Web安全技术

Web Security

1.4 HTTP与Cookie

刘潮歌

liuchaoge@iie.ac.cn

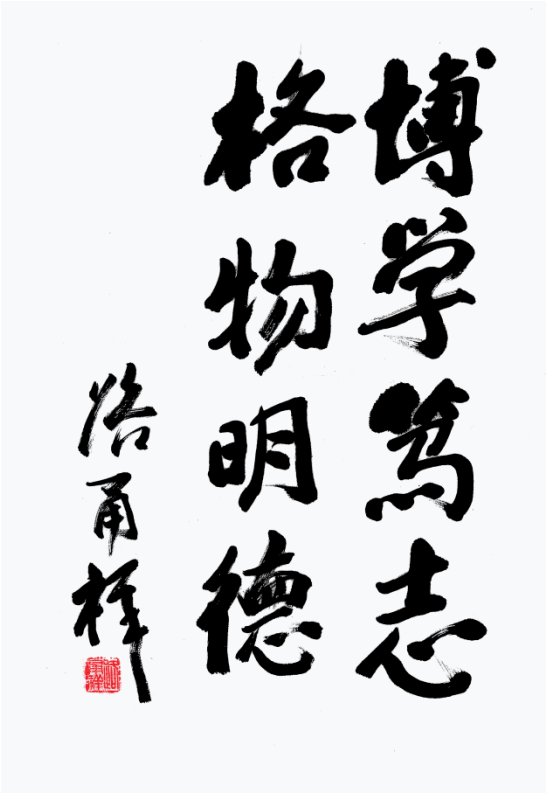
中科院信工所 第六研究室

一章一问

- HTTP协议和Cookie有哪些安全问题，现在有哪些解决办法？

本章大纲

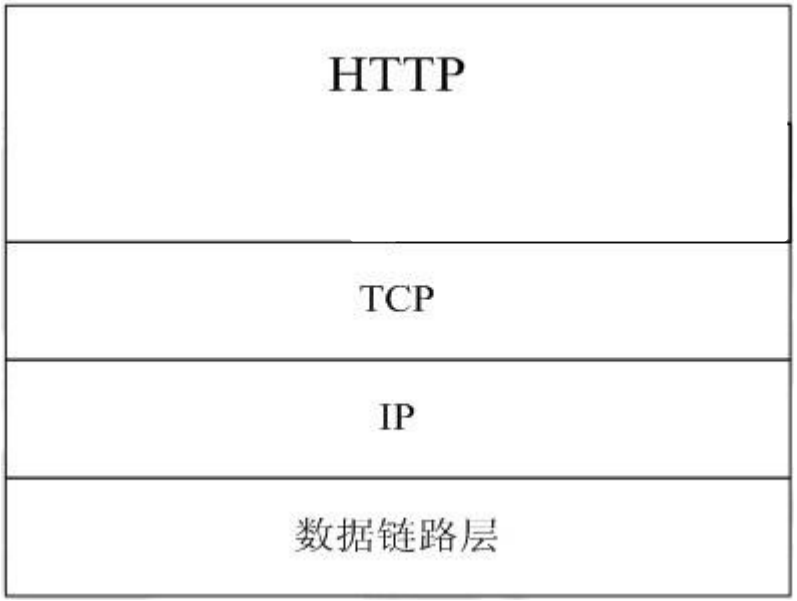
- **HTTP及安全问题**
  - 基础知识回顾
  - HTTP安全问题
  - HTTPS协议
- **Cookie及安全问题**
  - 基础知识
  - 安全策略
  - 安全问题
  - 顶会论文



基础知识回顾

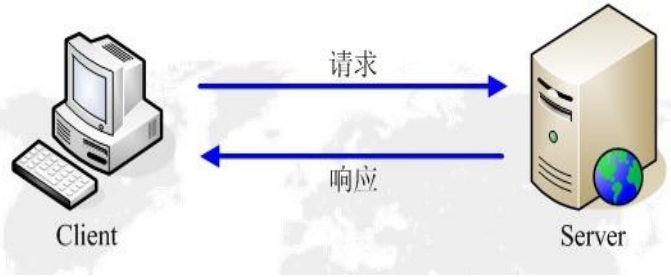
- **历史背景**
- HTTP（Hyper Text Transfer Protocol），超文本传输协议，应用层协议
- Tim Berners-Lee及其团队在1989年发明了最原始的HTTP协议：仅有一种GET请求方式，响应的内容也仅包含一个HTML文件
- 1991年将该HTTP定义为HTTP 0.9
- 1996年发布RFC1945，将HTTP协议进行拓展，形成新标准HTTP 1.0
- 1997年发布RFC2068，定义了新的标准HTTP 1.1
- 2007年发布RFC2616，修正HTTP1.1标准

基础知识回顾



- 工作流程
- HTTP是位于TCP层上的应用层协议
- 基于<请求-响应>工作模式
- 典型流程：

1. 客户端与运行HTTP服务器的主机建立TCP连接（三次握手）；
2. 客户端通过与TCP连接发出HTTP请求消息（GET/POST…）；
3. 服务器接受到请求后，解析并处理请求，然后发出HTTP响应消息；
4. 重复步骤3~4（HTTP 1.1）；
5. 关闭TCP连接（四次挥手）。



基础知识回顾

口 请求报文

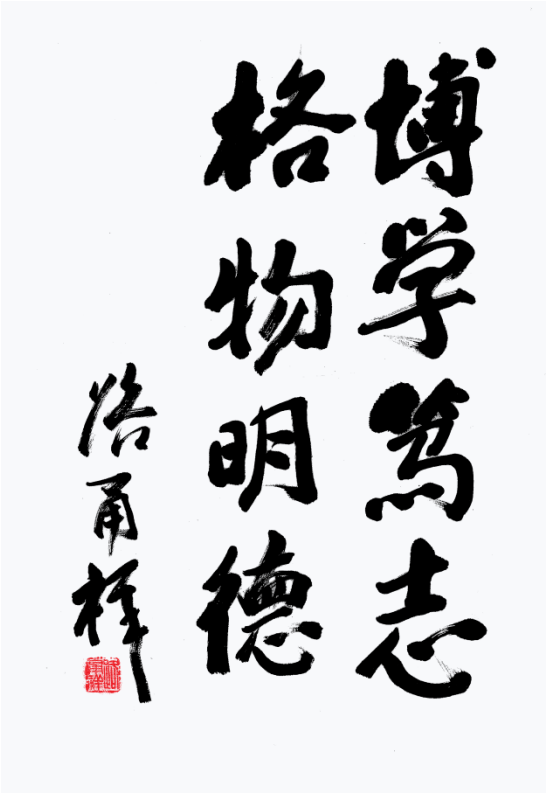
```
GET /jggk_101117/skjj/201501/t20150127_4305430.html HTTP/1.1
Host: www.iie.cas.cn
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.116 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0.8
Cookie: _gscu_1321407670=7442817828th0p10; _gscs_1321407670=74428178qfwzh610|pv:3

HTTP/1.1 200 OK
Date: Wed, 21 Sep 2016 03:23:58 GMT
Server: Apache/2.4.23 (Unix)
Accept-Ranges: bytes
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: text/html

873
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<META http-equiv=x-ua-compatible content="IE=7, IE=9">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>.....----.....</title>
<meta name="keywords"
content="..........
......"/>
<meta name="description"
```

本章大纲

- **HTTP及安全问题**
  - 基础知识回顾
  - HTTP安全问题
  - HTTPS协议
- **Cookie及安全问题**
  - 基础知识
  - 安全策略
  - 安全问题
  - 顶会论文



HTTP安全问题

- **数据明文传输**
- HTTP协议是明文传输协议，消息内容不经过加密处理
- 如果攻击者控制受害者传输网络，即可轻易嗅探或篡改传输内容

- **身份认证缺乏校验**
- 客户端与服务器通讯过程没有身份认证环节
- 客户端无法确认通信对方是正确的服务器

HTTP安全问题

- **数据明文传输**
- 网络嗅探与监听

- **身份认证缺乏校验**
- 中间人攻击

HTTP安全问题

- 网络嗅探与监听（Network Sniffer， Network Listening）
- 可以捕获网络报文
- Wireshark：经典的抓包工具

```
GET /jggk_101117/skjj/201501/t20150127_4305430.html HTTP/1.1
Host: www.iie.cas.cn
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.116 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0.8
Cookie: _gscu_1321407670=7442817828th0p10; _gscs_1321407670=74428178qfwzh610|pv:3

HTTP/1.1 200 OK
Date: Wed, 21 Sep 2016 03:23:58 GMT
Server: Apache/2.4.23 (Unix)
Accept-Ranges: bytes
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: text/html

873
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<META http-equiv=x-ua-compatible content="IE=7, IE=9">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>.....----.....</title>
<meta name="keywords"
content="..........
....."/>
<meta name="description"
```

传输信息
一览无遗

HTTP安全问题

```
POST /member.php?mod=logging&action=login&loginsubmit=yes&infloat=yes&lssubmit=yes&inajax=1 HTTP/1.1
Host: www.discuz.net
Connection: keep-alive
Content-Length: 82
Cache-Control: max-age=0
Origin: http://www.discuz.net
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.116 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://www.discuz.net/forum-3913-1.html
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.8
Cookie: t7asq_4ad6_saltkey=uKklDAxI; t7asq_4ad6_lastvisit=1474424790;
t7asq_4ad6_st_t=0%7C1474428390%7Ce34820e74c71f8dc9eaf2f6ee23ba897; t7asq_4ad6_forum_lastvisit=D_3913_1474428390;
t7asq_4ad6_sendmail=1; t7asq_4ad6_lastact=1474428391%09plugin.php%09; pgv_pvi=6450992444; pgv_info=ssi=s1256806880

fastloginfield=username&username=abc&password=123456&quickforward=yes&handlekey=lsHTTP/1.1 200 OK
Server: nginx
Date: Wed, 21 Sep 2016 03:26:46 GMT
Content-Type: text/xml; charset=gbk
Transfer-Encoding: chunked
Connection: keep-alive
Cache-Control: no-store, private, post-check=0, pre-check=0, max-age=0
Set-Cookie: t7asq_4ad6_lastact=1474428406%09member.php%09logging; expires=Thu, 22-Sep-2016 03:26:46 GMT; path=/; domain=.discuz.net
Expires: -1
Cache-Control: no-store, private, post-check=0, pre-check=0, max-age=0
Pragma: no-cache
comsenz_tag: 28d312ca960a0c75e4bd3707a49d8e90
Content-Encoding: gzip
```

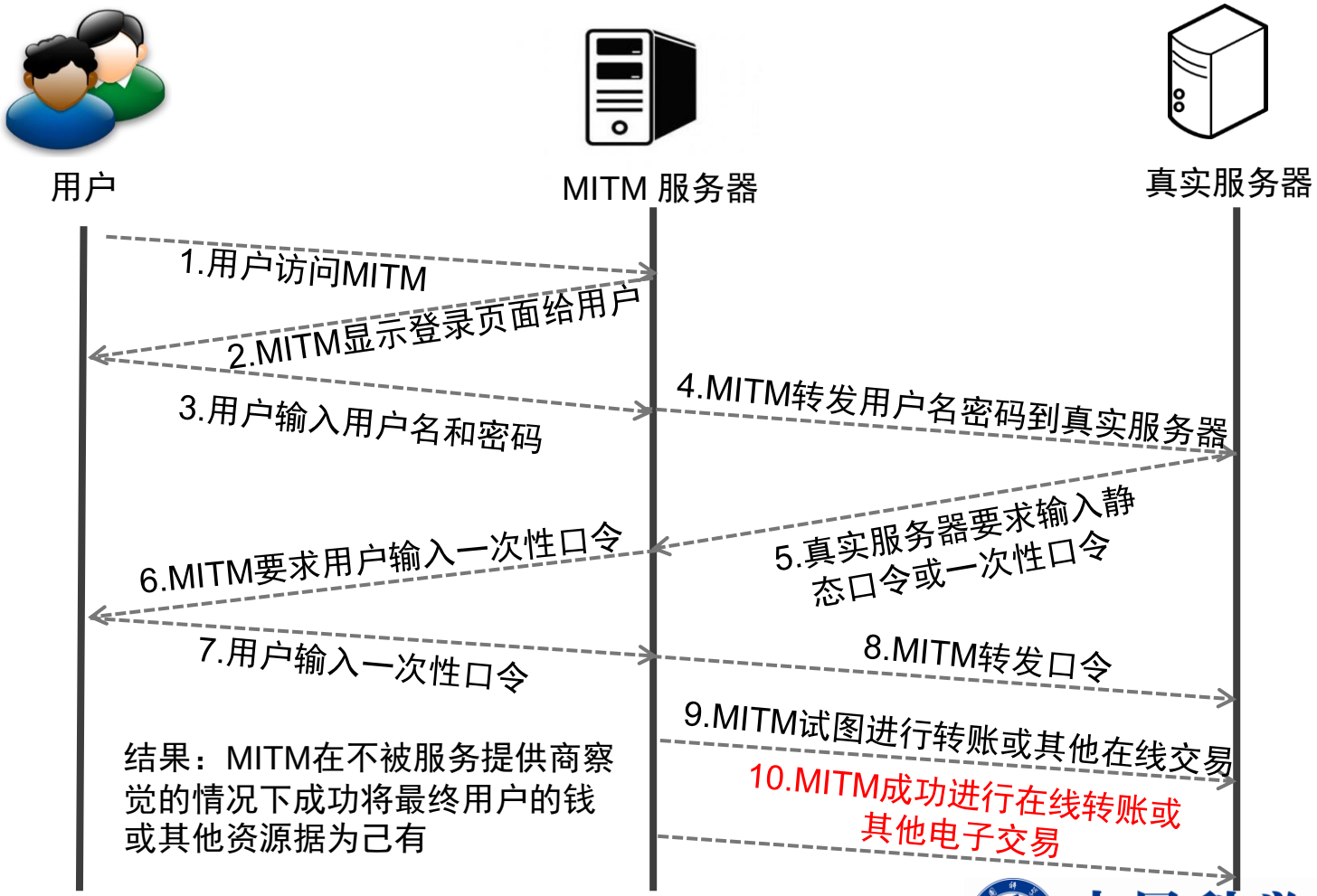
HTTP安全问题

- **中间人攻击（MITM，Man-in-the-middle attack）**
- 攻击者与通讯两端分别建立独立的联系，并交换其所收到的数据
- 通讯的两端认为他们正在直接与对方对话，但事际上整个会话都被攻击者完全控制
- 攻击成功的前提条件：攻击者能将自己伪装成全部参与会话的终端，并且不被识破

- 中间人攻击的重要原因是通信各端缺乏有效的认证

HTTP安全问题

- 中间人攻击（MITM，Man-in-the-middle attack）

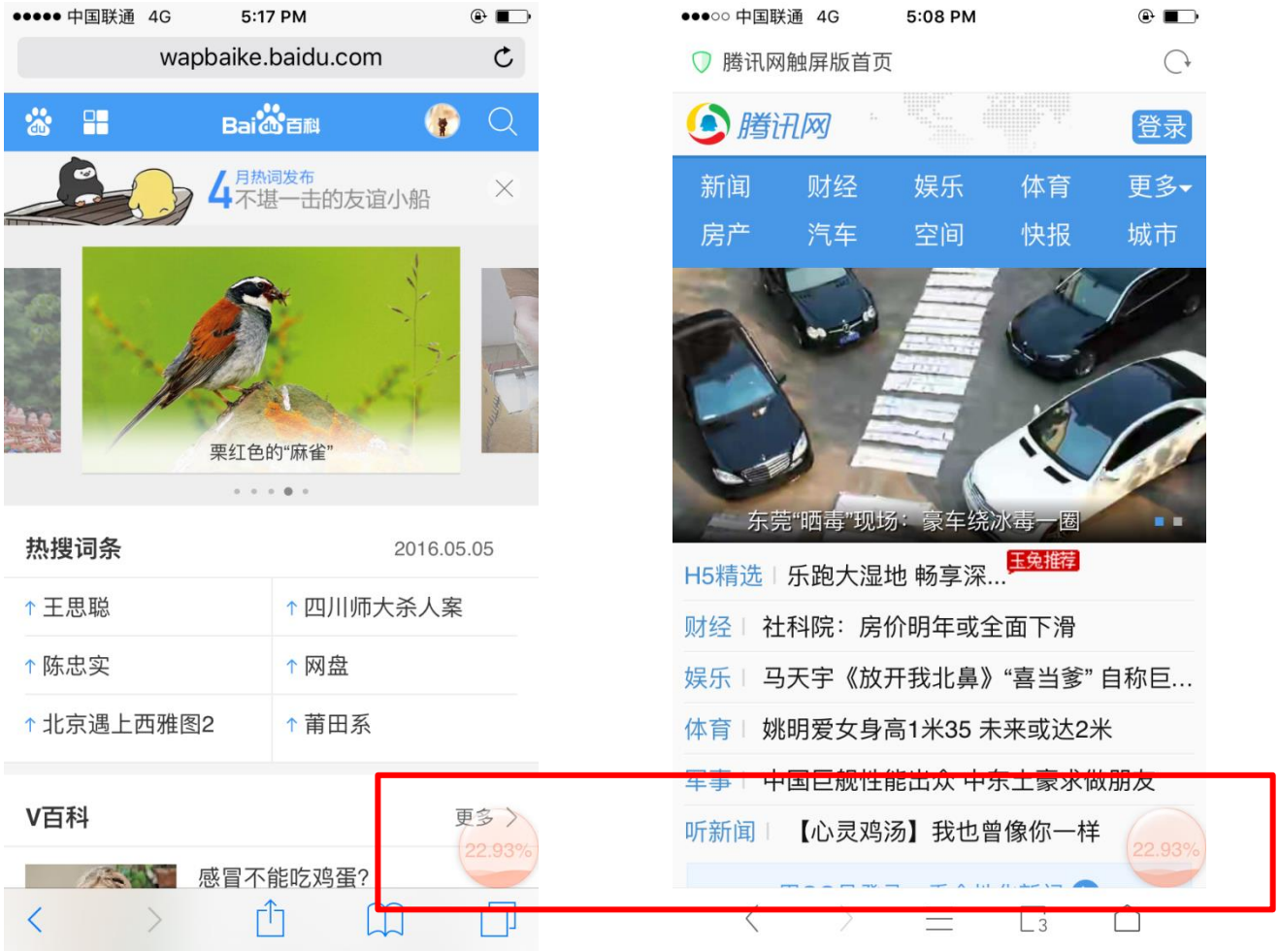


HTTP安全问题

- **中间人攻击**
- 流量篡改
- 修改返回数据内容
- 在返回流量中加入新的内容，如JavaScript脚本，iframe页面等

HTTP安全问题

- 中间人攻击
- 篡改流量



中国科学院大学 University of Chinese Academy of Sciences

HTTP安全问题

- **中间人攻击**
- 流量劫持
- 劫持网关设备、路由设备或者DNS，导致用户访问受非正确网站
- 恶意结果：网站不能访问，或访问至假网站
- 并非都是恶意行为

HTTP安全问题

- 中间人攻击
- 流量劫持

HTTP安全问题

- 中间人攻击
- 流量劫持

HTTP安全问题

- **中间人攻击**
- 典型工具——mitmproxy/mitmdump

- MITM神器：mitmproxy/mitmdump（交互版/非交互版）
- 开源托管在github，使用python开发，支持跨平台
- 能够捕获、分析、修改HTTP/HTTPS数据包

HTTP安全问题

- mitmproxy
- 访问example.com获得请求

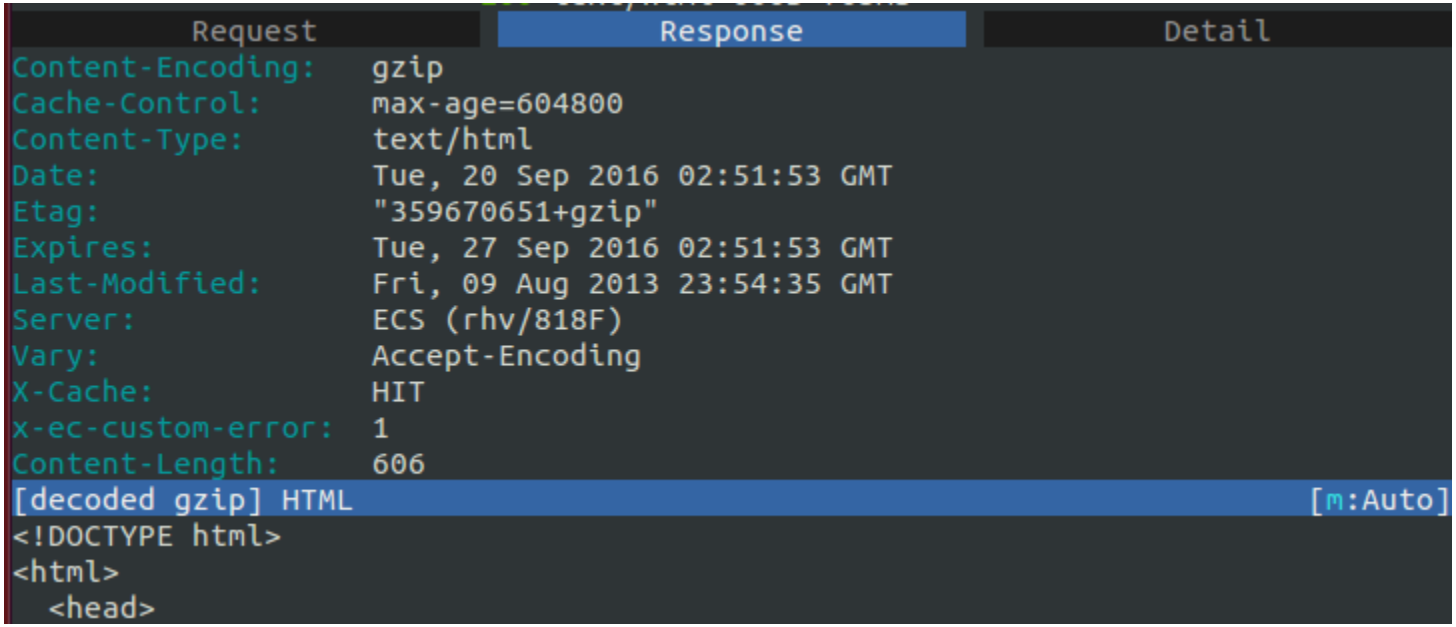
```
>> GET http://example.com/
     ←200 text/html 606B 765ms
   GET http://example.com/favicon.ico
     ←404 text/html 606B 300ms
   GET http://example.com/favicon.ico
     ←404 text/html 606B 309ms
   POST http://ocsp.digicert.com/
     ←200 application/ocsp-response 471B 357ms
[1/4]                                                    ?:help [*:8080]
```

- 查看具体数据包（请求包）

```
Request                 Response                Detail
Host:                   example.com
User-Agent:             Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:48.0)
                        Gecko/20100101 Firefox/48.0
Accept:                 text/html,application/xhtml+xml,application/xml;q=
                        0.9,*/*;q=0.8
Accept-Language:        en-US,en;q=0.5
Accept-Encoding:        gzip, deflate
Connection:             keep-alive
Upgrade-Insecure-Requests: 1
```

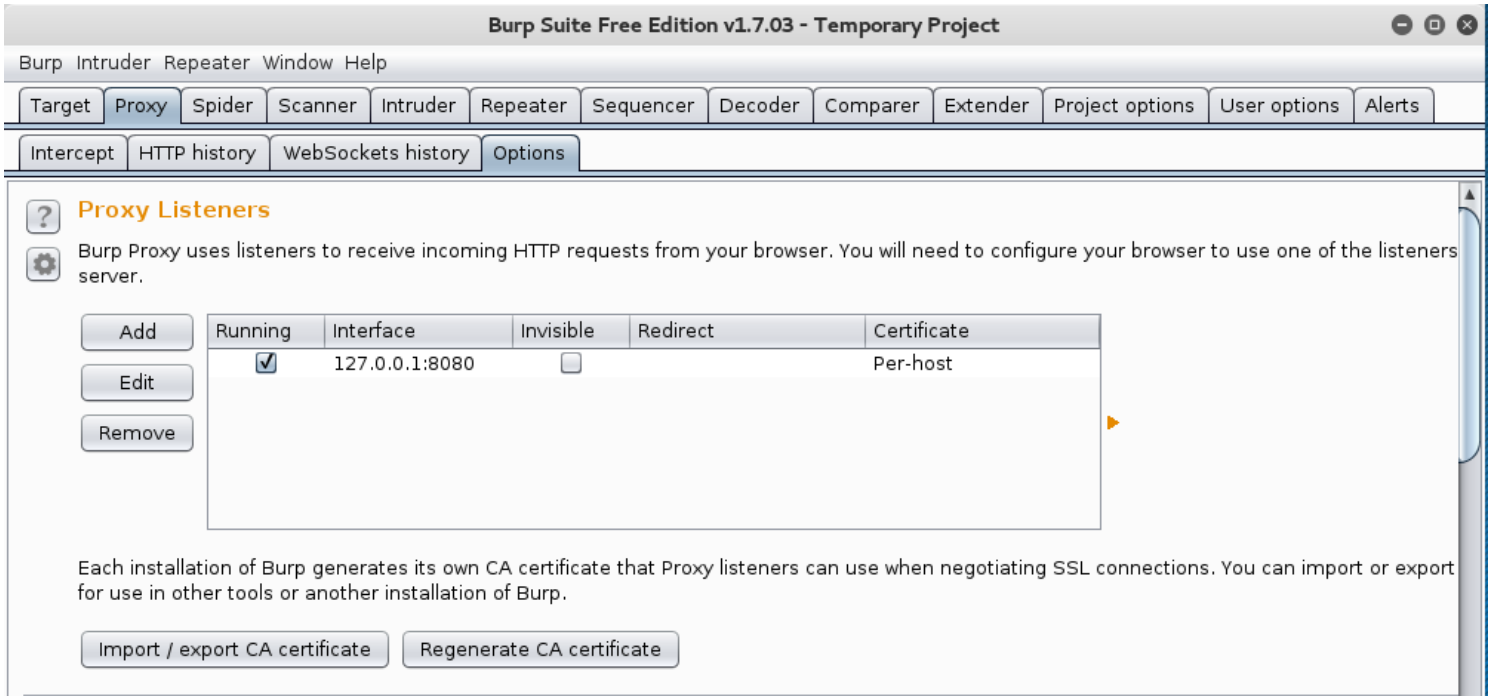
HTTP安全问题

- mitmproxy
- 响应包



HTTP安全问题

- 中间人攻击
- 典型工具——Burp Suite



HTTP安全问题

- **Burp Suite**
- 功能介绍

名称	功能说明
Proxy	是一个拦截HTTP/S的代理服务器，作为一个在浏览器和目标应用程序之间的中间人，允许你拦截，查看，修改在两个方向上的原始数据流。
Spider	是一个应用智能感应的网络爬虫，它能完整的枚举应用程序的内容和功能。
Scanner [仅限专业版]	是一个高级的工具，执行后，它能自动地发现web 应用程序的安全漏洞。
Intruder	是一个定制的高度可配置的工具，对web应用程序进行自动化攻击，如：枚举标识符，收集有用的数据，以及使用fuzzing 技术探测常规漏洞。

HTTP安全问题

- **Burp Suite**
- 功能介绍

名称	功能说明
Repeater	是一个靠手动操作来补发单独的HTTP 请求，并分析应用程序响应的工具。
Sequencer	是一个用来分析那些不可预知的应用程序会话令牌和重要数据项的随机性的工具。
Decoder	是一个进行手动执行或对应用程序数据者智能解码编码的工具。
Comparer	是一个实用的工具，通常是通过一些相关的请求和响应得到两项数据的一个可视化的“差异”。

HTTP安全问题

◘ Burp Suite

Response from http://www.example.com:80/ [93.184.216.34]

Forward | Drop | Intercept is on | Action

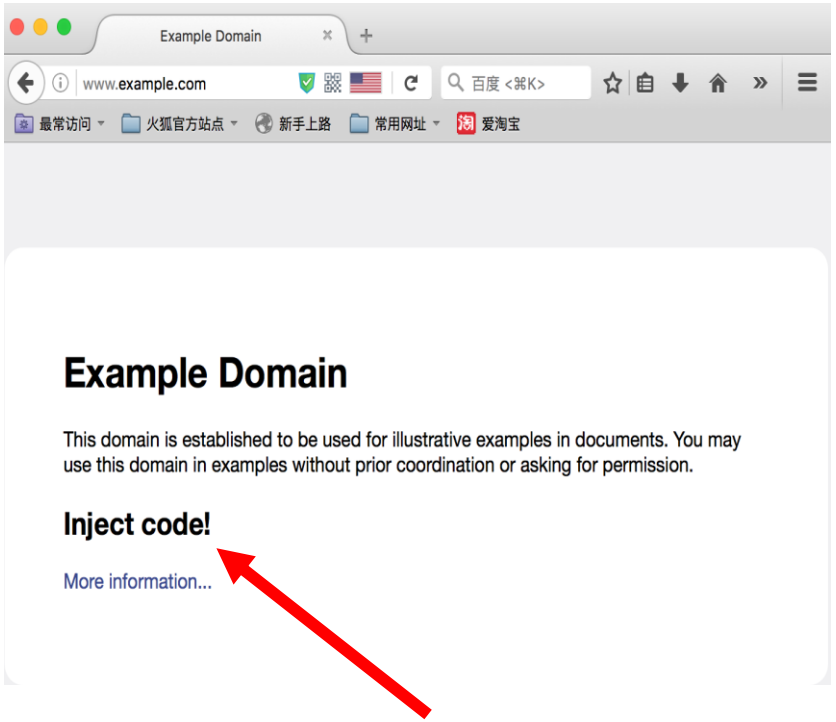
Raw | Headers | Hex | HTML | Render

```
        margin: 0;
        padding: 0;
        font-family: "Open Sans", "Helvetica Neue", Helvetica, Arial, sans-serif;

    }
    div {
        width: 600px;
        margin: 5em auto;
        padding: 50px;
        background-color: #fff;
        border-radius: 1em;
    }
    a:link, a:visited {
        color: #38488f;
        text-decoration: none;
    }
    @media (max-width: 700px) {
        body {
            background-color: #fff;
        }
        div {
            width: auto;
            margin: 0 auto;
            border-radius: 0;
            padding: 1em;
        }
    }
    </style>
</head>

<body>
<div>
    <h1>Example Domain</h1>
    <p>This domain is established to be used for illustrative examples in documents. You may use this
    domain in examples without prior coordination or asking for permission.</p>
    <p><h2>Inject code!</h2></p>
    <p><a href="http://www.iana.org/domains/example">More information...</a></p>
</div>
</body>
</html>
```

篡改报文



篡改后响应

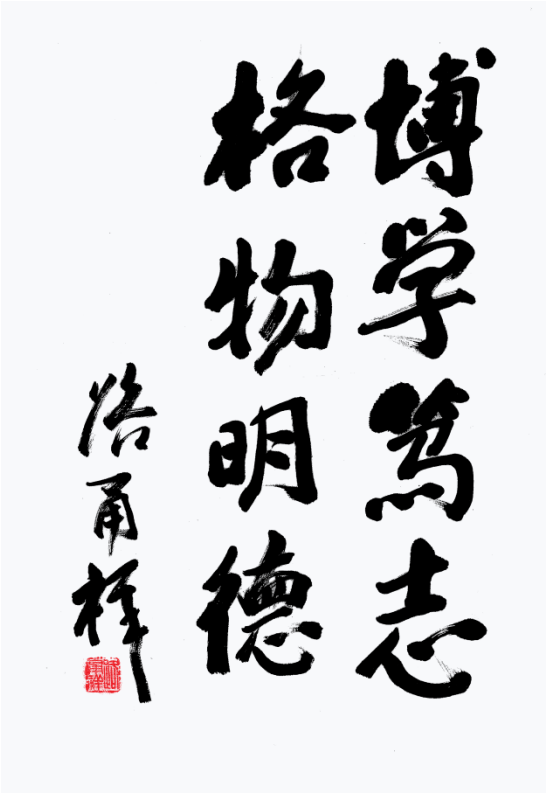
中国科学院大学 University of Chinese Academy of Sciences

HTTP安全问题

- **典型攻击**
- 网络嗅探与监听，中间人
- **难点在于将恶意设备插入到通信链路中！**
- 无线网络监听
- 网络设备劫持：路由器、交换机、防火墙
- ARP欺骗
- 恶意钓鱼设备：Wifi
- DNS劫持

本章大纲

- **HTTP及安全问题**
  - 基础知识回顾
  - HTTP安全问题
  - HTTPS协议
- **Cookie及安全问题**
  - 基础知识
  - 安全策略
  - 安全问题
  - 顶会论文



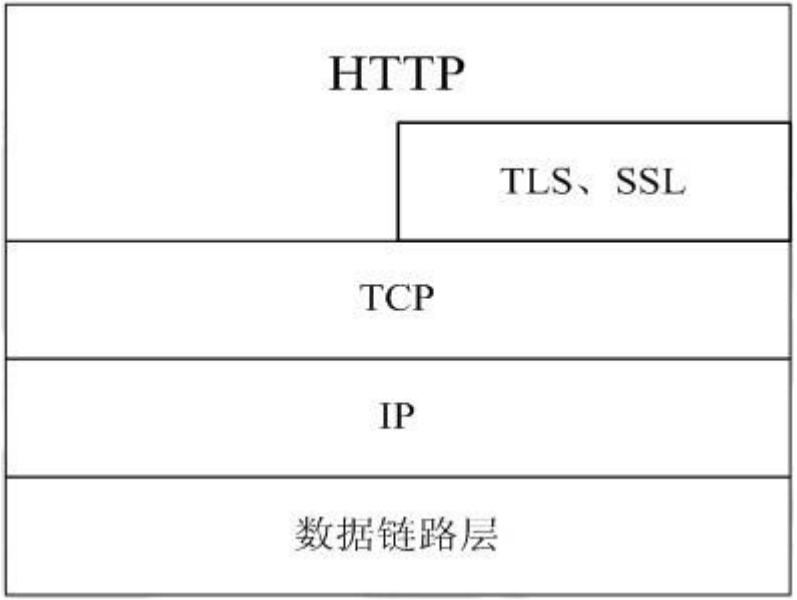
HTTPS协议

- **历史背景**
- Hyper Text Transfer Protocol over Security Socket Layer
- 1994年由Netscape创建，并在其Netscape Navigator浏览器中使用
- 初期HTTPS使用的是SSL协议，随着SSL协议逐渐演变成TLS协议，在2000年五月在RFC2818中正式确定了HTTPS标准

HTTPS协议

- 工作流程

- HTTPS同样也是应用层协议，相比HTTP多加入了SSL/TSL层
- 基于<请求-响应>工作模式
- 在正式传输数据前，先进行身份认证
- 认证成功后再协商传输的加密秘钥
- 后续的传输数据使用秘钥进行加密



HTTPS协议

- HTTPS通信中，客户端如何认证服务器身份？
- 通信密钥如何采用安全方式协商？

HTTPS协议

工作流程

1、客户端发起HTTPS请求

2、服务端准备公私钥和证书

3、服务端发送证书（含公钥）

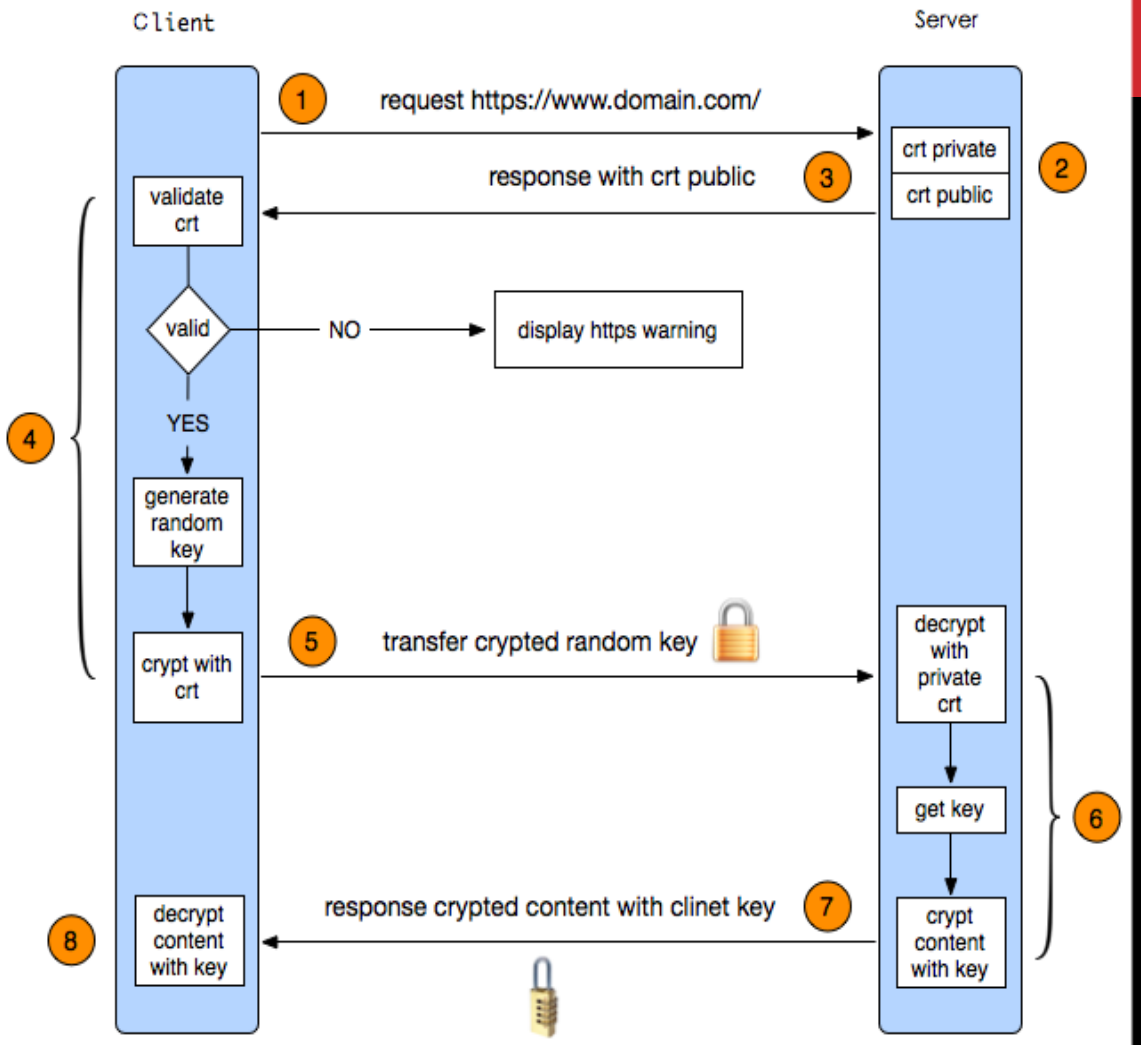
4、客户端解析证书

5、客户端生成对称密钥，并用公钥加密后，发送给服务端

6、服务端使用私钥解密，获得对称密钥

7、服务端使用对称密钥加密信息并发送

8、客户端使用对称密钥解密信息



HTTPS协议

- 要点
- 使用证书确认网站身份
  - 证书由专门的CA机构颁发，携带公钥信息，不可伪造或篡改
- 基于非对称密钥分发对称密钥
  - 客户端用公钥加密，服务器用私钥解密
  - 确保用于通信的对称密钥，不被攻击者监听获取
- 使用对称密钥通信
  - 加解密效率高

HTTPS协议

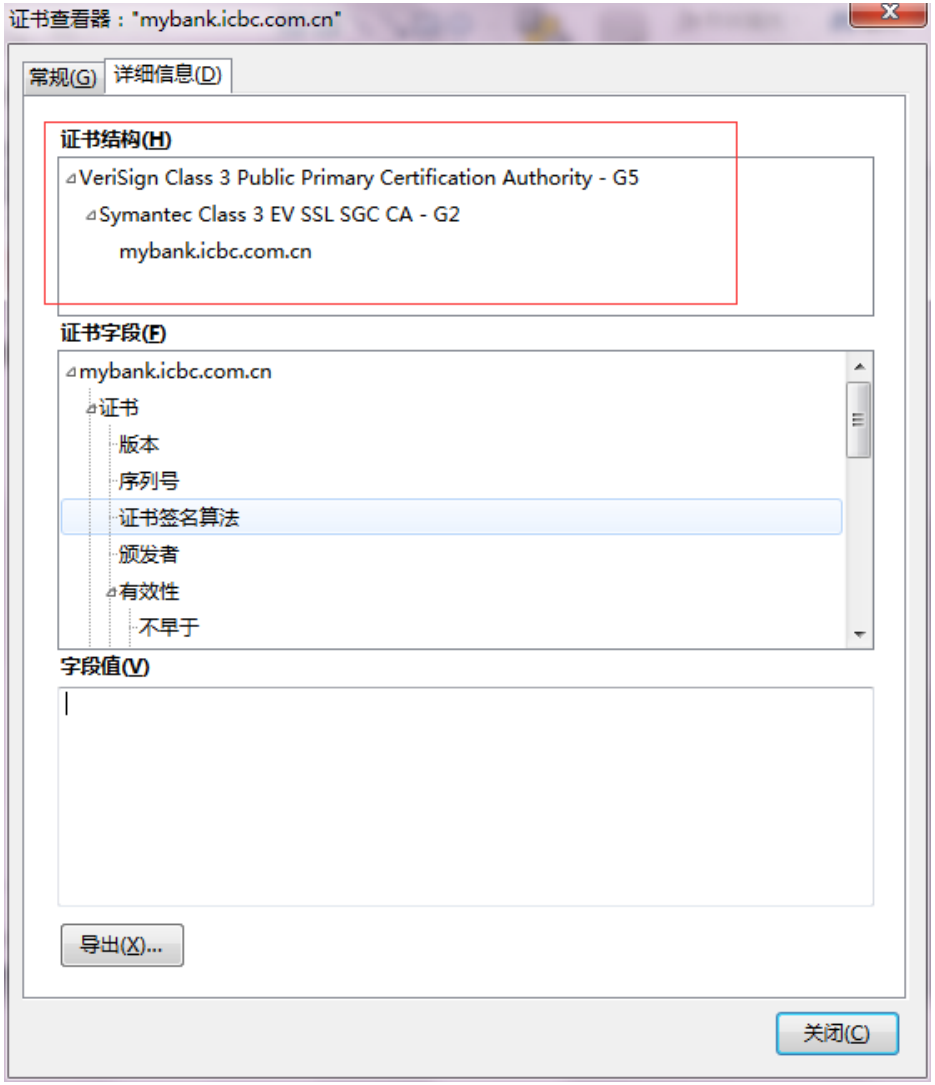
- **RSA非对称加密算法**
- 1977年由Ron Rivest，Adi Shamir，Leonard Adleman一起提出的
- 至今，还没有任何可靠的攻击RSA算法的方式；只要其钥匙的长度足够长，用RSA加密的信息实际上是不能被解破的
- RSA算法基于数论事实：将两个大质数相乘十分容易，但是想要对其乘积进行因式分解却极其困难
- RSA生成一对非对称密钥——公钥和私钥
- 公钥加密，只能私钥解密；私钥加密，只能公钥解密
- 私钥妥善保存，公钥可以公开

HTTPS协议

- **数字证书**
- 数字证书-验证公钥所属的用户身份
  - 网络世界里的“身份证”，信息不可篡改
- 证书管理机构（certificate authority，CA）
  - CA用其私钥对用户的身份信息（包括用户信息及其公钥等信息）进行签名，该签名和用户的身份信息一起就形成了证书。
  - 签名：对用户身份信息做哈希运算，对哈希值用私钥加密
- 证书信任链
  - 信任关系可传递，形成证书信任链

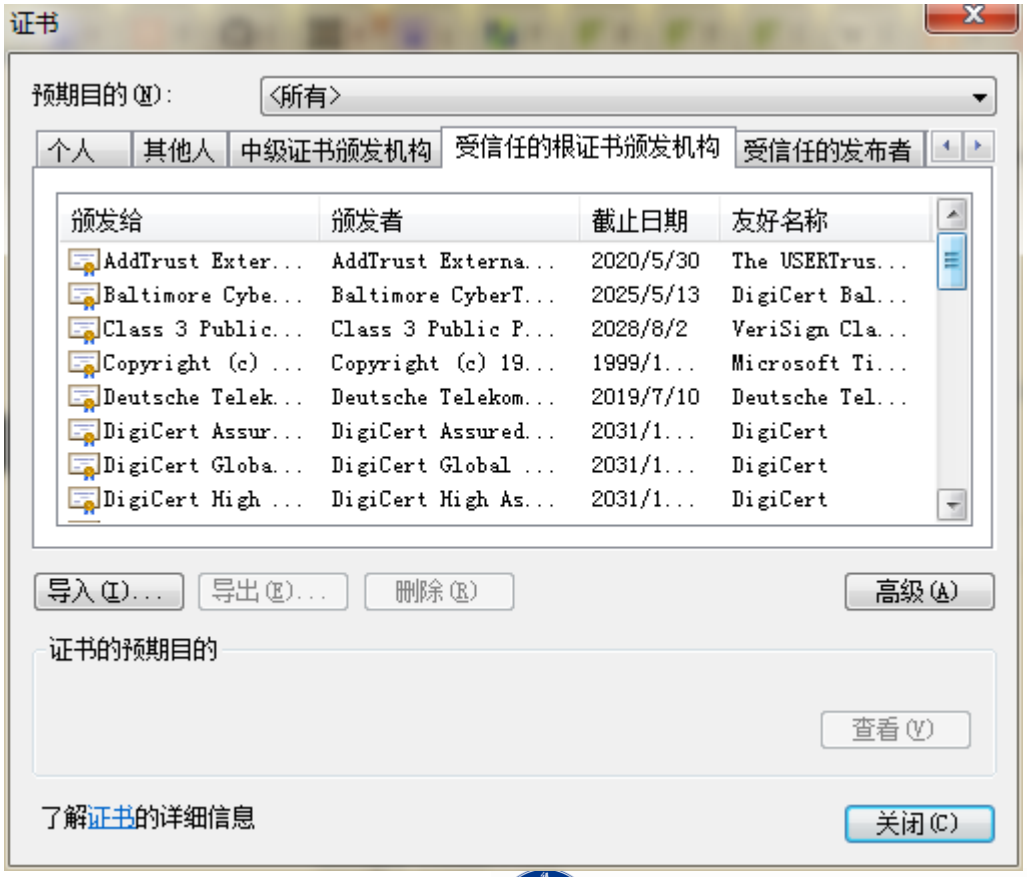
HTTPS协议

- 证书信任链



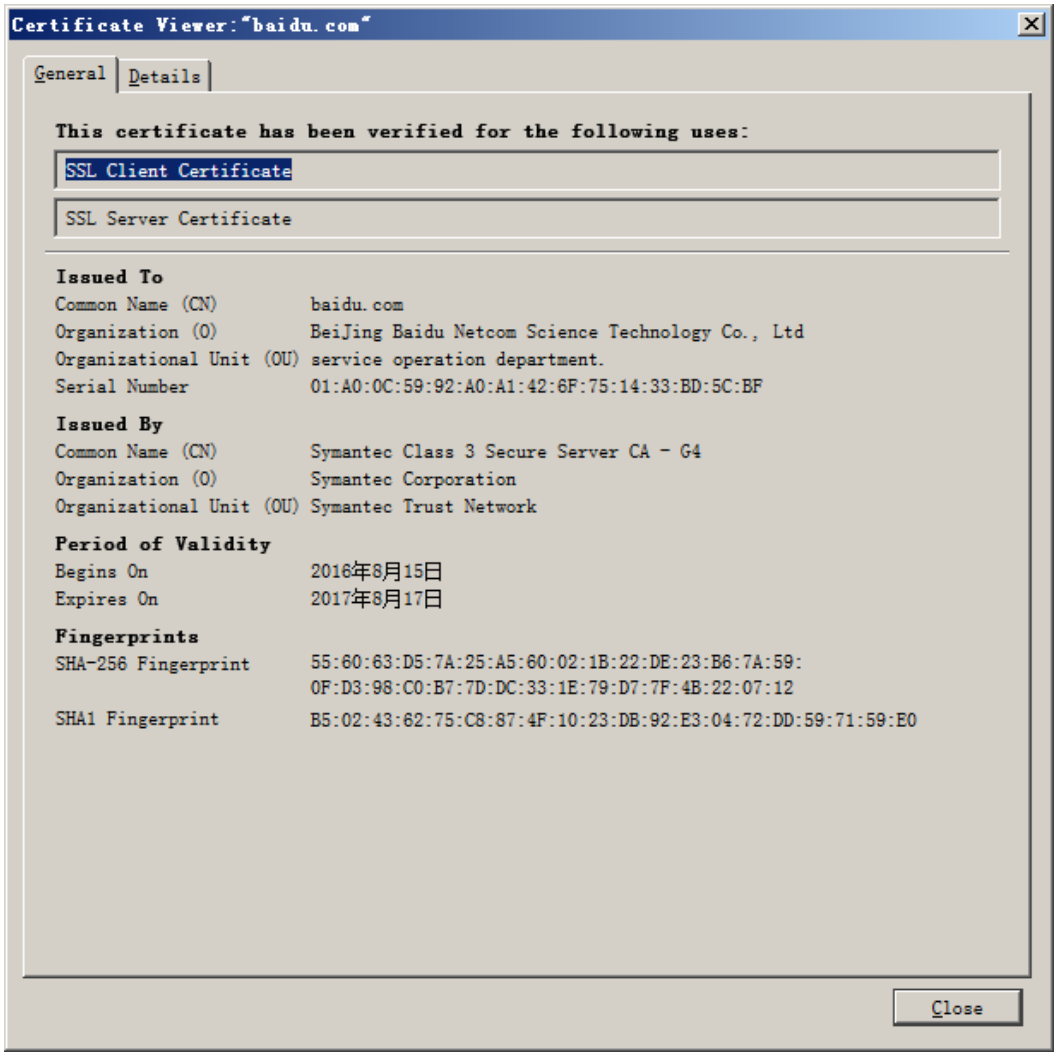
HTTPS协议

- 浏览器根证书
- 各大浏览器厂商预置在浏览器内
- 直接受浏览器信任的证书



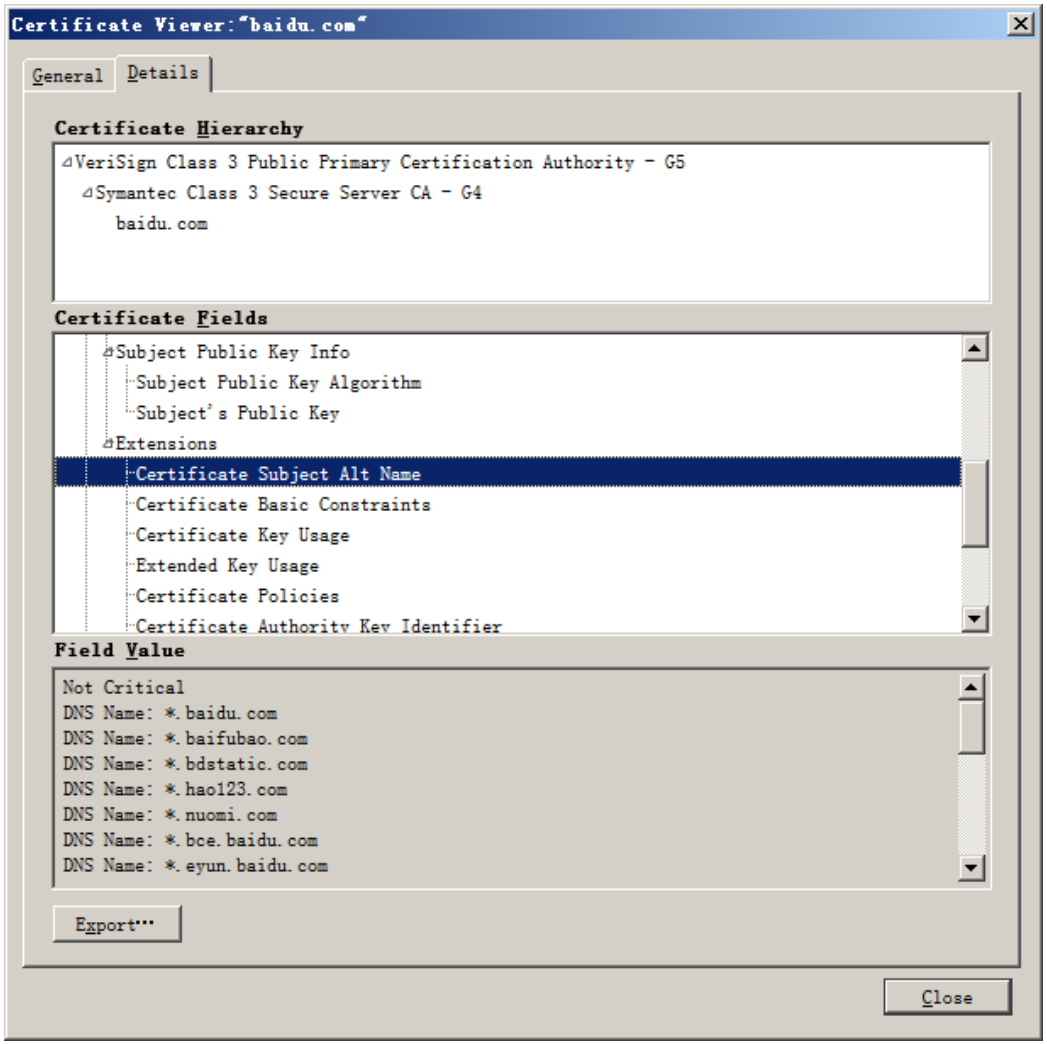
HTTPS协议

□ 数字证书



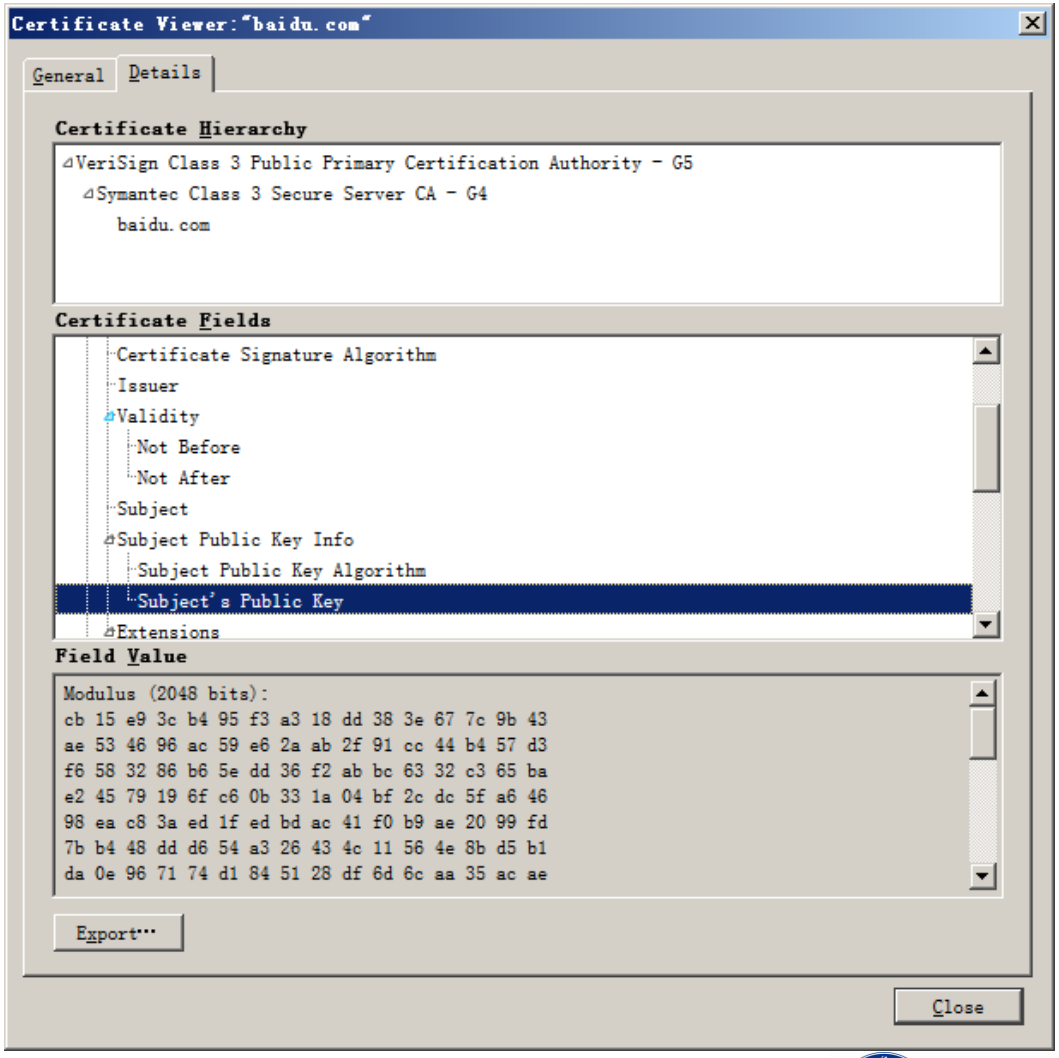
HTTPS协议

- 数字证书



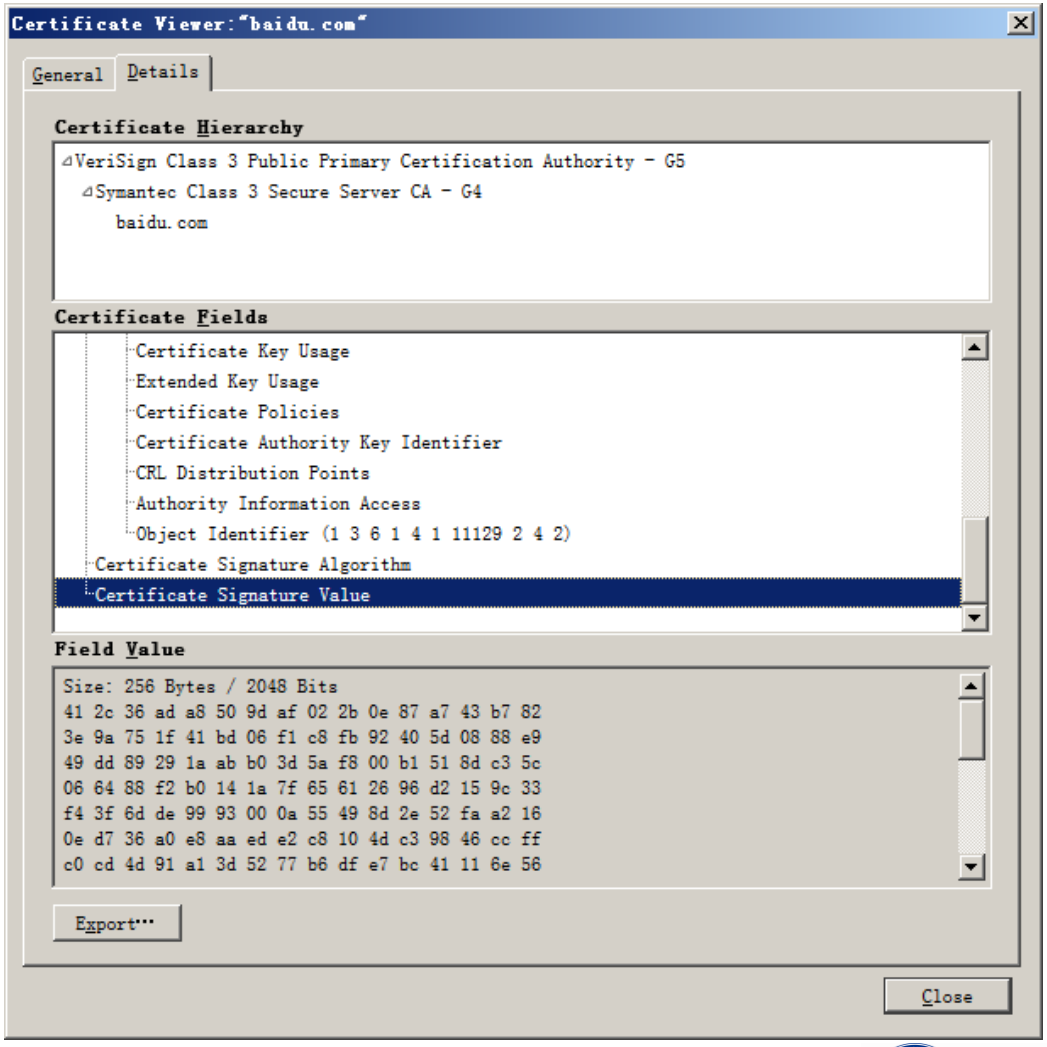
HTTPS协议

- 数字证书



HTTPS协议

- 数字证书



HTTPS协议

□ 数字证书

HTTPS协议

- 数字证书是安全的
- 证书生成
- 网站管理员向CA申请

UserInfo = {User, Domain, User_PubKey, HashAlgorithm…}

Hash = Hash_ Algorithm(UserInfo)

Signature = RSA_CA_Private(Hash, CA_PriKey)

Certificate = {UserInfo, Signature, HashAlgorithm…}

网站得到： Certificate， User_PubKey，User_PriKey

HTTPS协议

- 数字证书是安全的
- 证书校验
- 客户端操作

公开：CA_PubKey

Hash1 = Hash_ Algorithm(UserInfo)

Hash2 = RSA_CA_Public(Signature, CA_PubKey)

Compare(Hash1, Hash2)

- 比对成功则认为证书中的内容可信！

HTTPS协议

HTTPS是安全的

1、客户端发起HTTPS请求

2、服务端准备公私钥和证书

3、服务端发送证书（含公钥）

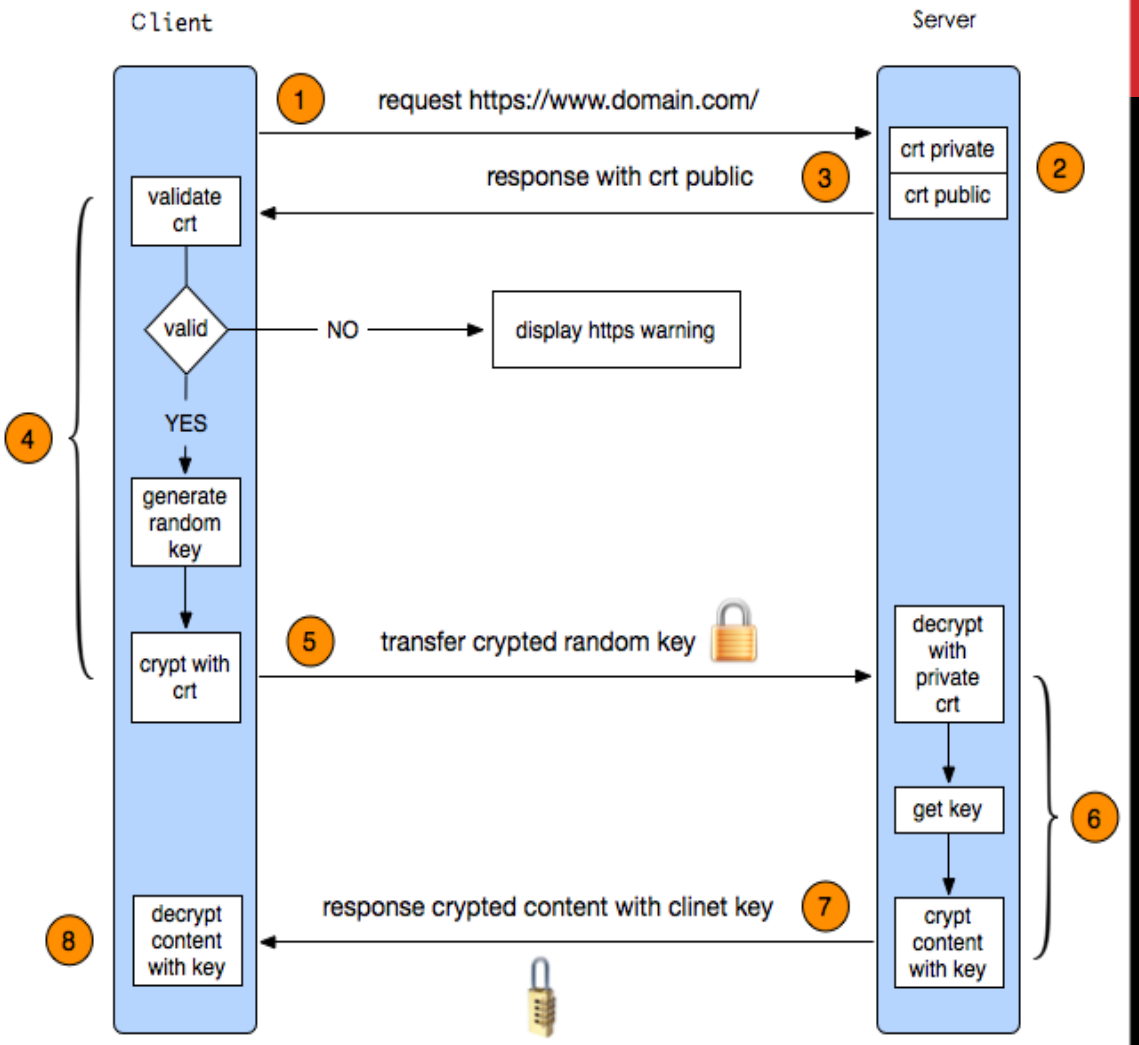
4、客户端解析证书

5、客户端生成对称密钥，并用公钥加密后，发送给服务端

6、服务端使用私钥解密，获得对称密钥

7、服务端使用对称密钥加密信息并发送

8、客户端使用对称密钥解密信息



HTTPS协议

HTTPS不足

- 安全方面也有小瑕疵
  - 2015年3月爆出的浏览器FREAK漏洞（Factoring RSA Export Keys， CVE-2015-0204)，由于密钥长度不足导致被破解
- 性能与速度方面
  - 由于需要认证，势必会导致在建立连接，身份验证等时候带来时间的开销
  - 由于加密，势必会加大服务器与客户端的运算量
- 成本方面
  - 证书需要购买
  - 现在诸如Let’s encrypt等开始提供免费的HTTPS证书服务

HTTPS协议

- 小结

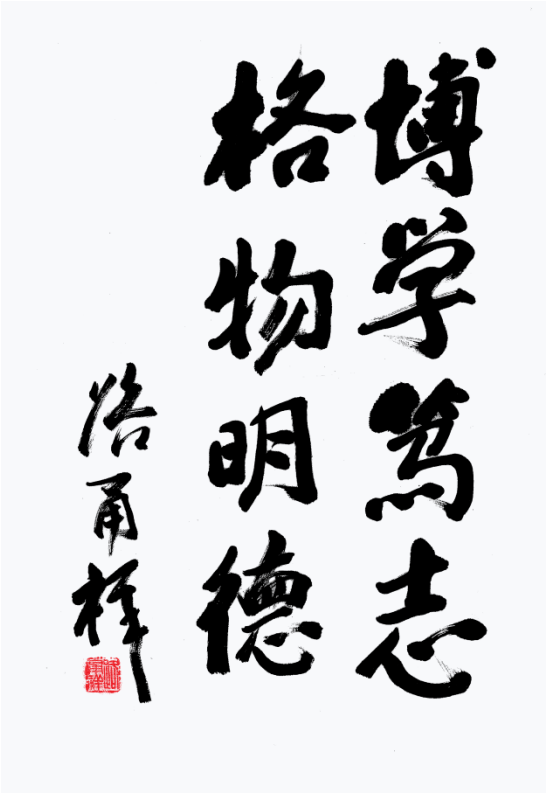
- 使用HTTPS协议可认证用户和服务器，确保数据发送到正确的客户机和服务器。
- HTTPS协议是由SSL+HTTP协议构建的可进行加密传输、身份认证的网络协议，要比http协议安全，可防止数据在传输过程中不被窃取、改变，确保数据的完整性。
- 浏览器嗅探与监听、中间人
- 算法实现上的漏洞/后门？
- 没有绝对安全！

HTTP及安全问题

- 小结
- HTTP协议在明文传输、身份校验等方面存在安全问题
- HTTP协议先天性无法抵御监听和中间人攻击
- HTTPS能够有效的解决HTTP存在的安全问题

本章大纲

- **HTTP及安全问题**
  - 基础知识回顾
  - HTTP安全问题
  - HTTPS协议
- **Cookie及安全问题**
  - 基础知识
  - 安全策略
  - 安全问题
  - 顶会论文



基础知识

历史背景

- 网站为了辨别用户身份与维持状态而存储在用户本地终端上的数据；
- 由Netscape前雇员Lou Montulli与1993年3月发明，定义了cookie最原始的标准，又称Netscape标准；
- W3C在在RFC2109中推出官方的Cookie标准；
- RFC2965尝试修补RFC2109的缺点，并最终取代了RFC2019；
- RFC6265又将RFC2965取代，添加了Http-Only字段；

基础知识

- **设计目的**

- HTTP协议是无状态的协议，导致服务器无法判断是谁在浏览网页；
- 网上商城的购物车、用户的登陆状态、用户曾经点过的按钮，访问过的页面等等均无法保存；
- 维持状态是很有必要，否则将严重影响交互性以及用户体验；
- cookie和session应运而生，二者的设计初衷都是为了维持状态；
- 简单来区分，session存于服务器，cookie存在客户端；

基础知识

工作过程

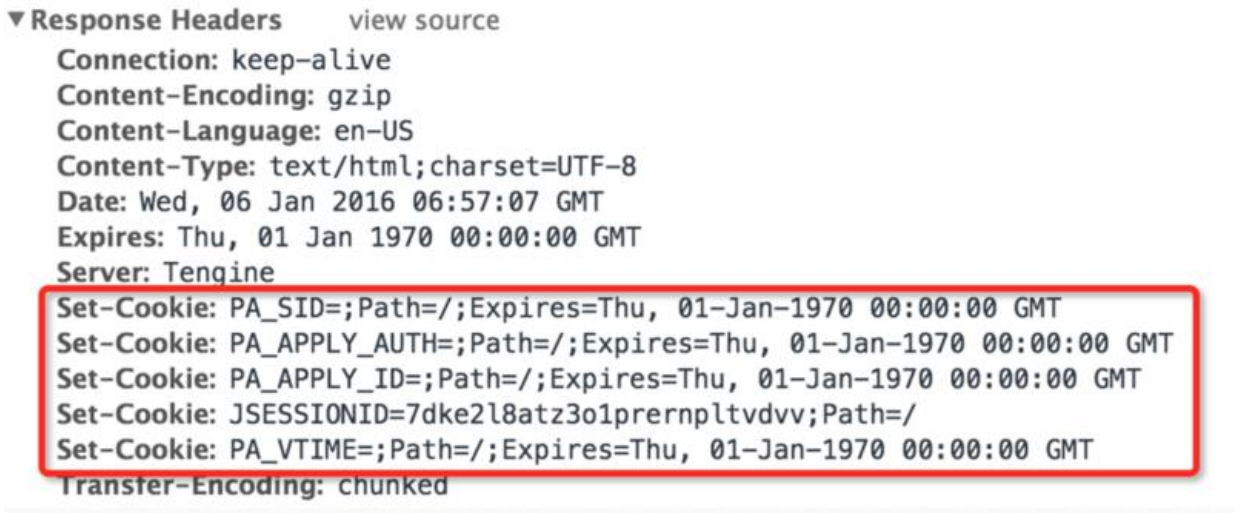
- 根据Expire（或max-age, 失效时间）属性值，将cookie区分为内存cookie和持久cookie
  - 内存cookie，只在本次会话中有效，会话结束后立即删除cookie；
  - 持久cookie，在客户端硬盘中存储，直至过了失效期；
- 后续的请求中，Cookie连同请求一起发送至服务器；
- 服务器根据cookie区分状态

基础知识

- **工作过程**
- 生成阶段：服务端（或本地JS脚本）根据不同的状态，生成特定的cookie，用以标示不同的身份或状态
- 设置阶段：服务端（或本地JS脚本）进行SetCookie，将生成的cookie设置与客户端中
  - 服务端：在响应头中以Set-Cookie字段返回cookie值，告知客户端保存
  - JS：调用document.cookie=“key=value”设置
- 使用阶段：
  - 客户端发送请求时，携带cookie，以维护会话状态

基础知识

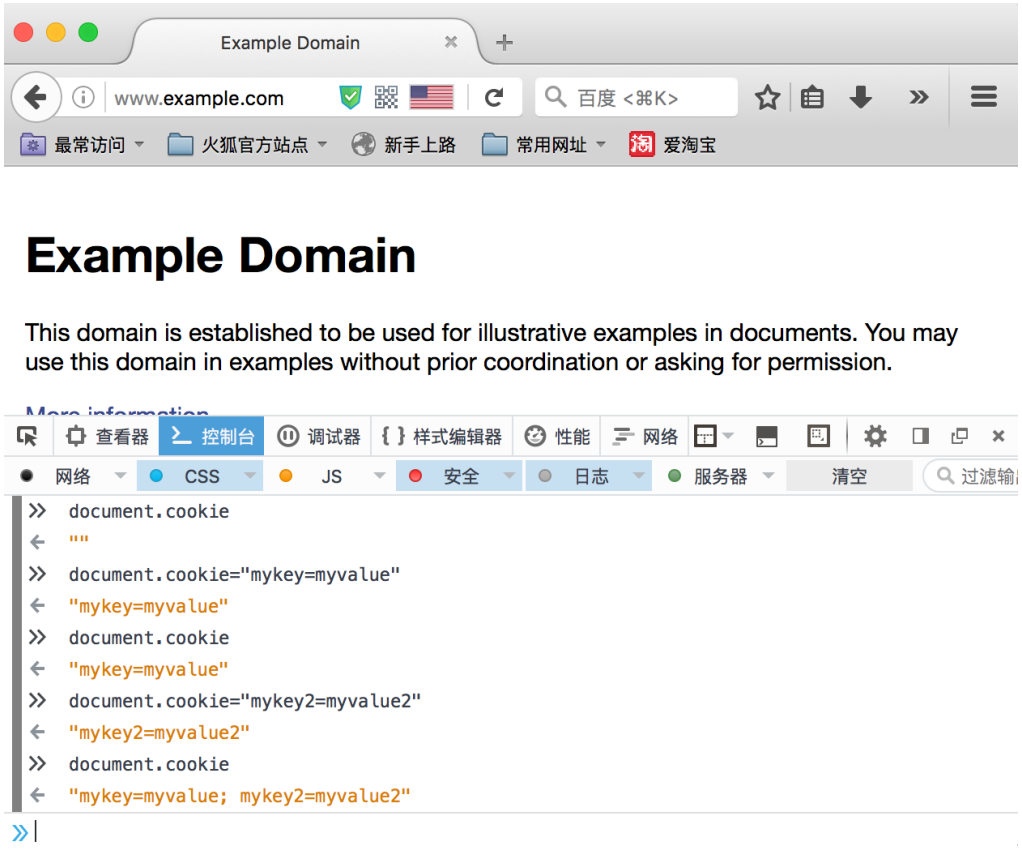
- 服务端设置
- 以PHP为例
- setcookie(name,value,expire,path,domain,secure)



基础知识

本地JS设置

操作document.cookie对象

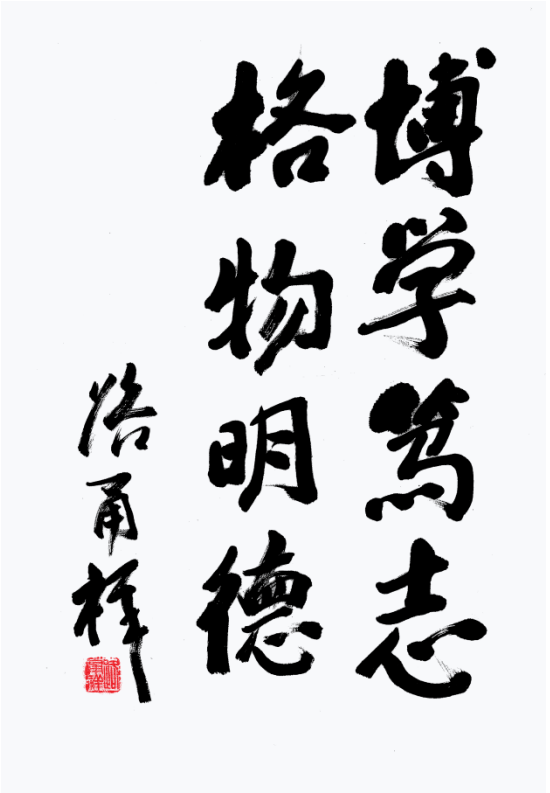


基础知识

- **cookie属性**
- Domain，指定cookie被绑定到哪台主机上，如：domain=.example.org，cookie仅会被发送到*.example.org上；
- Path，控制那些访问能够触发cookie的发送。如不指定path，cookie对全站生效。如path=/test，则只在访问/test的网页时才会被发送；
- Expire（max-age），指明了cookie失效的时间，进一步将cookie划分为内存cookie和持久cookie；
- Secure，如指定则该cookie只能通过安全通道传输（即SSL通道）；
- Httponly，JS无法读取和修改标记为HttpOnly的Cookie，这样Cookkie可免受脚本的攻击（如XSS）；

本章大纲

- **HTTP及安全问题**
  - 基础知识回顾
  - HTTP安全问题
  - HTTPS协议
- **Cookie及安全问题**
  - 基础知识
  - 安全策略
  - 安全问题
  - 顶会论文



安全策略

- **Cookie与同源**
- Cookie不应该被无关页面篡改
- Cookie的domain属性设计初衷是限制Cookie的有效范围，但实际上并不精准

foo.example.com的Cookie domain参数设置为：	实际的cookie范围	
	非IE浏览器	IE浏览器
完全未设置	foo.example.com(精确匹配）	*.foo.example.com
bar.foo.example.com	Cookie 未能设置成功：因为domain值比当前主机名范围更窄	
foo.example.com	*.foo.example.com	
baz.example.com	Cookie 未能设置成功：因为域名不匹配	
example.com	*.example.com	
ample.com	Cookie 未能设置成功：因为域名不匹配	
.com	Cookie 未能设置成功：因为域名太宽泛，会带来安全风险	

安全策略

- **安全的cookie设计**
- 一个相对安全的cookie登陆状态设计方案：

1、用户登陆根据信息组合生成token，存储在数据库中

2、生成token的信息包括用户名、ip、UA、expiration、salt等

3、信息组合再进行可逆加密算法进行加密

4、用户发送的token，解密得出用户名，并与数据库中对应的token比对

5、如果相同则视为登陆状态，继续操作；否则下线操作，要求重新登陆

安全策略

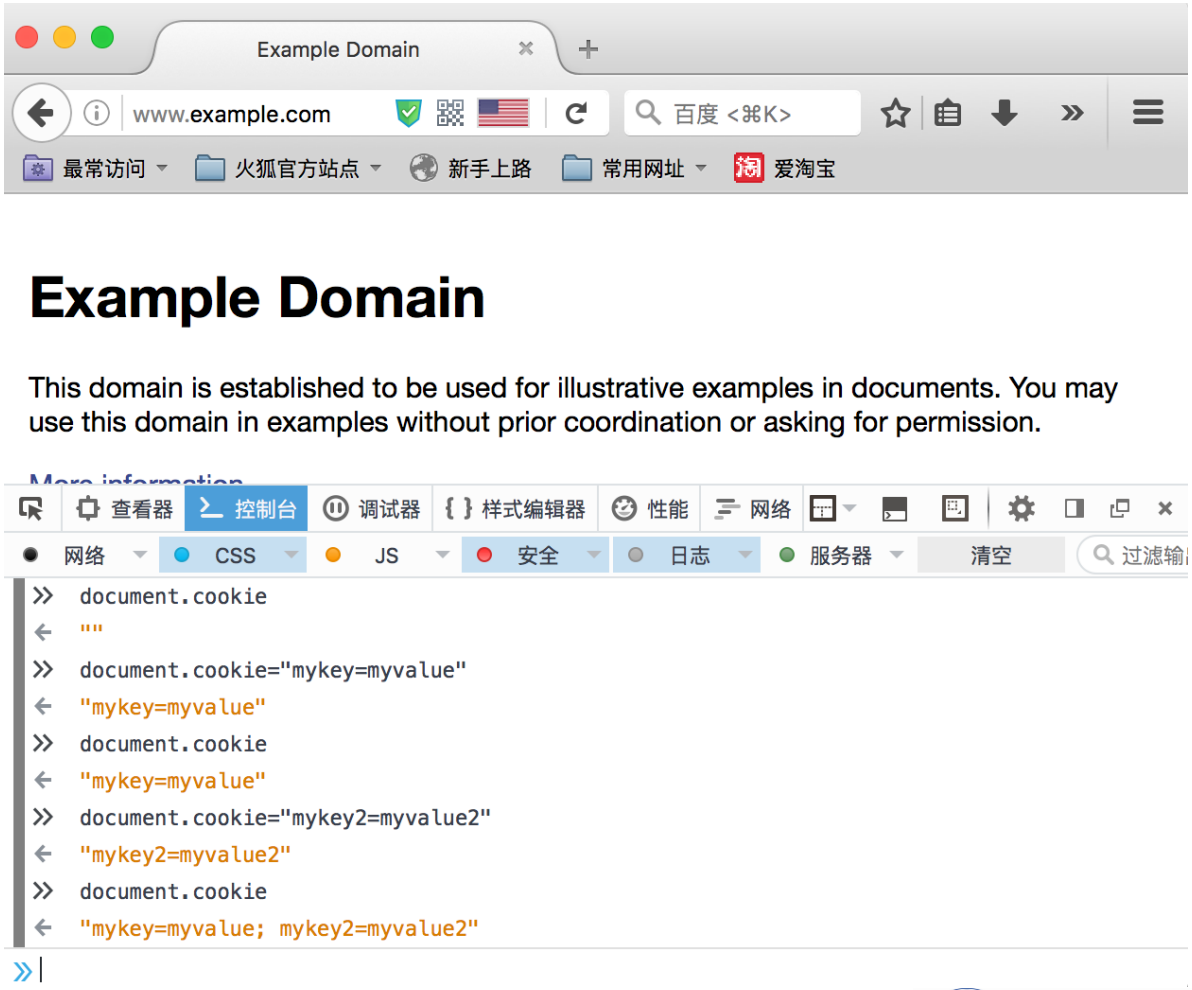
- **安全的cookie设计**
- salt字段属于用户不可知字段，防止直接伪造
- 加密算法，salt的使用，使猜解cookie的难度剧增
- IP，UA的限制，一定程度防止直接窃取cookie在其他终端登陆

安全策略

- **HttpOnly保护**
- HttpOnly最早由微软提出，并在IE6中实现，至今已经逐渐成为一个浏览器标准。
- 浏览器将禁止页面的JavaScript访问带有HttpOnly属性的Cookie
- 各种浏览器对HttpOnly支持情况：
  - ✓Microsoft IE 6 SP1+
  - ✓Mozilla Firefox 2.0.0.5+
  - ✓Mozilla Firxfox 3.0.0.6+
  - ✓Google Chrome
  - ✓Apple Safari 4.0+
  - ✓Opera 9.5+

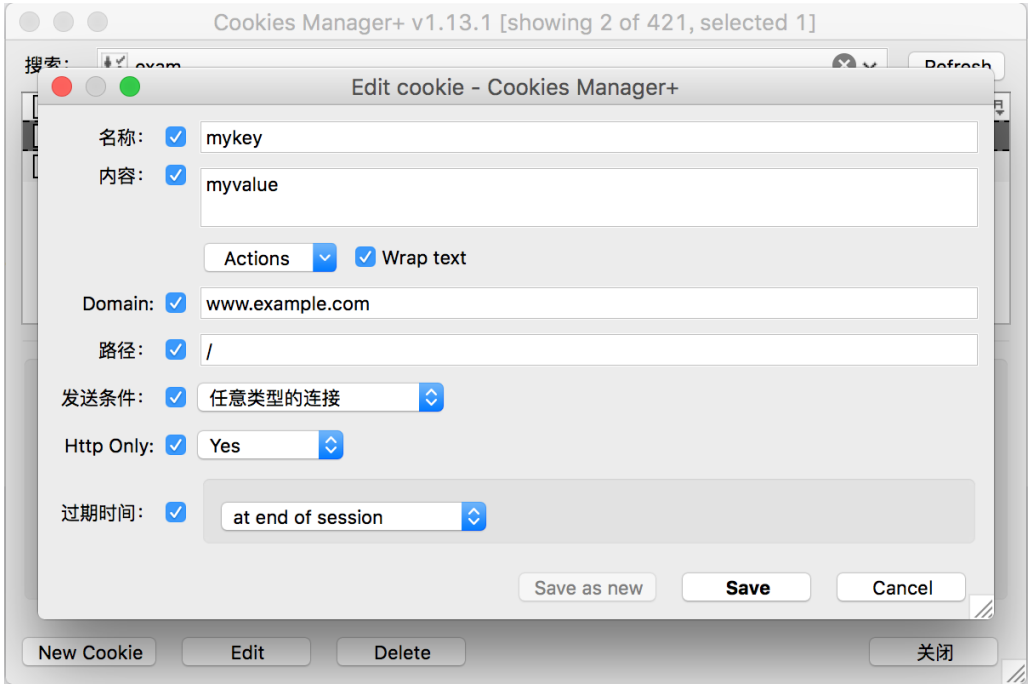
安全策略

HttpOnly保护



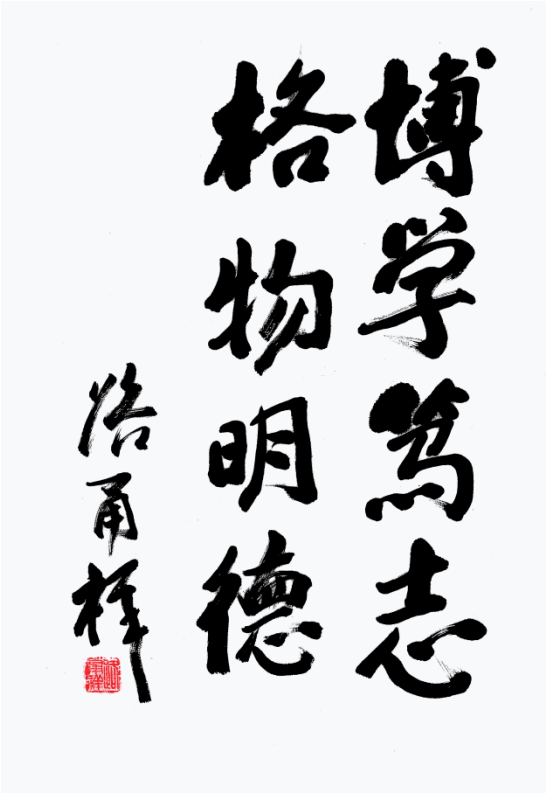
安全策略

□ HttpOnly保护



本章大纲

- **HTTP及安全问题**
  - 基础知识回顾
  - HTTP安全问题
  - HTTPS协议
- **Cookie及安全问题**
  - 基础知识
  - 安全策略
  - 安全问题
  - 顶会论文



安全问题

- **cookie窃取**
- 窃取到cookie并原样发送，同样可以进行身份欺骗
- 典型手段
  - ✓基于HTTP明文，可在传输过程窃取；
  - ✓通过XSS利用脚本窃取；

安全问题

- **cookie猜解**

- 通过暴力破解或计算分析的办法，猜解cookie

- 假设教务系统网站使用setcookie：account=studentnum+timestamp来标识学生登陆状态，其中timestamp为登陆时的时间戳;
- 在已经学号和大致登陆时间（选课时间、期末查分）情况下，通过暴力枚举，即可实现身份欺骗；

安全问题

- **隐私信息泄露**
- 网站使用cookie来保存用户的活动轨迹
  - 看过某个商品，将某商品加入购物车
- 对数据进一步分析，甚至可以得到用户的生活习惯，兴趣爱好，从而产生商业的利益

安全问题

- **Cookies标识**

- 用户浏览某网站时，网站存储在用户浏览器上的一段信息，在 Web 服务器和浏览器之间传递，用于辨别用户身份与维持状态

- Cookie可以记录用户ID、密码、浏览过的网页等多种信息

名称	Internet 地址	截止期限
cookie:administrator@360.cn/	Cookie:administrator@360.cn/	2046/6/10 23:19
cookie:administrator@accounts.google.com/	Cookie:administrator@accounts.google....	2026/8/14 23:22
cookie:administrator@apis.google.com/	Cookie:administrator@apis.google.com/	2016/7/17 23:23
cookie:administrator@baidu.com/	Cookie:administrator@baidu.com/	2017/10/1 19:35

安全问题

- Cookies标识
- 第一次访问网站

① 用户首次访问京东，浏览器在本地未找到相应的Cookie

② 浏览器向服务器发送不携带用户Cookie的HTTP请求

③ 服务器没有检测到Cookie：创建新用户ID，保存到数据库；响应请求，并在客户端写入Cookies

④ 用户选择商品加入购物车，用户Cookie和商品信息发送到服务器

⑤ 服务器从数据库中检索到用户ID，记录购物车信息

安全问题

- Cookies标识
- 第二次访问网站

① 用户再次访问京东，浏览器在本地找到相应的Cookie

② 浏览器向服务器发送携带Cookie的HTTP请求

③ 服务器从数据库中检索到用户标识，将其购物车信息返回给用户，并根据用户浏览记录推荐商品

- 用户没有登录，关闭浏览器再次打开京东时，但京东依然知道购物车中的商品——Cookie！

网络对用户进行了Cookie追踪！

安全问题

□ Cookies标识

您还没有登录！登录后购物车的商品将保存到您账号中 立即登录

全部商品 1　京东大药房　配送至：北京朝阳区三环以内

全选　商品　单价(元)　数量　小计(元)　操作

京东自营　钻石会员满¥79.00免运费，其他会员满¥99.00免运费

联想（Lenovo）小新310经典版14英寸　颜色：新310经典版【i7-75...
笔记本电脑（i7-7500U 4G 500G 2G独

4399.00　1　4399.00　删除

购买增值保障

用户关闭浏览器，第二次访问时，仍可以看到上次添加到购物车中的商品

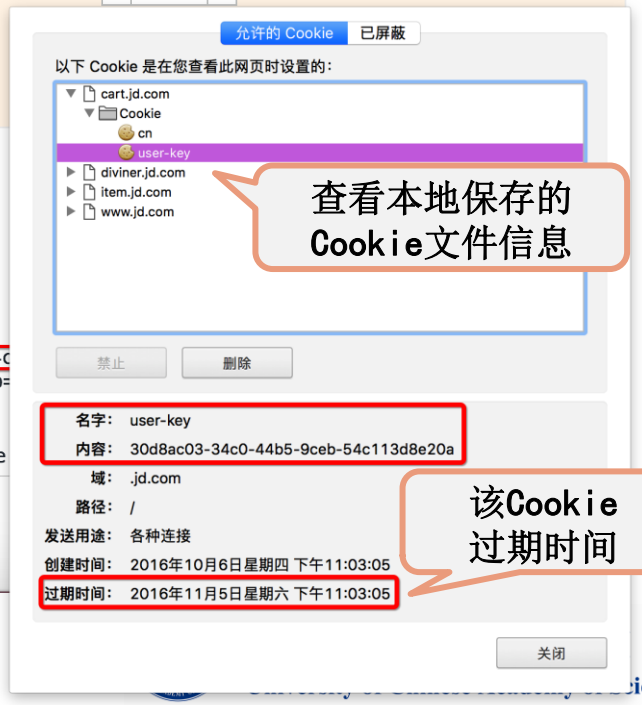
▼ Request Headers　view source

Accept: */*
Accept-Encoding: gzip, deflate, sdch, br
Accept-Language: zh-CN,zh;q=0.8
Cache-Control: max-age=0
Connection: keep-alive
Cookie: __jdv=122270672|direct|-|none|-|1475766179910; user-key=30d8ac03-34c0-44b5-9ceb-54c
jd=1-72-2799-0; cn=1; __jda=122270672.657335033.1475766180.1475766180.1475766180.1; __jdb=
Host: ai.jd.com
Referer: https://www.jd.com/
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_0) AppleWebKit/537.36 (KHTML, like

用户第二次访问时，浏览器发送的Cookie信息，关键信息为user-key

▼ Query String Parameters　view source　view URL encoded

callback: cathot
pin: 0
uuid: 657335033



查看本地保存的Cookie文件信息

该Cookie过期时间

安全问题

□ Cookies标识

① 用户访问网站，浏览/购买/查询商品

② 用户访问其它新闻、娱乐网站

③ 按用户喜好，进行广告插入、推广

安全问题

□ Cookies标识

安全问题

- **Evercookie**

- Evercookie is a Javascript API that produces extremely persistent cookies in a browser. Its goal is to identify a client even after they‘ve removed standard cookies, Flash cookies (Local Shared Objects or LSOs), and others.

- Evercookie将cookie通过多种机制保存到系统多个地方，如果用户删除其中某几处的cookie， Evercookie仍然可以恢复cookie

- 如果开启本地共享对象(Local Shared Objects)，Evercookie甚至可以跨浏览器传播

野火烧不尽，春风吹又生

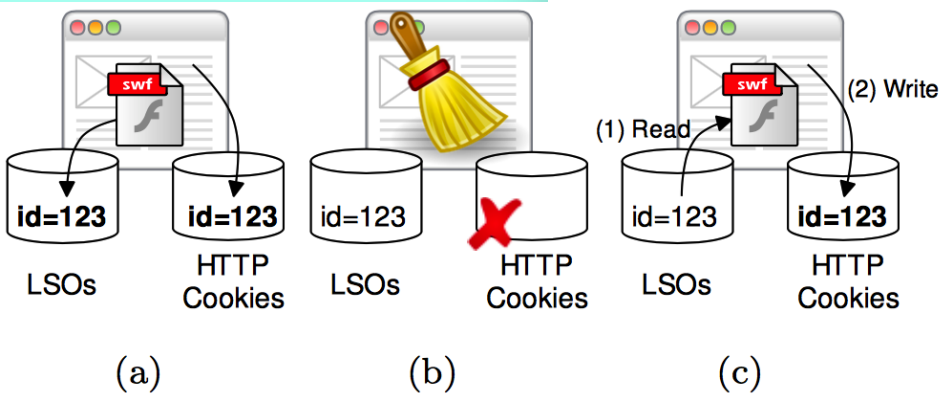
安全问题

Evercookie

Specifically, when creating a new cookie, it uses the following storage mechanisms when available:

- Standard HTTP Cookies
- HTTP Strict Transport Security (HSTS) Pinning
- Local Shared Objects (Flash Cookies)
- Silverlight Isolated Storage
- Storing cookies in RGB values of auto-generated, force-cached PNGs using HTML5 Canvas tag to read pixels (cookies) back out
- Storing cookies in Web History
- Storing cookies in HTTP ETags
- Storing cookies in Web cache
- window.name caching
- Internet Explorer userData storage
- HTML5 Session Storage
- HTML5 Local Storage
- HTML5 Global Storage
- HTML5 Database Storage via SQLite
- HTML5 IndexedDB
- Java JNLP PersistenceService
- Java CVE-2013-0422 exploit (applet sandbox escaping)

Evercookie可以采用以下的存储机制记录cookie



- 从所有的存储机制中删除Cookie，对于经验的用户也很困难

安全问题

- **Evercookie**
- 开源：https://github.com/samyk/evercookie
- Samy Kamkar用了不到一天时间，开发的开源JavaScript API
- 思考：
- 需要引起警惕：相信有人在使用了！
- 并没使用新技术——**把简单的事做到极致！**

安全问题

- Cookies同步

- 网站example.com嵌入A.com追踪代码

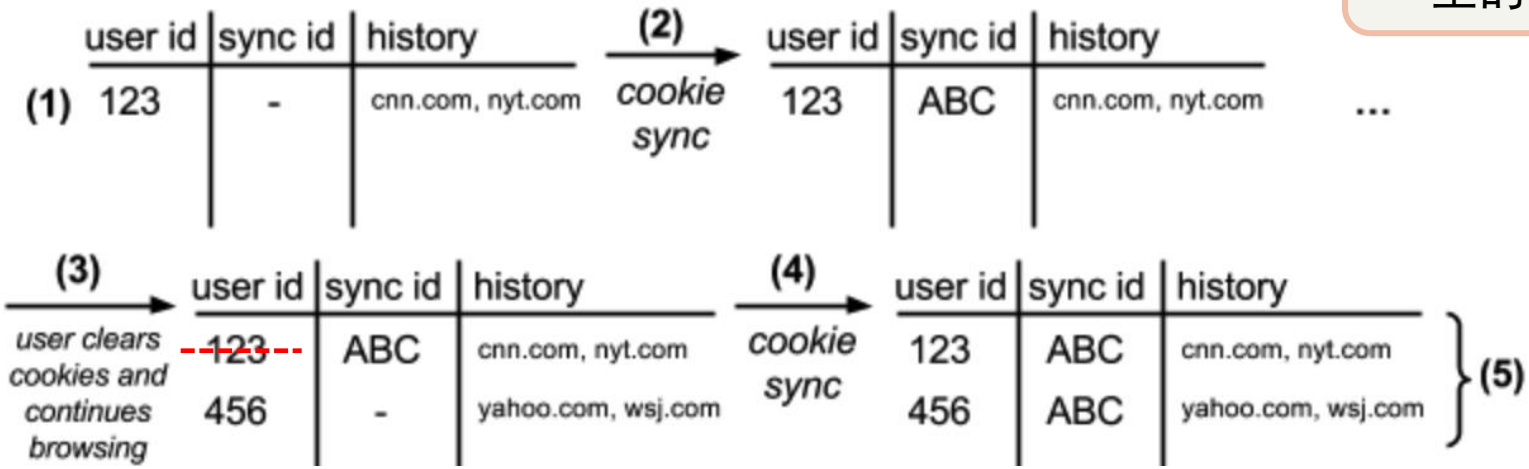
- 访问example.com时，还可能发生一系列用户毫无察觉的动作：

安全问题

Cookies同步

- Cookie同步，即使用户清除Cookie，仍可追踪用户

服务器数据库里的情况



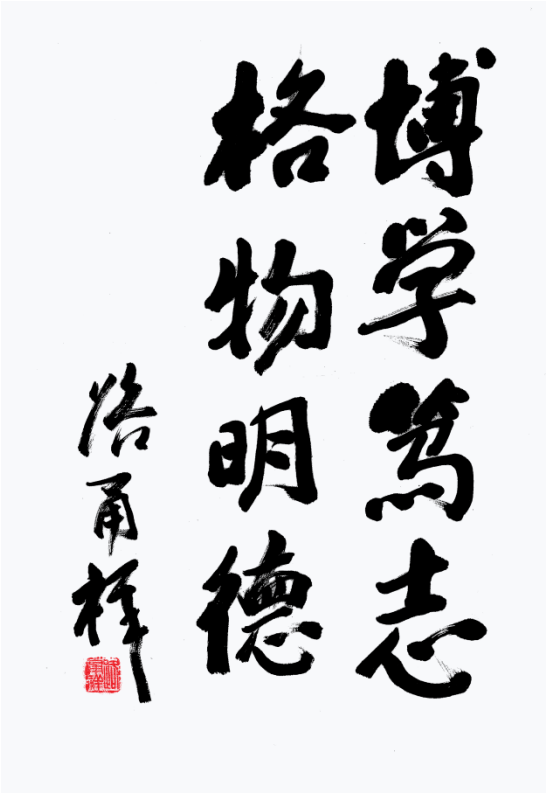
- 只要互相同步的任意一方恢复了Cookie，各方通过Cookie同步都可以恢复追踪

安全问题

- **Cookie追踪“落寞”**
- Cookie容易被清除，且越来越受限制
- 不少安全工具甚至是浏览器都允许或引导用户关闭Cookie

本章大纲

- **HTTP及安全问题**
  - 基础知识回顾
  - HTTP安全问题
  - HTTPS协议
- **Cookie及安全问题**
  - 基础知识
  - 安全策略
  - 安全问题
  - 顶会论文



顶会论文

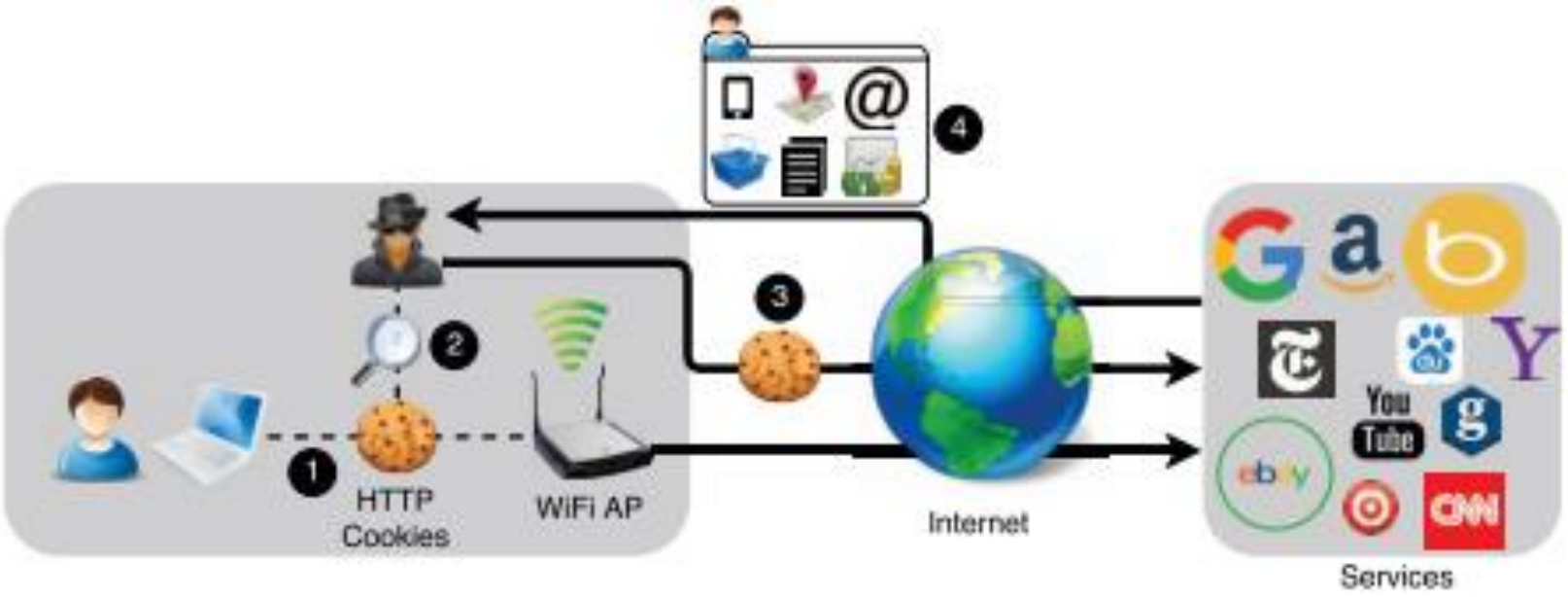
- **The Cracked Cookie Jar: HTTP Cookie Hijacking and the Exposure of Private Information**
- *Suphannee Sivakorn, Iasonas Polakis and Angelos D. Keromytis*
- *Department of Computer Science Columbia University, New York, USA*
- 2016 IEEE Symposium on Security and Privacy（S&P 2016）

顶会论文

- **安全领域四大顶会**
- S&P: IEEE Symposium on Security and Privacy
- CCS: ACM Conference on Computer and Communications Security
- USENIX: USENIX Security Symposium
- NDSS: The Network and Distributed System Security Symposium

顶会论文

- **HTTP cookie劫持**
- **网站未使用HTTPS**
- **网站部分使用HTTPS**



顶会论文

- 威胁模型

- 有害无线接入点或者安全性弱的路由，监听**cookie**，或强制用户浏览器访问某些脆弱网站
- 若**cookie**中没有设置**httponly**属性，可以通过**XSS**获取**cookie**
- 利用向主流网站中插入广告的网站进行**cookie**劫持
- 国家级第三方

顶会论文

STATISTICS OF OUTGOING CONNECTIONS FROM A SUBSET OF OUR CAMPUS' PUBLIC WIRELESS NETWORK FOR 30 DAYS.

Protocol	Connections	Requests	Vulnerable Requests*	Exposed Accounts
HTTP	685,500,365	1,398,044,178	29,908,099	282,459
HTTPS	772,562,024	–	–	–

*HTTP requests to domains that we have audited and found to be vulnerable.

顶会论文

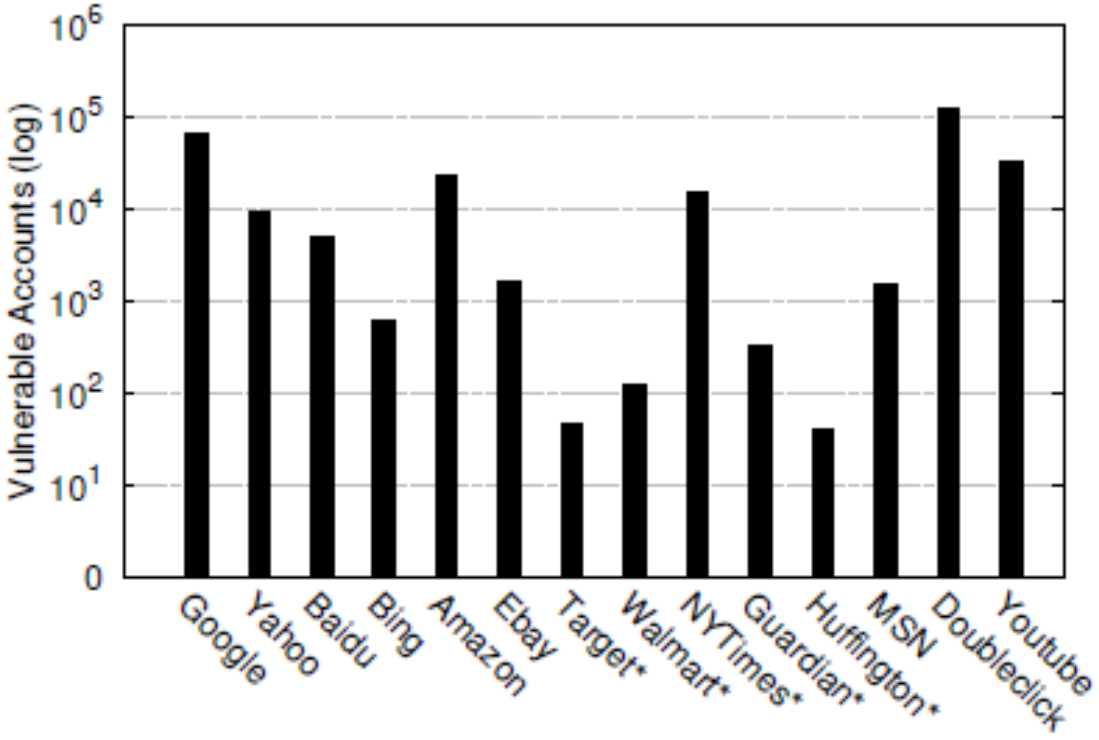


Fig. 2. Number of exposed accounts per service. Services marked with "*" have an explicit userID cookie (or field) that allows us to differentiate users.

顶会论文

OVERVIEW OF THE AUDITED WEBSITES AND SERVICES, THE FEASIBILITY OF COOKIE HIJACKING ATTACKS, AND THE TYPE OF USER INFORMATION AND ACCOUNT FUNCTIONALITY THEY EXPOSE.

Service	HTTPS Adoption	Cookie Hijacking	XSS Cookie Hijacking	Information and Account Functionality Exposed
Google	partial	✓	✗	first and last name, username, email address, profile picture, home and work address, search optimization, click history of websites returned in search results
Baidu	partial	✓	✓	username, email address, profile picture, entire search history, address of any saved location
Bing	partial	✓	✓	first name, profile photo, view/edit search history (incl. images and videos), links clicked from search results, frequent search terms, saved locations, information in interest manager, edit interest manager
Yahoo	partial	✓	✓	username, full name, email address, view/edit search history, view/edit/post answers and questions in Yahoo Answers (anonymous or eponymous), view/edit finance portfolio, view subject and sender of latest incoming emails, extract contact list and send email as user
Youtube	partial	✓	✗	view and change (through pollution attacks) recommended videos and channels
Amazon	partial	✓	✓	view user credentials (username, email address or mobile number), view/edit profile picture, view recommended items, view user wish lists, view recently browsed items, view recently bought items, view/edit items in cart, view shipping name and city, view current balance, view user's review (even anonymous), send email of products or wishlist on behalf of user, obtain email addresses of previously emailed contacts
Ebay	partial	✓	✓	delivery name and address, view/edit items in cart, view/edit purchase history, view items for sale, view previous bids, view user's messages, view/edit watch list and wish lists
MSN	partial	✓	✓	first and last name, email address, profile picture
Walmart	partial	✓	✓	first name, email address, view/edit items in cart, view delivery postcode, write product review
Target	partial	✓	✓	first name, email address, view/edit items in cart, recently viewed items, view and modify wish list, send email about products or wish list
CNN	partial	✓	✓	view/edit profile (full name, postal address, email address, phone number, profile picture) view/edit linked Facebook account, write/delete article comments, recently viewed content on iReport
New York Times	partial	✓	✓	username, email address, view/edit basic profile (display name, location, personal website, bio, profile picture) username, email address, view/edit list of saved articles, share article via email on behalf of user
Huffington Post	partial	✓	partial	profile can be viewed and edited (login name, profile photo, email address, biography, postal code, location, subscriptions, fans, comments and followings). change account password, delete account
The Guardian	partial	✓	✓	username, view public section of profile (profile picture, bio, interests), user's comments, replies, tags and categories of viewed articles, post comments on articles as user
Doubleclick	partial	✓	✓	ads show content targeted to user's profile characteristics or recently viewed content

顶会论文

- Google
- 用户cookie可能会暴露的情况：
- 重定向功能缺陷：虽然google会自动把用户的http连接请求重定向成https，但是在用户最初请求时，浏览器会使用http协议，并发送cookie。
- 用户在地址输入栏访问网页的情况下，浏览器会发送http cookies。

顶会论文

BROWSER BEHAVIOR FOR USER INPUT IN ADDRESS BAR.

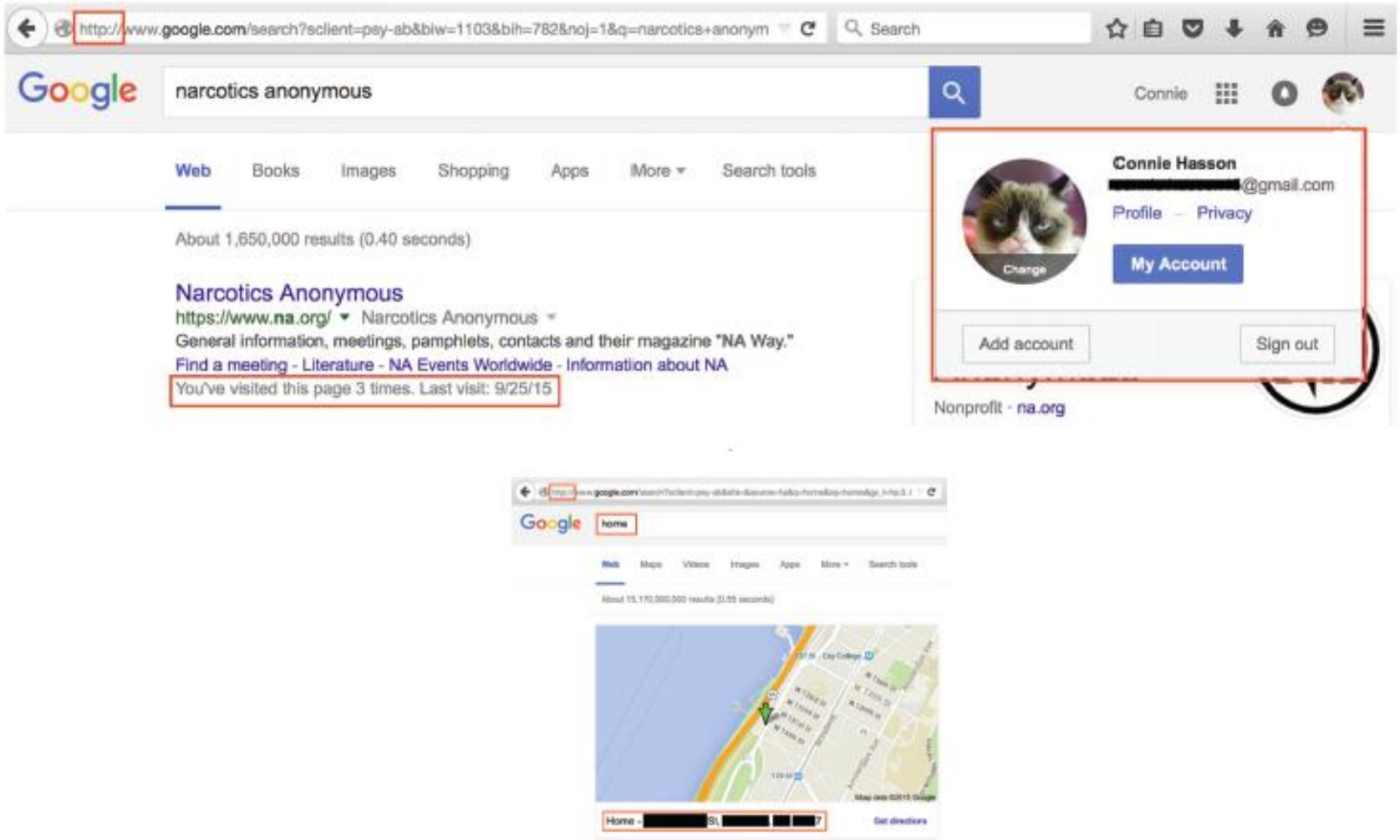
Browser	Connect over HTTP
Desktop	
Chrome (v. 45)	✓
Firefox (v. 41)	✓
Safari (v. 8.0)	✓
Internet Explorer (v. 11)	✓
Opera (v. 32)	✓
Mobile	
Safari (iOS 9)	✓
Chrome (v.46, Android 5.1.1)	✗ (conditionally)

***user input:** {google.com, www.google.com}

顶会论文

- **Google**
- **信息泄露**
  - 账户信息——用户名，谷歌邮箱，头像图片
  - 位置信息——家庭住址，工作单位地址
  - 浏览历史——浏览某网页的次数以及最后访问时间
  - 个性化搜索信息——利用谷歌的搜索优化功能（给用户推荐个性化信息）进一步抽取用户个性化信息
  - 购物网站信息——用户姓名，邮箱句柄，谷歌个人资料
- **可利用进行的攻击——搜索历史污染攻击**

顶会论文



顶会论文

- **Amazon**
- **信息泄露及危害**
  - 账户信息——用户名，邮箱，手机号码，用户姓名以及所在城市，同时攻击者拥有改变用户信息以及提交评论的权限
  - 账户历史记录——了解用户购物偏好
  - 购物车——攻击者可以查看用户购物车或者修改购物车中的物品
  - 邮箱联系人——攻击者可以利用亚马逊提供的邮件转发功能向用户联系人发送垃圾邮件且不易被检测

顶会论文

- **Amazon**
- **信息泄露及危害**
- **勒索诈骗**
  - 攻击者可能会从用户浏览历史中标识出令人尴尬的物品，并给受害者发送邮件，勒索受害者不交赎金就将信息泄露
  - 攻击者给用户发送邮件勒索，并威胁如果不交赎金，就向用户的购物车里添加令人尴尬的物品，并给受害者的好友发送邮件

顶会论文

- **Bing**
- **Yahoo**
- **Baidu**
- **Youtube**
- **Walmart**
- **Ebay**
- **Target**

顶会论文

- **广告网站——信息间接暴露**

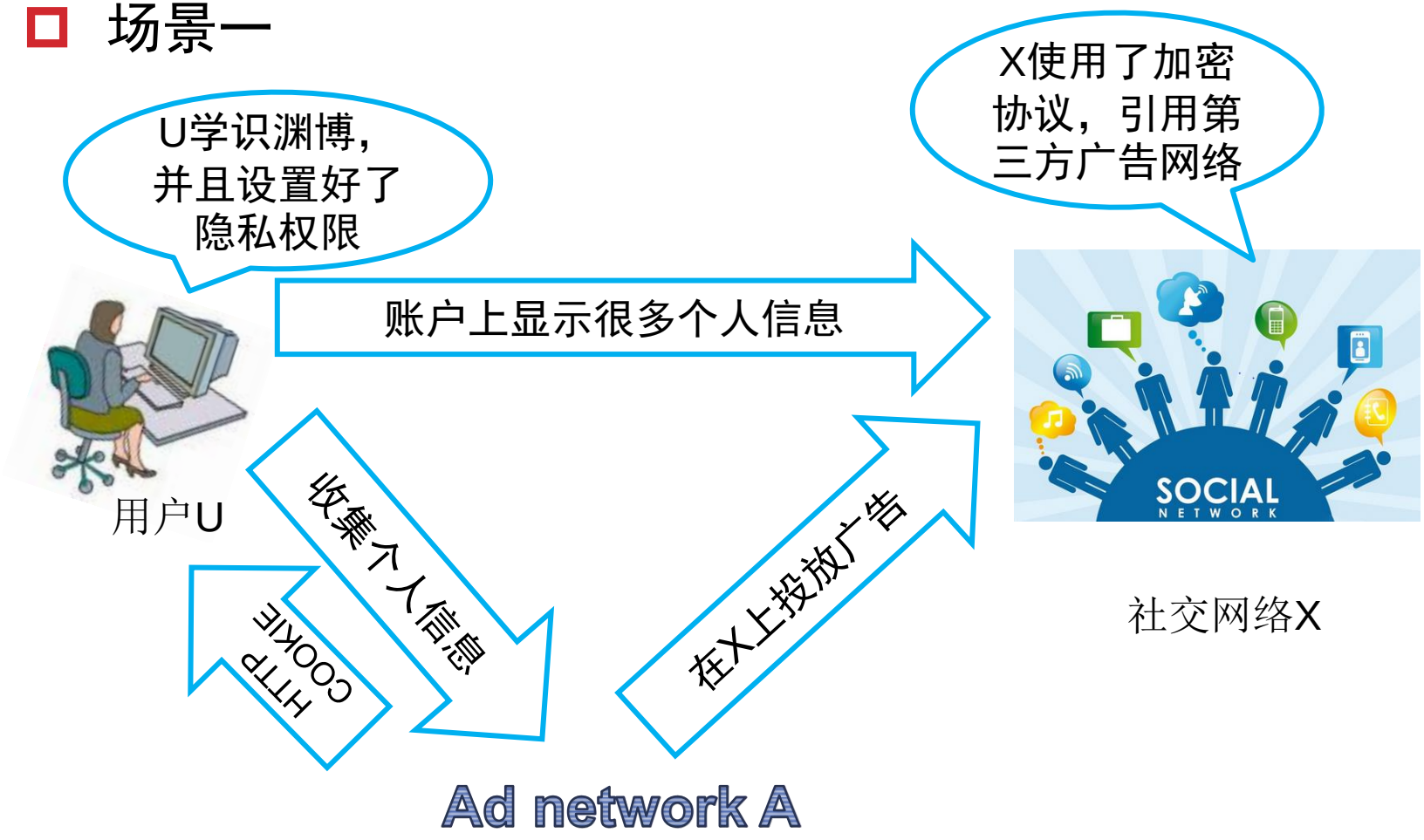
- 广告的投放都是根据个人特征进行定制的，包含很多隐私信息，例如性别、健康情况

- 并非直接某个网站的cookie，而是窃取第三方广告提供商的cookie，并且通过cookie推测敏感信息。

- 有两种场景

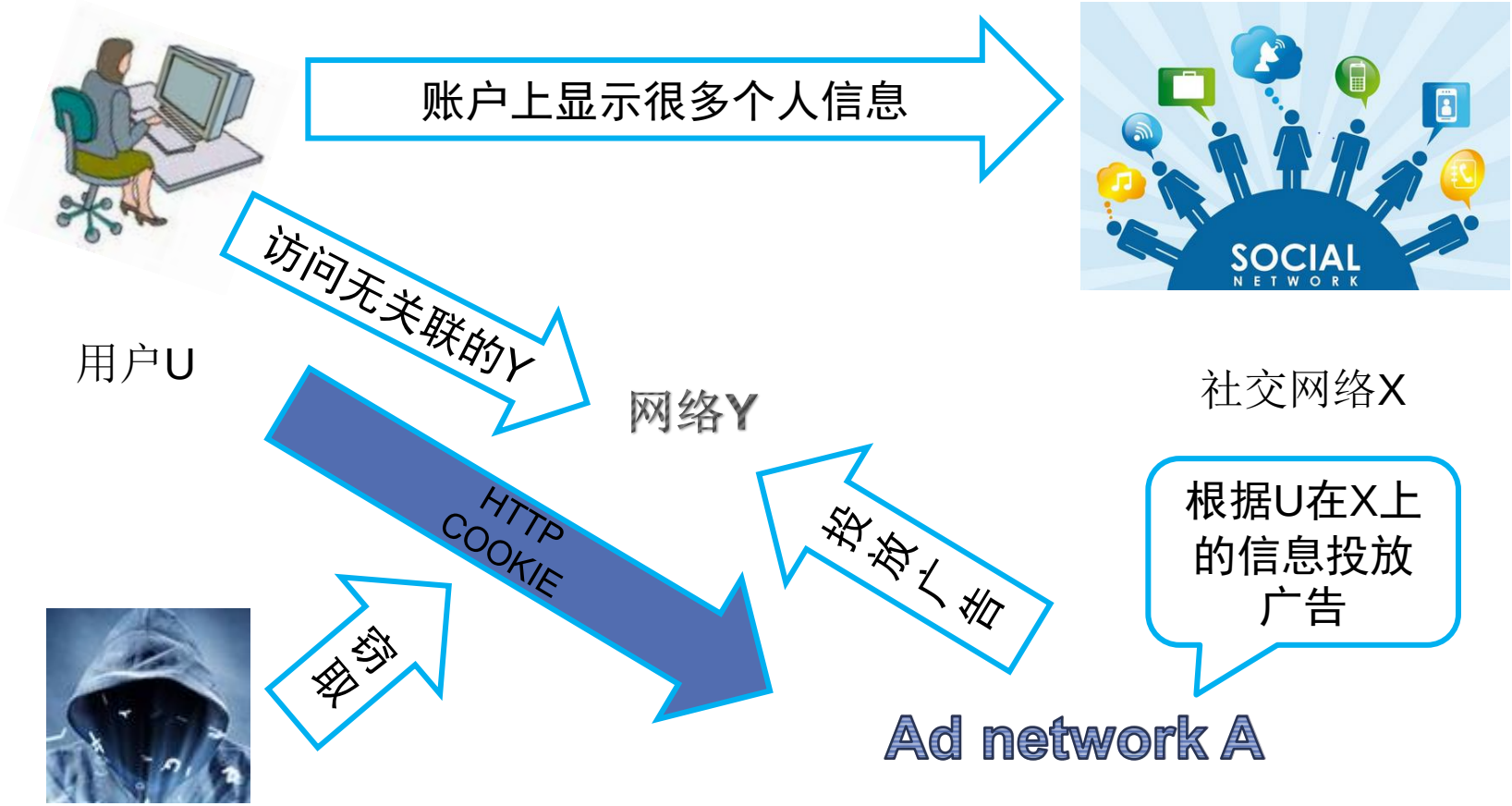
顶会论文

- 场景一



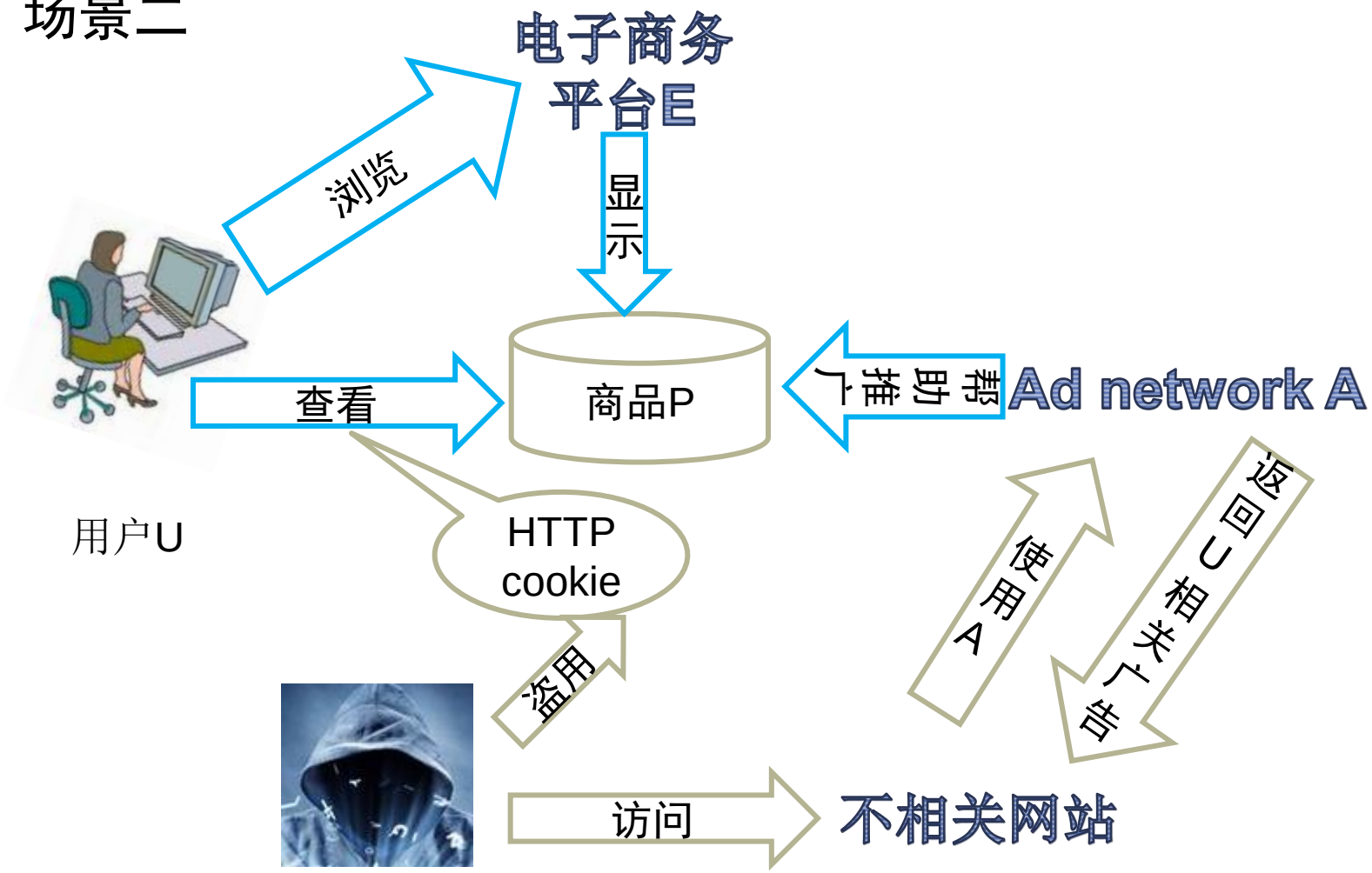
顶会论文

- 场景一



顶会论文

- 场景二


电子商务
平台E
浏览
显示
查看
商品P
广告
Ad network A
用户U
HTTP
cookie
使用A
返回C相关广告
盗用
访问
不相关网站

顶会论文

- XSS
- 大量网站存在XSS劫持cookie的风险

Site	HttpOnly	non—HttpOnly	XSS Hijacking
Amazon	—	x-main	✓
Bing	—	_U, WLS	✓
Baidu	—	BDUSS	✓
CNN	—	CNNid, authid	✓
Doubleclick	—	id	✓
Ebay	—	cid, nonsession	✓
Google	HSID	SID	✗
Guardian	—	GU_U	✓
HuffingtonPost	huffpost_s	huffpost_user huffpost_user_id last_login_username	✗
MSN	MSNRPSAuth	—	✗
New York Times	—	NYT-S	✗
Target	—	WC_PERSISTENT guestDisplayName UserLocation	✓
Walmart	—	customer, CID	✓
Yahoo	F	T, Y	partial
Youtube	VISITOR_INFO1_LIVE	—	✗

顶会论文

- **浏览器扩展泄露cookie**
  - 对3万多的谷歌扩展进行调查，很大一部分要求使用HTTP连接而并非HTTPS
  - 拥有270万次下载量的谷歌词典，也被证实存在cookie劫持的风险
  - 连同两个默认搜索框在一起的火狐的扩展，亦未经加密进行连接
  - 浏览器扩展使得数百万用户的隐私面临危险！

顶会论文

移动设备

- 据思科数据，45%的手机流量是通过WiFi连接的。在另一份报道中表明，72%的人会使用公共WiFi上网
- 劫持到的手机cookie不仅暴露搜索和浏览记录，更会得到个人敏感性息
- iOS系统上的Spotlight，虽然在进行搜索请求时会建立HTTPS连接，但点击“Show more in Bing”按钮，则会进行HTTP连接，这将会暴露用户的HTTP Bing cookie。

顶会论文

- **对策**

- 任何与用户账户和个性化体验相关的服务，必须强制进行加密处理
- 设置cookie中secure标识为true，HttpOnly也应设置好
- 正确部署HSTS（HTTP Strict Transport Security，强制要求使用HTTPS访问）
- 在浏览器上使用HTTPS Everywhere
- 在不可信的公网络上使用VPN

顶会论文

总结

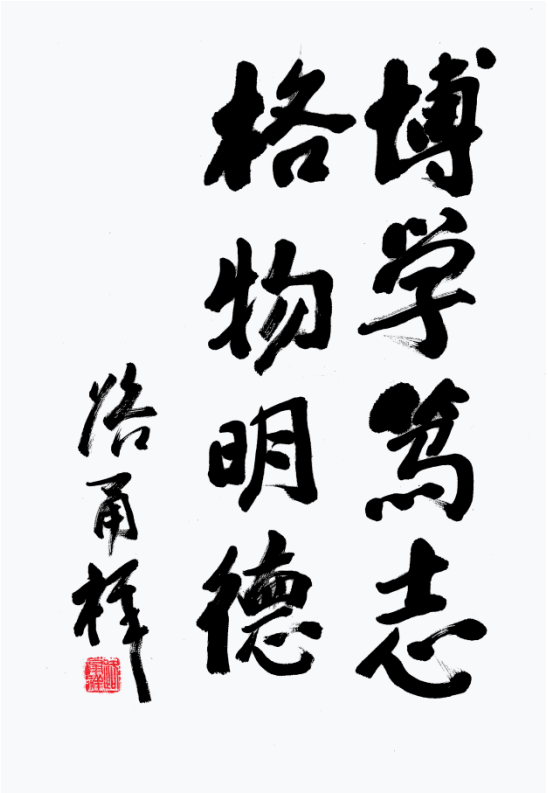
- 论文主要对攻击者盗取cookie造成的用户敏感信息泄露问题进行了深入的研究。
- 论文对当前世界上用户数量众多的主流网站进行了审计，发现cookie劫持攻击并不仅限于个别网站，可用范围非常广泛。研究表面除了存有用户信息的网站，这项威胁同样存在于浏览器扩展，移动app等网络应用。同时评估了Cookie劫持对于Tor用户去匿名化的影响。
- 论文评估了主流浏览器对与安全机制的支持，发现安全机制可以减轻攻击对用户的影响，但如果网站不支持全部加密的协议还是不能够完全保证用户隐私信息安全。

COOKIE及安全问题

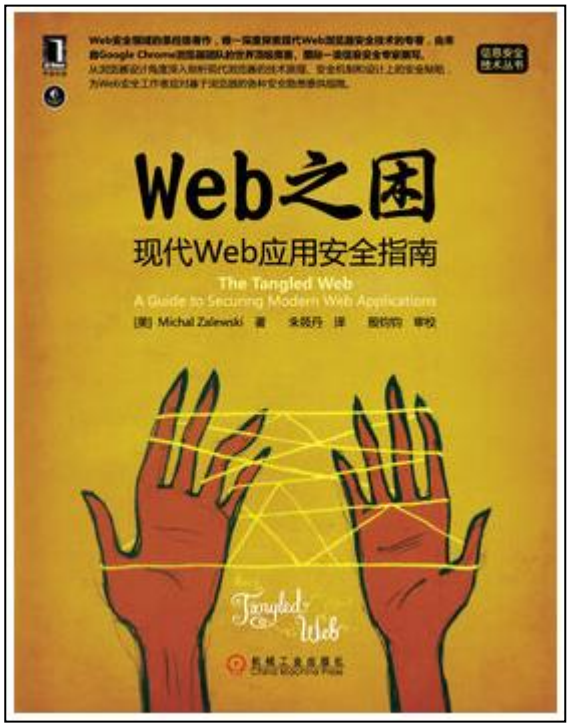
- 小结
- Cookie用于维护会话，极大地提高了交互性和用户体验，同时也带来了安全问题
- Cookie的同源策略并不十分严格，httponly是一种比较成功的安全机制
- Cookie的安全问题仍然十分严重，可能造成严重的用户隐私泄露

本章大纲

- **HTTP及安全问题**
  - 基础知识回顾
  - HTTP安全问题
  - HTTPS协议
- **Cookie及安全问题**
  - 基础知识
  - 安全策略
  - 安全问题
  - 顶会论文



参考文献



The Cracked Cookie Jar: HTTP Cookie Hijacking and the Exposure of Private Information

Suphannee Sivakorn*, Iasonas Polakis* and Angelos D. Keromytis
Department of Computer Science
Columbia University, New York, USA
{suphannee, polakis, angelos}@cs.columbia.edu
*Joint primary authors

***Abstract*—The widespread demand for online privacy, also fueled by widely-publicized demonstrations of session hijacking attacks against popular websites, has spearheaded the increasing deployment of HTTPS. However, many websites still avoid ubiquitous encryption due to performance or compatibility issues. The prevailing approach in these cases is to force critical functionality and sensitive data access over encrypted connections, while allowing more innocuous functionality to be accessed over HTTP. In practice, this approach is prone to flaws that can expose sensitive information or functionality to third parties.**

In this paper, we conduct an in-depth assessment of a diverse set of major websites and explore what functionality and information is exposed to attackers that have hijacked a user's HTTP cookies. We identify a recurring pattern across websites with partially deployed HTTPS; service personalization inadvertently results in the exposure of private information. The separation of functionality across multiple cookies with different scopes and inter-dependencies further complicates matters, as imprecise access control renders restricted account functionality accessible to non-session cookies. Our cookie hijacking study reveals a

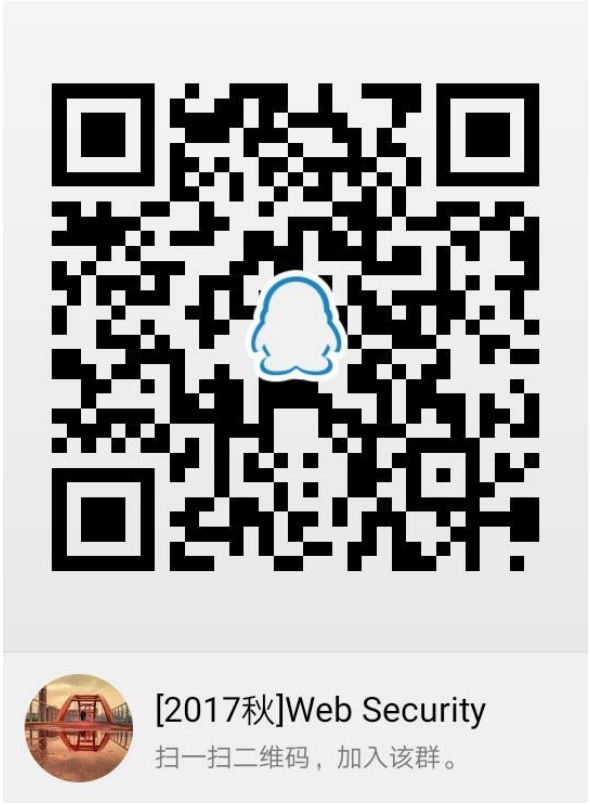
the necessity of securing web connections from prying eyes. The publicity garnered by the Firesheep extension [1], which demonstrated how easily attackers can hijack a user's session, was a catalyst in expediting migration of critical user activity to mandatory HTTPS connections in major services (e.g., transmitting user credentials during the log-in process).

Nonetheless, many major websites continue to serve content over unencrypted connections, which exposes the users' HTTP cookies to attackers monitoring their traffic. Not enforcing ubiquitous encrypted connections may be attributed to various reasons, ranging from potential increases to infrastructure costs and the loss of in-network functionality [2] to maintaining support for legacy clients. If access control policies correctly separated privileges of authenticated (e.g., session cookies) and non-authenticated cookies (e.g., persistent tracking cookies), stolen HTTP cookies would not allow attackers to obtain any personal user information. However, that is not the case

中国科学院大学
University of Chinese Academy of Sciences

后续课程内容

- **第一部分：基础知识**
- **介绍Web安全定义与内涵，国内外现状与趋势、近年来重大网络安全事件等，以及本课程可参考的书籍和网络资源；介绍本课程所需掌握的基础知识，包括HTTP/HTTPS协议、Web前后端编程语言、浏览器安全特性等。**
- 1.3 同源策略
- 1.4 HTTP与Cookie
- 2.1 OWASP Top Ten
- 2.2 XSS与CSRF



谢谢大家

刘潮歌

liuchaoge@iie.ac.cn

中科院信工所 第六研究室

中国科学院大学 University of Chinese Academy of Sciences