Http协议基础

问题：
在上一节课中，请求包与响应包中有一行行的数据，这些数据代表着什么？

学院介绍

学院宗旨：专注网安人才实战技能培养

学院官网：https://edu.hetianlab.com/

合天网安实验室：https://www.hetianlab.com/

主打课程：

《web安全》：OWASP TOP 10漏洞原理及测试

《渗透测试》：渗透测试流程及工具的使用

《安全开发》：用python写一个综合的扫描器

《CTF-PWN》：CTF中的PWN相关

《CTF-WEB》：CTF中WEB相关

目录
CONTENTS

/01 Http介绍

1.1 Http简介

HTTP(HyperText Transfer Protocol): 超文本传输协议。是互联网上应用最广泛的一种网络协议。所有www文件都必须遵守的一个标准，是以 ASCII 码传输，建立在 TCP/IP 协议之上的应用层规范。

1.2 计算机相互之间的通信

互联网的关键技术就是TCP/IP协议。两台计算机之间的通信是通过TCP/IP协议在因特网上进行的。实际上这个是两个协议：

TCP : Transmission Control Protocol 传输控制协议

IP： Internet Protocol 网际协议。

1.2.1 IP：计算机之间的通信

IP协议是计算机用来相互识别的通信的一种机制，每台计算机都有一个IP.用来在internet上标识这台计算机。 IP 负责在因特网上发送和接收数据包。通过 IP，消息（或者其他数据）被分割为小的独立的包，并通过因特网在计算机之间传送。IP 负责将每个包路由至它的目的地。

IP协议仅仅是允许计算机相互发消息，但它并不检查消息是否以发送的次序到达而且没有损坏（只检查关键的头数据）。为了提供消息检验功能，直接在IP协议上设计了传输控制协议TCP.

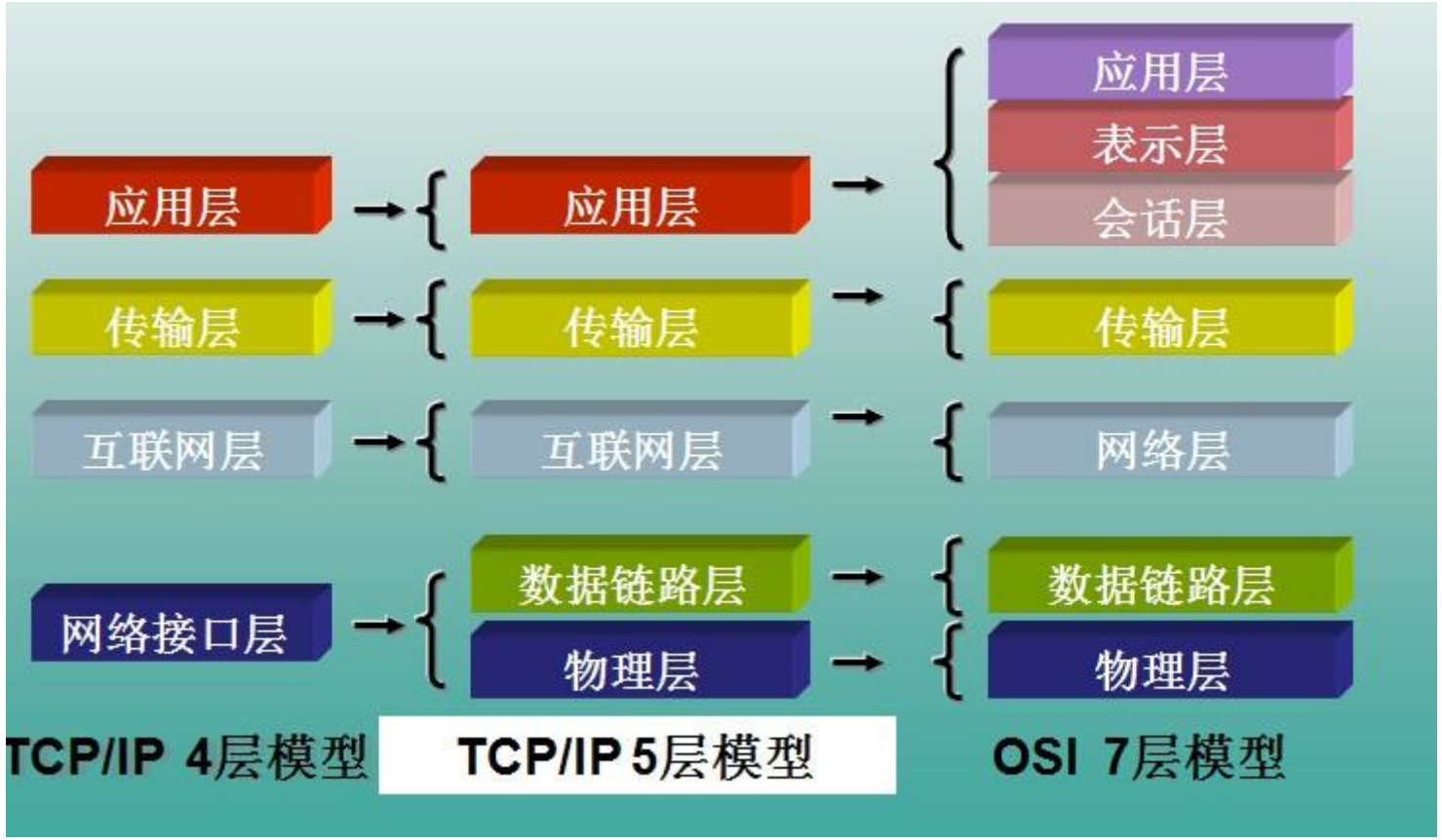
1.2.1 TCP：应用程序之间的通信

TCP确保数据包以正确的次序到达，并且尝试确认数据包的内容没有改变。TCP在IP地址之上引端口（port），它允许计算机通过网络提供各种服务。一些端口号为不同的服务保留，而且这些端口号是众所周知。

服务或者守护进程：在提供服务的机器上，有程序监听特定端口上的通信流。例如大多数电子邮件通信流出现在端口25上，用于wwww的HTTP通信流出现在80端口上。

当应用程序希望通过 TCP 与另一个应用程序通信时，它会发送一个通信请求。这个请求必须被送到一个确切的地址。在双方“握手”之后，TCP 将在两个应用程序之间建立一个全双工 (full-duplex) 的通信，占用两个计算机之间整个的通信线路。TCP 用于从应用程序到网络的数据传输控制。TCP 负责在数据传送之前将它们分割为 IP 包，然后在它们到达的时候将它们重组。

1.3 Http所在的协议层

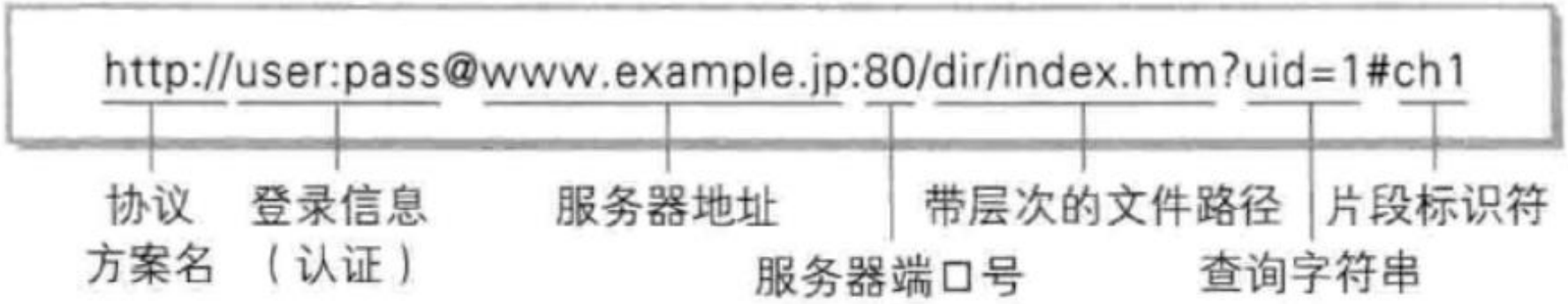


1.4 Http工作过程

域名解析 -> 三次握手 -> 发起HTTP请求 -> 响应HTTP请求并得到HTML代码 -> 浏览器解析HTML代码 -> 浏览器对页面进行渲染呈现给用户

1.5 统一资源定位符

URL（统一资源定位符）：我们常说的网址，包含了用于查找资源的足够的信息，而一个完整的URL包含下面几部分：



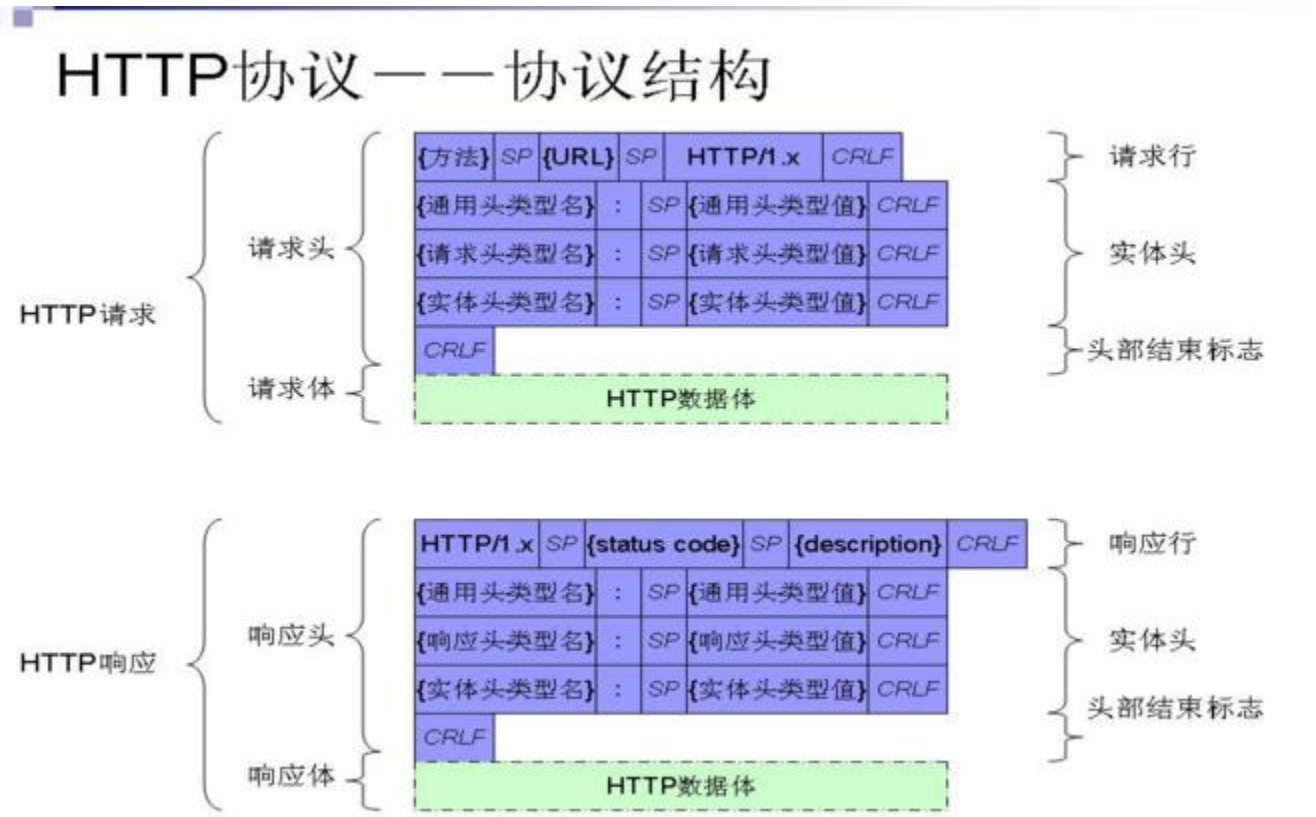
/02 Http消息结构解析

2.1 Http协议结构

无论是请求还是响应，都包括http头部和正文信息。

Http头部：发送的是一些附加信息，如内容类型，服务器发送响应的日期，http状态码。

正文：用户提交的表单数据。



2.2 客户端请求消息


请求方法
空格
URL
空格
协议版本
回车符
换行符
请求行
头部字段名
:
值
回车符
换行符
...
请求头部
回车符
换行符
请求数据


POST /Login.action HTTP/1.1
请求行
Host: hetianlab.com
Content-Length: 316
Accept: application/json, text/javascript, */*; q=0.01
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/81.0.4044.69 Safari/537.36 Edg/81.0.416.34
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Origin: http://hetianlab.com
Referer: http://hetianlab.com/loginLab.do
请求头部字段
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cookie: JSESSIONID=68C618EFCAE4538CAF6A1684E082053E.jvm0; register=; platform=os; __qc_wId=914;
pgv_pvid=3574541960
Connection: close
空行
请求数据
username=12345%40qq.com&password=1f7402fc3e35feb707cf58436bec075893f9593cbe1e7cb3a37af71b8a1de1d7
84763b1e790685daa98f04c02e966af35888dd6a9b90f080a0cd54debf0f2ab7affac0faf39cb6dcb95dd5b6d777b3106
32e3cb79da830e6d8295808e61c9a60cfa0ed3014cbc85923b0dd994511458c30a799acb819e8a6703e3ad7f8da592b&v
alidateCode=&rtnJson=true

2.2.1 请求方法

GET：获取资源

POST：用于向指定资源发送数据，指定的资源会对数据进行处理，然后将处理结果返回给客户端，常用于表单提交、文件上传

方法	描述
GET	请求指定的页面信息，并返回实体主体。
HEAD	类似于get请求，只不过返回的响应中没有具体的内容，用于获取报头
POST	向指定资源提交数据进行处理请求（例如提交表单或者上传文件）。数据被包含在请求体中。POST请求可能会导致新的资源的建立和/或已有资源的修改。
PUT	从客户端向服务器传送的数据取代指定的文档的内容。
DELETE	请求服务器删除指定的页面。
CONNECT	HTTP/1.1协议中预留给能够将连接改为管道方式的代理服务器。
OPTIONS	允许客户端查看服务器的性能。
TRACE	回显服务器收到的请求，主要用于测试或诊断。

2.2.1 请求头部

允许客户端传递关于自身的信息和希望的响应形式。

Header头部	解释	示例
Host	指定请求的服务器的域名和端口号	Host: hetianlab.com
Content-Length	请求的内容长度	Content-Length: 316
Accept	指定客户端能够接收的内容类型	Accept: application/json, text/javascript, */*; q=0.01
X-Requested-With	Ajax 异步请求	X-Requested-With: XMLHttpRequest
User-Agent	包含发出请求的用户信息	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
Content-Type	请求的与实体对应的MIME信息	Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Origin	指明当前请求来自于哪个站点	Origin: http://hetianlab.com
Referer	先前网页的地址，当前请求网页紧随其后	Referer: http://hetianlab.com/loginLab.do
Accept-Encoding	指定浏览器可支持的web服务器返回内容压缩编码类型。	Accept-Encoding: gzip, deflate
Accept-Language	浏览器可接受的语言	Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cookie	HTTP请求发送时，会把保存在该请求域名下的所有cookie值一起发送给web服务器。	Cookie: JSESSIONID=68C618EFCAE4530CAF6A1684E082053E.jvm0
Connection	表示是否需要持久连接	Connection: close

2.2.1 空行、请求数据

空行：

表示请求头结束，请求正文（请求体）开始

请求数据：

GET方法：提交数据时，数据参数会做为URL的一部分放在文件路径后面发送给服务器，被称为查询字符串

POST方法：发送的数据在请求体中

2.3 服务端响应消息


```
HTTP/1.1 200 OK
Server: nginx/1.5.6
Date: Mon, 23 Mar 2020 04:10:20 GMT
Content-Type: application/json;charset=UTF-8
Connection: close
Set-Cookie: JSESSIONID=68C618EFCAE4598CAF6A1684E082053E.jvm0
Content-Length: 77

{"result":"failure","message":"□□□□□□□□","requireCode":false}
```

状态行

消息报头

空行

响应正文

2.3.1 状态码

200 OK //客户端请求成功

302 //url跳转

400 Bad Request //客户端请求有语法错误，不能被服务器所理解

401 Unauthorized //请求未经授权，这个状态代码必须和 WWW-Authenticate 报头域一起使用

403 Forbidden //服务器收到请求，但是拒绝提供服务

404 Not Found //请求资源不存在，eg：输入了错误的URL

500 Internal Server Error //服务器发生不可预期的错误

503 Server Unavailable //服务器当前不能处理客户端的请求，一段时间后可能恢复正常

2.3.2 响应头部

Header头部	解释	示例
Server	web服务器软件名称	Server: nginx/1.5.6
Data	原始服务器消息发出的时间	Date: Mon, 23 Mar 2020 03:57:16 GMT
Content-Type	返回内容的MIME类型	Content-Type: text/html
Set-Cookie	设置Http Cookie	Set-Cookie: JSESSIONID=68C6…053E.jvm0; Path=/; HttpOnly
Content-Length	响应体的长度	Content-Length: 12
ETag	请求变量的实体标签的当前值	ETag: W/"91992-1583487900000"
refresh	应用于重定向或一个新的资源被创造	Refresh: 5; url= http://www.zcmhi.com/archives/94.html
WWW-Authenticate	表明客户端请求实体应该使用的授权方案	WWW-Authenticate: Basic
Allow	对某网络资源的有效的请求行为，不允许则返回405	Allow: GET, HEAD
Content-Type	返回内容的MIME类型	Content-Type: application/json;charset=UTF-8
Location	用来重定向接收方到非请求URL的位置来完成请求或标识新的资源	Location: http://hetianlab.com/loginLab.do

2.3.3 空行、响应数据

空行：

表示响应头结束，响应正文开始

响应数据：

服务器返回的资源内容

{"result":"success","message":null}

/03 Cookie与Session

3.1 Cookie简介

HTTP是一个基于请求与响应，无状态的，应用层的协议。

cookie 是一种发送到客户浏览器的文本串句柄，并保存在客户机硬盘上，可以用来在某个WEB站点会话间持久的保持数据。

3.2 Cookie特点

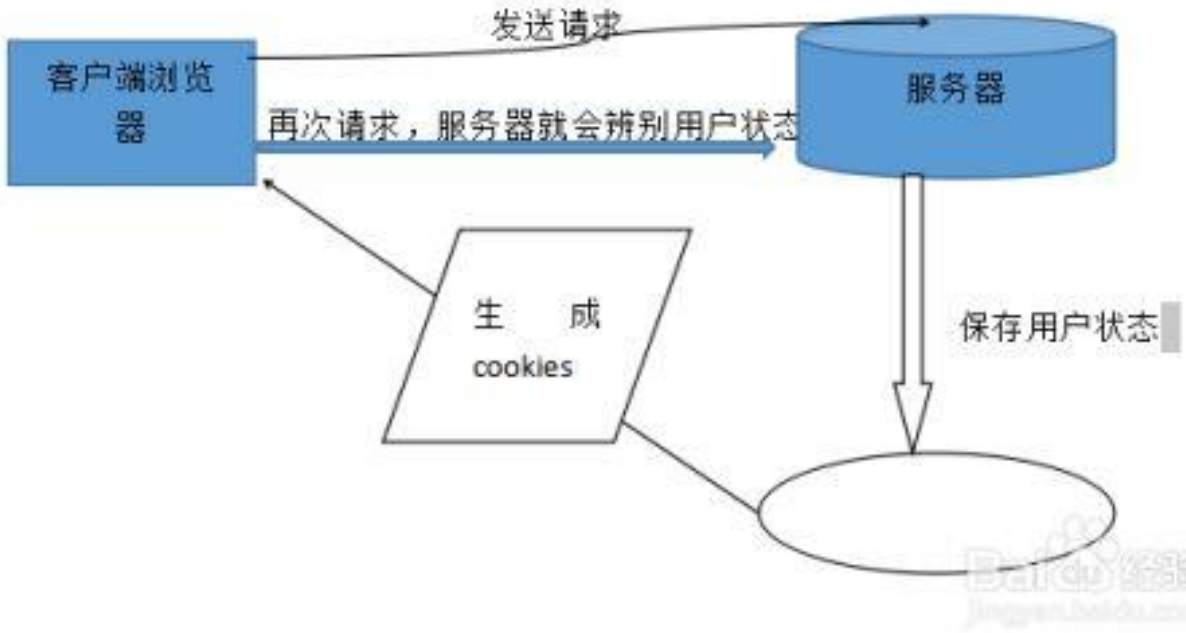
Cookie总是保存在客户端中，按在客户端中的存储位置，可分为内存Cookie和硬盘Cookie：

内存Cookie由浏览器维护，保存在内存中，浏览器关闭后就消失了，其存在时间是短暂的。

硬盘Cookie保存在硬盘里，有一个过期时间，除非用户手工清理或到了过期时间，硬盘Cookie不会被删除，其存在时间是长期的。

3.3 Cookie工作原理

首先当我们访问某个网站时，服务器首先根据浏览器的编号生成一个cookie 返回给客户端。客户端下次再访问时就会将自己本地的cookie 加上url访问地址一同给服务器。服务器读出来以此来辨别用户的状态。



3.4 Session简介

Session代表服务器与浏览器的一次会话过程，这个过程是连续的，也可以时断时续的。Session是一种服务器端的机制，Session 对象用来存储特定用户会话所需的信息。

Session由服务端生成，保存在服务器的内存、缓存、硬盘或数据库中。

3.5 Session工作原理

当用户访问到一个服务器，如果服务器启用Session，服务器就要为该用户创建一个SESSION，并生成一个与此SESSION相关的SESSION ID，这个SESSION ID是唯一的、不重复的、不容易找到规律的字符串，这个SESSION ID将被在本次响应中返回到客户端保存，而保存这个SESSION ID的正是COOKIE，这样在交互过程中浏览器可以自动的按照规则把这个标识发送给服务器。

3.1 Cookie与Session的区别

1、存放位置不同

Cookie保存在客户端，

Session保存在服务端。

2 、存取方式的不同

Cookie中只能保管ASCII字符串，假如需求存取Unicode字符或者二进制数据，需求先进行编码。

Session中能够存取任何类型的数据

3、安全性不同

Cookie存储在浏览器中，对客户端是可见的，客户端的一些程序可能会窥探、复制以至修正Cookie中的内容。

Session存储在服务器上，对客户端是透明的，不存在敏感信息泄露的风险。

感谢您的聆听

学习工具、资料及课程回放

扫码免费领取