😀 ob混淆格式化检测存在内存爆破，存在三处内存爆破的地方;

一、定位内存爆破：

☐ 第1处内存溢出的位置：

```
var h = function() {  h = ƒ ()
    var k = new RegExp('\x5cw+\x20*\x5c(\x5c)\x20*{\x5cw+\x20*[\x27|\x22].+[\x27|\x22];?\x20*}');
    return k['test'](g['removeCookie']['toString']());  g = {data: {…}, setCookie: ƒ, removeCookie: ƒ, getCookie: ƒ, updateCookie: ƒ}
};
g['updateCookie'] = h;  g = {data: {…}, setCookie: ƒ, removeCookie: ƒ, getCookie: ƒ, updateCookie: ƒ}, h = ƒ ()
var i = '';  i = ""
var j = g['updateCookie']();  j = false, g = {data: {…}, setCookie: ƒ, removeCookie: ƒ, getCookie: ƒ, updateCookie: ƒ}
if (!j) {
    g['setCookie'](['*'], 'counter', 0x1);
} else if (j) {
    i = g['getCookie'](null, 'counter');
} else {
    g['removeCookie']();
}
```

```
14              },
15              'setCookie': function(k, l, m, n) {  k = ['*'], l = "counter", m = 1, n = {}
16                  n = n || {};
17                  var o = l + '=' + m;
18                  var p = 0x0;
19                  for (var q = 0x0, r = k['length']; q < r; q++) {
20                      var s = k[q];
21                      o += ';\x20' + s;
22                      var t = k[s];
23                      k['push'](t);
24                      r = k['length'];
25                      if (t !== !![]) {
26                          o += '=' + t;
27                      }
28                  }
29                  n['cookie'] = o;
30              },
```

这是一个死循环，会导致页面直接卡死

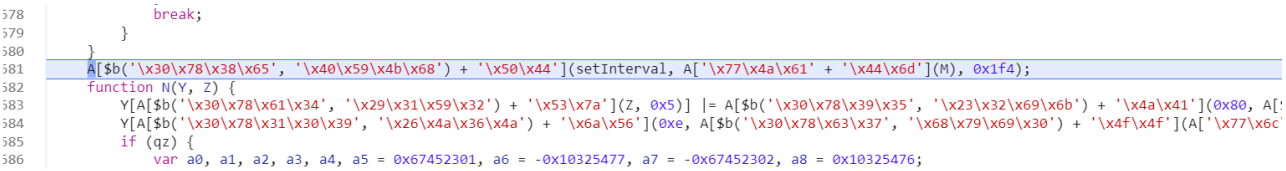
☐ 第2处内存溢出的位置：

```
(function $c(k) {  k = undefined
    var y = {};  y = {}
    y[$b('\x30\x78\x63\x37', '\x21\x46\x77\x37') + '\x69\x74'] = function(Y, Z) {
        return Y + Z;
    }
    ;
    y[$b('\x30\x78\x64\x30', '\x4e\x41\x33\x5b') + '\x45\x42'] = function(Y, Z) {
        return Y & Z;
    }
    ;
    y[$b('\x30\x78\x63\x64', '\x67\x23\x6c\x26') + '\x4f\x4b'] = function(Y, Z) {
```

再次进入死循环

```
        ;
        f['prototype']['ZqzWkS'] = function(g) {
            if (!Boolean(~g)) {
                return g;
            }
            return this['VoeCCy'](this['iZaYAB']);
        }
        ;
        f['prototype']['VoeCCy'] = function(g) {  g = ƒ (a, b)
            for (var h = 0x0, j = this['drtgZu']['length']; h < j; h++) {
                this['drtgZu']['push'](Math['round'](Math['random']()));
                j = this['drtgZu']['length'];
            }
            return g(this['drtgZu'][0x0]);
        }
        ;
        new f($b)['BXodrO']();
        $b['SLYmaz'] = !![];
    }
```

☐ 第3处内存溢出的位置：



解决方案：打 script 断点之后 hook RegExp

```
RegExp.prototype.test = function(){
    return true
}
```

二、确定加密参数：

抓包工具Fiddler数据重放，发现cookie缺失 m 参数, 请求数据就会失败。

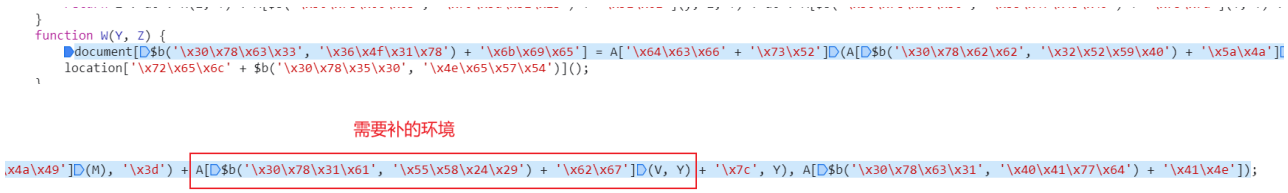
```
POST https://www.python-spider.com/api/challenge3 HTTP/1.1
Host: www.python-spider.com
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.0.0 Safari/537.36
Cookie:  sessionid=aq42cy0z4twwx6nd7z5vwpk805dtbkdn; m=ef089578918aa25d3a38f3339ea3624d|1699448092000;

page=1
```

处理方法是hook cookie 的 m 参数, 向上找堆栈找函数入口

```
(function () {
    var cookieVal = '';
    Object.defineProperty(document, 'cookie', {
    set: function (val) {
      if (val.indexOf('m') != -1) {
        debugger;
```

```
        }
        console.log('Hook捕获到cookie设置->', val);
        cookieVal = val;
        return val;
      }, get: function () {
        return cookieVal;
      },
      });
})();
```

三、找函数入口:



开始抠代码。。。