CVE-2021-30057 : A stored HTML injection vulnerability exists in Knowage Suite version 7.1. An attacker can inject arbitrary HTML in "/restful-services/2.0/analyticalDrivers" via the 'LABEL' and 'NAME' parameters.

Overview

Knowage (https://www.knowage-suite.com) is the Open Source Business Analytics Suite combining traditional and big data sources into valuable and meaningful information.

Description

The vulnerability is present in the '**/restful-services/2.0/analyticalDrivers**', and can be exploited throuth a POST request via the '**LABEL**' and '**NAME**' parameters.

Impact

An attacker can send HTML code through any vulnerable form field to change the design of the website or any information displayed to the user, saving the information persistently on the site (e.g. database). As a result, the user will see the data sent by the attacker every time he calls up the vulnerable page.

Timeline

- **2021-02-09**: Discovered and reported to Knowage
- **2021-02-09**: Got instant response from Knowage development team, "Thanks for your analysis report. We will evaluate your finding and get back to you soon with our feedback.
- **2021-03-22**: Knowage Team fixed this issue in Knowage version 7.4.0
- **2021-04-05**: I have obtained the CVE-2021-30057 and published the PoC

Discovered by

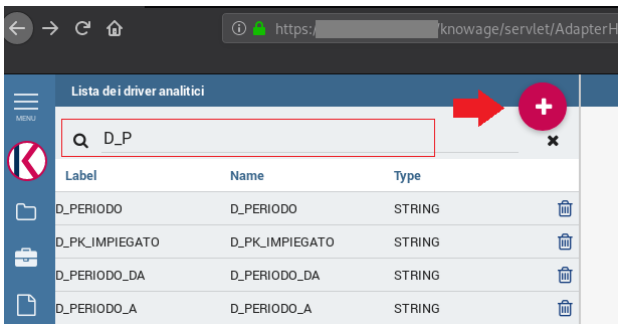
Gianluca Palma (@piuppi) of Engineering Ingegneria Informatica S.p.A.

Antonio Scibilia of Cybertech S.r.l.

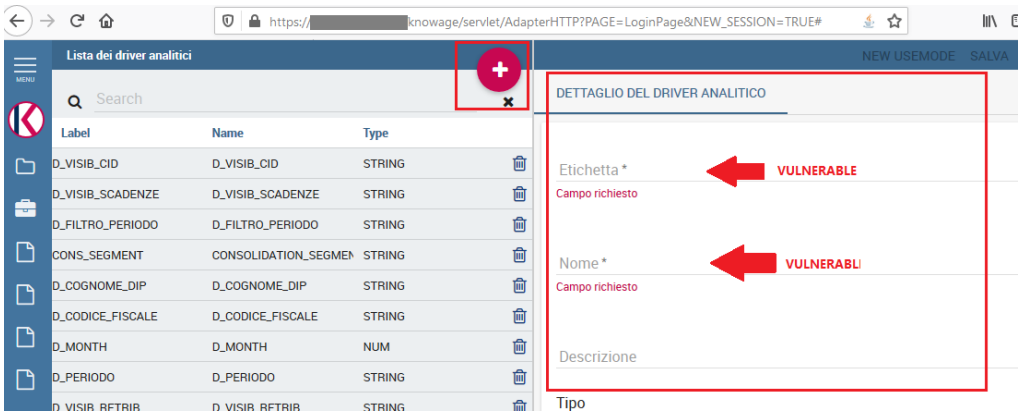
Proof of concept (POC)

Reproducing Steps

After logging into the **Knowage Suite** portal with an administration profile, you can manage the list of **analytical drivers** where you can search for, create or edit a specific driver.



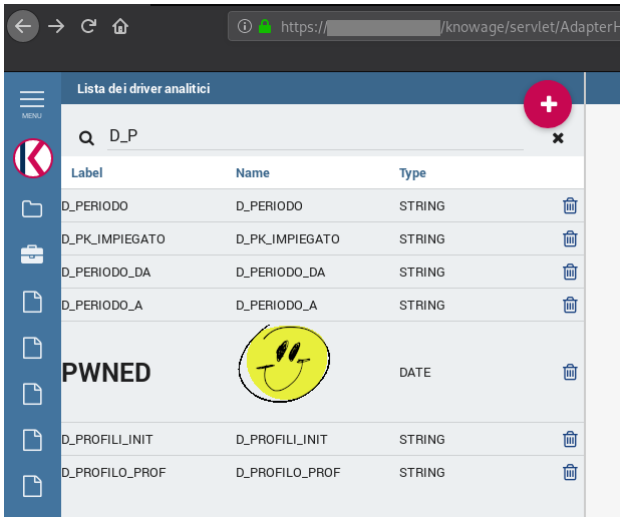
By editing, or creating a new analytical driver, it is possible to give it a **label** and a **name**. These fields are vulnerable to stored HTML injection, as shown below:



As can be seen from the following evidence, the content of the injection was correctly saved on the page (**on the database**) and executed each time the analytical driver in question is searched or called up internally by the application.

Request:

Response:



Suggestions

In most situations where user-controllable data is copied into application responses, HTML Injection attacks can be prevented using two layers of defenses:

- Input should be validated as strictly as possible on arrival, given the kind of content that it is expected to contain. For example, personal names should consist of alphabetical and a small range of typographical characters, and be relatively short; a year of birth should consist of exactly four numerals; email addresses should match a well-defined regular expression. Input which fails the validation should be rejected, not sanitized.
- User input should be HTML-encoded at any point where it is copied into application responses. All HTML metacharacters, including < > " ' and =, should be replaced with the corresponding HTML entities (< > etc). In cases where the application's functionality allows users to author content using a restricted subset of HTML tags and attributes (for example, blog comments which allow limited formatting and linking), it is necessary to parse the supplied HTML to validate that it does not use any dangerous syntax; this is a non-trivial task.