第二章 客户端脚本安全

同源策略

同源策略（Same Origin Policy）是一种约定，是浏览器最核心最基本的安全功能

同源策略：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

同源定义：域名，协议，端口相同

URL	Outcome	Reason
http:// store.company.com/dir2/other.html	Success	
http:// store.company.com/dir/other2.html	Success	
https:// store.company.com/dir2/other.html	Failure	Different protocol
http:// store.company.com:81/dir2/etc.html	Failure	Different port
http:// news.company.com/dir2/other.html	Failure	Different host

同源策略限制：DOM、Cookie、XMLHttpRequest以及第三方插件（Flash、Java Applet等）

- Flash为例：Flash为网站提供crossdomain.xml文件判断是否允许当前“源”的Flash跨域访问目标资源
  - 其他任何域加载Flash，如果对www.qq.com发起访问请求，Flash会先检查www.qq.com上此策略文件是否存在
  - crossdomain.xml中只有来自 .qq.com 和 .gtimg.com 域的请求是被允许的

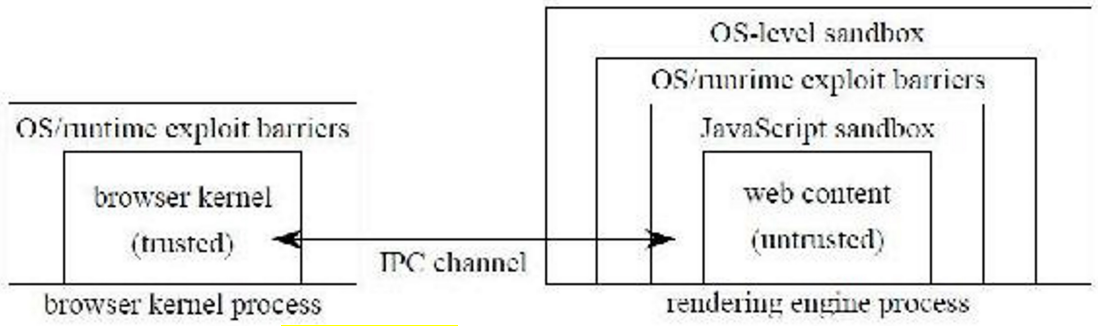
<script><iframe>等有src属性的标签可以加载恶意地址

浏览器沙盒

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

浏览器多进程架构：浏览器的各个功能模块分开，各个浏览器实例分开，一个进程崩溃不会影响到其他进程

- Google Chrome是第一个采取多进程架构的浏览器
  - 主要进程分为：浏览器进程、渲染进程、插件进程、扩展进程。

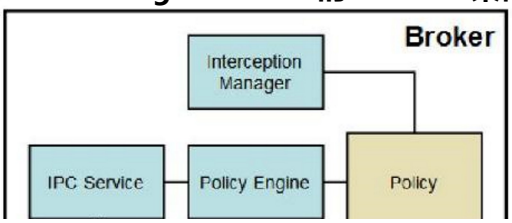


  - 渲染引擎由Sandbox隔离，网页代码要与浏览器内核通信、与操作系统通信都要通过IPCchannel，其中会进行安全检查

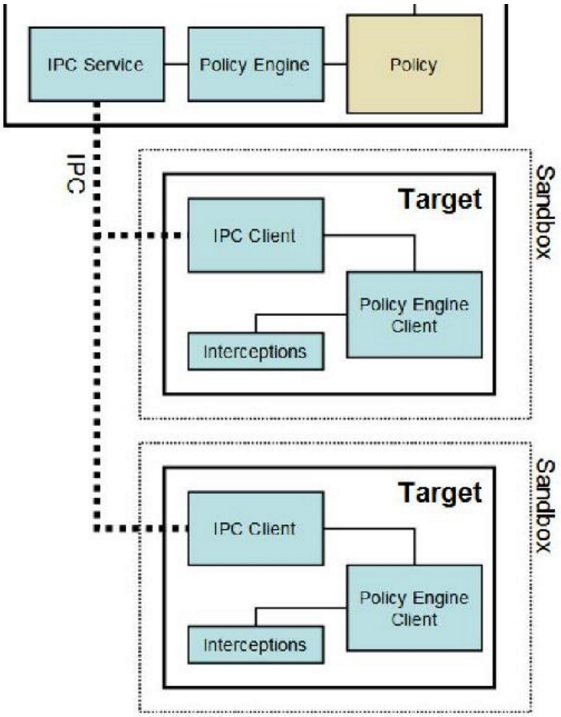
Sandbox：资源隔离类模块，设计目的是为了让不可信任的代码运行在一定的环境中，限制不可信任的代码访问隔离区之外的内容

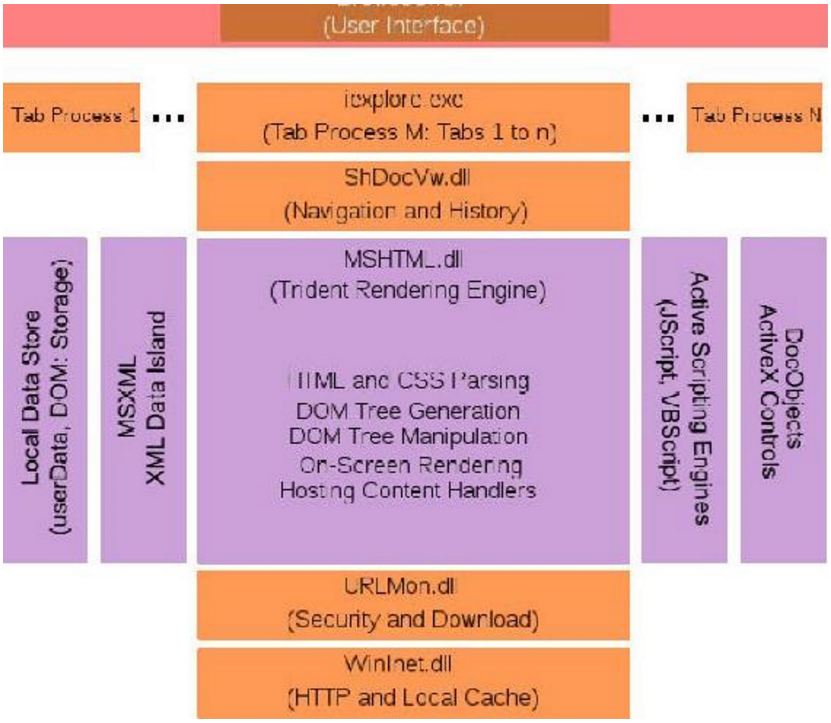
多进程架构好处：崩溃时只奔溃当前Tab页，提升用户体验

Google Chrome的Sandbox架构



IE 8的架构





恶意网站拦截

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，如果用户上网时访问的网址在此黑名单中就会弹出警告页面

恶意网址分类：

- 挂马网站：包含恶意脚本如JS，Flash，通过浏览器漏洞执行shellcode，在用户电脑植入木马
- 钓鱼网站：模仿知名网站的相似页面欺骗用户

PhishTank：互联网上免费提供恶意网址黑名单的组织之一，黑名单由世界各地志愿者提供，更新频繁

SafeBrowsing API：Googel的恶意网址库，任何组织或个人都可以在产品中接入

EV SSL证书(Extended ValidationSSL Certificate): 全球数字证书颁发机构与浏览器厂商一起打造的证书

高速发展的浏览器安全

微软率先在IE8中推出XSS Filter功能对抗反射型XSS

- 当用户访问的URL中包含XSS攻击脚本时，IE会修改其中的关键字符使其攻击无法完成

火狐随后在Firefox4中推出了Content Security Policy（CSP）

- 服务器返回一个HTTP头，在其中描述页面应该遵守的安全策略
- 因为XSS攻击在没有第三方插件帮助的情况下，无法控制HTTP头，所以这项措施可行
- 插入一个HTTP返回头：`X-Content-Security-Policy: policy`
- policy的描述极其灵活比如：`X-Content-Security-Policy: allow 'self' *.mydomain.com`
  //浏览器将信任来自mydomain.com及其子域下的内容
  `X-Content-Security-Policy: allow 'self'; img- src *; media-src media1.com; script-src userscripts.example.com`
  //浏览器除了信任自身的来源外，还可以加载任意域的图片、来自media1.com的媒体文件，以及userscripts.example.com的脚本，其他的则一律拒绝
- CSP设计理念出色，但是配置复杂，很难一个个配置起来，维护成本大，未能很好推广

浏览器的用户体验越来越好，但是有安全隐患

- IE和Chrome可以解析“www.google.com\abc”，Firefox不行
- Firefox能解析“[http://www.cnn.com]” “[http://]www.cnn.com” “[http://www].cnn.com”

浏览器插件越来越丰富，但是扩展和插件的权限高于页面JS，可以惊醒跨域网络请求，所以有安全隐患