

Web 系统测试

4.11 渗透测试—浏览器安全

➤SQL注入漏洞概述

●什么是SQL注入漏洞

- 通过把SQL命令插入到Web表单提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**

➤SQL注入方法

●通过字符串注入

●猜测：猜表名，猜列名，猜数据库名等等

●后台身份验证绕过漏洞

➤ 怎样测试SQL注入漏洞

- 猜字段、猜数据库名字、猜用户名、猜数据库版本
- 猜当前操作系统、猜数据库表名、获取用户名和密码

➤ 怎样防御SQL注入漏洞

- 使用参数化的过滤性语句
- 输入验证
- 错误消息处理
- 加密处理
- 存储过程来执行所有的查询
- 使用专业的漏洞扫描工具
- 确保数据库安全

➤ Burp Suite 中 Target 和 Spider 的使用

目录

- 浏览器安全概述
- 同源策略
- 浏览器沙箱
- 恶意网址拦截
- 高速发展的浏览器安全

➤什么是浏览器安全

- 浏览器端具备安全功能

➤为什么提浏览器安全

- 作为客户端，如果具备安全功能，就可以像安全软件一样对用户上网起到较好的保护作用
- 浏览器安全也成为浏览器厂商之间竞争的底牌，能够针对安全建立起技术门槛，以获得竞争优势

目录

➤ 浏览器安全概述

➤ 同源策略

➤ 浏览器沙箱

➤ 恶意网址拦截

➤ 高速发展的浏览器安全

➤什么是同源

- 两个页面地址中的协议，域名(或IP)，子域名，端口号一致，则表示同源

URL	是否同源	原因
http://store,xompany.com/dir2/orther.html	是	
http://store,xompany.com/dir/inner/another.html	是	
https://store,xompany.com/secure.html	不是	协议不同
http://store,xompany.com:81/dir/etc.html	不是	端口不同
http://news,xompany.com/dir/orther.html	不是	子域名不同

➤ 同源策略的意义

- 限制了来自不同源的“document”或脚本，对当前“document”读取或设置某些属性

➤ 举例：

- 如果没有同源策略，可能a.com的一段JS脚本，在b.com未曾加载此脚本时，也可以随意涂改b.com的页面（在浏览器的显示中）。为了不让浏览器的页面行为发生混乱，浏览器提出了“Origin”（源）这一概念来自不同Origin的对象无法互相干扰

➤为什么浏览器要使用同源策略

- 主要目的是为了安全，浏览器中JS的同源策略决定了，当浏览器认为来自不同源时，请求被拒绝
- 如果没有同源限制，在浏览器中的cookie等其他数据可以任意读取，不同域下的DOM任意操作，ajax任意请求其他网站的数据，包括隐私数据

➤注意：对于当前页面来说，页面内存放JS文件的域并不重要，重要的是加载JS页面所在的域是什么

- a.com通过以下代码

- <script src = <http://b.com/b.js></script>加载了b.com上的b.js，但是b.js是运行在a.com页面中的，因此对于当前打开的页面（a.com）来说，b.js的Origin就应该是a.com而非b.com

➤ 浏览器中哪些标签可以加载资源

- `<script>`、``、`<iframe>`、`<link>`等标签可以跨域，不受同源策略的限制
- 这些带“src”属性的标签每次加载时，实际上是浏览器发起了一次GET请求，浏览器限制了JS的权限，使其不能读、写返回的内容
- XMLHttpRequest可以访问来自同源对象的内容，不能跨域访问资源

➤对于浏览器来说，哪些内容会受到同源策略的限制

- DOM、Cookie、XMLHttpRequest

- 第三方插件也有自己的控制策略：Flash、Java Applet、Silverlight、Google Gears

- 举例如：Flash，主要通过目标网站提供的crossdomain.xml判断是否允许当前“源”的Flash跨域访问目标资源

```
<cross-domain-policy>  
<allow-access-from domain="*.qq.com"/>  
<allow-access-from domain="*.gtimg.com"/>  
</ cross-domain-policy >
```

只有来自*.qq.com和*.gtimg.com域的请求是被允许的。依靠这种方式，从Origin的层面上控制了Flash行为的安全性

➤攻击者是否可以使用其他方式控制Flash行为

- 攻击者可以通过上传crossdomain.xml文件控制Flash的行为，绕过同源策略
- Flash9及其之后的版本中，实现了MIME检查以确认crossdomain.xml是否合法，比如：查看服务器返回HTTP头的Content-Type是否是text/*、application/xml、application/xhtml+xml
- Flash还会检查crossdomain.xml是否在根目录下（使上传文件攻击失效）

➤浏览器有了同源策略就一定坚不可摧了吗？

- 不一定

- 例如IE8的CSS跨域漏洞

- www.a.com/test.html:

<body>

{}body{font-family:

aaaaaaaaaaaaaaaaaaaa

bbbbbbbbbbbbbbbbbb}

</body>

➤ www.b.com/test2.html:

<style>

@import url("http://ww.a.com/test.html");

</style>

<script>

setTimeout(function(){

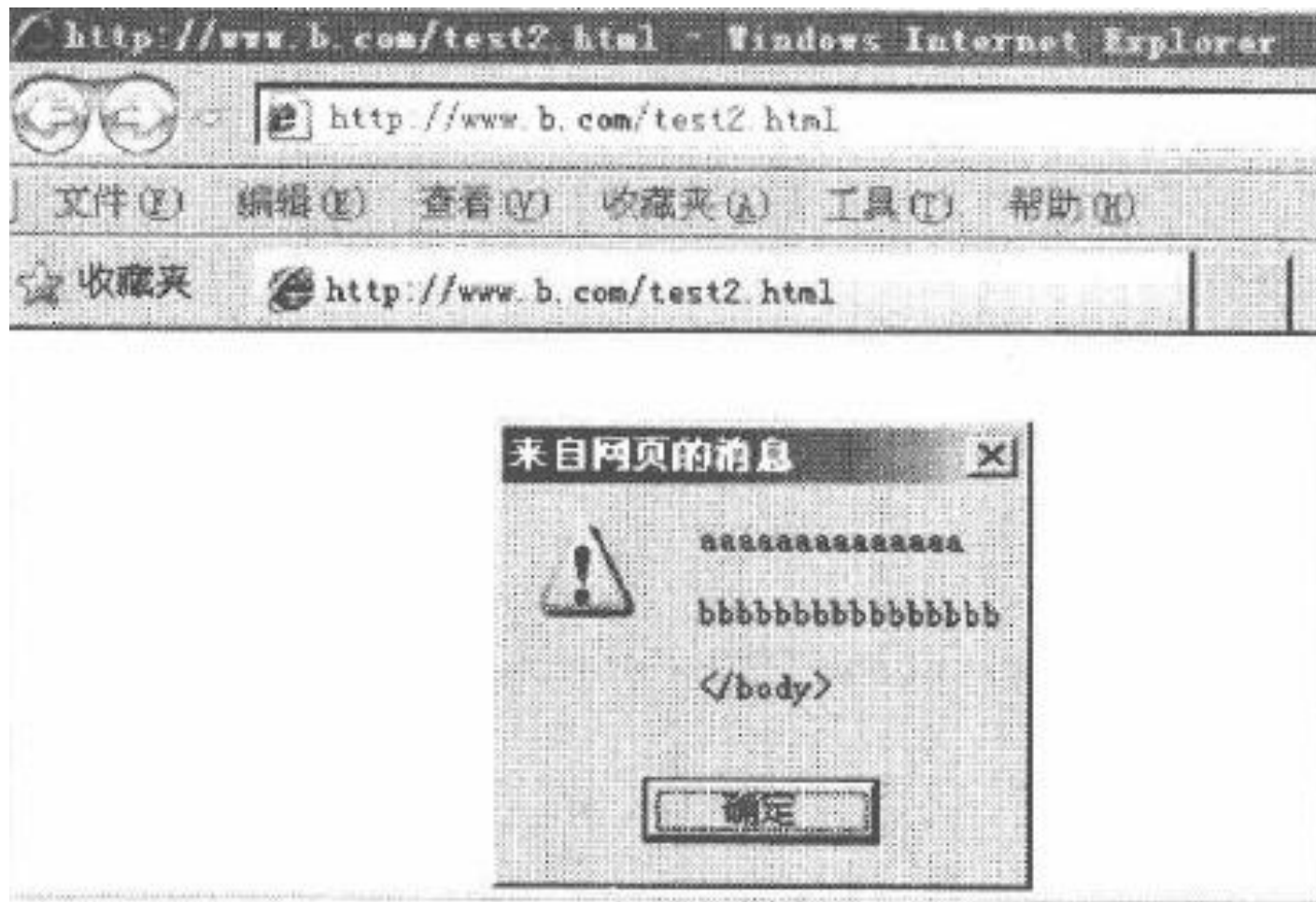
var t = document.body.currentStyle.fontFamily;

alert(t);

},2000);

</script>

- 在www.b.com/test2.html中通过@import加载了<http://www.a.com/test.html>的CSS文件，渲染进入当前页面DOM，同时通过document.body.currenStyle.fontFamily访问此内容
- 问题发生在IE的CSS解析过程中，IE将fontFamily后面的内容当做value，从而可以读取www.a.com/test.html的页面内容



➤ 这个漏洞能够跨域读取
页面内容，因此绕过了
同源策略，成为一个跨
域漏洞

➤同源策略总结：

- 浏览器的同源策略是浏览器安全的基础，理解同源策略对于客户端脚本攻击有着重要意义。同源策略一旦出现漏洞被绕过，也将带来非常严重的后果，很多基于同源策略制定的安全方案都将失去效果

目录

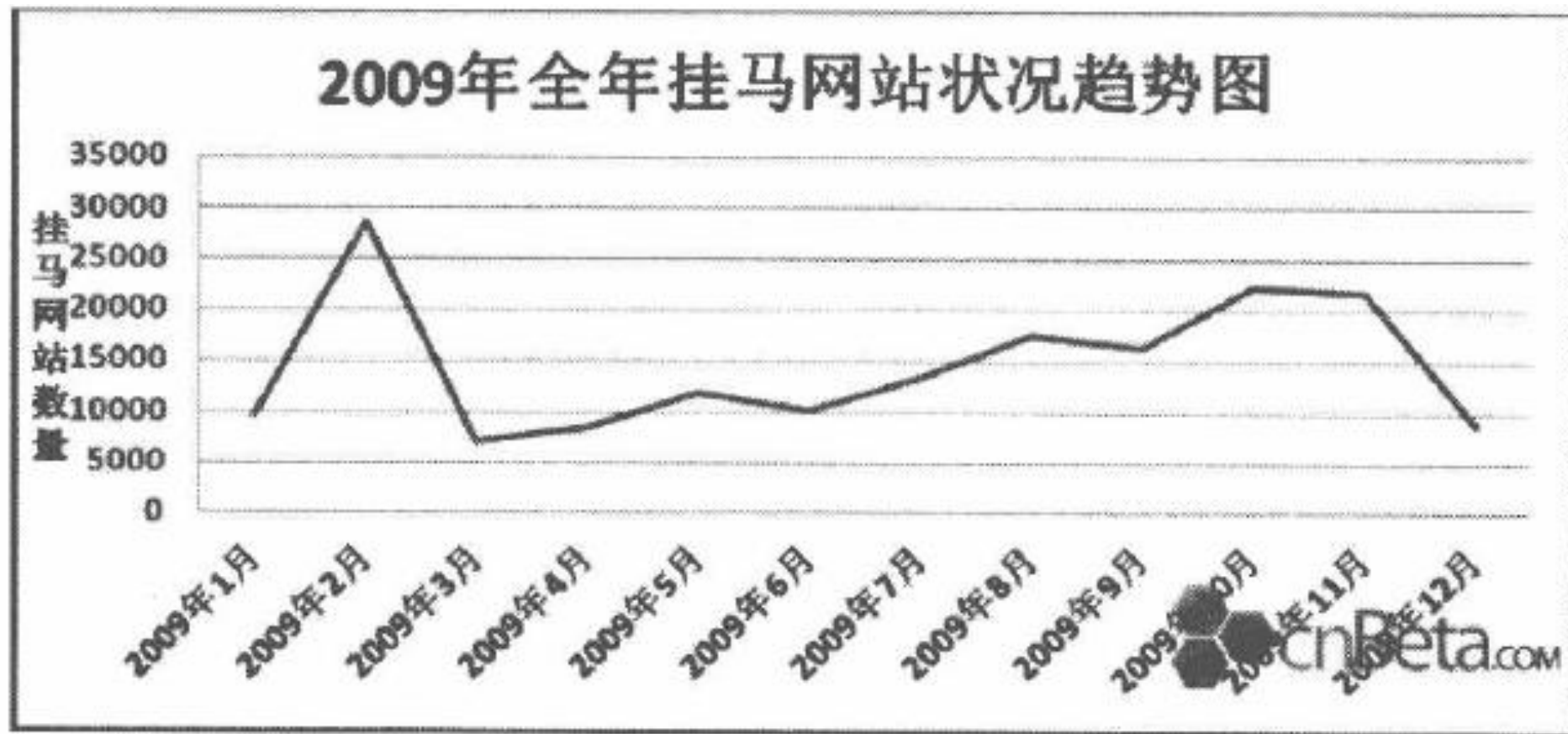
➤ 浏览器安全概述

➤ 同源策略

➤ 浏览器沙箱

➤ 恶意网址拦截

➤ 高速发展的浏览器安全



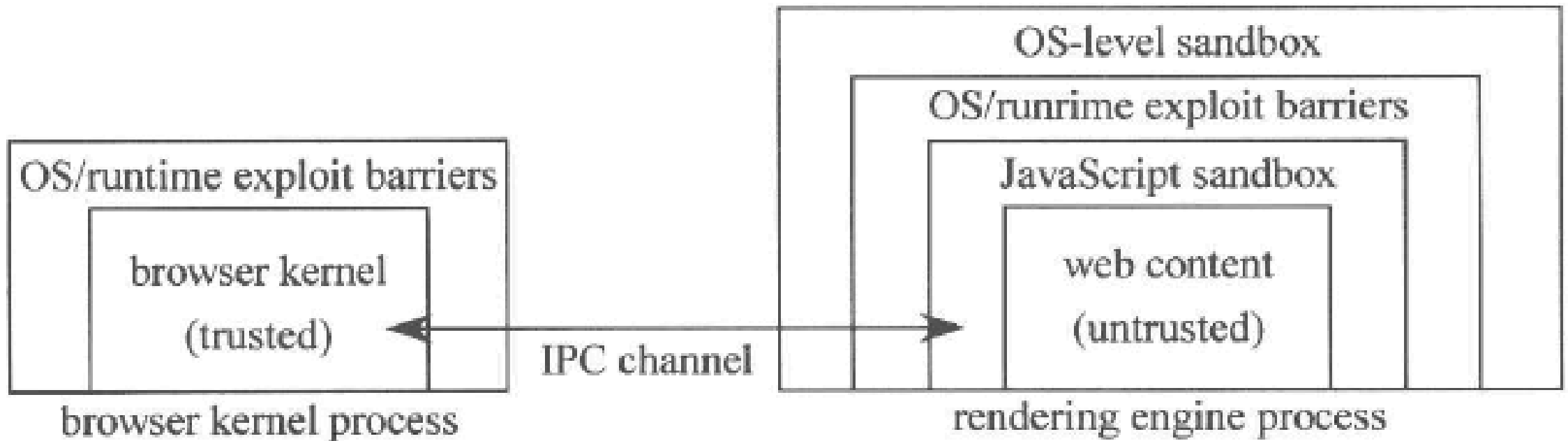
2009 年全年挂马网站状况趋势图

- 挂马：在网页中插入一段恶意代码，利用**浏览器漏洞**执行任意代码的攻击方式，被称为“挂马”
- “挂马”是浏览器需要面对的一个主要威胁，近年来，独立于杀毒软件之外，浏览器厂商根据挂马的特点研究出了一些对抗挂马的技术

➤ 典型对抗挂马的技术

● 多进程架构

- 将浏览器的各个功能模块分开，各个浏览器实例分开，当一个进程崩溃时，不会影响到其他的的进程
- Google Chrome主要进程有：浏览器进程、渲染进程、插件进程、扩展进程等
- 插件进程如flash、java、pdf等于浏览器进程严格隔离，因此不会互相影响



Google Chrome 的架构

- 渲染引擎由Sandbox隔离，网页代码要与浏览器内核进程通信、与操作系统通信都需要通过IPC channel，在其中进行一些安全检查

➤沙箱（Sandbox）：泛指“资源隔离类模块”的代名词

➤设计沙箱的目的：

- 让不可信任的代码运行在一定的环境中，限制不可信任的代码访问
隔离去之外的资源
- 如果一定要跨越沙箱边界产生数据交换，则只能通过指定的数据通道，比如经过封装的API来完成，在这些API中会严格检查请求的
合法性

➤多进程架构浏览器有什么好处？

- 相对于单进程浏览器，在发生崩溃时，多进程浏览器只会崩溃当前的Tab页，而单进程浏览器则会崩溃整个浏览器进程，对于用户体验是很大提升

➤多进程和沙箱保护就一定安全吗？

- 不一定
- 如第三方插件（Flash、Java、PDF、.NetFramework等）出现安全漏洞，第三方插件近年来也成为浏览器攻击的热点

目录

➤ 浏览器安全概述

➤ 同源策略

➤ 浏览器沙箱

➤ 恶意网址拦截

➤ 高速发展的浏览器安全

- 很多时候“挂马”攻击在实施时会在一个正常的网页中通过<script>或者<iframe>等标签加载一个恶意网址
- 除了加载恶意网址外，浏览器端还有没有别的威胁
 - 钓鱼网站
 - 诈骗网站
- 为了保护用户安全，浏览器厂商纷纷推出各自的拦截恶意网址功能
 - 目前各个浏览器的拦截恶意网址的功能都是基于“黑名单”的

➤ 拦截恶意网址的工作原理：

- 浏览器周期性地从服务器端获取一份最新的恶意网址黑名单，如果用户上网访问的网址存在于此黑名单中，浏览器就会弹出一个警告页面



➤ 常见恶意网址类型

● 挂马网站

- 通常包含恶意脚本如JS或Flash，通过利用浏览器的漏洞（插件或控件漏洞）执行shellcode，在用户电脑中植入木马

● 钓鱼网站


- 通过模仿知名网站的相似页面来欺骗用户

➤ 怎样识别这两种网站

- 目前只是以推送恶意网址黑名单为主，浏览器收到黑名单后，对用户访问的黑名单进行拦截
- 与专业安全厂商合作，由安全厂商或机构提供恶意网址黑名单

- Google和微软有自建安全团队做恶意网址识别工作，用以提供浏览器所使用的黑名单
- PhishTank是互联网上免费提供恶意网址黑名单的组织之一，它的黑名单由世界各地的志愿者提供，且更新频繁

恶意网址拦截

**PhishTank**[®] Out of the Net, into the Tank.

username [Sign in](#)

[Register](#) | [Forgot Password](#)

[Home](#) [Add A Phish](#) [Verify A Phish](#) [Phish Search](#) [Stats](#) [FAQ](#) [Developers](#) [Mailing Lists](#) [My Account](#)

Join the fight against phishing

[Submit](#) suspected phishes. [Track](#) the status of your submissions.
[Verify](#) other users' submissions. [Develop](#) software with our free API.

Found a phishing site? Get started now — see if it's in the Tank:

[Is it a phish?](#)

Recent Submissions

You can help! [Sign in](#) or [register](#) (free! fast!) to verify these suspected phishes.

ID	URL	Submitted by
6256722	https://motiv8radiofm.com/groups/123movies-motherl...	evannto
6256721	http://kontrola.newfreesongs.com/usec/aper.php	secopspishing
6256719	http://oplatnosci24.com/pkopay/	CERTPKOBP
6256718	https://montrealnewyearsevetickets.com/usec/aper.p...	secopspishing
6256717	https://login.bblockchaim.com	chdh
6256708	https://o-9.top/jp/	knack
6256707	https://o-9.top/	knack

What is phishing?

Phishing is a fraudulent attempt, usually made through email, to steal your personal information.
[Learn more...](#)

What is PhishTank?

PhishTank is a collaborative clearing house for data and information about phishing on the Internet. Also, PhishTank provides an open API for developers and researchers to integrate anti-phishing data into their applications at no charge.
[Read the FAQ...](#)

➤除了拦截网址黑名单功能外，主流浏览器都支持SSL证书，以增强对安全网站的识别

➤证书

- 提供了一种在网上进行身份验证的方法，是用来标志和证明网络通信双方身份的数字信息文件
- 概念类似日常生活中的司机驾照或身份证
- 数字签名主要用于发送安全电子邮件、访问安全站点、网上招标与投标、网上签约、网上订购、网上公文安全传送、网上办公、网上缴费、网上缴税以及网上购物等安全的网上电子交易活动

目录

- 浏览器安全概述
- 同源策略
- 浏览器沙箱
- 恶意网址拦截
- 高速发展的浏览器安全

➤ 微软率先在IE8中推出了XSS Filter功能，用以**对抗反射型XSS**

- 一直以来，业界认为XSS是服务器端应用的漏洞，应该由服务器端应用在代码中修补，而微软率先推出这一功能，使得IE8在安全领域极具特色
- 当用户访问URL中包含了XSS攻击的脚本时，IE就会修改其中的关键字符，使得攻击无法成功完成，并对用户弹出提示框，这些规则可以捕获URL中XSS攻击

➤Firefox4 推出了 Content Security Policy（内容安全政策）

- 服务器添加 Content-Security-Policy 响应头来指定规则
- HTML 中添加 <meta> 标签来指定 Content-Security-Policy 规则
- 如：
 - X—Content—Security—Policy:allow ‘self’ ; *.mydomain.com
 - 浏览器信任来自mydomain.com 及其子域下的内容
 - X—Content—Security—Policy:allow ‘self’; img—src *;media-src medial.com;script-src userscripts.example.com
 - 浏览器除了信任自身来源外，还可以加载任意域的图片，来自medial.com的媒体文件，以及userscripts.example.com的脚本，其他一律拒绝

- CSP设计理念是出色的，但CSP的规则配置较为复杂，在页面较多的情况下，很难一个个配置起来，且后期维护成本巨大，这些原因导致CSP未能得到很好的推广

- 浏览器安全概述
- 同源策略
- 浏览器沙箱
- 恶意网址拦截
- 高速发展的浏览器安全

Question