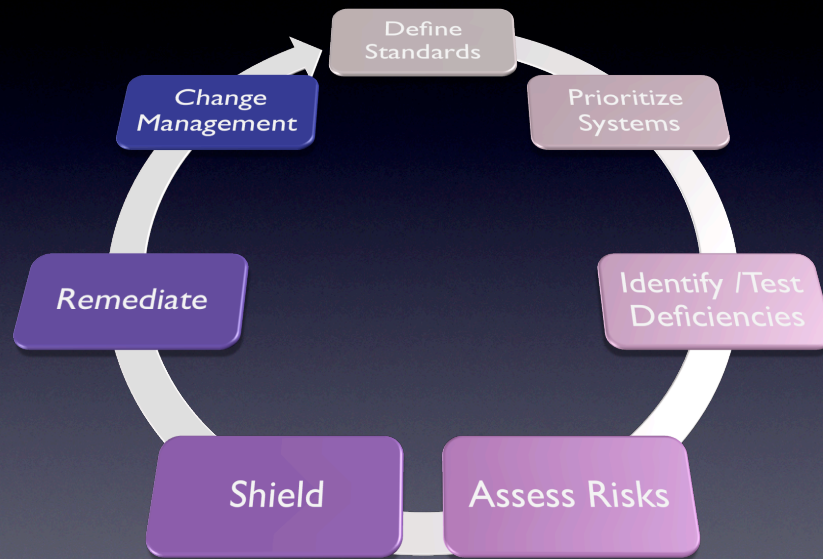
+ Or - “/” =



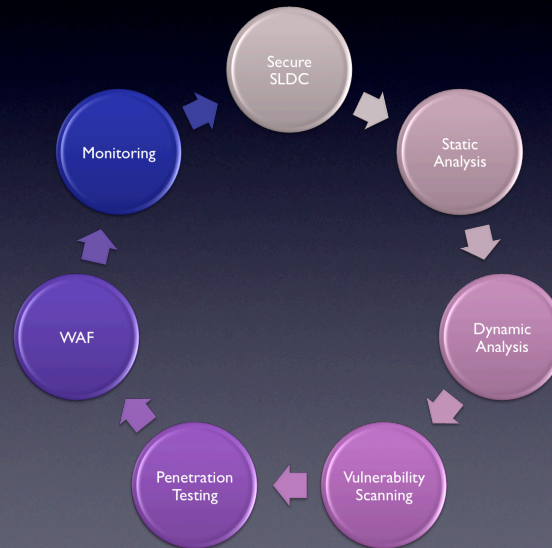
How we used to manage web applications



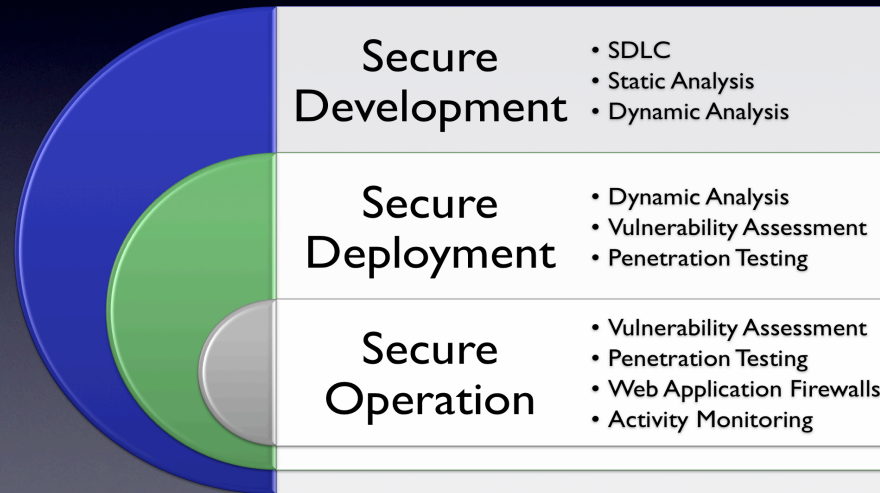
Vulnerability Management



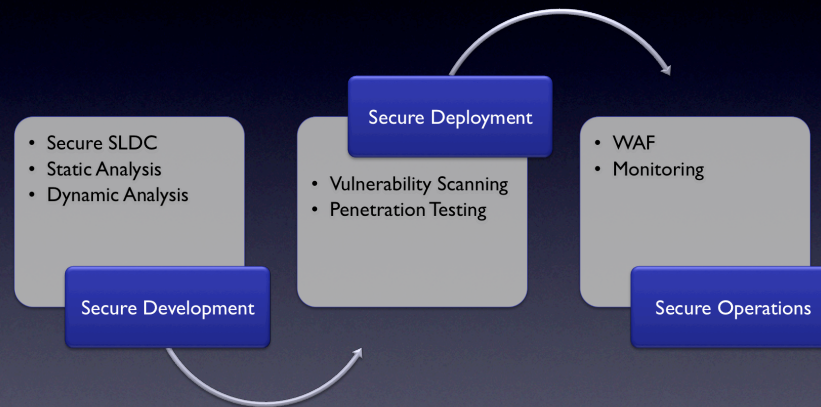
Web Application Security Program Overview



Application Security Lifecycle



Development Phases



Integration

```
graph TD; CM[Change Management] --> DS[Define Standards]; DS --> PS[Prioritize Systems]; PS --> ITD[Identify /Test Deficiencies]; ITD --> AR[Assess Risks]; AR --> S[Shield]; S --> R[Remediate]; R --> CM; CM --> SDLC[Secure SDLC]; DS --> SDLC; PS --> SDLC; ITD --> SDLC; AR --> SDLC; S --> SDLC; R --> SDLC; SDLC --> CM; SDLC --> DS; SDLC --> PS; SDLC --> ITD; SDLC --> AR; SDLC --> S; SDLC --> R; SDLC --> PT[Penetration Testing]; PT --> CM; PT --> DS; PT --> PS; PT --> ITD; PT --> AR; PT --> S; PT --> R; PT --> SDLC;
```

The diagram illustrates a circular flow of security and development processes, centered around the **Secure SDLC** (Software Development Life Cycle). The processes are arranged in a circle, with arrows indicating a clockwise flow between them. The central box is labeled **Secure SDLC**.

The processes shown are:

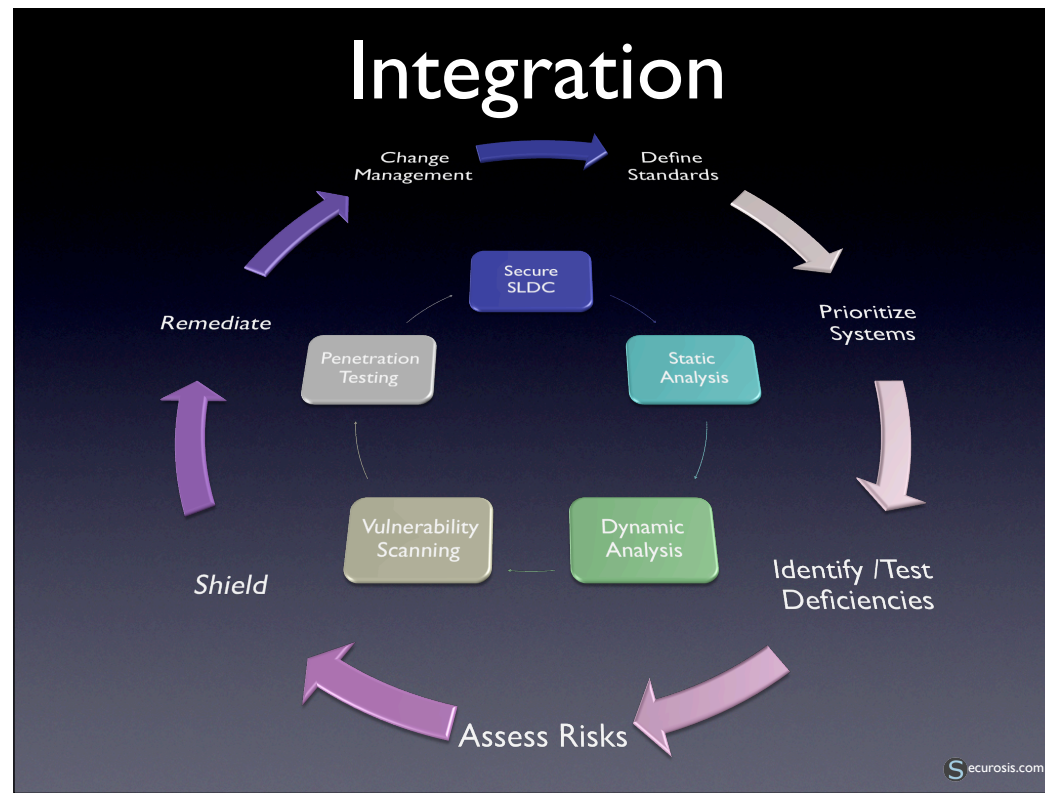
- Change Management** (top left)
- Define Standards** (top right)
- Prioritize Systems** (right)
- Identify /Test Deficiencies** (bottom right)
- Assess Risks** (bottom)
- Shield** (bottom left)
- Remediate** (left)
- Penetration Testing** (middle left)
- Static Analysis** (middle right)
- Dynamic Analysis** (bottom center)
- Vulnerability Scanning** (middle left)

The flow is as follows:

- Change Management** leads to **Define Standards**.
- Define Standards** leads to **Prioritize Systems**.
- Prioritize Systems** leads to **Identify /Test Deficiencies**.
- Identify /Test Deficiencies** leads to **Assess Risks**.
- Assess Risks** leads to **Shield**.
- Shield** leads to **Remediate**.
- Remediate** leads to **Change Management**.
- Change Management** also leads to **Secure SDLC**.
- Define Standards** also leads to **Secure SDLC**.
- Prioritize Systems** also leads to **Secure SDLC**.
- Identify /Test Deficiencies** also leads to **Secure SDLC**.
- Assess Risks** also leads to **Secure SDLC**.
- Shield** also leads to **Secure SDLC**.
- Remediate** also leads to **Secure SDLC**.
- Penetration Testing** also leads to **Secure SDLC**.
- Static Analysis** also leads to **Secure SDLC**.
- Dynamic Analysis** also leads to **Secure SDLC**.
- Vulnerability Scanning** also leads to **Secure SDLC**.

The **Secure SDLC** box is the central hub, with arrows pointing to all other processes, indicating its foundational role in the entire cycle.

Securosis.com



Platform vulns

Limitations of static analysis/scanning

- Can't catch everything
- No validation
- No exploitability/Impact
- Miss logic flaws
- Fire and forget
- The bad guys don't use them

Best Practices for Web App Pen Testing

- Begun testing in the development process.
- Use a combination of tools and manual process.
- Include traditional pen testing of the underlying platform.
- Perform periodic testing post-deployment, especially as new exploits appear.

Adapting your program for the long term

- Understand the different requirements of web application vulnerability management.
- Establish web application configuration standards and begin enforcement during development.
- Include code and vulnerability scanning, but you cannot skip penetration testing.

Integrating Web Application Penetration Testing into Your Vulnerability Management Program

Rich Mogull
Securosis, L.L.C.
<http://securosis.com>
rmogull@securosis.com