CTF-web 第一部分 MD5

原创

iamsongyu 于 2018-10-08 15:31:49 发布 7392 收藏 12

分类专栏： 理论知识 编程实践 文章标签： CTF

本文链接：https://blog.csdn.net/iamsongyu/article/details/82968660

理论知识 同时被 2 个专栏收录

109 篇文章 7 订阅

订阅专栏

编程实践

19 篇文章 0 订阅

订阅专栏

一． 哈希解密与攻击

```
哈希就是把任意长度的输入（又叫做预映射pre-image）通过散列算法变换成固定长度的输出，通常用来进行文件摘要或者信息加密。虽说很难具有相同哈希的文件，但是某些特意构造的
```

信息还是会满足相同的哈希，而且有些时候可以使用字典的方式进行解密和使用哈希攻击。

0x00 SHA

```
$_GET['name'] == $_GET['password']
sha1($_GET['name']) === sha1($_GET['password'])
```

其实最简单的是报错，false，至于为什么，其实仔细研究SHA1加密你就发现，其要求参数不能为数组，那我将传入的参数改成数组，两边return的结果不就都为false，从而，满足不等与相等了么。

0X01 MD5加密

salt型的md5加密可以考虑哈希长度拓展攻击，详情见我的博客哈希拓展攻击详解
第二种情况，md5加密后的比较，其实也是利用了"=="弱类型的判断（详情见我的博客php弱类型总结）
将加密后开头为0e的可以直接判等，原理这里不做解释了
特殊子串举例如下：
240610708、QNKCDZO、aabg7XSs、aabC9RqS

0x02 urldecode注意点

```
其实也没啥好说的，就是注意一点$_GET['name']，$_POST['name']均相当于进行了urldecode，如果存在
$a = $_GET['name'];
$b = urldecode($a);
```

$b相当于被解码了两次，注意区分

（1）MD5碰撞

要求是两者拥有相同的哈希摘要，在网页校验方面的小题目可以碰到。原理来讲MD5并不是十分可靠，不同的图像和php文件，hex数据等可以有相同的摘要，但是肯定是认为特意设计的，一般来讲是不会发生碰撞的。
常见的比如 name[]=a 和password[]=b 是相同的

（2）MD5解密

在某些情况下，会将数据进行MD5加密，这时候我们可以采用工具进行破解
http://www.cmd5.com/

（3）hash扩展长度攻击及hashdump使用 python支持 库hashpumpy

对与某些情况，服务器会根据提交的信息进行一定的加密得到B值。但是我们知道的是sha1(B+filename)的数值，简而言之，我们知道指定格式的哈希值，需要推导出指定的哈希与数据

```
      $password $user $secret=(15位) 已知 md5($secret，urluncode("admin","admin"))=571580b26c65f306376d4f64e53cb5c7
      而要求 username==="admin" password!="admin"
      并且md5($secret，urluncode($username，$password))==$COOKIE["getmein"]
      需要我们设置cookie 并且满足以上关系
```

使用hashdump 参数 -s 哈希值 -a 追加到末尾的内容 -k secret长度 -d 使用的数据 最终产生一个哈希 和 攻击串

```
 hashdump -s 571580b26c65f306376d4f64e53cb5c7 -d adminadmin -k 15 -a xxx 也可以将第一个admin视作secret一部分 因为不可更改
 hashdump -s 571580b26c65f306376d4f64e53cb5c7 -d admin -k 20 -a pcat 得到的结果如下：
       3e67e8f0c05e1ad68020df30bbc505f5 --》即为新的哈希

 admin\x80\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\xc8\x00\x00\x00\x00\x00\x00\x00pcat --》password值
```

（4）哈希截断爆破

用了多线程 跑起来很快## 标题

```
import hashlib
from multiprocessing.dummy import Pool as ThreadPool

# MD5截断数值已知 求原始数据
# 例子 substr(md5(captcha), 0, 6)=60b7ef

def md5(s):  # 计算MD5字符串
    return hashlib.md5(str(s).encode('utf-8')).hexdigest()


keymd5 = '8e6d35'   #已知的md5截断值
md5start = 0   # 设置题目已知的截断位置
md5length = 6

def findmd5(sss):    # 输入范围 里面会进行md5测试
    key = sss.split(':')
    start = int(key[0])   # 开始位置
    end = int(key[1])    # 结束位置
    result = 0
    for i in range(start, end):
        # print(md5(i)[md5start:md5length])
        if md5(i)[0:6] == keymd5:            # 拿到加密字符串
            result = i
            print(result)    # 打印
            break


list=[]  # 参数列表
for i in range(10):   # 多线程的数字列表 开始与结尾
    list.append(str(10000000*i) + ':' + str(10000000*(i+1)))
pool = ThreadPool()    # 多线程任务
pool.map(findmd5, list) # 函数 与参数列表
pool.close()
pool.join()
```