با ورود به به لینک به ادرس زیر ریدارکت میشیم که یک مقدار hmac دارد!

http://weshgrow.challs.nuitduhack.com/?hmac=ca8473d35a80a5ca4e9f3555c2869f71

از اونجا که hmac روشی برای امضا چیزی با استفاده از هشه پس این هش هم امضا چیزی هست احتمالا

در همان صفحه اول هم لینک دیگری هست:

http://weshgrow.challs.nuitduhack.com/admin?hmac=fac0887096a54ac497d968daf4c4fe0b

همچنین اگر به ادرس /flag بدون hmac یا با hmac الکی هم بریم به ادرسی مارو ریدایرکت میکنه که اخرش "#missinghmac" هست

بنابراین این hmac امضایی برای صفحات باید باشه و ما باید برای /flag hmac ولید ی به دست بیاریم

با بررسی کد صفحه متوجه نحوه لاگین ادمین هم میشوید که هش پسوردش ارسال میشود و تابعی که این کار را میکند در فایل bhe.js موجوده که همان "Best Hash Ever" است و باید در hmac هم استفاده شده باشد

کد آن به صورت زیر است:

```
var BestHashEver = function() {
    this.state = [
        str2bigInt('1336226589', 10),
        str2bigInt('251977347', 10),
        str2bigInt('716107527', 10),
        str2bigInt('1774966033', 10),
    ];
};

BestHashEver.prototype.bhe_round = function(byte) {
    var c = str2bigInt('162888806', 10);
    console.log(c);
    for (var i=3; i>0; i--) {
        this.state[i] = mod(add(mult(this.state[0], this.state[i]), mult(this.state[0],
int2bigInt(byte, 10))), str2bigInt('4294967295', 10));
        console.log(this.state[i]);
    }
    this.state[0] = mod(add(mult(this.state[0], c), mult(this.state[1], int2bigInt(byte, 10))),
str2bigInt('4294967295', 10));
};

BestHashEver.prototype.dword2hex = function(dw) {
    var hexchars = ["0", "1","2","3","4","5","6","7","8","9","a","b","c","d","e","f"];
    var output = '';
    var c1 = '', c2=''
    for (var i = 0; i < 4; i++) {
        byte = dw & 0x0000000F;
        c1 = hexchars[byte];
        dw = (dw >> 4);
        byte = dw & 0x0000000F;
        c2 = hexchars[byte];
        dw = (dw >> 4);
        output = output + c2 + c1;
    }
    console.log(output);
    return output;
};

BestHashEver.prototype.hash = function(data) {
    for (var i = 0; i < data.length; i++){
        this.bhe_round(data.charCodeAt(i));
        console.log(this.state);
    }

    /* Generate output. */
    var digest = [
        parseInt(bigInt2str(this.state[0], 10)),
        parseInt(bigInt2str(this.state[1], 10)),
        parseInt(bigInt2str(this.state[2], 10)),
        parseInt(bigInt2str(this.state[3], 10)),
    ];
    return this.dword2hex(digest[0]) + this.dword2hex(digest[1]) + this.dword2hex(digest[2]) +
this.dword2hex(digest[3]);
};

function hmac(data) {
    _bhe = new BestHashEver();
    return _bhe.hash(data);
}
```

این هش شبیه به md5 چهار متغیر حالت دارد و تعدادی round که در هر round با توجه به ورودی و مقادیر قبلی، با چند فرمول متغیر های حالت مقدار دهیه جدید میشوند ، در ابتدا هم متغیر های حالت مقادیر پیشفرضی دارند(default initialize state value).

در این هش هر round به ازای یک کارکتر ورودی است و در اخر با تبدیل متغیر های حالت نهایی به مقادیر هگز و چسباندشان به هم خروجی هش را میسازد.

مشکلی که در این روش وجود دارد در انتهایی هش اقدام خاصی مثل تاثیر دادن طول ورودی یا انجام نمیشه ...

این یعنی اگر ما مقادیر متغیر های حالت برای رشته ی s را داشته باشیم ، با ادامه دادن دور ها میتوان هش برای رشته های s+x را برای هر x بدست بیاوریم که به این حمله hash len extension attack گفته میشود.

واضح است که از hash(s) میتوان متغیر های حالت را به دست اورد ،تنها این نکته وجود دارد که تابع ای که در جاوااسکریپت برای تبدیل اعداد به هگر استفاده شده big endian byte order است.

Hash(s+x, default initialize state value)=Hash(s, state value of hash(s))

خب

با توجه به این آسیب پذیری و اینکه یکی از معروف ترین توابع hmac به صورت زیر است ، سناریو حمله مشخصه ...

Hmac(message)=hash(seret+message)

که در اینجا اسم صفحه به عنوان message به تابع داده میشه ، همانطور که در ابتدا دیدید ما hmac ولید برای صفحه خالی را داردیم بنا بر این:

Hmac(“”)=hash(secret)

Hmac(“flag”)= Hash(s, state value of Hmac (“”))

و تنها کاری که میمونه کد زدنه :

من تو پایتون زدم ولی میشود مقادر اولیه را در کد جاوا اسکریپت تغییر داد و کار کمتری کرد ، ولی جاوااسکریپ رو مخه دی:

```
def convert2be(s):#convert to big endian byte order
	return s[6:8]+s[4:6]+s[2:4]+s[0:2]

def myhex(s):
	return convert2be(hex(s)[2:-1].rjust(8,'0'))

def hash(inp,state=[1336226589,251977347,716107527,1774966033]):
	a,b,c,d=state[0],state[1],state[2],state[3]
	m=4294967295
	for x in inp:
		x=ord(x)
		aa=(a*(162888806+x*(b+x)))%m
		bb=(a*(b+x))%m
		cc=(a*(c+x))%m
		dd=(a*(d+x))%m
		a,b,c,d=aa,bb,cc,dd
	return myhex(a)+myhex(b)+myhex(c)+myhex(d)

emptyhmac="ca8473d35a80a5ca4e9f3555c2869f71" #hmac("")
import re
emptystate=map(lambda x:int(convert2be(x),16),re.findall('.'*8,emptyhmac))

print hash("flag",emptystate)
```

که خروجی کد **'f6933240ae234edddc27544d949238c3'** را به ما میدهد و با لینک زیر فلگ نمایش داده میشود

/flag?hmac=3f6933240ae234edddc27544d949238c