启用源代码分析技术处理大数据

Checkmarx中国区技术专家

陈安明

OWASP 中国

The Open Web Application Security Project

陈安明介绍

OWASP 中国
The Open Web Application Security Project

- **端玛科技总经理，独立应用安全风险分析师，Checkmarx中国区技术专家。**
- 是中国最早从事源代码分析技术调查和研究人员，专门从事应用软件安全风险评估、风险消除、培训、教育和软件安全生命开发周期SDL咨询。其优秀的软件安全方案、产品及专业化的软件安全开发生命周期SDL服务已进入金融银行、保险、电信、汽车、媒体娱乐、软件、服务和军事等财富1000的企业

CHECKMARX
SOURCE CODE ANALYSIS TECHNOLOGIES
Dumasoft 端玛科技

议题概述

OWASP 中国
The Open Web Application Security Project

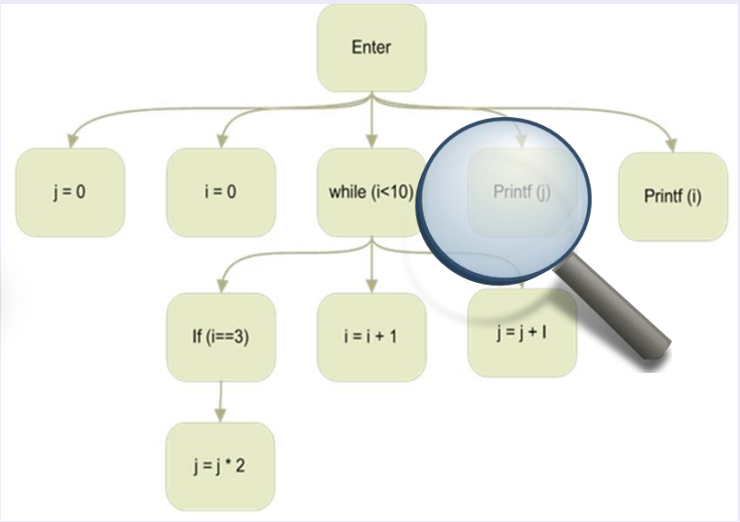
- 传统以安全为导向的源代码分析工具只能检测到黑客明显可以利用的漏洞，而且这些工具所找到的安全漏洞的数量非常多，即使这些结果是精确的，都很难在短时间修复，这样一来，我们就不得不面临两个现实问题：
  - 一、我们如何对付那些工具没有覆盖到的代码？
  - 二、我们怎样才能提高安全漏洞修复的能力？
- 为了应对这些挑战，我们把研究的侧重点放在了大数据分析领域，将大数据的先进技术与我们的研究整合到一起。借这次交流的机会，我想与大家分享一下我们的研究方法以及我们的成果。

源代码分析的历史

OWASP 中国
The Open Web Application Security Project

- **第一代源代码分析**
  - **系统安全知识是通过绑定静态的规则体现。**静态规则依据原始或者标准语言的缺陷来制定的，对用户而言，技术是不可见的。
  - **用户代码架构和框架适应能力差。**几乎无法适应在开发语言基础上用户私有的架构和框架代码封装的扫描。规则主要细节不公开，用户很难自定义或者调整规则满足用户自身的系统架构和代码封装的需求。
  - **使用依赖操作系统环境和编译器**

```
void main()
{
   int j = 0;
   int i = 0;

   while (i < 10){
      if (i == 3){
         j=j*2;
      }
      j = j + i;
      i = i + 1;
   }

   printf ("%d\n", j);
   printf ("%d,n", i);
}
```



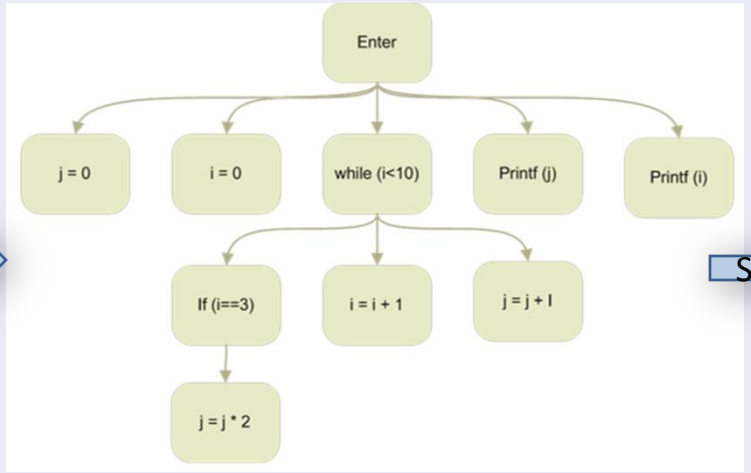
源代码分析的历史(续)

OWASP 中国
The Open Web Application Security Project

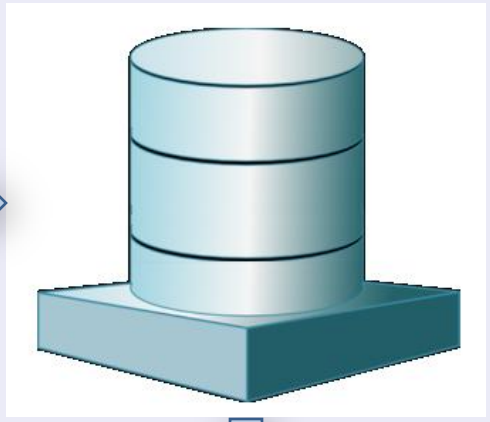
- **新一代源代码分析**
  - **系统的安全知识是通过绑定静态的规则体现。**静态规则依据原始或者标准语言技术架构和框架的缺陷来制定的，**公开规则实现的技术和细节。**
  - **用户代码架构和框架适应能力强。**适应在开发语言基础上用户私有的架构和框架代码的扫描。规则主要细节完全公开，用户很容易自定义或者调整规则满足用户自身的系统架构和代码封装的需求。
  - **能够任意添加自己需要的有关业务逻辑和代码质量相关的查询**
  - **使用简便！虚拟编译器，无须代码编译。**无需依赖操作系统环境和编译。
  - 分析范围：SQL 注入-〉恶意后门-〉代码质量缺陷

```
void main()
{
   int j = 0;
   int i = 0;

   while (i < 10){
      if (i == 3){
         j=j*2;
      }
      j = j + i;
      i = i + 1;
   }

   printf ("%d\n", j);
   printf ("%d,n", i);
}
```

Abstract



Store



示例

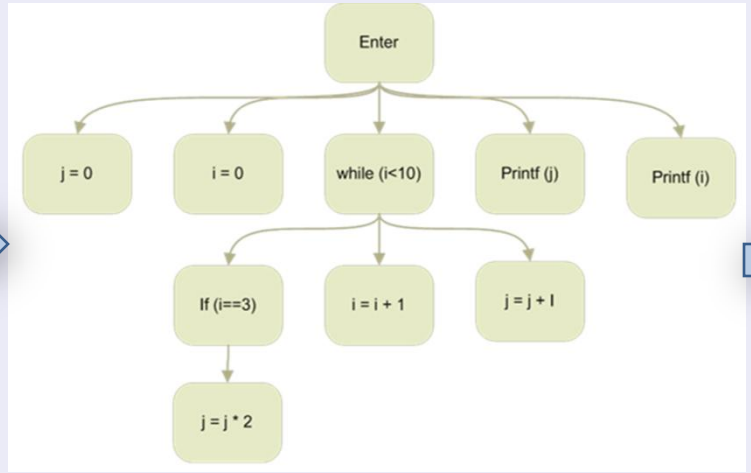
OWASP 中国

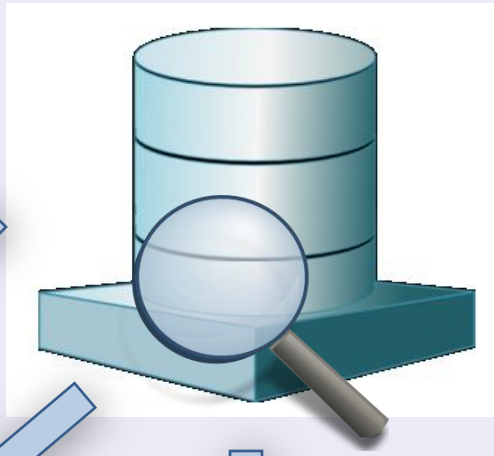
The Open Web Application Security Project

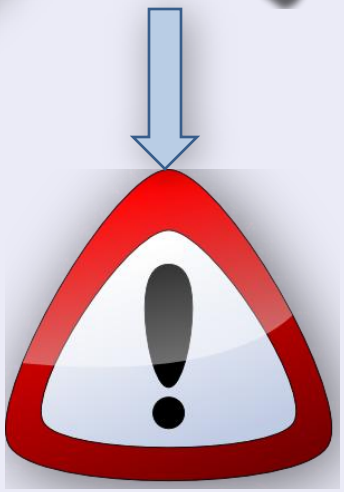
应用程序智能分析

OWASP 中国
The Open Web Application Security Project

```
void main()
{
   int j = 0;
   int i = 0;

   while (i < 10){
      if (i == 3){
         j=j*2;
      }
      j = j + i;
      i = i + 1;
   }

   printf ("%d\n", j);
   printf ("%d,n", i);
}
```







OWASP 中国
The Open Web Application Security Project

SCKD

源代码知识发现

"使用群体的智慧"（大数据）

通过代码的不规则性来识别安全漏洞

零日？零配置？

OWASP 中国
The Open Web Application Security Project

- 如果我们连自己要问什么问题都不清楚，该怎么办呢？
- 如果我们没办法对系统进行配置，怎么办？
- 我们需要一位“大师”，
  - 来替我们问问题。
  - 替我们配置系统。
  - 替我们找到漏洞。
  - 给我们提供指导。

有这样一位大师

OWASP 中国
The Open Web Application Security Project

- 是你！
- 是你！
- 是你！
- 还是你！！！
- 我们大家 – 形成集体智慧
- 大多数开发人员在大多数时间都能编写出好的、标准 、高质量代码

群体

OWASP 中国
The Open Web Application Security Project

- 我们可以根据代码统计来设定一个基准，并发现偏差。

源代码知识发现-SCKD

- 源代码知识发现– 时下最为活跃 的研究课题之一
(数据库中的知识发现- http://en.wikipedia.org/wiki/Knowledge_extraction)

> *“知识发现描述的是一个自动搜索大规模数据模式的过程，而该模式可以被称之为有关数据的知识。通常我们称之为来自于输入数据的知识。从方法和术语两个层面来说，知识发现与其发掘来源数据领域的关系都非常紧密。”*

技术实现

OWASP 中国
The Open Web Application Security Project

- 建立参考数据、
- 寻找共同序列
- 查找违规情况

获取数据

OWASP 中国
The Open Web Application Security Project

```
M = input();
If (isValid(M))
{
   …
      response.write(M);
   …
}
```

```
A = input();
If (isValid(A))
{
   …
      response.write(A);
   …
}
```

查找偏差，设立基准


```
* = input();
If (isValid(*))
{
   …
      response.write(*);
   …
}
```

```
v = input();
```

X

?

```
…
response.write(v);
…
```

后门 – 若我的名字是Maty，登录

OWASP 中国
The Open Web Application Security Project

```
If (isAuthenticated(user)) || user.name == “maty”)
   {
        ....
   }
```

增值– 利用应用云服务？

OWASP 中国
The Open Web Application Security Project

寻找不同应用之间的相似之处，建立一个内部标准。

使用零定义！

只要我们能修复一些应用就行，这些应用会帮助我们找到那些没被修复的。

VAT = 1.05

…

VAT = 1.08

…

VAT = 1.08

…

VAT = 1.08

我们的优势

OWASP 中国
The Open Web Application Security Project

- 总体来说：
  - 我们能够找到群体中隐藏的知识，给它命名，并找到违规情况。
- 安全：
  - 确定在每一个页面都对客户进行验证
  - 自动识别消毒程序
  - 后门（“if (isValid(user) or user== “Maty”)…”）
  - 业务逻辑(“if (qty > 0) {charge (qty*amnt)}”）
- 质量
  - 发布的永远都是具体资源
  - 最佳编码实践（自动识别策略）
  - 变量初始化
- 群体的智慧
  - 对于大型企业和代码库作用更为明显

OWASP 中国
The Open Web Application Security Project

图形可视化

优化代码修复活动

“使用智能图形方法识别安全漏洞交汇的连接节点及最佳的修复位置”

代码扫描结果修复问题

OWASP 中国
The Open Web Application Security Project

- 找到数以千计的准确结果，但其实并不是我们希望看到的。
- 例如Webgoat有大约220个跨站脚本和SQL 注入漏洞。
- 假设我们需要30分钟来修复一个漏洞+30分钟来验证修复，那就会需要220个小时，几乎是一个月的工作量
- 我们把这些工作缩到16个地方
- 约耗费1/14 的时间
- 这样一来，我们就有时间去打打高尔夫球了☺

代码扫描分析结果现状

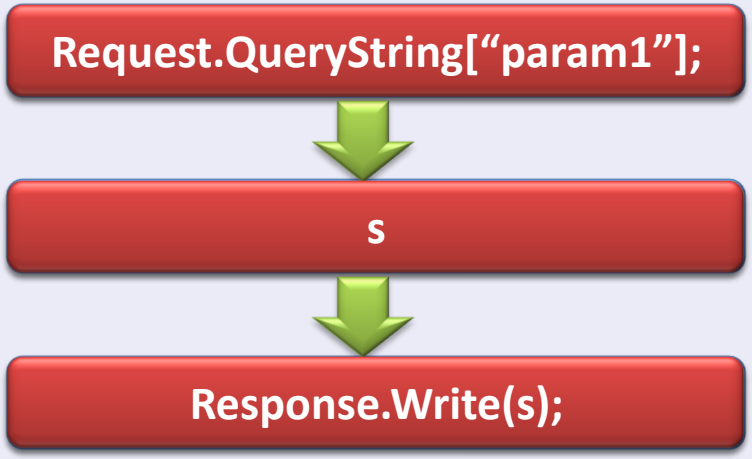
OWASP 中国
The Open Web Application Security Project

- 每一个代码扫描结果都有一个独立于其它调查结果的数据流。

单一数据流路径 – 跨站脚本

OWASP 中国
The Open Web Application Security Project

```
String s = Request.QueryString[“param1”];
...
Response.Write(s);
```

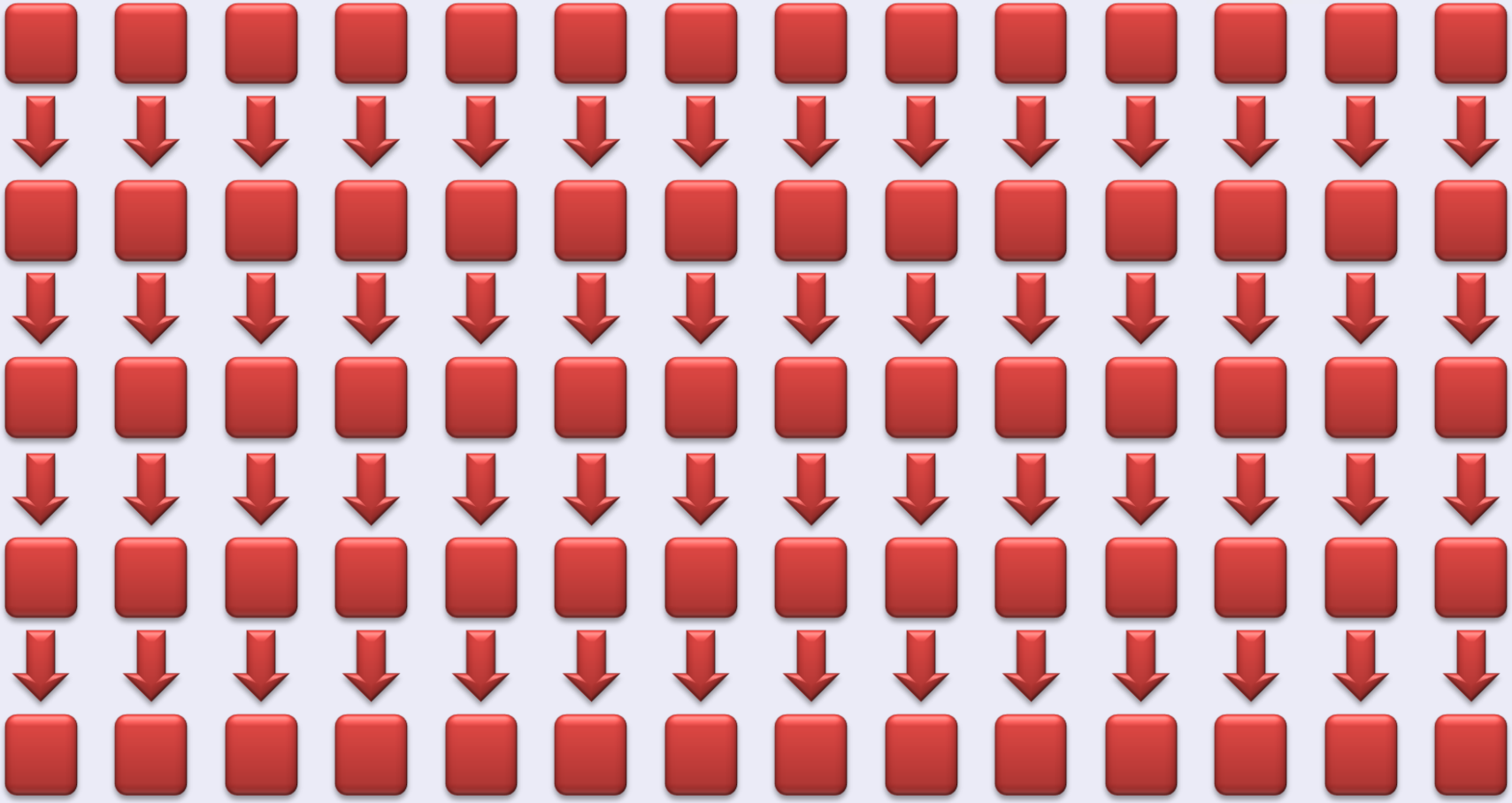


代码扫描分析结果现状

OWASP 中国
The Open Web Application Security Project

- 一个漏洞，很好解决。
- 14个同时出现，怎么办？

多个单一路径– 跨站脚本 – 大量工作



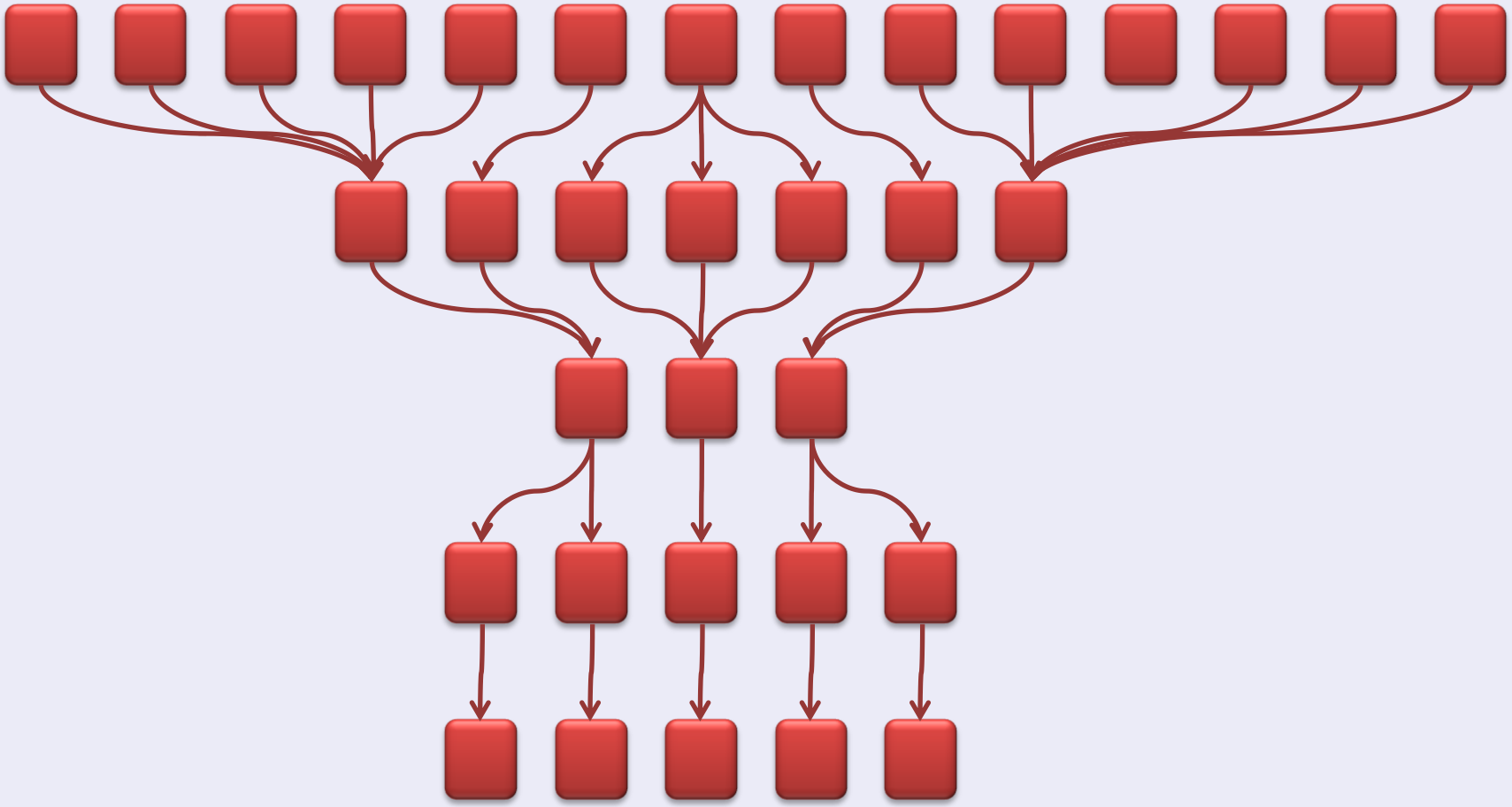
但。。。

OWASP 中国
The Open Web Application Security Project

- 他们有什么共同点？
- 这些扫描结果之间有关联吗？

组合路径

OWASP 中国
The Open Web Application Security Project



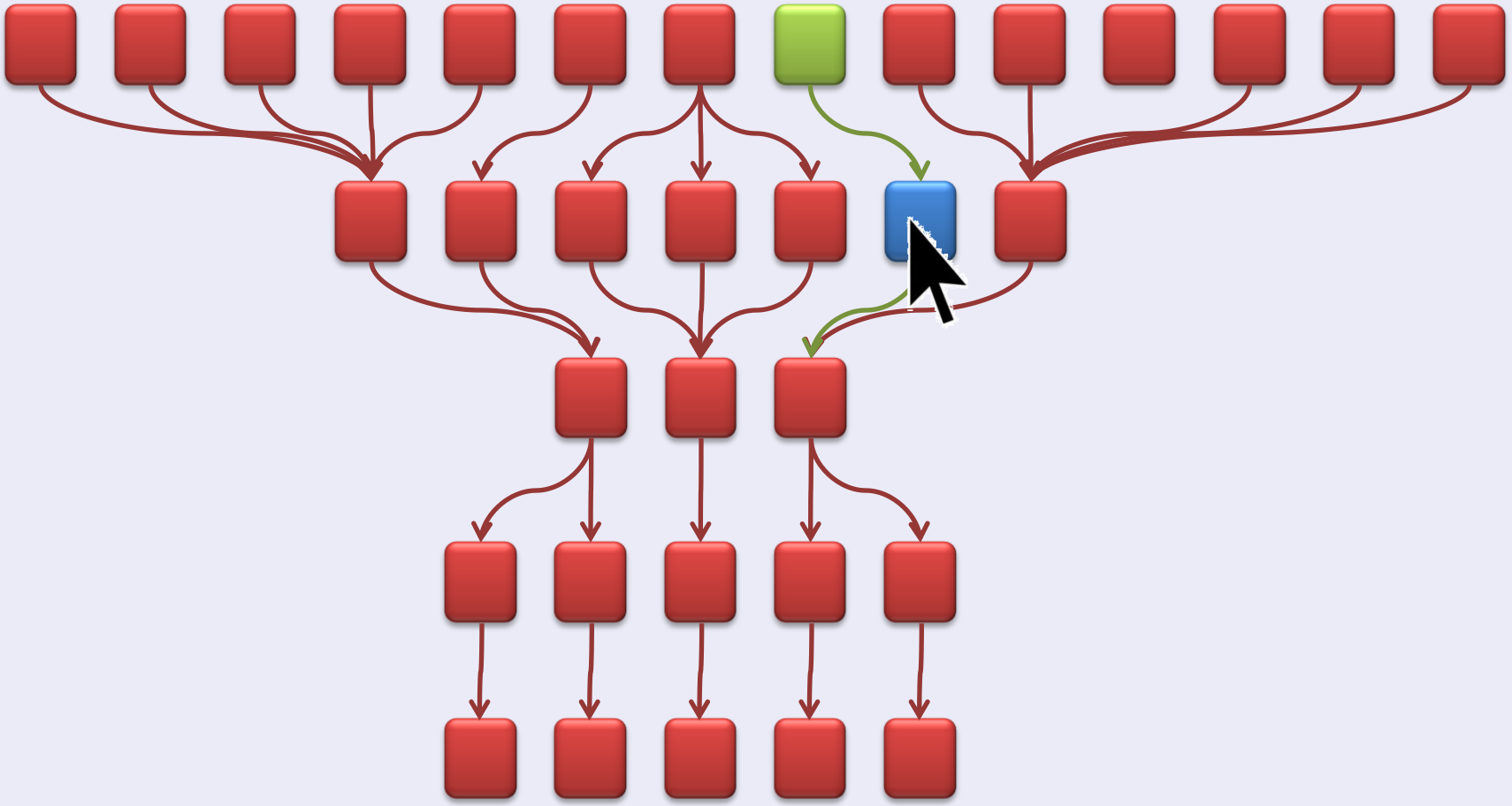
我们就能够。。。

OWASP 中国
The Open Web Application Security Project

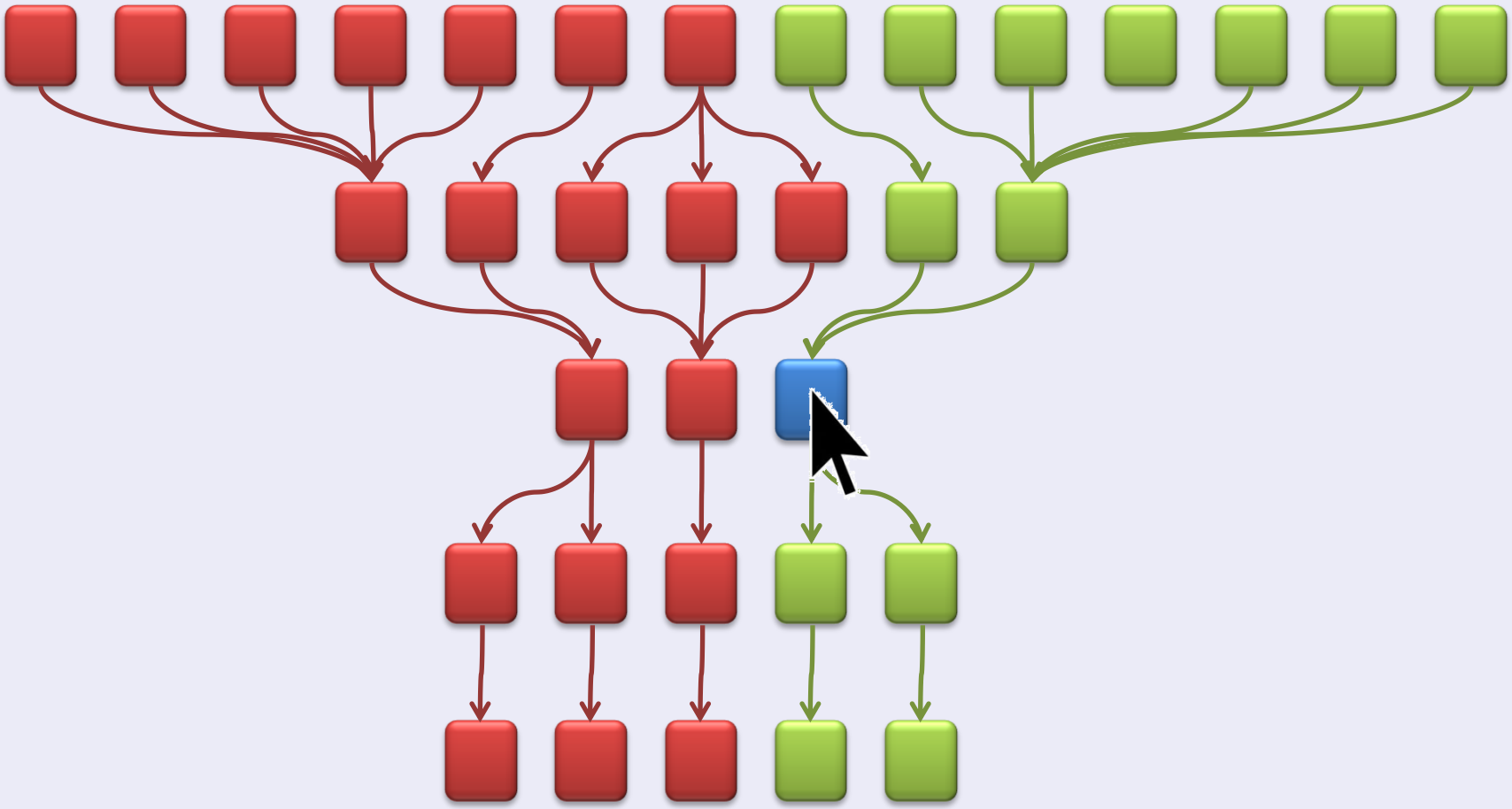
- 指出、点击、检查，甚至连源代码都不用读吗 “
- 我修复这里可以吗？” “或者这里？”

我修复这里可以吗？

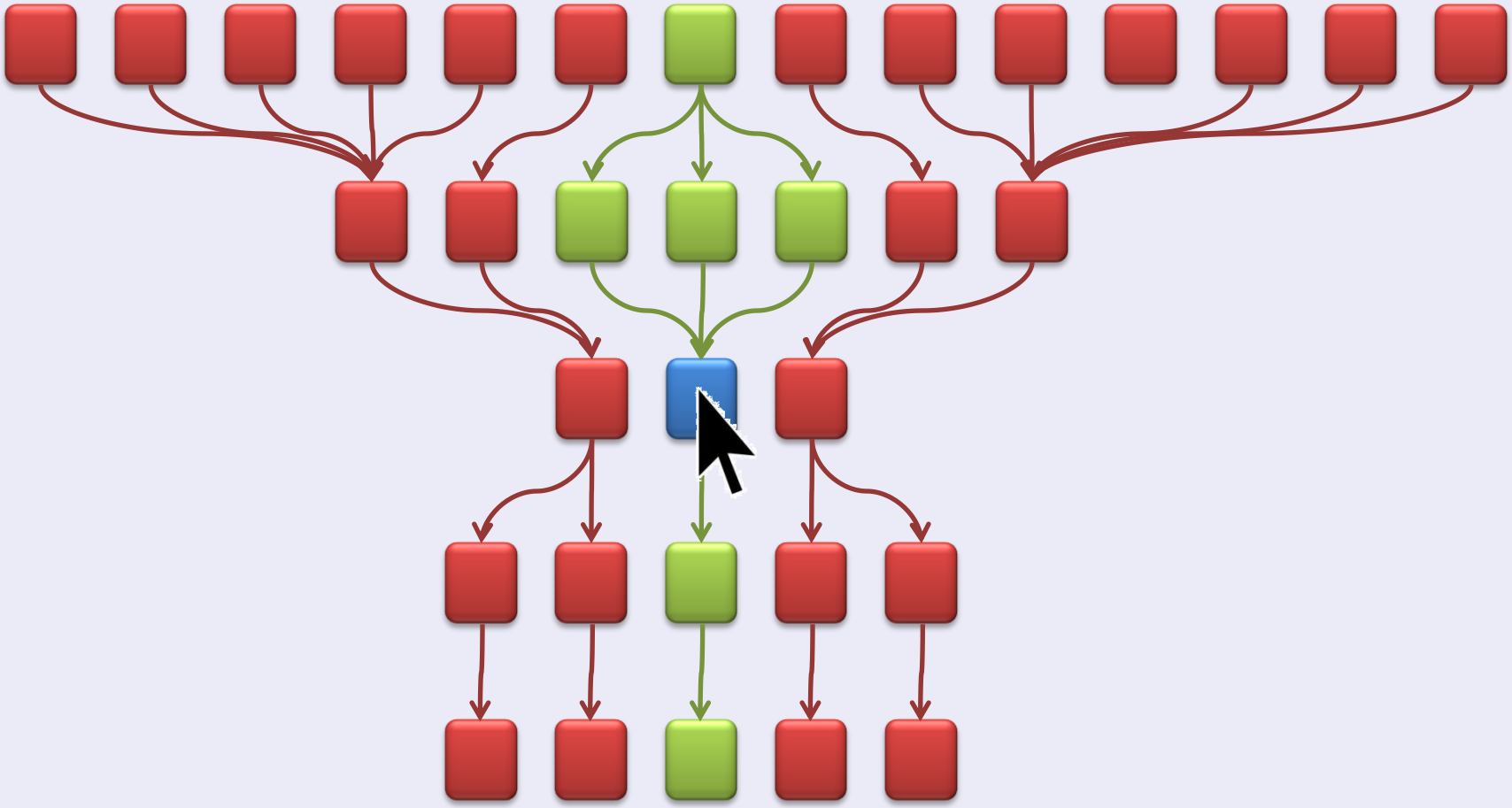
OWASP 中国
The Open Web Application Security Project



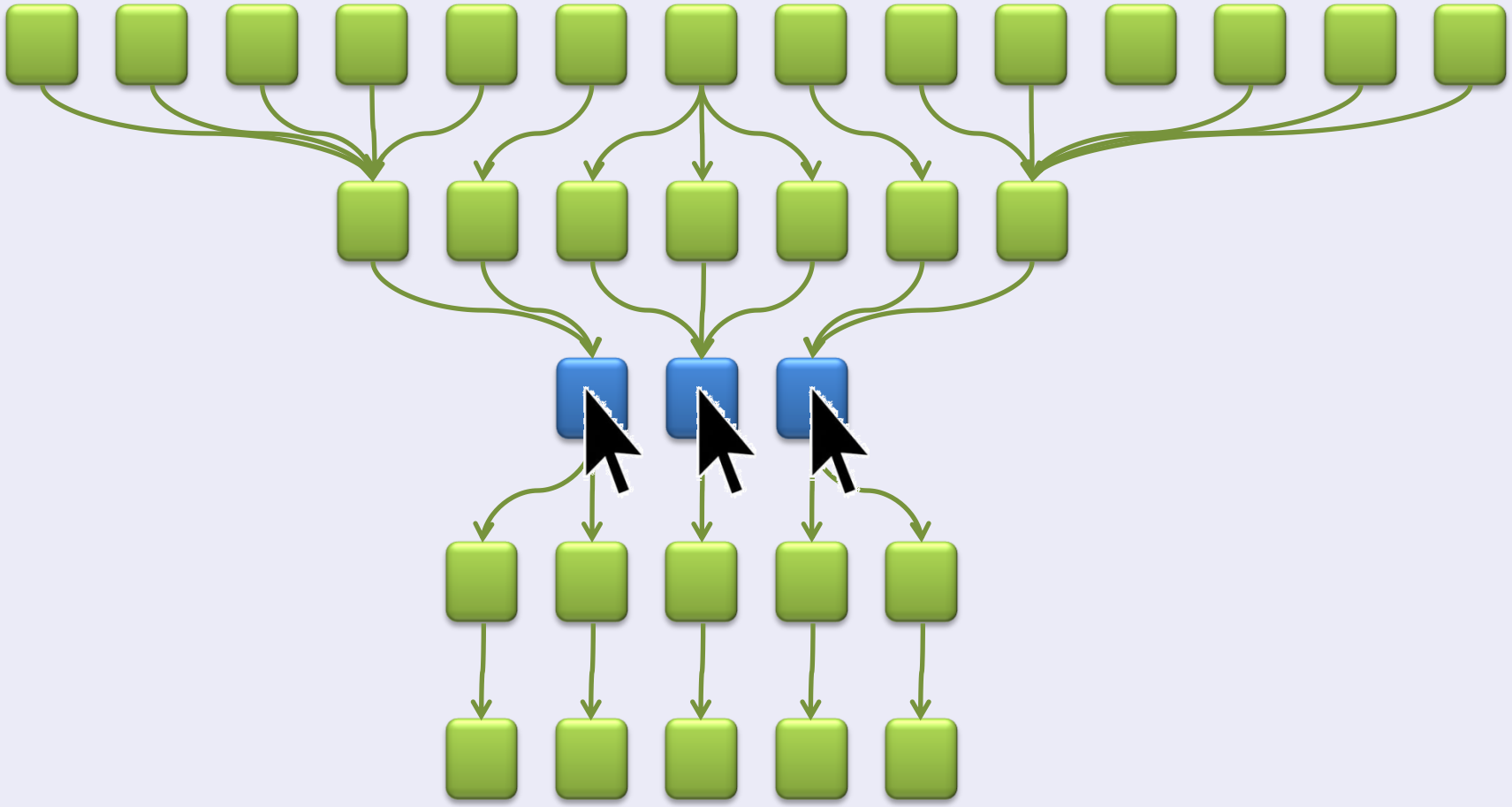
修复这里其实效率更高



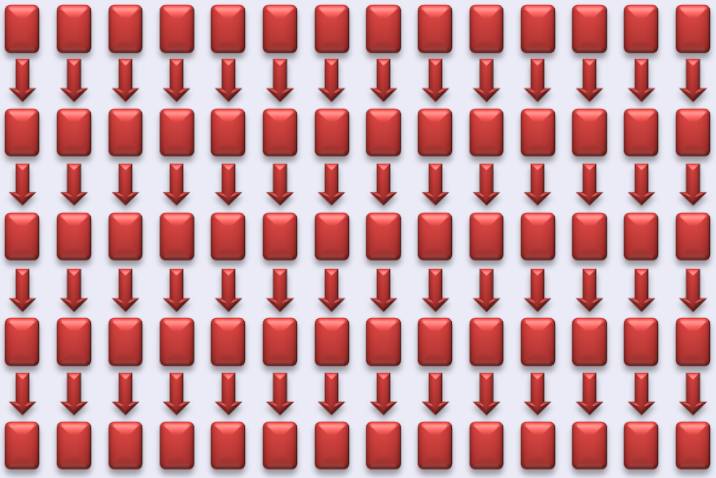
还有这？

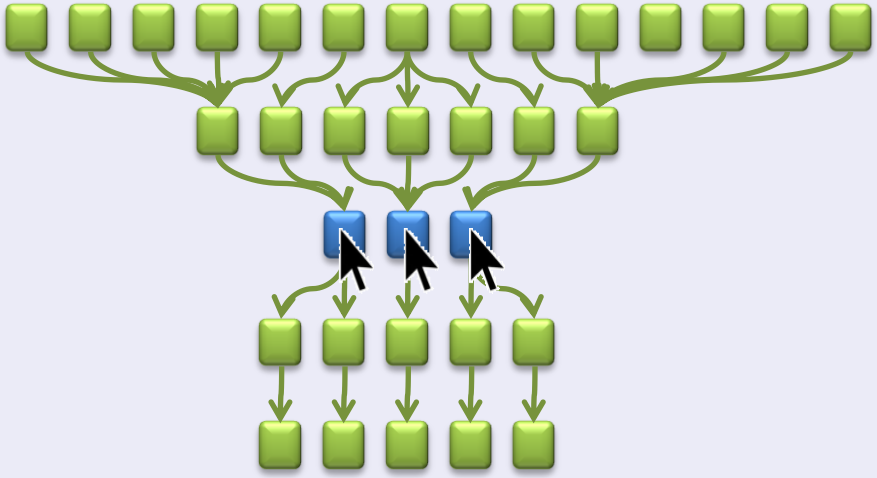


自动提出“假设”问句=>找到最佳的修复位置



比较这两种情况：





图形可视化修复建议的优势

OWASP 中国
The Open Web Application Security Project

- 展示相同安全漏洞类型和不同问题调用之间的相关性。
- 处理的并不是个人或者单个问题路径的发现– 事实上是整个系统。
- 让您得以更好地利用时间

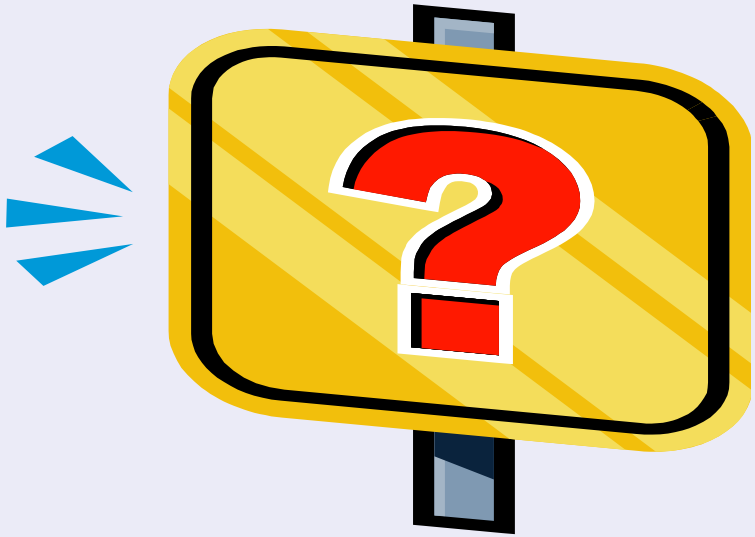
Webgoat 220个修复位置示例

OWASP 中国
The Open Web Application Security Project

- 只要轻轻一点，我们就可以把220个位置缩减到16个。
- 结果越多，我们的解决方式就越能体现其效率

Q&A

- Any Question？



代码安全漏洞和质量缺陷扫描云服务中心

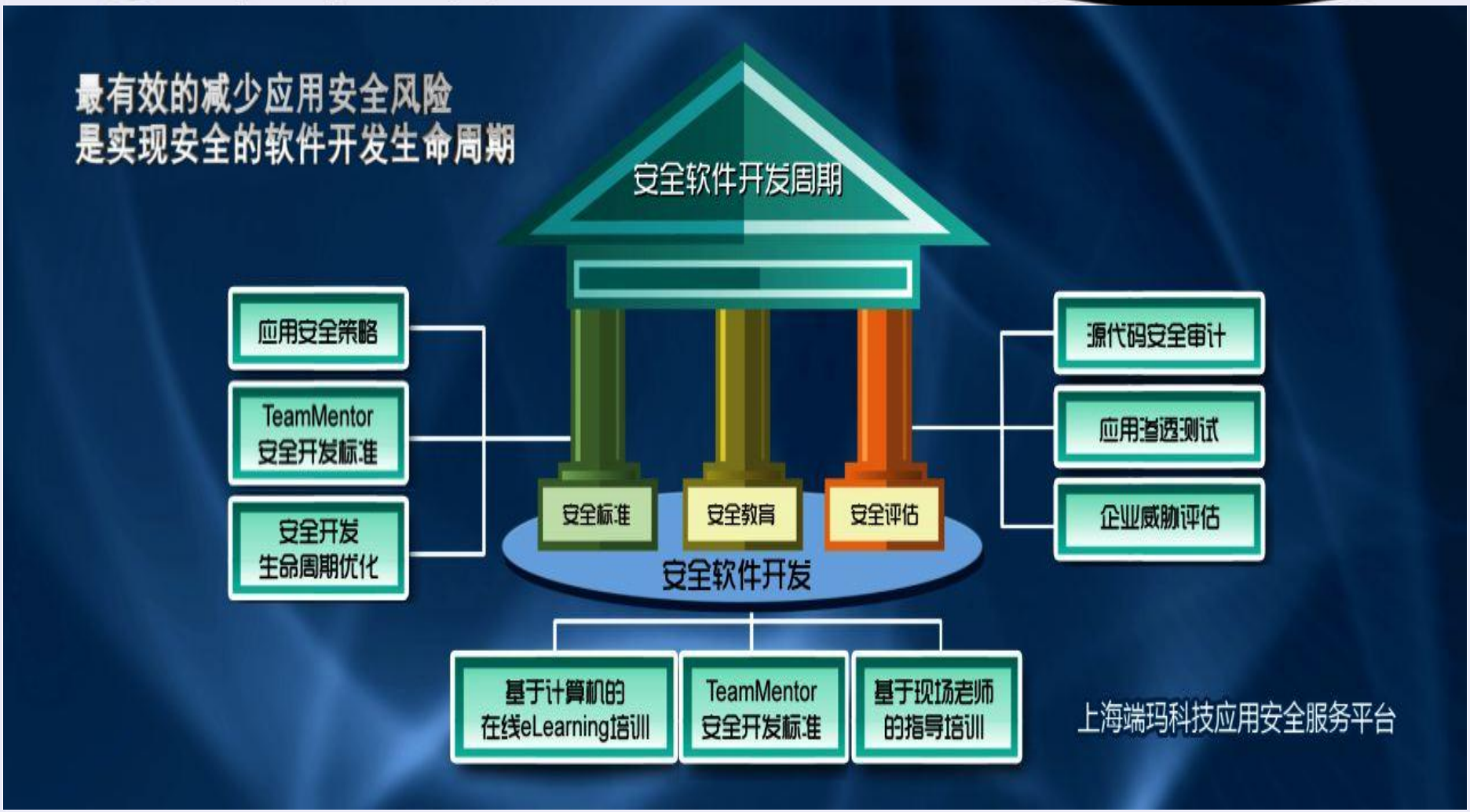
OWASP 中国
The Open Web Application Security Project

- 为使用Java、JSP、JavaSript、VBSript、C# 、ASP.net 、VB.Net、VB6、C/C++ 、ASP 、PHP, Ruby、Perl 、PL/SQL、Android、OWASP ESAPI、MISRA、和Objective-C (iOS) .(AppExchange platform)、API to 3rd party languages 等多种语言开发的软件开发企业和项目提供源代码安全漏洞和质量缺陷扫描和分析，并提供结果审计 、管理 和报表生成。

www.dumascacloud.com

应用安全服务平台框架及内容

OWASP 中国
The Open Web Application Security Project



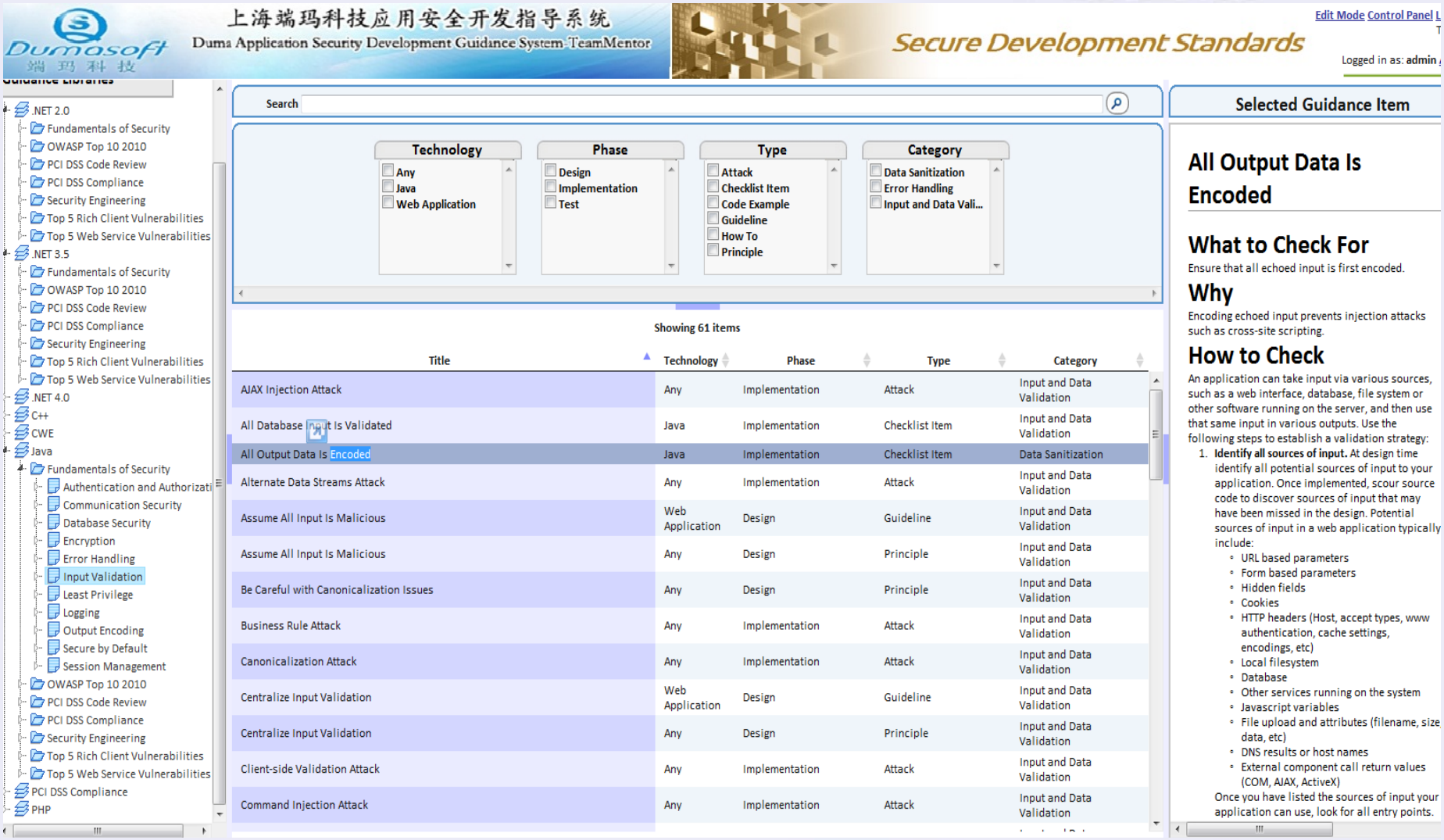
应用安全在线eLearning培训

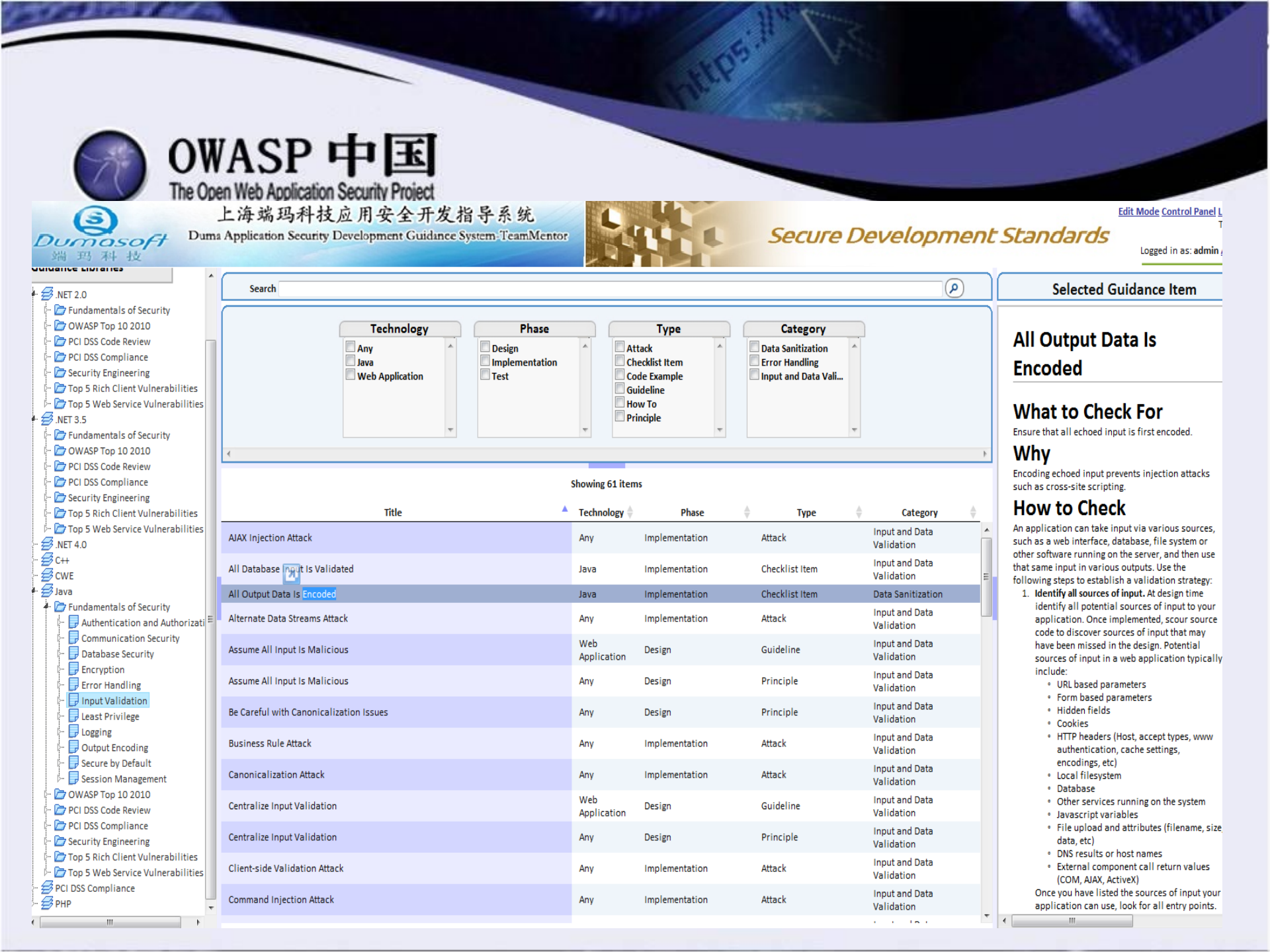
OWASP 中国
The Open Web Application Security Project

- 在线培训课程是基于世界顶级的应用安全讲师和咨询专家的丰富的安全实践和教学经验而组织，帮助开发组织快速了解和普及应用软件安全方面的知识、技能和成熟软件安全分析方法和最佳实践。以期他们在最短的时间里学习到最需要的软件安全知识，从而有效地应用到软件开发的生命周期里

应用安全开发标准指导系统

OWASP 中国
The Open Web Application Security Project




OWASP 中国
The Open Web Application Security Project
上海端玛科技应用安全开发指导系统
Duma Application Security Development Guidance System-TeamMentor
Dumasoft
端玛科技
Secure Development Standards
Edit Mode Control Panel
Logged in as: admin
.NET 2.0
Fundamentals of Security
OWASP Top 10 2010
PCI DSS Code Review
PCI DSS Compliance
Security Engineering
Top 5 Rich Client Vulnerabilities
Top 5 Web Service Vulnerabilities
.NET 3.5
Fundamentals of Security
OWASP Top 10 2010
PCI DSS Code Review
PCI DSS Compliance
Security Engineering
Top 5 Rich Client Vulnerabilities
Top 5 Web Service Vulnerabilities
.NET 4.0
C++
CWE
Java
Fundamentals of Security
Authentication and Authorizati
Communication Security
Database Security
Encryption
Error Handling
Input Validation
Least Privilege
Logging
Output Encoding
Secure by Default
Session Management
OWASP Top 10 2010
PCI DSS Code Review
PCI DSS Compliance
Security Engineering
Top 5 Rich Client Vulnerabilities
Top 5 Web Service Vulnerabilities
PCI DSS Compliance
PHP
Search
Technology
Any
Java
Web Application
Phase
Design
Implementation
Test
Type
Attack
Checklist Item
Code Example
Guideline
How To
Principle
Category
Data Sanitization
Error Handling
Input and Data Vali...
Showing 61 items
Title
Technology
Phase
Type
Category
AJAX Injection Attack
Any
Implementation
Attack
Input and Data Validation
All Database Input Is Validated
Java
Implementation
Checklist Item
Input and Data Validation
All Output Data Is Encoded
Java
Implementation
Checklist Item
Data Sanitization
Alternate Data Streams Attack
Any
Implementation
Attack
Input and Data Validation
Assume All Input Is Malicious
Web Application
Design
Guideline
Input and Data Validation
Assume All Input Is Malicious
Any
Design
Principle
Input and Data Validation
Be Careful with Canonicalization Issues
Any
Design
Principle
Input and Data Validation
Business Rule Attack
Any
Implementation
Attack
Input and Data Validation
Canonicalization Attack
Any
Implementation
Attack
Input and Data Validation
Centralize Input Validation
Web Application
Design
Guideline
Input and Data Validation
Centralize Input Validation
Any
Design
Principle
Input and Data Validation
Client-side Validation Attack
Any
Implementation
Attack
Input and Data Validation
Command Injection Attack
Any
Implementation
Attack
Input and Data Validation
Selected Guidance Item
All Output Data Is Encoded
What to Check For
Ensure that all echoed input is first encoded.
Why
Encoding echoed input prevents injection attacks such as cross-site scripting.
How to Check
An application can take input via various sources, such as a web interface, database, file system or other software running on the server, and then use that same input in various outputs. Use the following steps to establish a validation strategy:
1. Identify all sources of input. At design time identify all potential sources of input to your application. Once implemented, scour source code to discover sources of input that may have been missed in the design. Potential sources of input in a web application typically include:
◦ URL based parameters
◦ Form based parameters
◦ Hidden fields
◦ Cookies
◦ HTTP headers (Host, accept types, www authentication, cache settings, encodings, etc)
◦ Local filesystem
◦ Database
◦ Other services running on the system
◦ Javascript variables
◦ File upload and attributes (filename, size data, etc)
◦ DNS results or host names
◦ External component call return values (COM, AJAX, ActiveX)
Once you have listed the sources of input your application can use, look for all entry points.