Strategic Information Security.

Attacking and Defending Web Services

Presented By:

David W. Green, CISSP

dgreen@securityps.com

SECURITY PS

STRATEGIC INFORMATION SECURITY

About Security PS

Application Security Assessments

Network Security Assessments

Security Compliance Consulting

Security Training and Awareness

Agenda

- Background for Discussion
- Attacks and Defenses
  - Information Gathering
  - Denial of Service
  - Message Confidentiality
  - Authentication
  - Access Control
  - Data Validation and Encoding
- Conclusions

Background for Discussion

What is a Web Service?

Current Risk Considerations

What is a Web Service?

Definition for Today:

Application to application communication over XML

Common Uses:

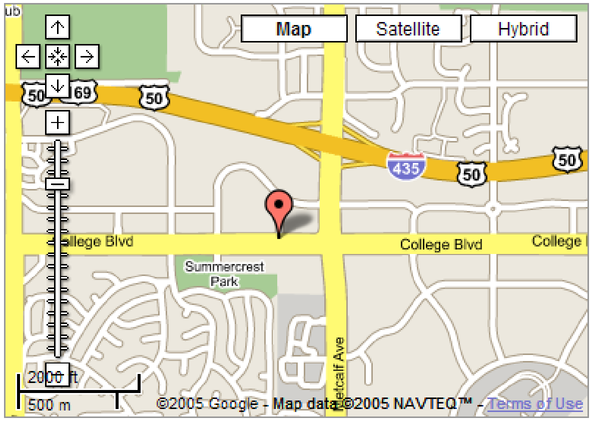
B2B communication

Middleware

Interface to legacy systems

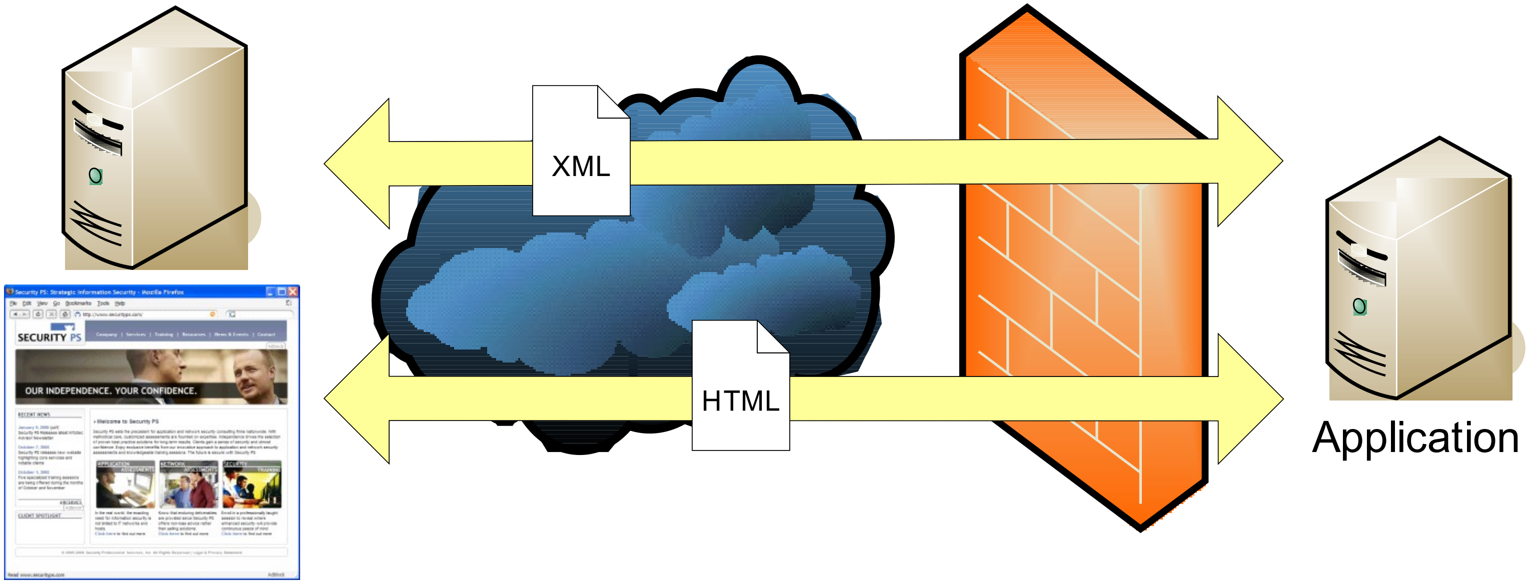
AJAX (maps.google.com)

APIs to add functionality



What is a Web Service?

Web Service Client



Application

Browser

Current Risk Considerations

- Increased popularity
  - The use of web services has increased dramatically in recent years
- It’s still a web application
  - Web app security principles still apply
- Emerging technologies
  - Supporting standards are still being developed
  - “Closed door” solutions are currently common

A Few of the Standards…

XML XACML ebXML

WS–Security SOAP REST XSL

XAML SAML XKMS

WSDL CORBA XrML XSD UBR

X.509

UDDI XKMS XLANG

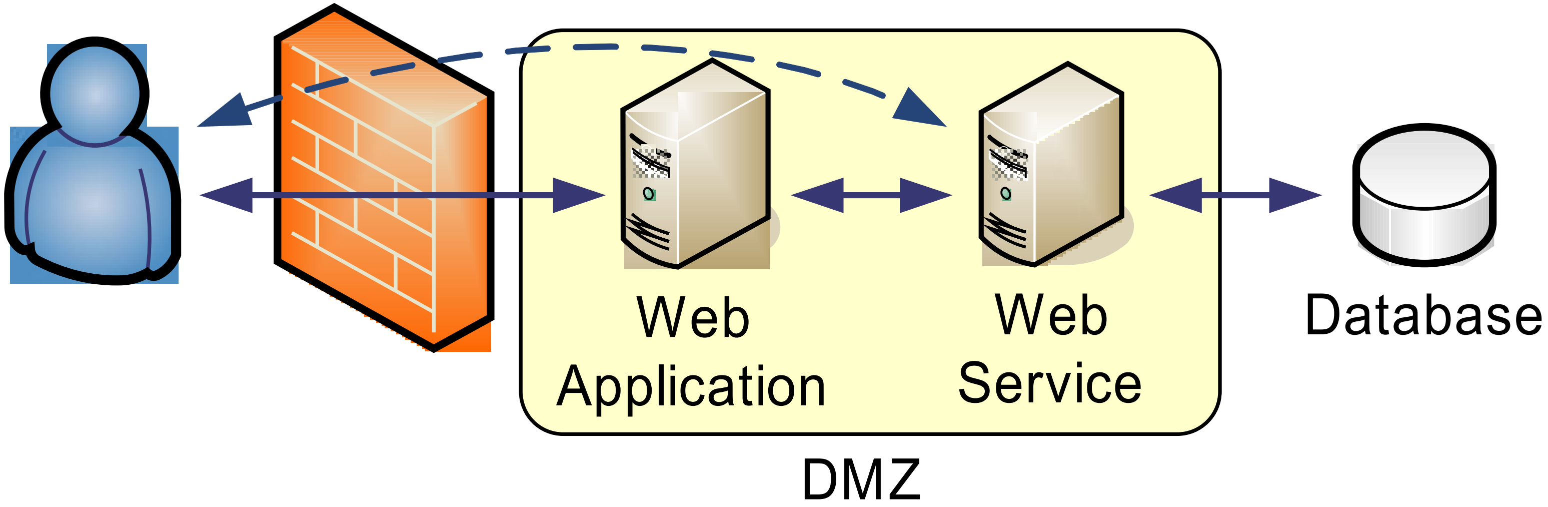
Attacks and Defenses

Information Gathering
Denial of Service
Message Confidentiality
Authentication
Access Control
Data Validation and Encoding

Information Gathering (Exposure)

Visibility

Network firewalls and common ports



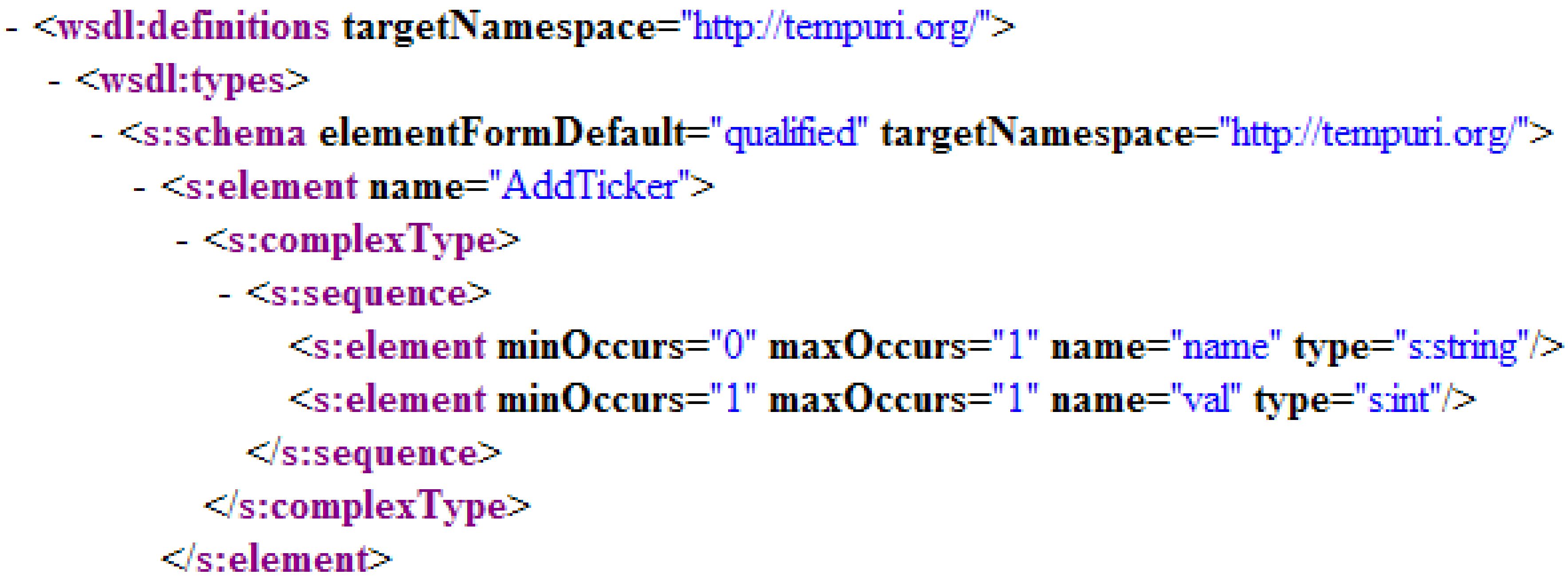
Information Gathering (Exposure)

- Discovery
  - Google
  - UDDI Business Registry
  - Third Party Registries (xmethods.com)
  - WSDL

Information Gathering (Exposure)

WSDL: A hacker’s reference manual

http://www.example.com/service.asmx?wsdl



```
- <wsdl:definitions targetNamespace="http://tempuri.org/">
  - <wsdl:types>
    - <s:schema elementFormDefault="qualified" targetNamespace="http://tempuri.org/">
      - <s:element name="AddTicker">
        - <s:complexType>
          - <s:sequence>
              <s:element minOccurs="0" maxOccurs="1" name="name" type="s:string"/>
              <s:element minOccurs="1" maxOccurs="1" name="val" type="s:int"/>
            </s:sequence>
          </s:complexType>
        </s:element>
```

Avoiding Information Gathering

- Reduce exposure
- Privately exchange WSDL
- Don’t assume!

Denial of Service Attack

DTD Interpretation

- Servers can accept and interpret DTDs provided by clients
- Complex/large/recursive DTD can overload parser

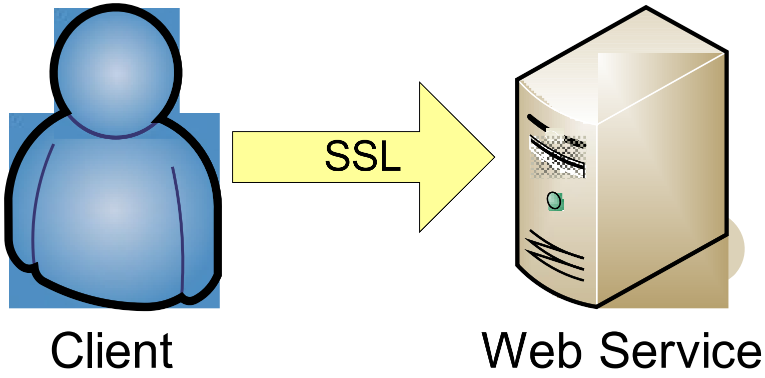
Solution Options

- Disable support for DTDs, use XSD instead
- Ideally, don’t accept any form of schema definition from the client

Message Confidentiality

Option 1: SSL/TLS

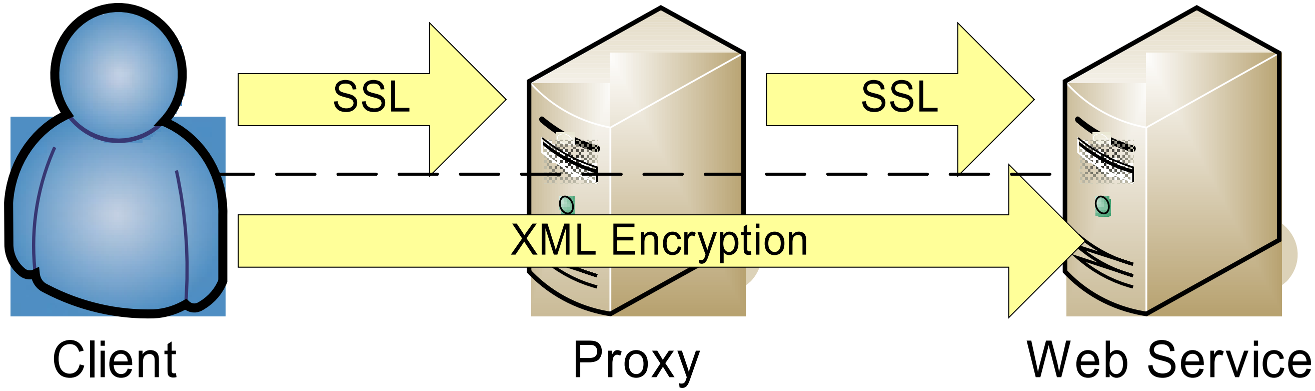
- Common, widely supported
- Fine for single hops
- Point to point encryption



Message Confidentiality

Option 2: XML Encryption

- End to end encryption
- Intermediate servers are able to see the request for routing
- Encrypts only a specific portion of the data



Authentication

- Federated Identity
  - Leverage existing solution
  - Can use SAML to communicate assertions
- Single-Use Authentication
  - Options available from architecture
  - Custom solutions

Access Control

Forced requests for:

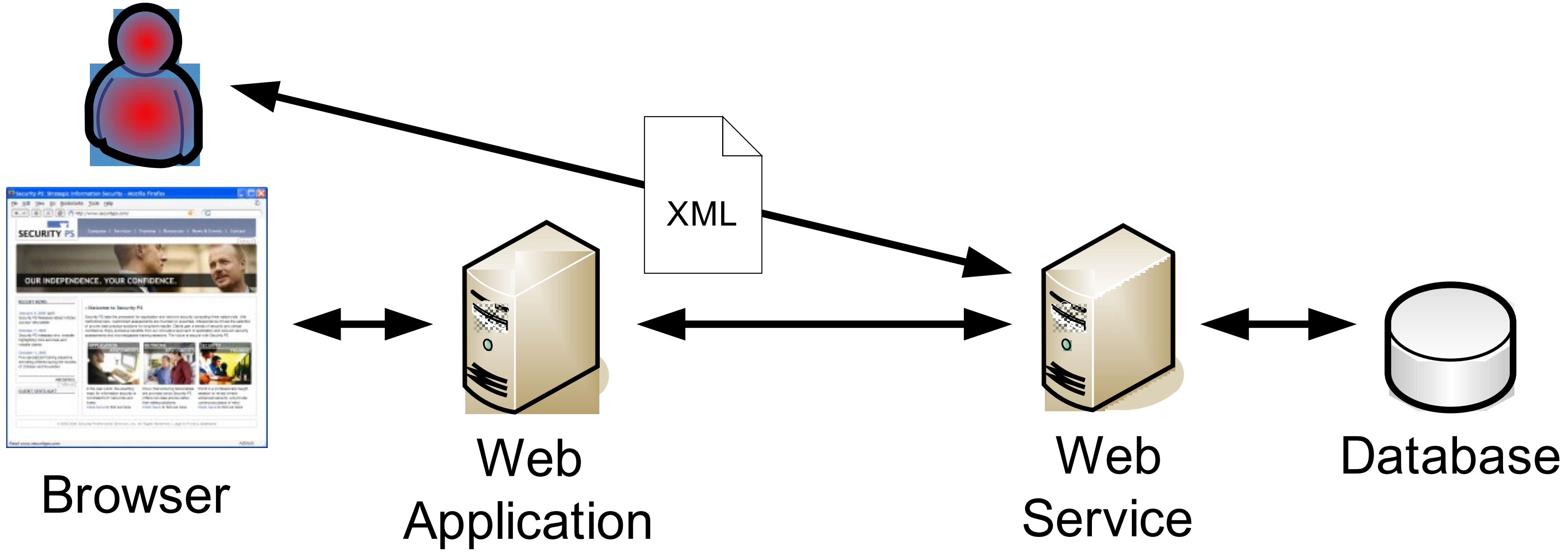
- Services: by URL
- Methods: by modifying service method
- Data: by manipulating parameter values

Results

- Access to web service without authentication
- Escalation of privileges
- Crossed permission boundaries

Unauthorized Access to a Web Service

- Network firewalls may not help
- Access control must be coded into the application or provided by the web server



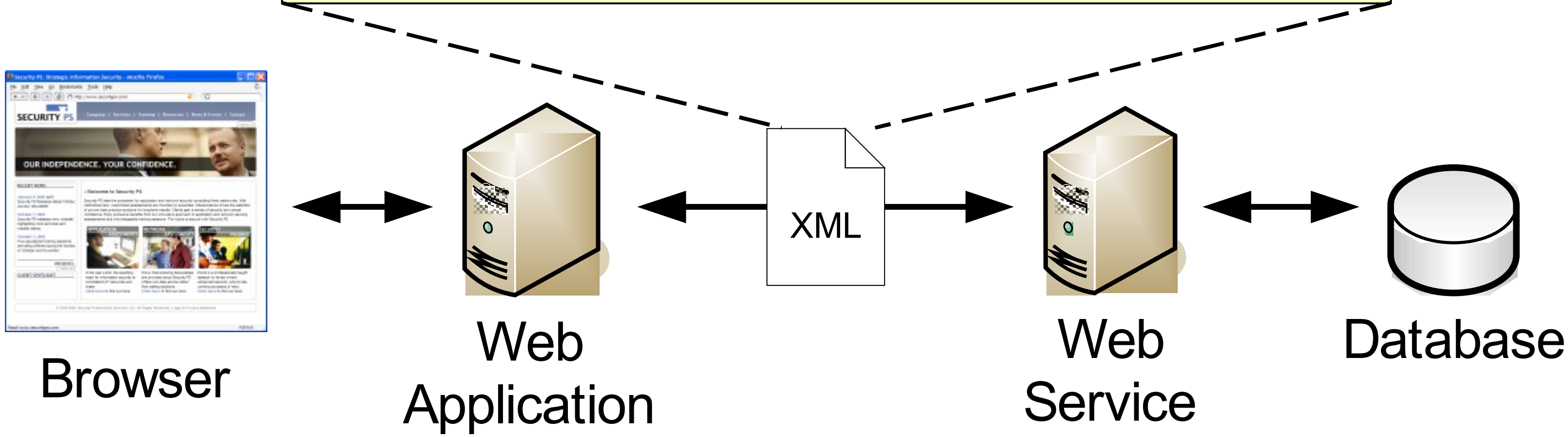
Data Validation and Encoding

- XML Data Injection
- XML Parser Command Injection
- SQL Injection
- Cross Site Scripting

XML Data Injection

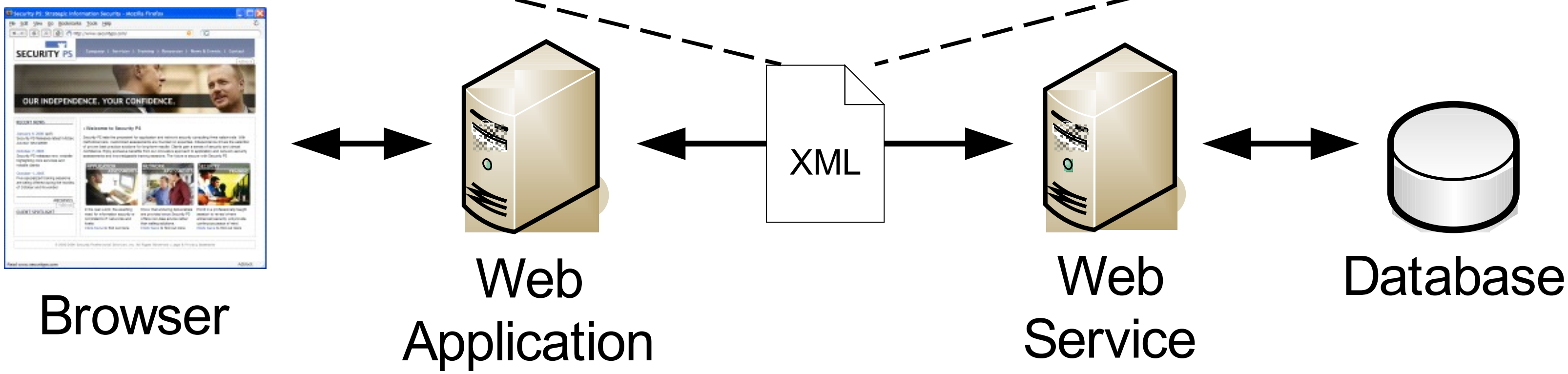
Example: User Account Creation

```
<UserRecord>
  <UserID>983</UserID>
  <Name>Dave Green</Name>
  <Email>dgreen@securityps.com</Email>
  <Phone>913-888-2111</Phone>
</UserRecord>
```

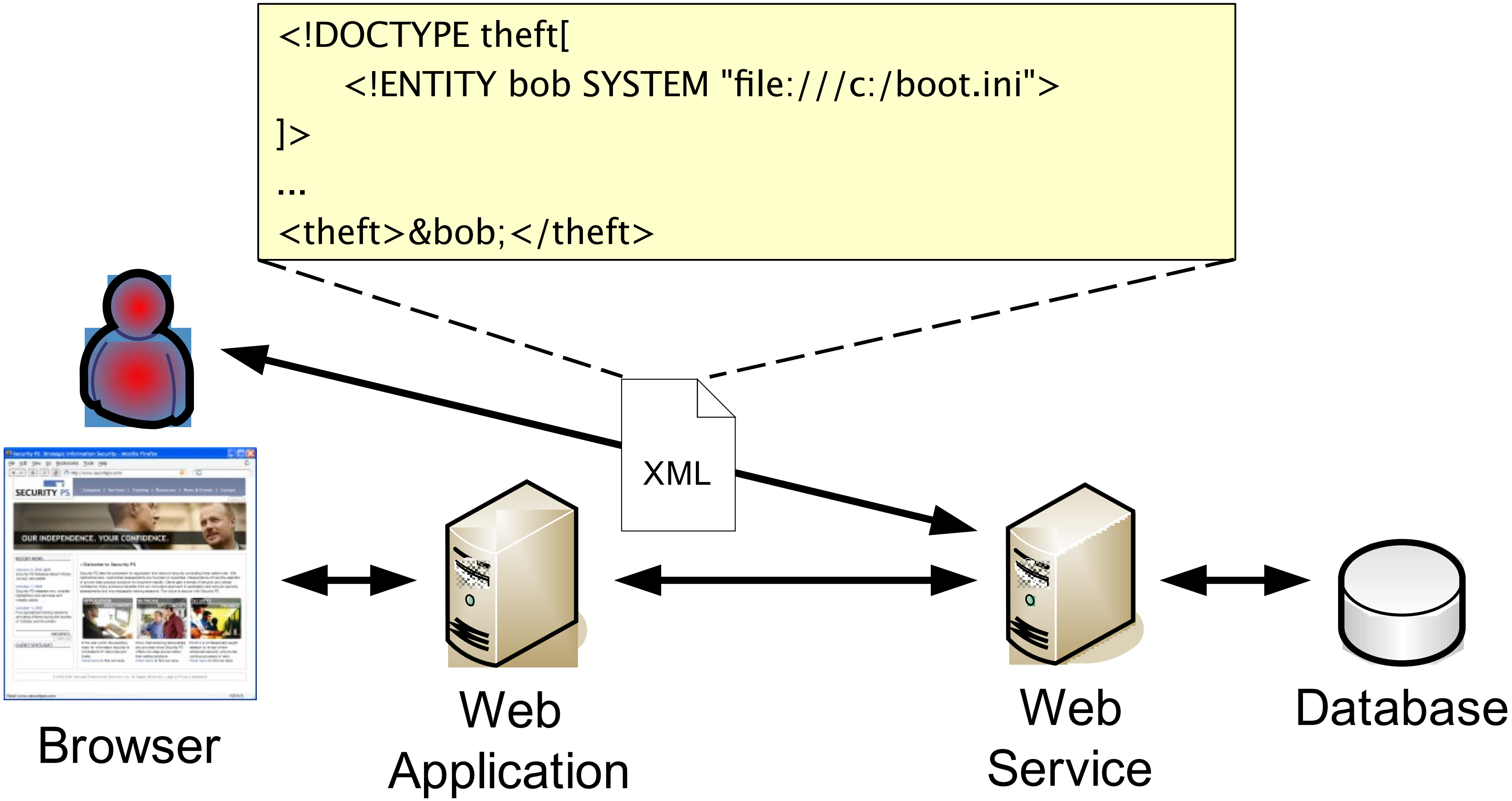


XML Data Injection Attack

```
<UserRecord>
  <UserID>859</UserID>
  <Name>Mr. Evildoer</Name>
  <Email>evil@3mu.us</Email><UserID>1</UserID><Email>evil@3mu.us</Email>
  <Phone>913-234-6789</Phone>
</UserRecord>
```

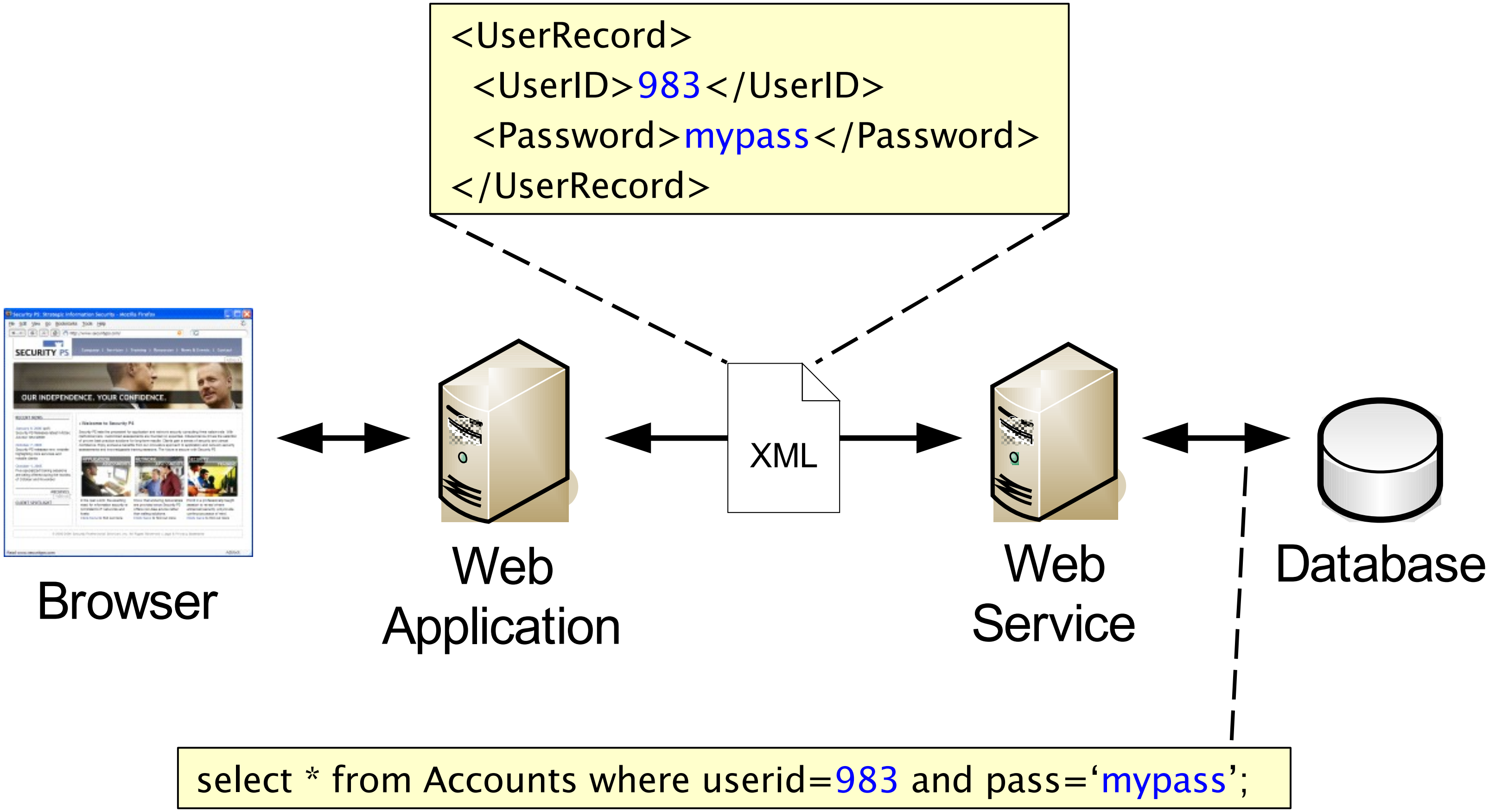


XML Parser Command Injection



```
<!DOCTYPE theft[
      <!ENTITY bob SYSTEM "file:///c:/boot.ini">
]>
...
<theft>&bob;</theft>
```

SQL Injection

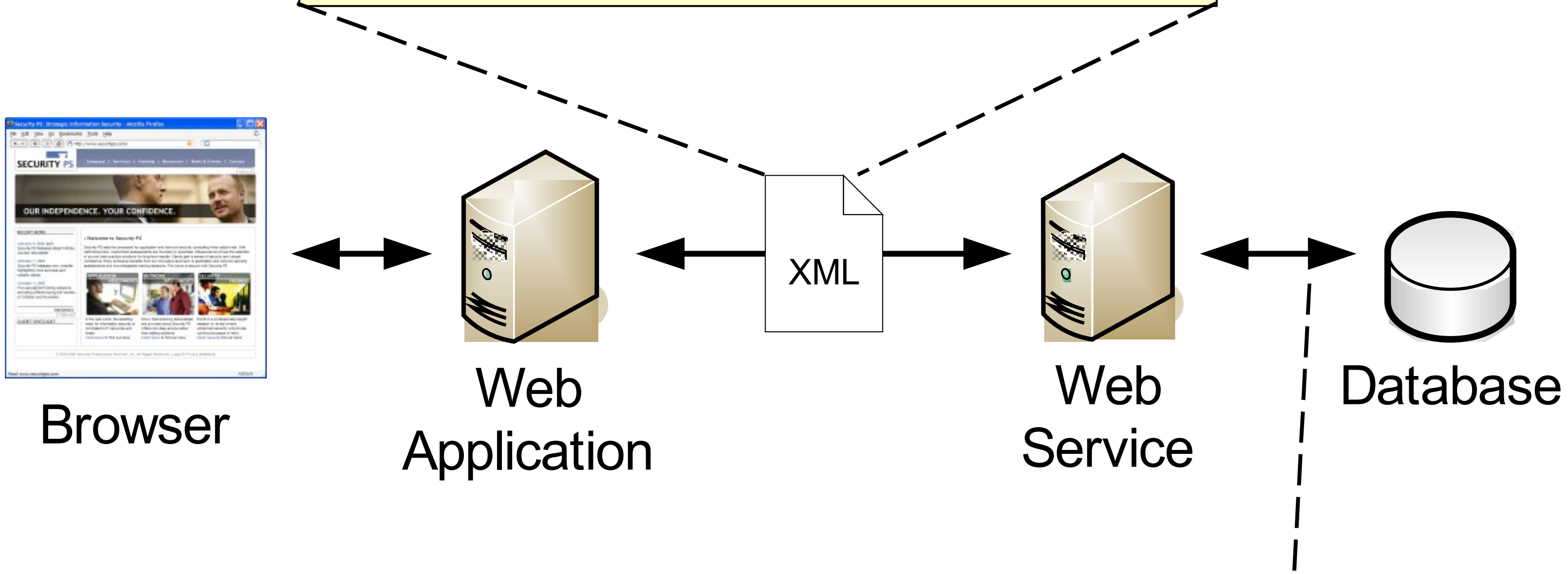


```
<UserRecord>
  <UserID>983</UserID>
  <Password>mypass</Password>
</UserRecord>
```

```
select * from Accounts where userid=983 and pass='mypass';
```

SQL Injection Attack

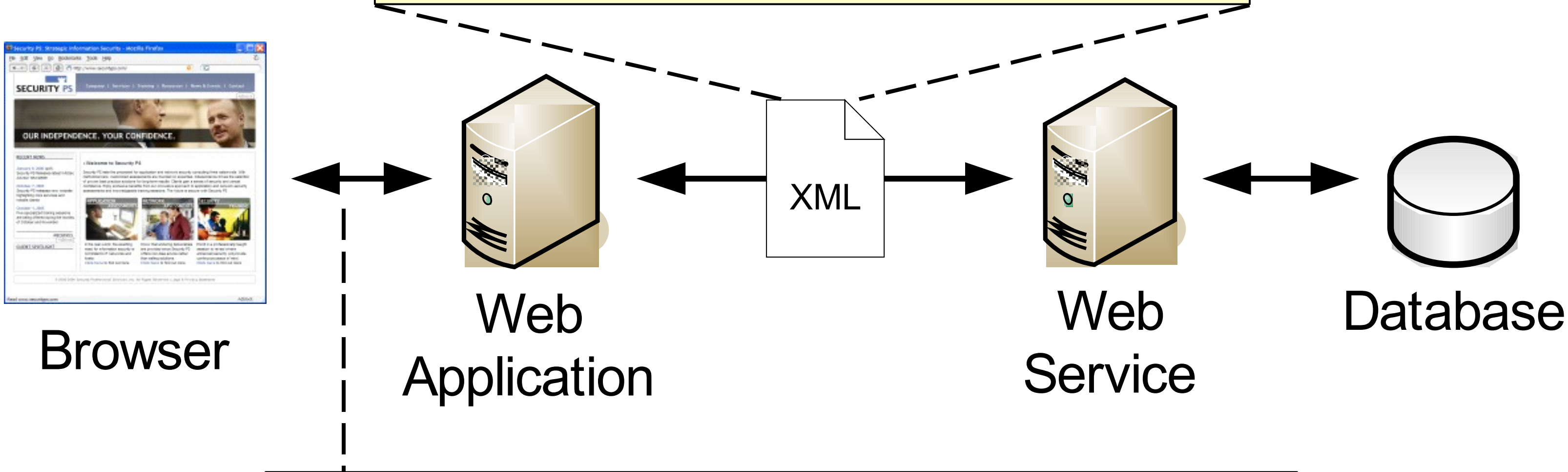
```
<UserRecord>
  <UserID>983</UserID>
  <Password>a’ or ‘a’=‘a</Password>
</UserRecord>
```



```
select * from Accounts where userid=983 and pass=‘a’ or ‘a’=‘a’;
```

Cross Site Scripting

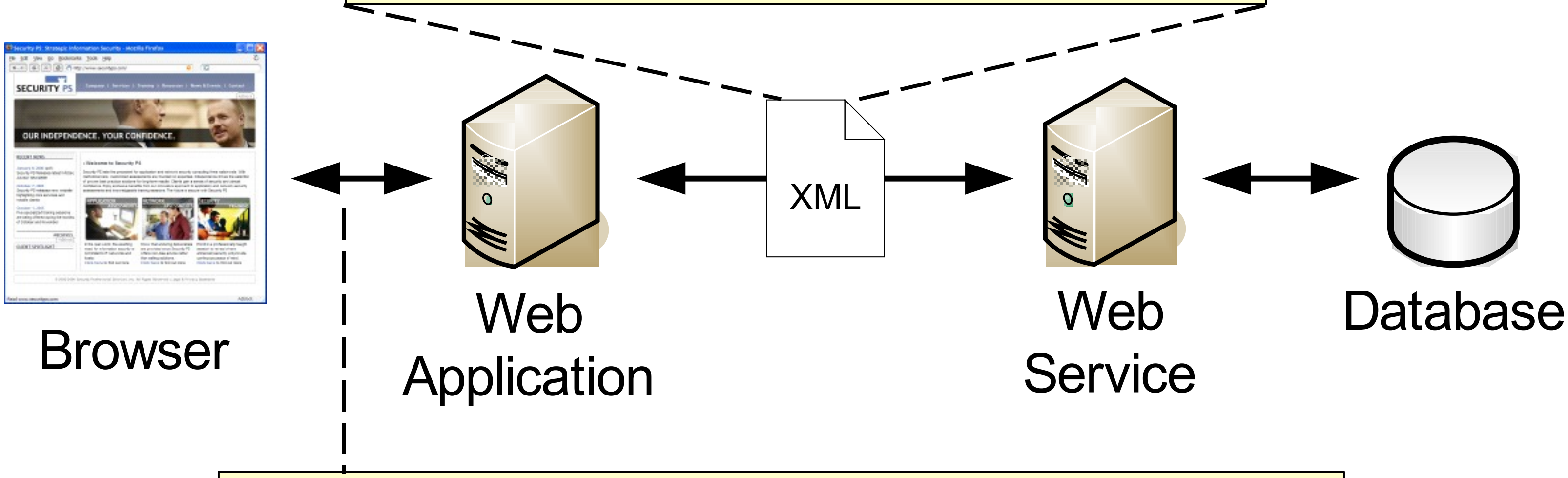
```
<UserRecord>
  <UserID>983</UserID>
  <Name>Dave Green</Name>
  <Email>dgreen@securityps.com</Email>
  <Phone>913-888-2111</Phone>
</UserRecord>
```



```
Dave Green<br>
<a href="mailto:dgreen@securityps.com">Email</a>
```

Cross Site Scripting Attack

```
<UserRecord>
  <UserID>983</UserID>
  <Name>Dave Green</Name>
  <Email>” onMouseOver=“alert(‘xss’);</Email>
  <Phone>913-888-2111</Phone>
</UserRecord>
```



```
Dave Green<br>
<a href=“mailto:” onMouseOver=“alert(‘xss’);”>Email</a>
```

Data Validation and Encoding Solutions

- Integrity
  - XML Signature
  - Other Cryptography
- Datatyping
  - XML schema definitions – must be detailed
- Application Logic
  - Input validation: size, type, sanity
  - Output encoding: ensure data stays data

Conclusions

Summary of Risks

Risk Mitigation Strategy

- Security Frameworks
- Web Application Security Products
- Effective App Security: The SDLC

- Summary of Risks
  - Information Gathering
  - Denial of Service
  - Message Confidentiality
  - Authentication
  - Access Control
  - Data Validation and Encoding

Summary of Risks (cont.)

No browser, but still need to defend against common web application attacks

OWASP Top 10 are still valid

Unvalidated Input

Broken Authentication and Session Management

Buffer Overflows

Improper Error Handling

Denial of Service

Broken Access Control

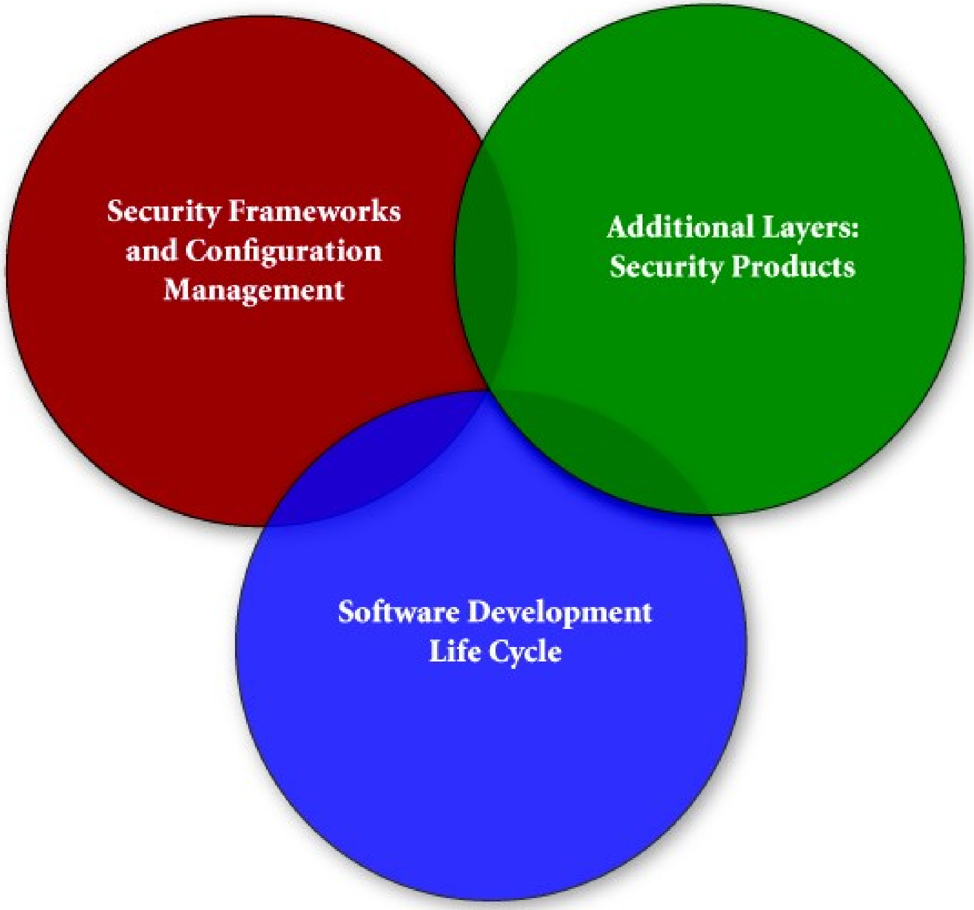
Insecure Configuration Management

Injection Flaws

Insecure Storage

Cross Site Scripting

Risk Mitigation Strategy


Security Frameworks and Configuration Management
Additional Layers: Security Products
Software Development Life Cycle

Security Framework Features

Many security frameworks available today provide effective high-level access to important functions such as:

User Authentication/Access Controls

Auditing

Encryption

Key Management, Certificate Management

General object permission controls

Additional Security Layers: Products

In the wake of a large number web application security problems, many products have been introduced to help limit risk of vulnerable applications.

- Automated vulnerability scanning tools
- Incoming filters/proxies (App firewalls)
- Outgoing filters/validators
- Back-end filters/proxies
- Hybrids or multi-purpose systems

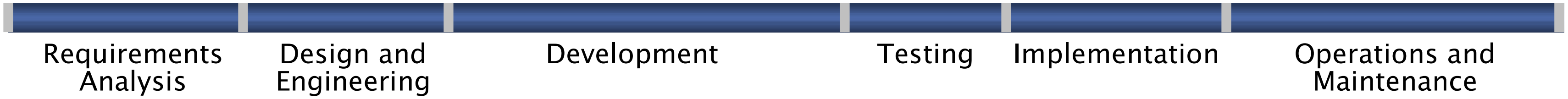
In Perspective

These devices:

- When used correctly, can reduce specific risks.
- Provide only one line of defense.
- Do <u>**not**</u> replace the need for secure application design/development.
- Should not be the only application security layer

Conclusions

- Effective Application Security Efforts
  - Effectiveness: Results, Cost, Longevity/ROI
  - Consider the entire **Software Development Lifecycle (SDLC)**



Requirements Analysis

Design and Engineering

Development

Testing

Implementation

Operations and Maintenance

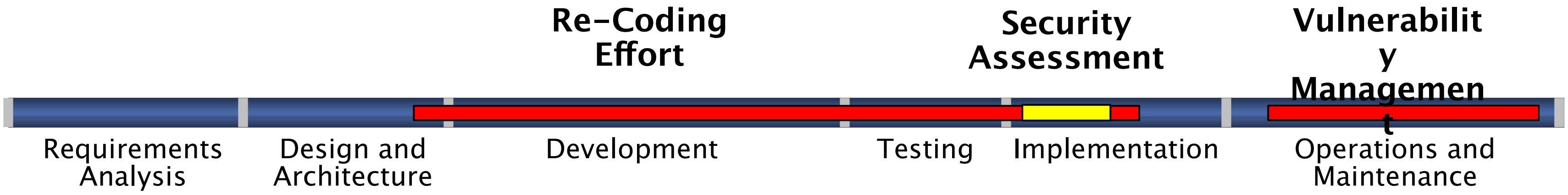
Conclusions

"Tactical Only" Approach

Design and develop first, secure later

General results:

- Apps deployed with risk
- Re-writes are costly, significant at this stage
- Vulnerabilities addressed
- Root cause rarely addressed



Strategic View of Application Security

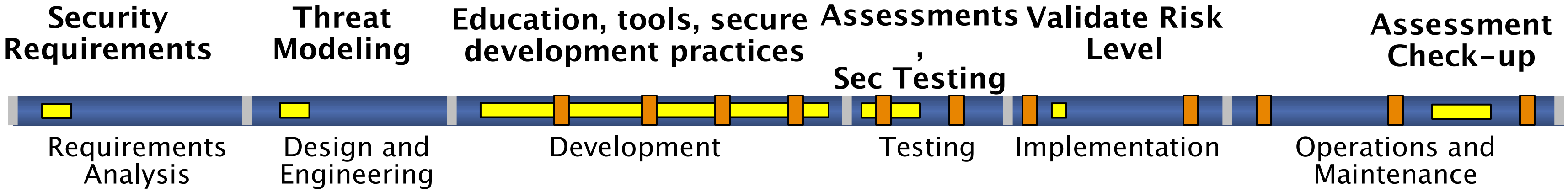
Security is a process, not a task

Early security results in less cost and strong ROI

Address the root cause, not the symptom

Address the practices first, vulnerabilities last

Incorporate proven practices into all phases of the software development lifecycle



Conclusions

Activities	Core	Security
Planning		
Requirements and Analysis	Functional Requirements Non Functional Requirements Technology Requirements	Security Objectives
Architecture and Design	Design Guidelines Architecture and Design Review	Security Design Guidelines Threat Modeling Security Architecture and Design Review
Development	Unit Tests Code Review Daily Builds	Security Code Review
Testing	Integration Testing System Testing	Security Testing
Deployment	Deployment Review	Security Deployment Review
Maintenance		

Microsoft’s Key Security Activities Mapped to the SDLC

Questions

&

Discussion