6.1.3. IDX-028: Inyección JSON

INYECCIÓN JSON					
ID	IDX-028	**SEVERIDAD**	CRITICA	**CVSS V3.1**	9.4
VECTOR CVSS	AV:NETWORK/AC: LOW/PR: NONE/UI: NONE/S: UNCHANGED/C: HIGH/I: HIGH/A: LOW				
TIPO DE VULNERABILIDAD	A03 2021 - Inyección				

Descripción

Las inyecciones en estructuras JSON se producen cuando:

1. Los datos entran en un programa desde una fuente que no es de confianza.
2. Los datos se escriben en una estructura JSON.

Las aplicaciones suelen utilizar el formato JSON para almacenar datos o enviar mensajes. Cuando se utiliza para almacenar datos, JSON se trata a menudo como datos en caché y puede contener potencialmente información sensible. Cuando se utiliza para enviar mensajes, JSON se utiliza a menudo en conjunción con un servicio `RESTful` y puede ser utilizado para transmitir información sensible, como credenciales de autenticación.

La semántica de los documentos y mensajes JSON puede verse alterada si una aplicación construye una estructura JSON a partir de una entrada no validada. En un caso relativamente benigno, un atacante puede ser capaz de insertar elementos con claves arbitrarias que causen que una aplicación lance una excepción mientras analiza un documento o petición JSON. Por otro lado, en un escenario más crítico, como los que involucran inyección en estructuras JSON, un atacante puede ser capaz de insertar elementos arbitrarios que permitan la manipulación predecible de valores críticos dentro de un documento o petición JSON que sean utilizados posteriormente. En algunos casos, la inyección en formatos JSON puede conllevar una vulnerabilidad de "*Cross-Site Scripting*" o a la evaluación de código de forma dinámica.

Evidencias

```
$.pkp.controllers.UploaderHandler.prototype.
                uploadComplete = function(caller, pluploader, file, response) {
        var jsonData = $.parseJSON(response.response), filename = file.name;


        if (!jsonData.status) {
```

Se encontraron evidencias en las siguientes rutas:

- lib/pkp/js/controllers/UploaderHandler.js (Línea 172)
- lib/pkp/js/controllers/wizard/fileUpload/form/FileUploadFormHandler.js (Línea 203)
- lib/pkp/js/controllers/form/FileUploadFormHandler.js (Línea 134)

Recomendaciones

Cuando se escriban datos suministrados por el usuario en estructuras JSON, se recomienda seguir estas directrices:

1. No cree atributos JSON con nombres derivados de la entrada del usuario.
2. Asegúrese de que toda la serialización a JSON se realiza utilizando una función de serialización segura que delimite los datos no fiables entre comillas simples o dobles y escape cualquier carácter especial.

Por ejemplo, el siguiente código JavaScript implementa una serialización a JSON verificando el campo del nombre contra una lista de valores permitidos, rechazando el valor y estableciendo el nombre a "*guest*" en caso de no cumplir la condición.

```
var str = document.URL;
var url_check = str.indexOf('name=');
var name = null;
if (url_check > -1) {
name = decodeURIComponent(str.substring((url_check+5), str.length));
}
function getName(name){
var regexp = /^[A-z0-9]+$/;
var matches = name.match(regexp);
if (matches == null){
return "guest";
} else {
return name;
}
}
$(document).ready(function(){
if (name !== null){
var obj = jQuery.parseJSON('{"role": "user", "name" : "' + getName(name) + '"}');
...
}
});
```

Aunque en este caso está bien utilizar una lista de valores permitidos de esta manera, en otros casos es mejor utilizar valores sin establecer ningún tipo de control.

Referencias

https://owasp.org/Top10/A03_2021-Injection/
https://cwe.mitre.org/data/definitions/74.html