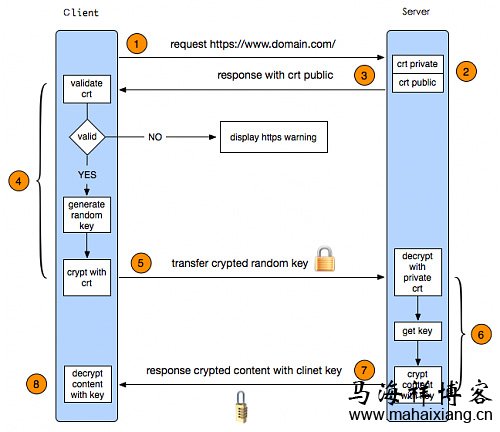
Http和Https的区别

超文本传输协议HTTP协议被用于在Web浏览器和网站服务器之间传递信息，HTTP协议以明文方式发送内容，不提供任何方式的数据加密，如果攻击者截取了Web浏览器和网站服务器之间的传输报文，就可以直接读懂其中的信息，因此，HTTP协议不适合传输一些敏感信息，比如：信用卡号、密码等支付信息。

为了解决HTTP协议的这一缺陷，需要使用另一种协议：安全套接字层超文本传输协议HTTPS，为了数据传输的安全，HTTPS在HTTP的基础上加入了SSL协议，SSL依靠证书来验证服务器的身份，并为浏览器和服务器之间的通信加密。

* Http与Https有什么区别？

HTTP协议传输的数据都是未加密的，也就是明文的，因此使用HTTP协议传输隐私信息非常不安全，为了保证这些隐私数据能加密传输，于是网景公司设计了SSL（Secure Sockets Layer）协议用于对HTTP协议传输的数据进行加密，从而就诞生了HTTPS。、

简单来说，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。

HTTPS和HTTP的区别主要如下：

1、https协议需要到ca申请证书，一般免费证书较少，因而需要一定费用。

2、http是超文本传输协议，信息是明文传输，https则是具有安全性的ssl加密传输协议。

3、http和https使用的是完全不同的连接方式，用的端口也不一样，前者是80，后者是443。

4、http的连接很简单，是无状态的；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，比http协议安全。

* Https的工作原理

HTTPS能够加密信息，以免敏感信息被第三方获取，所以很多银行网站或电子邮箱等等安全级别较高的服务都会采用HTTPS协议。



**客户端发起HTTPS请求**

这个没什么好说的，就是用户在浏览器里输入一个https网址，然后连接到server的443端口。

**2、服务端的配置**

采用HTTPS协议的服务器必须要有一套数字证书，可以自己制作，也可以向组织申请，区别就是自己颁发的证书需要客户端验证通过，才可以继续访问，而使用受信任的公司申请的证书则不会弹出提示页面(startssl就是个不错的选择，有1年的免费服务)。

这套证书其实就是一对公钥和私钥，如果对公钥和私钥不太理解，可以想象成一把钥匙和一个锁头，只是全世界只有你一个人有这把钥匙，你可以把锁头给别人，别人可以用这个锁把重要的东西锁起来，然后发给你，因为只有你一个人有这把钥匙，所以只有你才能看到被这把锁锁起来的东西。

**3、传送证书**

这个证书其实就是公钥，只是包含了很多信息，如证书的颁发机构，过期时间等等。

**4、客户端解析证书**

这部分工作是有客户端的TLS来完成的，首先会验证公钥是否有效，比如颁发机构，过期时间等等，如果发现异常，则会弹出一个警告框，提示证书存在问题。

如果证书没有问题，那么就生成一个随机值，然后用证书对该随机值进行加密，就好像上面说的，把随机值用锁头锁起来，这样除非有钥匙，不然看不到被锁住的内容。

**5、传送加密信息**

这部分传送的是用证书加密后的随机值，目的就是让服务端得到这个随机值，以后客户端和服务端的通信就可以通过这个随机值来进行加密解密了。

**6、服务端解密信息**

服务端用私钥解密后，得到了客户端传过来的随机值(私钥)，然后把内容通过该值进行对称加密，所谓对称加密就是，将信息和私钥通过某种算法混合在一起，这样除非知道私钥，不然无法获取内容，而正好客户端和服务端都知道这个私钥，所以只要加密算法够彪悍，私钥够复杂，数据就够安全。

**7、传输加密后的信息**

这部分信息是服务端用私钥加密后的信息，可以在客户端被还原。

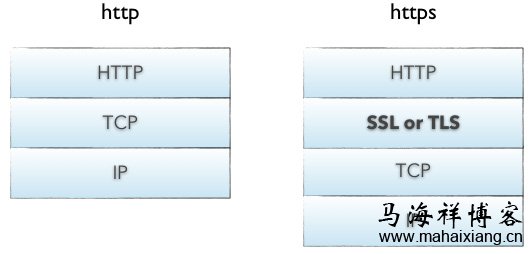
**8、客户端解密信息**

客户端用之前生成的私钥解密服务段传过来的信息，于是获取了解密后的内容，整个过程第三方即使监听到了数据，也束手无策。

//Todo：发起Http请求的几种方法

* Https要比HTTP多用多少服务器资源？

HTTPS其实就是建构在SSL/TLS之上的 HTTP协议，所以，要比较HTTPS比HTTP多用多少服务器资源，主要看SSL/TLS本身消耗多少服务器资源。



HTTP使用TCP三次握手建立连接，客户端和服务器需要交换3个包（具体可查看博客《[HTTP服务的七层架构技术解析及运用](http://www.mahaixiang.cn/internet/801.html)》的相关介绍）；HTTPS除了TCP的三个包，还要加上ssl握手需要的9个包，所以一共是12个包。

HTTP建立连接，按照下面链接中针对Computer Science House的测试，是114毫秒；HTTPS建立连接，耗费436毫秒，ssl部分花费322毫秒，包括网络延时和ssl本身加解密的开销（服务器根据客户端的信息确定是否需要生成新的主密钥；服务器回复该主密钥，并返回给客户端一个用主密钥认证的信息；服务器向客户端请求数字签名和公开密钥）。

当SSL连接建立后，之后的加密方式就变成了3DES等对于CPU负荷较轻的对称加密方式，相对前面SSL建立连接时的非对称加密方式，对称加密方式对CPU的负荷基本可以忽略不记，所以问题就来了，如果频繁的重建ssl的session，对于服务器性能的影响将会是致命的，尽管打开HTTPS保活可以缓解单个连接的性能问题，但是对于并发访问用户数极多的大型网站，基于负荷分担的独立的SSL termination proxy就显得必不可少了，Web服务放在SSL termination proxy之后，SSL termination proxy既可以是基于硬件的，譬如F5；也可以是基于软件的，譬如维基百科用到的就是Nginx。

* **HTTPS的优点**

正是由于HTTPS非常的安全，攻击者无法从中找到下手的地方，从站长的角度来说，HTTPS的优点有以下2点：

**1、SEO方面(搜索引擎算法)**

谷歌曾在2014年8月份调整搜索引擎算法，并称“比起同等HTTP网站，采用HTTPS加密的网站在搜索结果中的排名将会更高”。

**2、安全性**

尽管HTTPS并非绝对安全，掌握根证书的机构、掌握加密算法的组织同样可以进行中间人形式的攻击，但HTTPS仍是现行架构下最安全的解决方案，主要有以下几个好处：

（1）、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；

（2）、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。

（3）、HTTPS是现行架构下最安全的解决方案，虽然不是绝对安全，但它大幅增加了中间人攻击的成本。

**HTTPS的缺点**

虽然说HTTPS有很大的优势，但其相对来说，还是有些不足之处的，具体来说，有以下2点：

**1、SEO方面**

据ACM CoNEXT数据显示，使用HTTPS协议会使页面的加载时间延长近50%，增加10%到20%的耗电，此外，HTTPS协议还会影响缓存，增加数据开销和功耗，甚至已有安全措施也会受到影响也会因此而受到影响。

而且HTTPS协议的加密范围也比较有限，在黑客攻击、拒绝服务攻击、服务器劫持等方面几乎起不到什么作用。

最关键的，SSL证书的信用链体系并不安全，特别是在某些国家可以控制CA根证书的情况下，中间人攻击一样可行。

**2、经济方面**

（1）、SSL证书需要钱，功能越强大的证书费用越高，个人网站、小网站没有必要一般不会用。

（2）、SSL证书通常需要绑定IP，不能在同一IP上绑定多个域名，IPv4资源不可能支撑这个消耗（SSL有扩展可以部分解决这个问题，但是比较麻烦，而且要求浏览器、操作系统支持，Windows XP就不支持这个扩展，考虑到XP的装机量，这个特性几乎没用）。

（3）、HTTPS连接缓存不如HTTP高效，大流量网站如非必要也不会采用，流量成本太高。

（4）、HTTPS连接服务器端资源占用高很多，支持访客稍多的网站需要投入更大的成本，如果全部采用HTTPS，基于大部分计算资源闲置的假设的VPS的平均成本会上去。

（5）、HTTPS协议握手阶段比较费时，对网站的相应速度有负面影响，如非必要，没有理由牺牲用户体验。

**站长如何搭建HTTPS站点？**

说到HTTPS站点的搭建，就不得不提到SSL协议，SSL是Netscape公司率先采用的网络安全协议，它是在传输通信协议（TCP/IP）上实现的一种安全协议，采用公开密钥技术，SSL广泛支持各种类型的网络，同时提供三种基本的安全服务，它们都使用公开密钥技术。

**1、SSL的作用**

（1）、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；

（2）、加密数据以防止数据中途被窃取；

（3）、维护数据的完整性，确保数据在传输过程中不被改变。

而SSL证书指的是在SSL通信中验证通信双方身份的数字文件，一般分为服务器证书和客户端证书，我们通常说的SSL证书主要指服务器证书，SSL证书由受信任的数字证书颁发机构CA（如VeriSign，GlobalSign，WoSign等），在验证服务器身份后颁发，具有服务器身份验证和数据传输加密功能，分为扩展验证型(EV)SSL证书、组织验证型(OV)SSL证书、和域名验证型（DV）SSL证书。

**2、SSL证书申请的3个主要步骤**

对于SSL证书的申请，主要有以下3个步骤：

**（1）、制作CSR文件**

所谓CSR就是由申请人制作的Certificate Secure Request证书请求文件，制作过程中，系统会产生2个密钥，一个是公钥就是这个CSR文件；另外一个是私钥，存放在服务器上。

要制作CSR文件，申请人可以参考WEB SERVER的文档，一般APACHE等，使用OPENSSL命令行来生成KEY+CSR2个文件，Tomcat，JBoss，Resin等使用KEYTOOL来生成JKS和CSR文件，IIS通过向导建立一个挂起的请求和一个CSR文件。

**（2）、CA认证**

将CSR提交给CA，CA一般有2种认证方式：

①、域名认证：一般通过对管理员邮箱认证的方式，这种方式认证速度快，但是签发的证书中没有企业的名称。

②、企业文档认证：需要提供企业的营业执照，一般需要3-5个工作日。

也有需要同时认证以上2种方式的证书，叫EV证书，这种证书可以使IE7以上的浏览器地址栏变成绿色，所以认证也最严格。

**（3）、证书的安装**

在收到CA的证书后，可以将证书部署上服务器，一般APACHE文件直接将KEY+CER复制到文件上，然后修改HTTPD.CONF文件；TOMCAT等，需要将CA签发的证书CER文件导入JKS文件后，复制上服务器，然后修改SERVER.XML；IIS需要处理挂起的请求，将CER文件导入。

获取Code

1. JavaScript中的定义

var box;//定义变量，但没有对它进行初始化(没有赋值,但变量的值为undefined)。

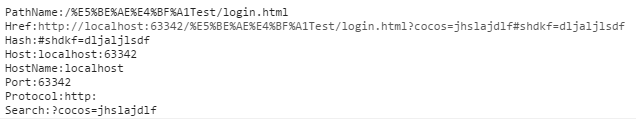
1. 获取JS当前页面的URL方法:

window.location对象下的属性

属性                  描述  
hash                设置或获取 href 属性中在井号“#”后面的分段，注意包含‘#’。  
host                 设置或获取 location 或 URL 的 hostname 和 port 号码。  
hostname 设置或获取 location 或 URL 的主机名称部分。  
href                  设置或获取整个 URL 为字符串。  
pathname      设置或获取对象指定的文件名或路径。  
port                  设置或获取与 URL 关联的端口号码。  
protocol          设置或获取 URL 的协议部分。  
search            设置或获取 href 属性中跟在问号后面的部分，注意包含‘？’。

示例:

URL为（http://localhost:63342/%E5%BE%AE%E4%BF%A1Test/login.html?cocos=jhslajdlf#shdkf=dljaljlsdf）



1. URL传参数

如果仅仅只有一个参数，可以使用window.location对象下的hash和search对象，直接在url的后面添加“？”或者“#”即可。

否则的话，可以在URL的后面添加的”code=ghjkhk&id=hdsj”等方式传递参数。

注意:

1. 每个浏览器对URL的长度限制不同;

* IE浏览器对URL的长度现限制为2048字节(自己测试最多为2047字节)。
* 360极速浏览器对URL的长度限制为2118字节。
* Firefox(Browser)对URL的长度限制为65536字节。
* Safari(Browser)对URL的长度限制为80000字节。
* Opera(Browser)对URL的长度限制为190000字节。
* Google(chrome)对URL的长度限制为8182字节。

1. 在URL中，一个汉字通过不同的编码方式的长度也不一致。

* 优点:

1. URL地址法简洁易用，可同时传递多个字符型参数；
2. URL地址法可以很方便的在页面之间切换并传递参数，无需额外的处理，基于正常情况比较不会性能损失；

* 缺点:

1. URL传递参数长度受限，最大为2K；
2. URL只能传递字符型参数；
3. 信息泄露：URL地址在客户端可见，导致会泄露信息，可能被黑客用来攻击系统。

* JS解析的几种方法:

1. 正则分析法:

*//获取连接中的Code***function** *getQueryString*(name) {  
 **var** reg = **new** RegExp(**"(^|&)"** + name + **"=([^&]\*)(&|$)"**, **"i"**);  
 **var** r = **window**.**location**.search.substr(1).match(reg);  
 **if** (r != **null**) **return** unescape(r[2]); **return null**;  
}

//使用

**var** access\_code= *getQueryString*(**'code'**);

1. location.search和for取值:

* 方法一:for

function GetRequest() {

   var url = location.search; //获取url中"?"符后的字串

   var theRequest = new Object();

   if (url.indexOf("?") != -1) {

      var str = url.substr(1);

      strs = str.split("&");

      for(var i = 0; i **<** **strs.length**; i ++) {

         theRequest[strs[i].split("=")[0]]=unescape(strs[i].split("=")[1]);

      }

   }

   return theRequest;

}

* 方法二:while

/\*\*

 \* 获取指定的URL参数值

 \* URL:http://www.quwan.com/index?name=tyler

 \* 参数：paramName URL参数

 \* 调用方法:getParam("name")

 \* 返回值:tyler

 \*/

function getParam(paramName) {

    paramValue = "", isFound = !1;

    if (this.location.search.indexOf("?") == 0 && this.location.search.indexOf("=") **>** 1) {

        arrSource = unescape(this.location.search).substring(1, this.location.search.length).split("&"), i = 0;

        while (i **<** **arrSource.length** && !isFound) arrSource[i].indexOf("=") **>** 0 && arrSource[i].split("=")[0].toLowerCase() == paramName.toLowerCase() && (paramValue = arrSource[i].split("=")[1], isFound = !0), i++

    }

    return paramValue == "" && (paramValue = null), paramValue

}

1. Json字符串和Json对象4种互转方式:
   * **jQuery插件支持的转换方式**：

$.parseJSON( jsonstr );

或者jQuery.parseJSON(jsonstr)

也可以将json字符串转换成json对象

* + **浏览器支持的转换方式**(Firefox，chrome，opera，safari，ie9，ie8)等浏览器:

JSON.parse(jsonstr); //可以将json字符串转换成json对象   
JSON.stringify(jsonobj); //可以将json对象转换成json对符串

注：ie8(兼容模式),ie7和ie6没有JSON对象，推荐采用JSON官方的方式，引入json.js,即也就是方法4。

* + JavaScript支持的转换格式:

eval('(' + jsonstr + ')');

可以将json字符串转换成json对象,注意需要在json字符外包裹一对小括号   
注：ie8(兼容模式),ie7和ie6也可以使用eval()将字符串转为JSON对象，但不推荐这些方式，这种方式不安全eval会执行json串中的表达式。

* + JSON官方的转换方式:

http://www.json.org/提供了一个json.js,这样ie8(兼容模式),ie7和ie6就可以支持JSON对象以及其stringify()和parse()方法；   
可以在https://github.com/douglascrockford/JSON-js上获取到这个js，一般现在用json2.js。

1. 操作Object对象:

* 打印Object对象:

function writeObj(obj){

 var description = "";

 for(var i in obj){

 var property=obj[i];

 description+=i+" = "+property+"\n";

 }

 alert(description);

}

* 将Object对象转换为string字符串:

function obj2string(o){

 var r=[];

 if(typeof o=="string"){

 return "\""+o.replace(/([\'\"\\])/g,"\\$1").replace(/(\n)/g,"\\n").replace(/(\r)/g,"\\r").replace(/(\t)/g,"\\t")+"\"";

 }

 if(typeof o=="object"){

 if(!o.sort){

  for(var i in o){

  r.push(i+":"+obj2string(o[i]));

  }

  if(!!document.all&&!/^\n?function\s\*toString\(\)\s\*\{\n?\s\*\[native code\]\n?\s\*\}\n?\s\*$/.test(o.toString)){

  r.push("toString:"+o.toString.toString());

  }

  r="{"+r.join()+"}";

 }else{

  for(var i=0;i<o.length;i++){

  r.push(obj2string(o[i]))

  }

  r="["+r.join()+"]";

 }

 return r;

 }

 return o.toString();

}

1. 字符串编码:

JavaScript中有三个可以对字符串编码的函数，分别是:

escape、encodeURI、encodeURIComponent

相应3个解码函数是:

unescape,decodeURI,decodeURIComponent 。

* escape()函数:
  + - 定义和用法   
      escape() 函数可对字符串进行编码，这样就可以在所有的计算机上读取该字符串。
    - 语法   
      escape(string)
    - 参数  描述    
      string  必需。要被转义或编码的字符串。
    - 返回值   
      已编码的 string 的副本。其中某些字符被替换成了十六进制的转义序列。
    - 说明   
      该方法不会对 ASCII 字母和数字进行编码，也不会对下面这些 ASCII 标点符号进行编码： - \_ . ! ~ \* ' ( ) 。其他所有的字符都会被转义序列替换。
* encodeURI()函数:
  + - 定义和用法   
      encodeURI() 函数可把字符串作为 URI 进行编码。
    - 语法   
      encodeURI(URIstring)
    - 参数  描述    
      URIstring  必需。一个字符串，含有 URI 或其他要编码的文本。
    - 返回值   
      URIstring 的副本，其中的某些字符将被十六进制的转义序列进行替换。
    - 说明   
      该方法不会对 ASCII 字母和数字进行编码，也不会对这些 ASCII 标点符号进行编码： - \_ . ! ~ \* ' ( ) 。

该方法的目的是对 URI 进行完整的编码，因此对以下在 URI 中具有特殊含义的 ASCII 标点符号，encodeURI() 函数是不会进行转义的：;/?:@&=+$,#

* encodeURIComponent()函数:
  + - 定义和用法   
      encodeURIComponent() 函数可把字符串作为 URI 组件进行编码。
    - 语法   
      encodeURIComponent(URIstring)
    - 参数  描述    
      URIstring  必需。一个字符串，含有 URI 组件或其他要编码的文本。
    - 返回值   
      URIstring 的副本，其中的某些字符将被十六进制的转义序列进行替换。
    - 说明   
      该方法不会对 ASCII 字母和数字进行编码，也不会对这些 ASCII 标点符号进行编码： - \_ . ! ~ \* ' ( ) 。

其他字符（比如 ：;/?:@&=+$,# 这些用于分隔 URI 组件的标点符号），都是由一个或多个十六进制的转义序列替换的。

* + - 提示和注释   
       提示：请注意 encodeURIComponent() 函数 与 encodeURI() 函数的区别之处，前者假定它的参数是 URI 的一部分（比如协议、主机名、路径或查询字符串）。因此 encodeURIComponent() 函数将转义用于分隔 URI 各个部分的标点符号。
* 总结:

通过对三个函数的分析，我们可以知道：

escape()除了 ASCII 字母、数字和特定的符号外，对传进来的字符串全部进行转义编码，因此如果想对URL编码，最好不要使用此方法。

而encodeURI() 用于编码整个URI,因为URI中的合法字符都不会被编码转换。

encodeURIComponent方法在编码单个URIComponent（指请求参 数）应当是最常用的，它可以讲参数中的中文、特殊字符进行转义，而不会影响整个URL。

1. 通过拼接的字符串转换为函数名，调用函数:

*//读取微信支付消息***function** *read100001*(nmBaseMessage){  
 **var** dataMap= **new** *Map*();  
 *//存放消息类型和消息名* dataMap.put(**"messageType"**, ***WEBCHAT\_GET\_LOGIN***+ 1);  
 dataMap.put(**"messageName"**, **"WebChatLogin"**);  
  
 dataMap.put(**"appID"**, nmBaseMessage.readUTF16());*//公众号ID* **return** dataMap;  
}

*//拼接字符串，转换为函数名。例如：消息ID为100001***var** funcName= **"read"**+ nmBaseMessage.getMsgType();  
*//调用在Global.js中自定义的函数read100001，用于读取不同消息中的用户数据*eval(funcName)(nmBaseMessage);

1. 去除字符串中的空格

/\*\*\*\*\*\*\*\*\*\*\*自定义实现方法\*\*\*\*\*\*\*\*\*\*\*\*\*/

//删除左右两端的空格

function trim(str){

    return str.replace(/(^\s\*)|(\s\*$)/g, "");

}

//删除左边的空格

function ltrim(str){

   return str.replace(/(^\s\*)/g,"");

}

 //删除右边的空格

function rtrim(str){

　return str.replace(/(\s\*$)/g,"");

}

使用方法:trim(str);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*JavaScript中的自定义函数\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

str.trim();//删除左右两端的空格

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*JQuery中的自定义函数\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

$.trim(str);

其中，$ == JQuery不过需要注意的是，有时引入的库文件太多，$存在被其他库污染的危险，因此要注意使用。

1. 判断是否为微信浏览器？

//JS实现

*//是否为微信浏览器***function** *isWebChatBrowser*(){  
 **var** ua = **window**.**navigator**.**userAgent**.toLowerCase();  
 **if**(ua.match(/MicroMessenger/i) == **'micromessenger'**){  
 **return true**;  
 }**else**{  
 **return false**;  
 }  
}

1. 更新标题

*//通过document对象访问title***document**.**title**= **"Document-Success!"**;  
*//通过DOM方式访问title***document**.getElementsByTagName(**'title'**)[0].**innerHTML**= **"DOM-Success!"**;

*//在微信浏览器上，比较特殊一些。*

有人说在微信浏览器上，更新标题不成功，但是自己测试上述两种方法，可以正常改动。反倒是，网上提供的下列方法，在安卓上正常，但是在苹果手机上，页面上，就会出现一部分边框。

因此不推荐:

*//更新页面Title***function** *updateTitle*()  
{  
 **document**.**title**= **"登陆成功—同趣同趣游戏"**;  
 **var** $body = **$**(**'body'**);  
 **var** $iframe = **$**(**'<iframe></iframe>'**);  
 $iframe.on(**'load'**,**function**() {  
 setTimeout(**function**() {  
 $iframe.off(**'load'**).remove();  
 }, 0);  
 }).**appendTo**($body);  
}

微信授权

微信登录

微信分享