Vulnerability Report

Archivo: dataTables.bootstrap5.min.js

Code Analyzed:

```
/*! DataTables Bootstrap 5 integration
 * 2020 SpryMedia Ltd - datatables.net/license
 */
!function(t){var n,r;"function"==typeof
define&&define.amd?define(["jquery","datatables.net"],function(e){return
t(e,window,document)}):"object"==typeof
exports?(n=require("jquery"),r=function(e,a){a.fn.dataTable||require("datatables.net")(e,a)},"
undefined"!=typeof window?module.exports=function(e,a){return
e=e||window,a=a||n(e),r(e,a),t(a,0,e.document)}:(r(window,n),module.exports=t(n,window,window.
document))):t(jQuery,window,document)}(function(x,e,r,i){"use strict";var
o=x.fn.dataTable;return x.extend(!0,o.defaults,{dom:"<'row'<'col-sm-12 col-md-6'l><'col-sm-12
col-md-6'f>><'row dt-row'<'col-sm-12'tr>><'row'<'col-sm-12 col-md-5'i><'col-sm-12 col-md-
7'p>>",renderer:"bootstrap"}),x.extend(o.ext.classes,{sWrapper:"dataTables_wrapper dt-
bootstrap5",sFilterInput:"form-control form-control-sm",sLengthSelect:"form-select form-
select-sm",sProcessing:"dataTables_processing card",sPageButton:"paginate_button page-
item"}),o.ext.renderer.pageButton.bootstrap=function(d,e,s,a,l,c){function u(e,a){for(var
t,n,r=function(e){e.preventDefault(),x(e.currentTarget).hasClass("disabled")||b.page()==e.data
.action||b.page(e.data.action).draw("page")},i=0,o=a.length;i<o;i++)if(t=a[i],Array.isArray(t)
)u(e,t);else{switch(f=p="",t){case"ellipsis":p="&#x2026;",f="disabled";break;case"first":p=g.s
First,f=t+(0<l?"":" disabled");break;case"previous":p=g.sPrevious,f=t+(0<l?"":"
disabled");break;case"next":p=g.sNext,f=t+(l<c-1?"":"
disabled");break;case"last":p=g.sLast,f=t+(l<c-1?"":"
disabled");break;default:p=t+1,f=l===t?"active":""}p&&(n=-
1!==f.indexOf("disabled"),n=x("<li>",{class:m.sPageButton+" "+f,id:0===s&&"string"==typeof
t?d.sTableId+"_"+t:null}).append(x("<a>",{href:n?null:"#","aria-controls":d.sTableId,"aria-
disabled":n?"true":null,"aria-label":w[t],"aria-role":"link","aria-
current":"active"===f?"page":null,"data-dt-idx":t,tabindex:d.iTabIndex,class:"page-
link"}).html(p)).appendTo(e),d.oApi._fnBindAction(n,{action:t},r))}}var p,f,t,b=new
o.Api(d),m=d.oClasses,g=d.oLanguage.oPaginate,w=d.oLanguage.oAria.paginate||{},e=x(e);try{t=e.
find(r.activeElement).data("dt-idx")}catch(e){}var
n=e.children("ul.pagination");n.length?n.empty():n=e.html("<ul/>").children("ul").addClass("pa
gination"),u(n,a),t!==i&&e.find("[data-dt-idx="+t+"]").trigger("focus")},o});
```

Analysis: ```html
Potential Security Vulnerabilities

DOM-Based Cross-Site Scripting (XSS)

Type: DOM-Based XSS
Approximate Line: Several lines within the `o.ext.renderer.pageButton.bootstrap` function, particularly where p (which is derived from external data) is used in `.html(p)` calls. Specifically the cases where language strings such as first, previous, next, last and ellipsis are inserted as HTML. Also the default case where `p=t+1`.
Description: The code directly injects potentially untrusted data (language strings and computed values) into the DOM using `.html()` without proper sanitization. An attacker could potentially control the content of `d.oLanguage.oPaginate` and `d.oLanguage.oAria.paginate` to inject malicious JavaScript code, especially if these values are sourced from user input or a compromised server. The expression p=t+1 can be exploited to generate numbers with malicious Javascript as well.
Mitigation:

- **Escape HTML Entities:** Before inserting p into the DOM via `.html()`, escape any HTML entities (e.g., <, >, &, ", ', /). You can use a dedicated escaping function or a library like DOMPurify (more robust) to sanitize the HTML before insertion.
- **Use `.text()` for Text Content:** If the content is intended to be plain text, use `.text()` instead of `.html()`. This will ensure that any HTML tags are treated as literal text and not interpreted as HTML.
- **Validate and Sanitize Input:** If the language strings originate from an external source (e.g., a configuration file or API), rigorously validate and sanitize the data before using it in the application. Ensure that the strings conform to the expected format and do not contain any malicious characters.

Code Quality Metrics

Complexity & Readability

Complexity: The `o.ext.renderer.pageButton.bootstrap` function is quite complex due to the nested conditional logic, array processing, and DOM manipulation.
Readability: The code is somewhat difficult to read due to the minified format and the density of operations within the function.
Coupling: The function is tightly coupled with the DataTables API and internal data structures (e.g., `d.oClasses`, `d.oLanguage`).
Improvements:

- **Decompose Functions:** Break down the large `o.ext.renderer.pageButton.bootstrap` function into smaller, more manageable sub-functions, each responsible for a specific task (e.g., generating a single page button, handling ellipsis, applying classes).
- **Add Comments:** Include detailed comments to explain the purpose of each section of code, the logic behind the conditional statements, and the role of the variables.
- **Use Meaningful Variable Names:** While minified, in non-minified versions use names that clarify their purpose (e.g., `pageButtonClass` instead of `f`, `pageNumber` instead of `p`).
- **Reduce Nesting:** Try to reduce the nesting level of the conditional statements. Consider using `continue` or `return` statements to simplify the control flow.

Duplication

Duplication: The logic for applying "disabled" class is duplicated across the "first", "previous", "next", and "last" cases in the switch statement.
Improvements:

- **Refactor Duplicated Logic:** Extract the common logic for applying the "disabled" class into a separate function or expression that can be reused across the different cases. This reduces code duplication and improves maintainability.

Proposed Solution

A more robust solution combines escaping with DOMPurify to sanitize the content before it is injected. A code snippet is provided below to illustrate this approach. This approach can be used to remove XSS attack vectors in the described vulnerability above.

```
function sanitizeHTML(html) {
  return DOMPurify.sanitize(html);
}

o.ext.renderer.pageButton.bootstrap = function(d, e, s, a, l, c) {
  // ... existing code ...

  function u(e, a) {
    for (var t, n, r = function(e) {
      e.preventDefault();
      x(e.currentTarget).hasClass("disabled") || b.page() == e.data.action ||
b.page(e.data.action).draw("page");
    }, i = 0, o = a.length; i  o; i++)
      if (t = a[i], Array.isArray(t)) u(e, t);
      else {
        switch (f = p = "", t) {
          case "ellipsis":
            p = "…", f = "disabled";
            break;
          case "first":
            p = g.sFirst, f = t + (0  l ? "" : " disabled");
            break;
          case "previous":
            p = g.sPrevious, f = t + (0  l ? "" : " disabled");
            break;
```

```
           case "next":
             p = g.sNext, f = t + (l  c - 1 ? "" : " disabled");
             break;
           case "last":
             p = g.sLast, f = t + (l  c - 1 ? "" : " disabled");
             break;
           default:
             p = t + 1, f = l === t ? "active" : "";
         }
         if (p) {
           n = -1 !== f.indexOf("disabled");
           // Sanitize the HTML before injecting it into the DOM
           var sanitizedHTML = sanitizeHTML(p); //Sanitize p using DOMPurify here.

           n = x("
• ", {
             class: m.sPageButton + " " + f,
             id: 0 === s && "string" == typeof t ? d.sTableId + "_" + t : null
           }).append(x("", {
             href: n ? null : "#",
             "aria-controls": d.sTableId,
             "aria-disabled": n ? "true" : null,
             "aria-label": w[t],
             "aria-role": "link",
             "aria-current": "active" === f ? "page" : null,
             "data-dt-idx": t,
             tabindex: d.iTabIndex,
             class: "page-link"
           }).html(sanitizedHTML)).appendTo(e);

           d.oApi._fnBindAction(n, {
             action: t
           }, r);
         }
       }
   }

   // ... rest of the function ...
 }
```

```
```