

■ **Cross-Site Scripting in Cookiebot WordPress plugin version 3.10.1 and older**

■ **Security advisory** 2021-07-07

Antoine Cervoise

Vulnerabilities description

Cookiebot

Cookiebot is a cloud-driven solution that automatically controls cookies and trackers, enabling full GDPR/ePR and CCPA compliance.¹

The issues

Synacktiv discovered a DOM based Cross-Site Scripting vulnerabilities in the *Cookie declaration* page (this page is inserted into Wordpress using the string `[cookie_declaration]`).

Affected versions

At the time this report is written, the version 3.10.1 was proven to be affected. The plugin itself is not affected but a remote script used by this version. All previous versions using the same remote script are also vulnerable.

The patch was set on the remote script. There is no need to update the WordPress plugin.

Timeline

Date	Action
2021-09-04	Confirmation of correction from CookieBot.
2021-07-09	Mail sent to mail@cookiebot.com .
2021-07-12	Reply from Cookiebot with the Vulnerability disclosure policy.
2021-07-13	Advisory sent to mail@cookiebot.com .

¹ <https://wordpress.org/plugins/cookiebot/>

Technical description and proof-of-concept

DOM based Cross-Site Scripting (XSS)

Cookie Bot plugin for *Wordpress* has the functionality to insert the cookie declaration on the website using this string:

```
[cookie_declaration]
```

This string will be replaced by the results of the function *show_declaration* from *cookiebot.php*.

```
/**
 * Cookiebot_WP Output declation shortcode [cookie_declaration]
 * Support attribute lang="LANGUAGE_CODE". Eg. lang="en".
 *
 * @version 2.2.0
 * @since 1.0.0
 */
function show_declaration( $atts = array() ) {
    $cbid = $this->get_cbid();
    $lang = '';
    [...]
    return '<script id="CookieDeclaration" src="https://consent.cookiebot.com/' .
    $cbid . '/cd.js"' . $lang . ' type="text/javascript" ' . $tagAttr . '></script>';
    } else {
        return esc_html__( 'Please add your Cookiebot ID to show Cookie Declarations',
        'cookiebot' );
    }
}
```

The HTML code on the final page is the following:

```
<script id="CookieDeclaration" src="https://consent.cookiebot.com/[ID]/cd.js"
type="text/javascript" async></script>
```

The *JavaScript* hosted at *https://consent.cookiebot.com/[ID]/cd.js* will call another *JavaScript* at *https://consent.cookiebot.com/[ID]/cdreport.js*. This second script (*cdreport.js*) contains the HTML code inserted in the final page, especially the following lines (the language changes with the plugin configuration).

```
<span style="display:block">Identifiant de votre consentement: <span
id=CookieDeclarationUserStatusLabelConsentId></span></span><span style="display:block">Date
de consentement: <span id=CookieDeclarationUserStatusLabelConsentDate></span></span>
```

The first script (*cd.js*) will insert HTML value into these line:

```
a=document.getElementById("CookieDeclarationUserStatusLabelConsentId")
[...]
a.innerHTML=CookieConsent.consentID
```

As the cookie is not sanitized, HTML and *JavaScript* could be injected into this field. For example, if the cookie contains the following code, malicious *JavaScript* code could be executed:

```
<svg><animate onbegin=alert(1) attributeName=x dur=1s>
```

The screenshot shows a web browser window with a URL ending in `.fr/confidentialite/`. A consent dialog is displayed, and a network storage tab is open, showing a table of cookies. The table has columns: Name, Value, Domain, Path, Expires / Max-Age, Size, HttpO..., Secure, Same..., and Last Accessed. The first row is highlighted, showing a cookie with a value containing an SVG payload: `{stamp:%27 [REDACTED] <svg><animate onbegin=alert(document.location) attributeN...`

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpO...	Secure	Same...	Last Accessed
Cookie...	{stamp:%27 [REDACTED] <svg><animate onbegin=alert(document.location) attributeN...	[REDACTED]	/	Tue, 05 Oct 202...	128	false	true	None	Mon, 05 Jul 2021...
wfwaf-a...	[REDACTED]	[REDACTED]	/	Mon, 05 Jul 2021...	138	true	true	None	Mon, 05 Jul 2021...
wordpr...	[REDACTED]	[REDACTED]	/	Session	191	true	true	None	Mon, 05 Jul 2021...
wordpr...	[REDACTED]	[REDACTED]	/wp-ad...	Session	185	true	true	None	Mon, 05 Jul 2021...
wordpr...	[REDACTED]	[REDACTED]	/wp-co...	Session	185	true	true	None	Mon, 05 Jul 2021...
wordpr...	[REDACTED]	[REDACTED]	/	Session	40	false	true	None	Mon, 05 Jul 2021...

Illustration 1: XSS vulnerability triggered.

Recommendation

Sanitize user cookie within the `https://consent.cookiebot.com/[ID]/cd.js` script.

Note: this update can be done without any modification in the *Wordpress* plugin code.