CTF_WriteUp_HTTP基本认证（Basic access authentication）

原创

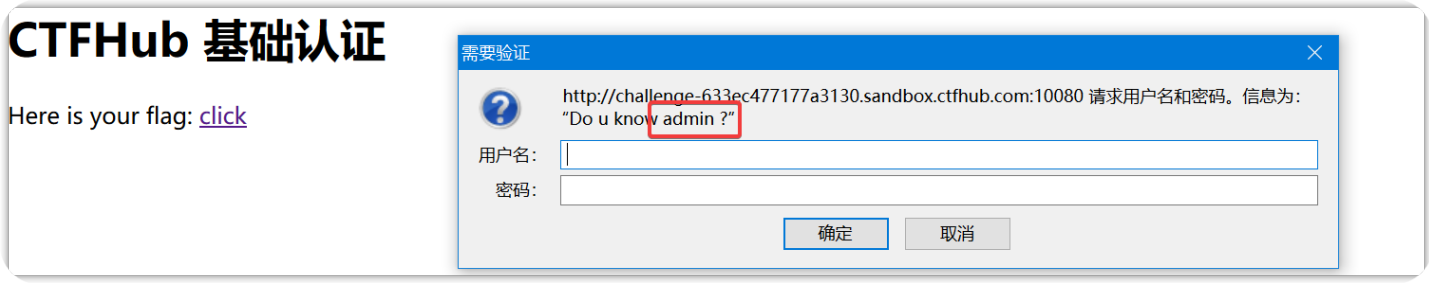
Art_Dillon 于 2020-03-26 11:11:21 发布 1553 收藏 1

分类专栏： CTF 文章标签： http

HTTP基本认证

在HTTP中，基本认证（英语：**Basic access authentication**）是允许http用户代理（如：网页浏览器）在请求时，提供用户名和密码 的一种方式。HTTP基本认证 。

题目描述



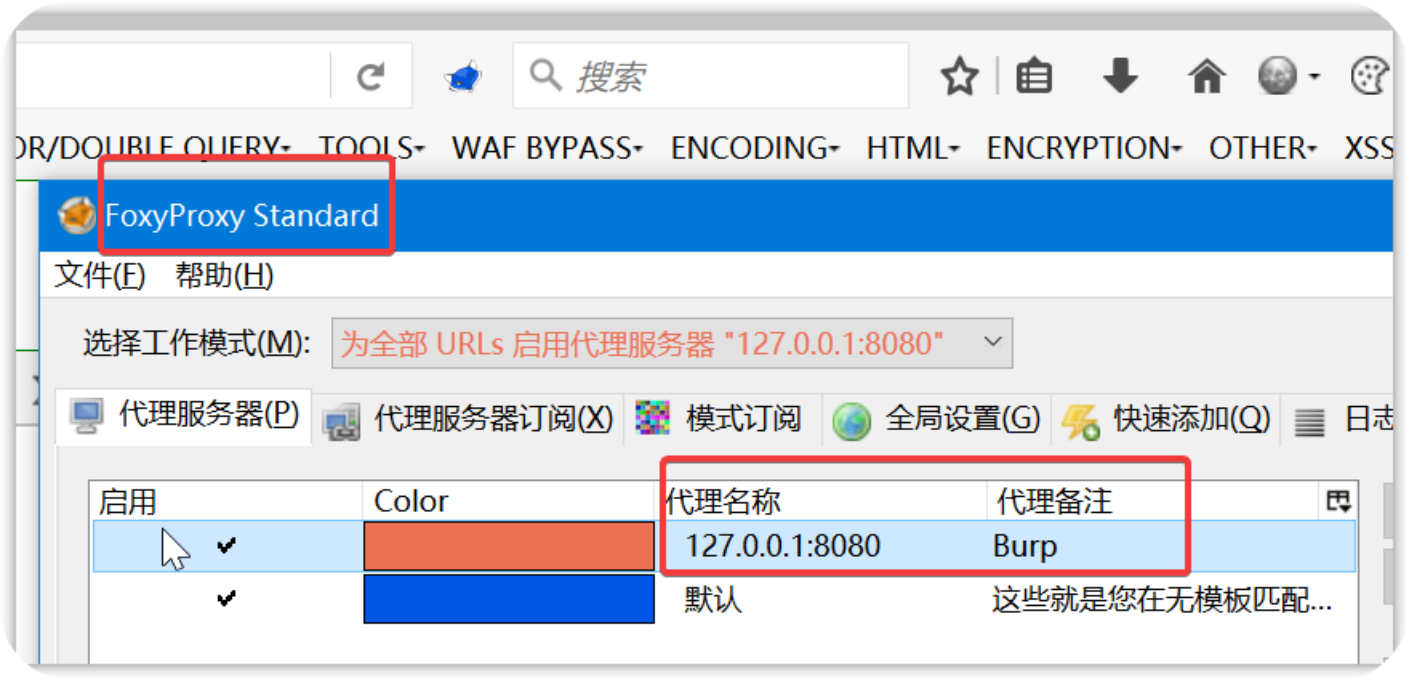
点击链接后，题目出现了让你输入**用户名和密码**的弹窗。而通过提示信息，我们知道用户名大概率是**admin**。而加之题目附件给出了字典。那么密码一定在给出的字典中。我们可以通过burp来进行爆破。

解题过程

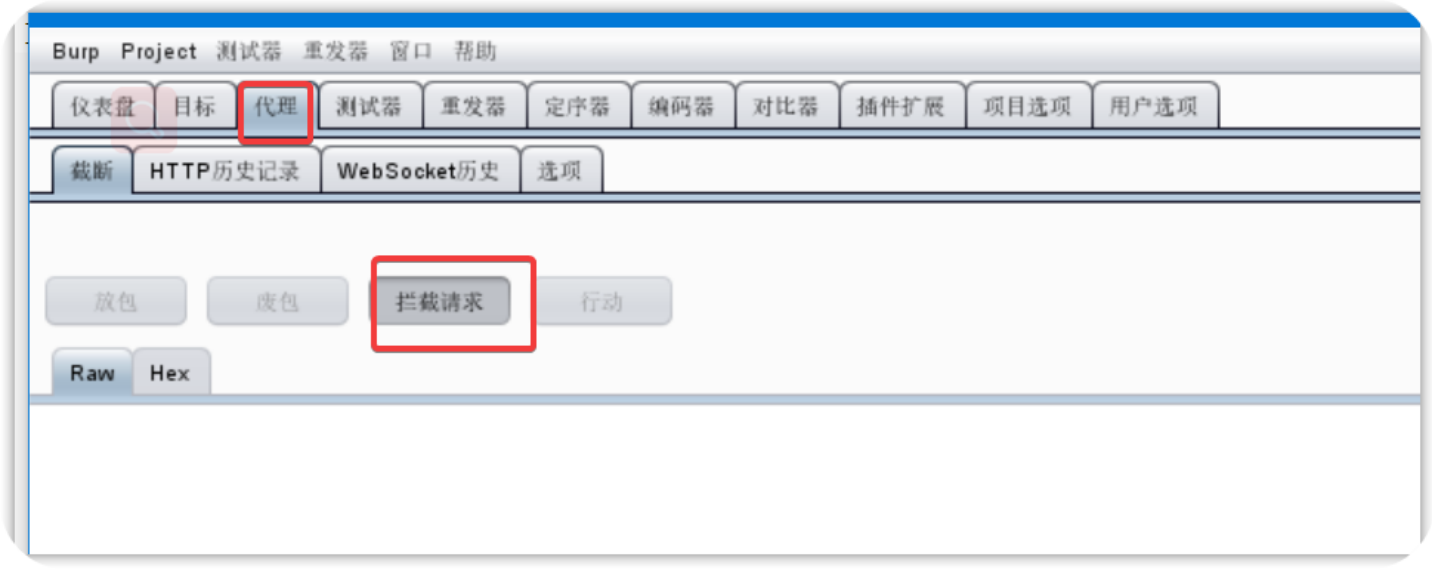
抓包

首先设置火狐的代理，让火狐的走burpsuit的代理**127.0.0.1：8080**

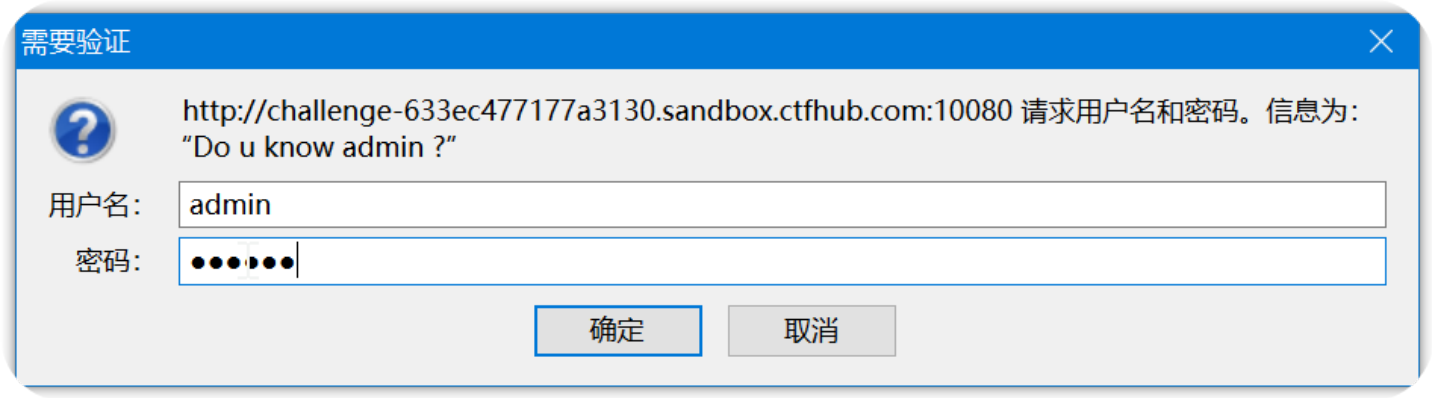
要想方便的话，还是要用火狐的插件，方便切换代理。



同时在Burp端设置好截断，监听**127.0.0.1：8080**



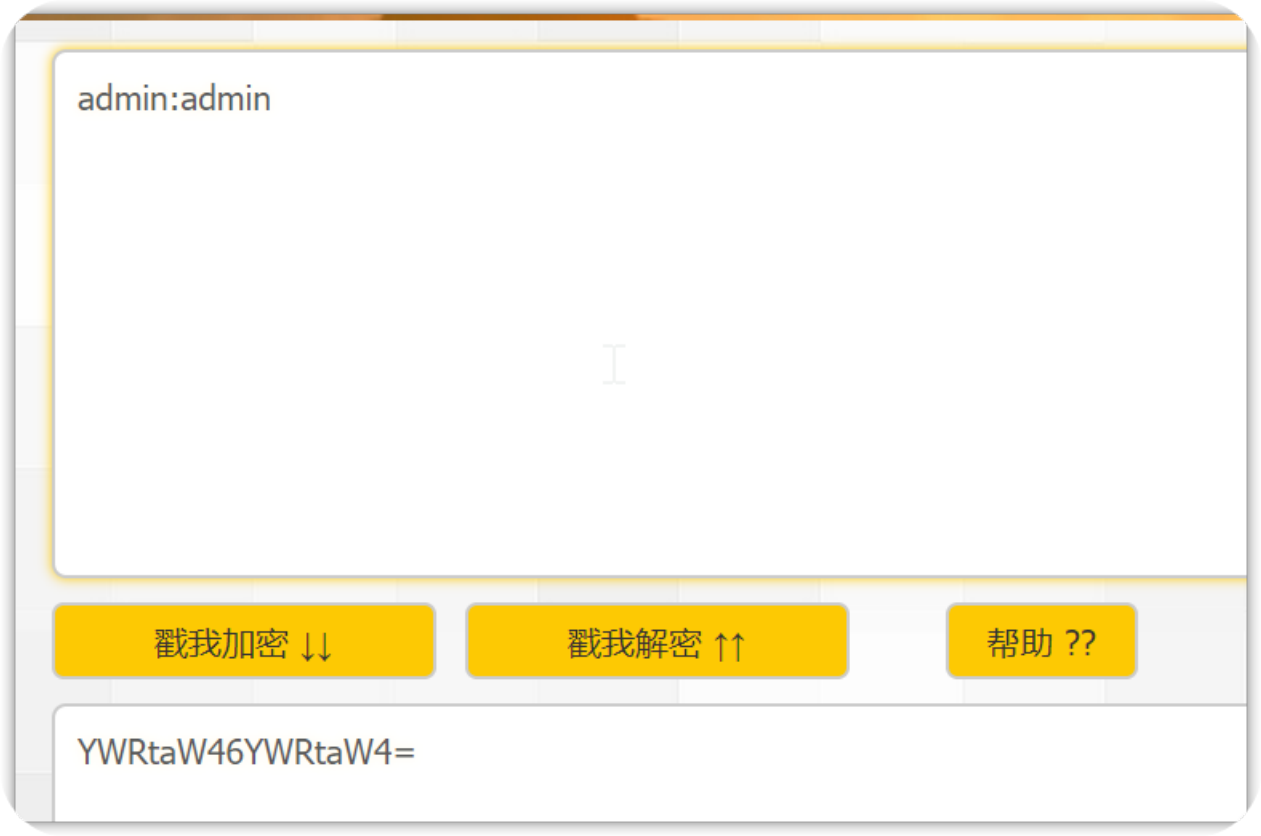
然后拦截我们输入密码的那一次请求



确定之后，我们会在burp的代理页面看到HTTP报文

```
GET /flag.html HTTP/1.1
Host: challenge-633ec477177a3130.sandbox.ctfhub.com:10080
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:46.0) Gecko/20100101 Firefox/46.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
DNT: 1
Referer: http://challenge-633ec477177a3130.sandbox.ctfhub.com:10080/
Connection: close
Authorization: Basic YWRtaW46YWRtaW4=
```

我们注意到最后一行，`Authorization : Basic`后面应该是一行base64编码。



解码之后我们就可以发现，我们刚刚输入的账号名的密码通过Base64 编码加密了。

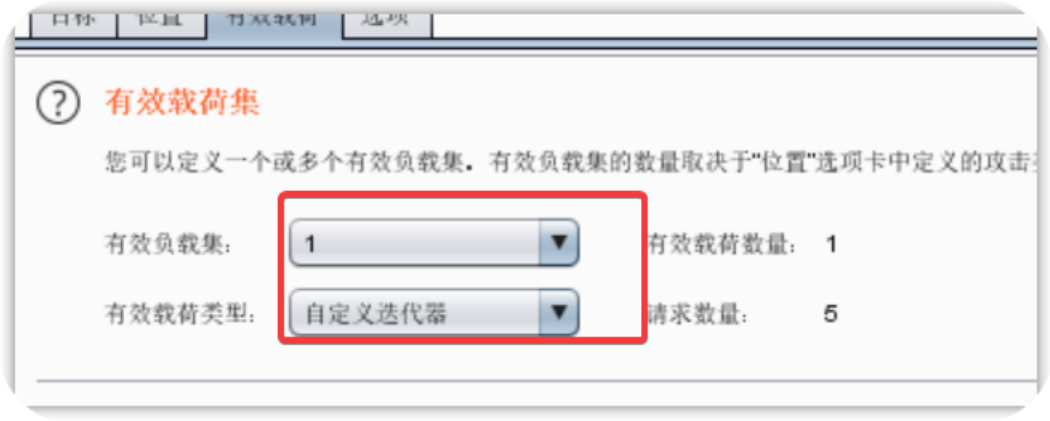
爆破

在代理的界面右键转发到测试器。

攻击位置，我们选择刚刚我们测试的**base64**。

然后我们设置payload

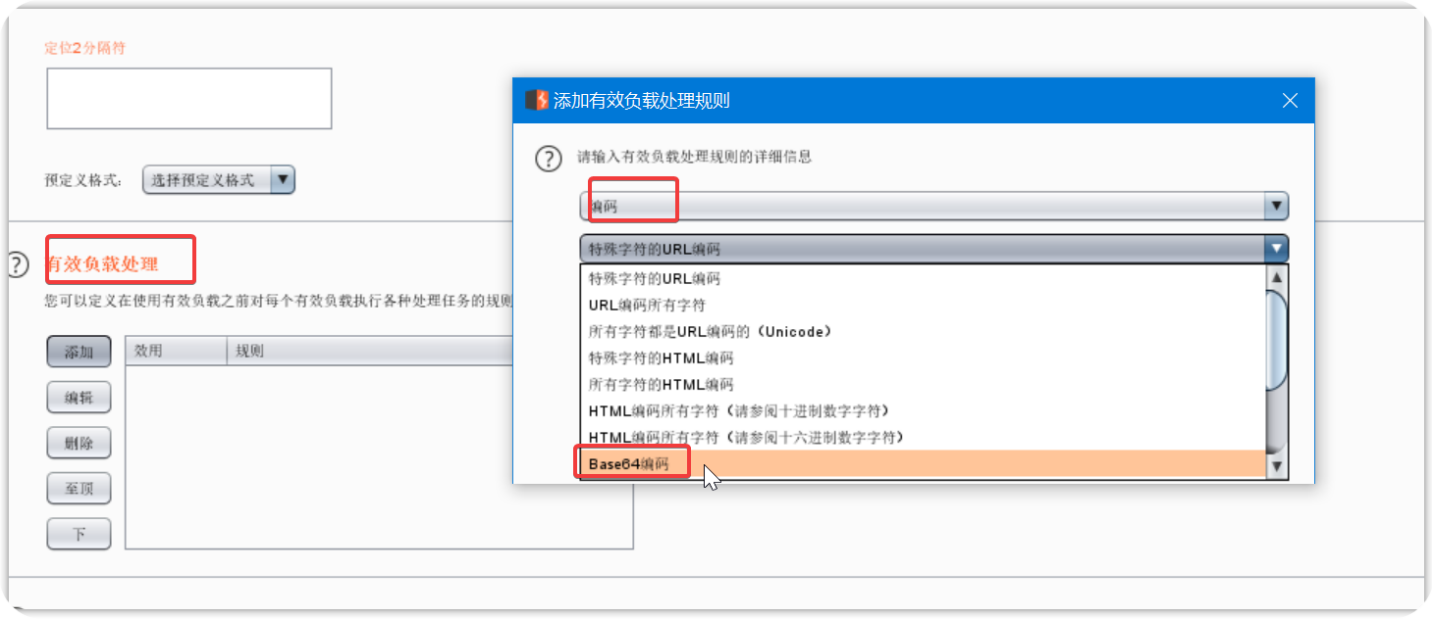
我们设置为自定义的迭代器



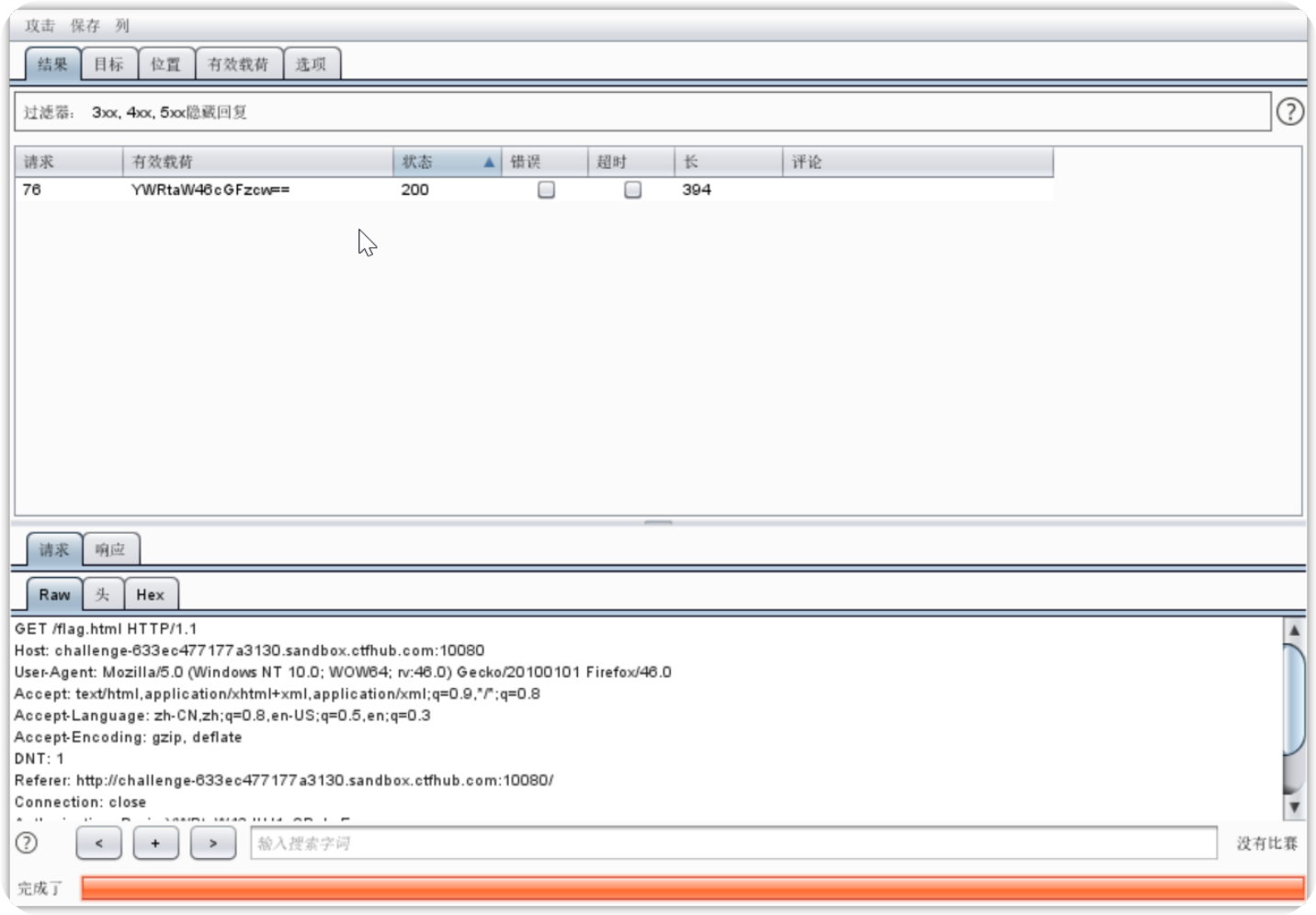
位置1

位置2

对我们的payload进行base64编码处理



之后我们就可以开始进行爆破了。



筛选出来唯一一条正确响应的那条记录

查看响应包。

```
HTTP/1.1 200 OK
Server: openresty/1.15.8.2
Date: Wed, 25 Mar 2020 13:57:01 GMT
Content-Type: text/html; charset=utf-8
Connection: close
Last-Modified: Wed, 25 Mar 2020 13:03:19 GMT
ETag: W/"5e7b5697-31"
Access-Control-Allow-Origin: *
Access-Control-Allow-Headers: X-Requested-With
Access-Control-Allow-Methods: *
Content-Length: 49

ctfhub{9d64d3704c05f66147f0ee91b75ec8f737440350}
```

获取flag。

相关知识

HTTP基本认证

HTTP基本认证 是一种十分简单的技术，使用的是 **HTTP头部字段** 强制用户访问网络资源，而不是通过必要的cookie、会话ID、登录页面等（非获取访问控制的）手段。

在进行基本认证的**过程**里，请求的**HTTP头字段会包含Authorization**字段，形式如下：`Authorization: Basic <凭证>`，该凭证是**用户和密码的组和的base64编码**。

过程

首先请求一个需要身份认证的网页，会有弹窗提示让你输入密码。如果没有提供用户名和密码，服务端会响应一个**401应答码**，并提供一个认证域（Access Authentication):**头部字段**为：`WWW-Authenticate`，该字段为要求客户端**提供适配的认证信息**。

```
GET /private/index.html HTTP/1.0
Host: localhost
```

```
HTTP/1.0 401 Authorization Required
Server: HTTPd/1.0
WWW-Authenticate: Basic realm="Secure Area"
Content-Type: text/html
Content-Length: 311

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
 "http://www.w3.org/TR/1999/REC-html401-19991224/loose.dtd">
<HTML>
  <HEAD>
    <TITLE>Error</TITLE>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=ISO-8859-1">
  </HEAD>
  <BODY><H1>401 Unauthorized.</H1></BODY>
</HTML>
```

注意上面的`WWW-Authenticate: Basic realm="Secure Area"`的这一字段。`Basic`为验证的模式，`realm="Secure Area"`为保护域，用于与其他请求URI作**区别**。

接到应答后，客户端**显示该认证域**（输入用户名和密码的框框）。给用户并提示输入用户名和密码。此时用户可以选择确定或取消。

用户输入了用户名和密码后，客户端软件将对其进行处理，并在原先的请求上增加**认证消息头（Authorization）**然后重新发送再次尝试。过程如下：

- 将用户名和密码拼接为`用户：密码`形式的字符串。
- 如果服务器WWW-Authenticate字段有指定编码，则将字符串编译成对应的编码（如：UTF-8）。
- 将字符串编码为base64。
- 拼接`Basic`，放入`Authorization头字段`，就像这样：`Authorization ： Basic 字符串`。
    - 这里注意**Base64编码并非加密算法**，其无法保证安全与隐私，仅用于将用户名和密码中的不兼容的字符转换为均与HTTP协议兼容的字符集。

```
GET /private/index.html HTTP/1.0
Host: localhost
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
```

服务端会进行基本认证，如果认证通过，则返回正确的页面。否则还是`401 unauthorized`

```
HTTP/1.0 200 OK
Server: HTTPd/1.0
Date: Sat, 27 Nov 2004 10:19:07 GMT
Content-Type: text/html
Content-Length: 10476
```

优点

简单，应用范围广

- HTTP基本认证是一种十分简单的技术，使用的是**HTTP头部字段强制用户访问网络资源**，而不是通过必要的cookie、会话ID、登录页面等（非获取访问控制的）手段

在可信网络环境中使用基本认证。

- 内部网络，或者对安全要求不是很高的网络。会结合HTTPS一起使用的，https保证网络的安全性，然后基本认证来做客户端身份识别。

缺点

- 基本认证 并没有为传送凭证（transmitted credentials）提供任何机密性的保护，**仅仅使用 Base64 编码**并传输，而没有使用任何**加密算法**。因此，基本认证**常常和HTTPS 一起使用，以提供机密性**。

参考资料

[1]. HTTP基本认证

[2]. 秒懂HTTP基本认证(Basic Authentication)