priyankark / **a11y-mcp** Public

Code Issues Pull requests Actions Projects Security and quality

Commit e3e11c9

priyankark and **claude** committed 2 weeks ago · ✓ 1 / 1

```
fix: add SSRF protection with URL validation (CVE pending)

Validate user-supplied URLs before passing to Puppeteer's page.goto()
to prevent server-side request forgery. Blocks loopback, private
(RFC1918), link-local, and cloud metadata addresses after DNS
resolution. Only http/https schemes are allowed.

Bumps version to 1.0.5.

Co-Authored-By: Claude Opus 4.6 (1M context) <noreply@anthropic.com>
```

main (#3)

1 parent 19fcc94 commit e3e11c9

2 files changed +91 -11 lines changed

Top

Filter files…

- package.json
- src
  - index.js

2 files changed +91 -11 lines changed

Search within code

package.json

```
@@ -1,6 +1,6 @@
{
  "name": "a11y-mcp",
-   "version": "1.0.4",
+   "version": "1.0.5",
  "main": "src/index.js",
```

```
5  5    "type": "module",
6  6    "bin": {
```

src/index.js

```
@@ -7,9 +7,83 @@ import {
7  7      ListToolsRequestSchema,
8  8      McpError,
9  9    } from '@modelcontextprotocol/sdk/types.js';
   10 + import { lookup } from 'node:dns/promises';
10 11   import puppeteer from 'puppeteer';
11 12   import { AxePuppeteer } from '@axe-core/puppeteer';
12 13
   14 + /**
   15 +  * Validate that a URL is safe to navigate to (SSRF protection).
   16 +  * Only allows http/https schemes and blocks requests to internal networks.
   17 +  */
   18 + async function validateUrl(urlString) {
   19 +   let parsed;
   20 +   try {
   21 +     parsed = new URL(urlString);
   22 +   } catch {
   23 +     throw new Error('Invalid URL format');
   24 +   }
   25 +
   26 +   // Only allow http and https schemes
   27 +   if (parsed.protocol !== 'http:' && parsed.protocol !== 'https:') {
   28 +     throw new Error(`Disallowed URL scheme: ${parsed.protocol}`);
   29 +   }
   30 +
   31 +   const hostname = parsed.hostname;
   32 +
   33 +   // Block obvious localhost/loopback hostnames
   34 +   const blockedHostnames = ['localhost', '127.0.0.1', '::1', '0.0.0.0',
      '[::1]'];
   35 +   if (blockedHostnames.includes(hostname.toLowerCase())) {
   36 +     throw new Error('URLs pointing to loopback addresses are not allowed');
   37 +   }
   38 +
   39 +   // Resolve the hostname and check the resulting IP
```

```
40 +   let address;
41 +   try {
42 +     const result = await lookup(hostname);
43 +     address = result.address;
44 +   } catch {
45 +     throw new Error(`Unable to resolve hostname: ${hostname}`);
46 +   }
47 +
48 +   if (isPrivateIP(address)) {
49 +     throw new Error('URLs pointing to private or internal network addresses are
     not allowed');
50 +   }
51 +
52 +   return parsed;
53 + }
54 +
55 + /**
56 +  * Check if an IP address belongs to a private, loopback, or link-local range.
57 +  */
58 + function isPrivateIP(ip) {
59 +   // IPv4 checks
60 +   const parts = ip.split('.').map(Number);
61 +   if (parts.length === 4 && parts.every(p => p >= 0 && p <= 255)) {
62 +     // 127.0.0.0/8 — loopback
63 +     if (parts[0] === 127) return true;
64 +     // 10.0.0.0/8 — private
65 +     if (parts[0] === 10) return true;
66 +     // 172.16.0.0/12 — private
67 +     if (parts[0] === 172 && parts[1] >= 16 && parts[1] <= 31) return true;
68 +     // 192.168.0.0/16 — private
69 +     if (parts[0] === 192 && parts[1] === 168) return true;
70 +     // 169.254.0.0/16 — link-local / cloud metadata
71 +     if (parts[0] === 169 && parts[1] === 254) return true;
72 +     // 0.0.0.0/8
73 +     if (parts[0] === 0) return true;
74 +   }
75 +
76 +   // IPv6 loopback
77 +   if (ip === '::1' || ip === '0:0:0:0:0:0:0:1') return true;
78 +   // IPv6 link-local
```

```
      79  +   if (ip.toLowerCase().startsWith('fe80:')) return true;
      80  +   // IPv6 unique local (fc00::/7)
      81  +   const first2 = ip.toLowerCase().slice(0, 2);
      82  +   if (first2 === 'fc' || first2 === 'fd') return true;
      83  +
      84  +   return false;
      85  + }
      86  +
 13   87    class A11yServer {
 14   88      constructor() {
 15   89        this.server = new Server(
          @@ -104,18 +178,21 @@ class A11yServer {
104  178        }
105  179
106  180        try {
      181  +       // Validate URL to prevent SSRF
      182  +       const validatedUrl = await validateUrl(args.url);
      183  +
107  184          const browser = await puppeteer.launch({
108  185            headless: 'new',
109  186            args: ['--no-sandbox', '--disable-setuid-sandbox'],
110  187          });
111  188          const page = await browser.newPage();
112       -
      189  +
113  190          // Set a reasonable viewport
114  191          await page.setViewport({ width: 1280, height: 800 });
115       -
116       -       // Navigate to the page
117       -       await page.goto(args.url, { waitUntil: 'networkidle2', timeout: 30000 });
118       -
      192  +
      193  +       // Navigate to the page using the validated URL
      194  +       await page.goto(validatedUrl.href, { waitUntil: 'networkidle2', timeout:
          30000 });
      195  +
119  196          // Run axe on the page
120  197          const axeOptions = {};
121  198          if (args.tags && args.tags.length > 0) {
```

```
@@ -192,18 +269,21 @@ class A11yServer {
192 269       }
193 270
194 271       try {
    272 +       // Validate URL to prevent SSRF
    273 +       const validatedUrl = await validateUrl(args.url);
    274 +
195 275         const browser = await puppeteer.launch({
196 276           headless: 'new',
197 277           args: ['--no-sandbox', '--disable-setuid-sandbox'],
198 278         });
199 279         const page = await browser.newPage();
200     -
    280 +
201 281         // Set a reasonable viewport
202 282         await page.setViewport({ width: 1280, height: 800 });
203     -
204     -       // Navigate to the page
205     -       await page.goto(args.url, { waitUntil: 'networkidle2', timeout: 30000 });
206     -
    283 +
    284 +       // Navigate to the page using the validated URL
    285 +       await page.goto(validatedUrl.href, { waitUntil: 'networkidle2', timeout:
        30000 });
    286 +
207 287         // Run axe on the page
208 288         const results = await new AxePuppeteer(page).analyze();
209 289
```

Comments 0

Please sign in to comment.