Analyzing JavaScript Malware

Direction – obfuscate

If I want get flag, I need to accomplish next operations

1. Step

Execute scripts for get javascript malware which exist in sequence decryption characters. It script include method document.write(str)which add text to HTML-page and method *unescape*(str) which create new line where hex-sequence characters %xx substitution on equals ASCII characters, method split("")- split up string for characters "," , method reverse()- invert string, and method join("")- assembly string

<script>  
document.write(*unescape*("%0a%3b%29%29%22%36%36%25%36%33%25%33%33%25%34%36%25%31%36%25%33%36%25%31%36%25%32%33%25%36%33%25%33%33%25%34%33%25%35%36%25%38%33%25%36%36%25%36%36%25%31%33%25%34%33%25%33%33%25%34%36%25%36%33%25%38%33%25%35%36%25%32%33%25%35%36%25%38%33%25%33%33%25%39%33%25%30%33%25%35%36%25%34%36%25%30%33%25%30%33%25%38%33%25%34%33%25%35%33%25%36%36%25%33%33%25%35%33%25%34%36%25%38%33%25%37%33%25%35%33%25%31%36%25%36%36%25%32%36%25%36%33%25%36%33%25%31%36%25%31%33%25%33%36%25%34%33%25%39%33%25%33%33%25%32%36%25%34%36%25%36%36%25%35%36%25%31%33%25%37%33%25%37%33%25%31%33%25%34%36%25%31%36%25%39%33%25%30%33%25%30%33%25%22%28%65%70%61%63%73%65%6e%75%28%67%61%6c%66%74%65%67%3d%6f%72%65%7a%0a%3b%29%31%74%70%69%72%63%73%28%74%72%65%6c%61%2f%2f%0a%7d%0a%0a%3b%29%30%35%2c%38%31%28%67%6e%69%72%74%73%62%75%73%2e%67%6e%69%72%74%73%65%20%6e%72%75%74%65%72%20%0a%7b%20%29%67%6e%69%72%74%73%65%28%67%61%6c%66%74%65%67%20%6e%6f%69%74%63%6e%75%66%0a%0a").split("").reverse().join(""));  
</script>

2. Step

After accomplish I can see next segment code which include function *getflag*(estring) with input variable String and it function return substring 18-50

function *getflag*(estring) { return estring.substring(18,50);} *//alert(script1);*zero=*getflag*(*unescape*("%30%30%39%61%64%31%37%37%31%65%66%64%62%33%39%34%63%31%61%36%36%62%66%61%35%37%38%64%35%33%66%35%34%38%30%30%64%65%30%39%33%38%65%32%65%38%36%64%33%34%31%66%66%38%65%34%33%36%32%61%63%61%64%33%36%66"));

Execute it script in method *alert*(*getflag*(zero)) which outputs modal window with a message

<script>  
function *getflag*(estring){ return estring.substring(18,50);}*//alert(script1);* zero=*getflag*(*unescape*("%30%30%39%61%64%31%37%37%31%65%66%64%62%33%39%34%63%31%61%36%36%62%66%61%35%37%38%64%35%33%66%35%34%38%30%30%64%65%30%39%33%38%65%32%65%38%36%64%33%34%31%66%66%38%65%34%33%36%32%61%63%61%64%33%36%66"));  
 *alert*(*getflag*(zero));</script>



Pic. 1 – javascript malware flag

Application decompilation

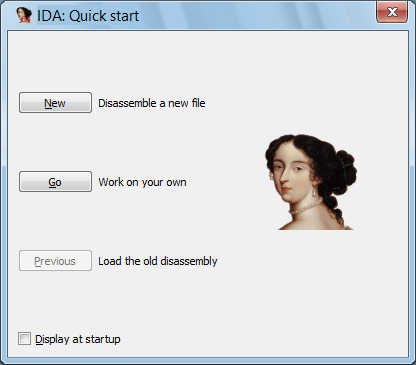
In begin I execute file challenge2.exe and get length of flag



Pic. 2.1 – flag length

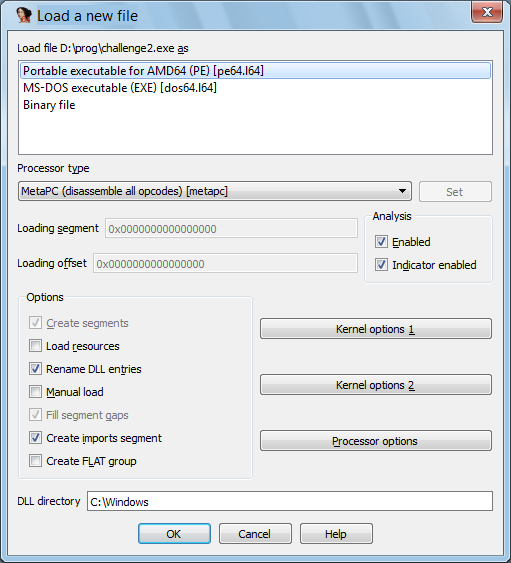
After I accomplish decompilation of program, because I want get value of flag.

I for decompilation use the program IDA Pro, start 🡪 New (Disassemble a new file) 🡪 select file challenge2.exe



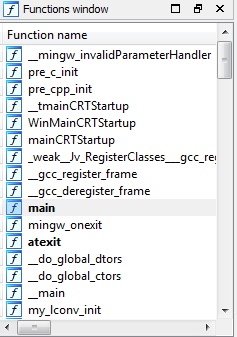
Pic. 2.2 – quick start window

Select Portable executable The PE format is a data structure that encapsulates the information necessary for the Windows OS loader to manage the wrapped executable code



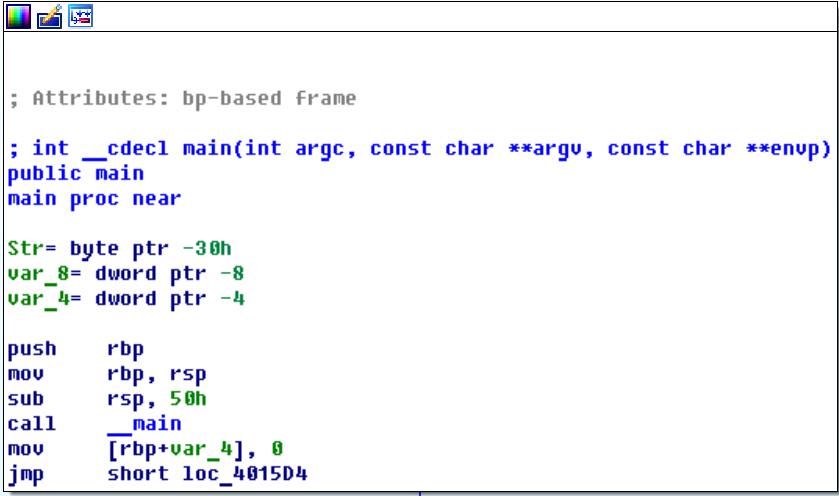
Pic. 2.3 – load a new file

After in Functions window I find and select main function



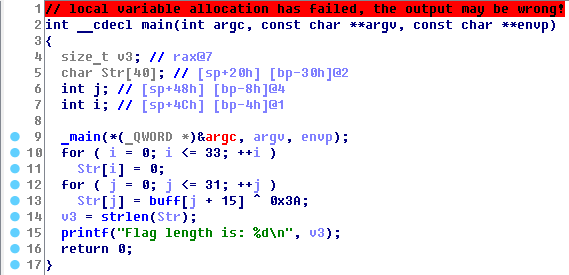
Pic. 2.4 – function window

The next step I see in general window block with main function



Pic. 2.5 – graph main function

I select it and press tab. This is how I find out pseudocode of this function



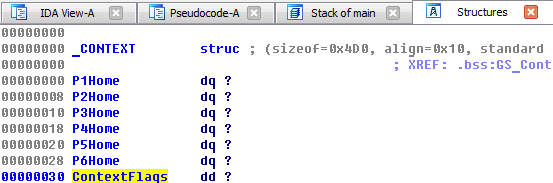
Pic. 2.6 – pseudocode main function

Inside a function I find char array Str[40] – it’s value of flag  
Select Str and in stack of main I find address value



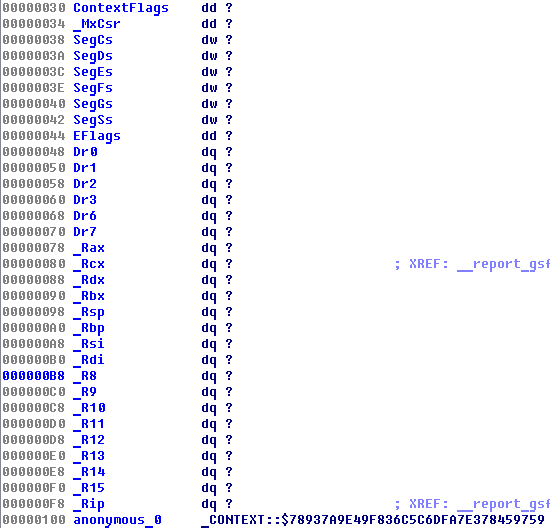
The next step on general window move on to the structures tab

and I find a variable ContextFlags at this address.



Pic. 2.7 – structure main function

Inside a \_CONTEXT structure I can see value of flag



Pic. 2.8 – flag value