PrefectHQ / **fastmcp** Public

Code Issues 211 Pull requests 12 Discussions Actions Projects

Commit `40bdfb6`

jlowin and **claude** authored on Mar 15 · ✕ 12 / 15 · Verified

```
fix: URL-encode path params to prevent SSRF/path traversal (GHSA-vv7q-7jx5-f767)
(#3507)

* fix: URL-encode path params in OpenAPI provider to prevent SSRF/path traversal

Co-authored-by: Claude <noreply@anthropic.com>

* Exempt too-long from core-category requirement in triage

* fix: also encode dots in path params to prevent bare .. traversal

* fix: only encode .. (not all dots) to preserve valid dotted values

* fix: encode all dots in path params to prevent single-dot normalization

* fix: check decoded path stays within prefix in double-encoding test

---------

Co-authored-by: Claude <noreply@anthropic.com>
```

main (#3507) · v3.2.4 ••• v3.2.0

1 parent c861862 commit 40bdfb6

2 files changed +134 -3

Top

Filter files…

- src/fastmcp/utilities/openapi
  - director.py
- tests/utilities/openapi
  - test_director.py

Search within code

src/fastmcp/utilities/openapi/director.py

```
@@ -1,7 +1,7 @@
"""Request director using openapi-core for stateless HTTP request building."""

from typing import Any
- from urllib.parse import urljoin
+ from urllib.parse import quote, urljoin

import httpx
from jsonschema_path import SchemaPath
@@ -205,12 +205,14 @@ def _build_url(
        Returns:
            Complete URL with path parameters substituted
        """
-        # Substitute path parameters
+        # Substitute path parameters with URL-encoding to prevent
+        # path traversal and SSRF via crafted parameter values
        url_path = path_template
        for param_name, param_value in path_params.items():
            placeholder = f"{{{param_name}}}"
            if placeholder in url_path:
-                url_path = url_path.replace(placeholder, str(param_value))
+                safe_value = quote(str(param_value), safe="").replace(".", "%2E")
+                url_path = url_path.replace(placeholder, safe_value)

        # Combine with base URL
        return urljoin(base_url.rstrip("/") + "/", url_path.lstrip("/"))
```

tests/utilities/openapi/test_director.py

```
@@ -1,5 +1,7 @@
"""Unit tests for RequestDirector."""

+ from urllib.parse import unquote
```

```
      4 +
3     5   import pytest
4     6   from jsonschema_path import SchemaPath
5     7
        ↕ @@ -460,3 +462,130 @@ def test_with_deepobject_spec(self, deepobject_spec):
460   462
461   463               assert request.method == "GET"
462   464               assert
      str(request.url).startswith("https://api.example.com/search")
      465 +
      466 +
      467 + class TestPathTraversalPrevention:
      468 +     """Test that path parameter values are URL-encoded to prevent SSRF/path
      traversal."""
      469 +
      470 +     @pytest.fixture
      471 +     def director(self, basic_openapi_30_spec):
      472 +         spec = SchemaPath.from_dict(basic_openapi_30_spec)
      473 +         return RequestDirector(spec)
      474 +
      475 +     @pytest.fixture
      476 +     def path_route(self):
      477 +         return HTTPRoute(
      478 +             path="/api/v1/users/{id}/profile",
      479 +             method="GET",
      480 +             operation_id="get_user_profile",
      481 +             parameters=[
      482 +                 ParameterInfo(
      483 +                     name="id",
      484 +                     location="path",
      485 +                     required=True,
      486 +                     schema={"type": "string"},
      487 +                 )
      488 +             ],
      489 +             flat_param_schema={
      490 +                 "type": "object",
      491 +                 "properties": {"id": {"type": "string"}},
      492 +                 "required": ["id"],
      493 +             },
```

```
494 +             parameter_map={"id": {"location": "path", "openapi_name": "id"}},
495 +         )
496 +
497 +     @pytest.mark.parametrize(
498 +         "malicious_id",
499 +         [
500 +             "../../../admin/delete-all?",
501 +             "../../secret",
502 +             "../../../etc/passwd",
503 +             "foo/../../../admin",
504 +             "..%2F..%2Fadmin",
505 +             "..%2f..%2fadmin",
506 +         ],
507 +     )
508 +     def test_path_traversal_encoded(self, director, path_route, malicious_id:
      str):
509 +         request = director.build(
510 +             path_route, {"id": malicious_id}, "https://api.example.com"
511 +         )
512 +         url = str(request.url)
513 +         assert "/admin" not in url
514 +         assert "/secret" not in url
515 +         assert "/etc/passwd" not in url
516 +         assert url.startswith("https://api.example.com/api/v1/users/")
517 +
518 +     def test_slash_in_param_is_encoded(self, director, path_route):
519 +         request = director.build(path_route, {"id": "a/b"},
      "https://api.example.com")
520 +         url = str(request.url)
521 +         assert "/a/b/" not in url
522 +         assert "a%2Fb" in url
523 +
524 +     def test_dot_dot_slash_is_encoded(self, director, path_route):
525 +         request = director.build(
526 +             path_route, {"id": "../admin"}, "https://api.example.com"
527 +         )
528 +         url = str(request.url)
529 +         assert "%2E%2E%2Fadmin" in url or "%2e%2e%2fadmin" in url
530 +         assert url.startswith("https://api.example.com/api/v1/users/")
531 +
```

```
532 +    def test_question_mark_encoded(self, director, path_route):
533 +        request = director.build(
534 +            path_route, {"id": "foo?bar=baz"}, "https://api.example.com"
535 +        )
536 +        url = str(request.url)
537 +        assert "foo%3Fbar%3Dbaz" in url or "foo%3fbar%3dbaz" in url
538 +
539 +    def test_hash_encoded(self, director, path_route):
540 +        request = director.build(
541 +            path_route, {"id": "foo#fragment"}, "https://api.example.com"
542 +        )
543 +        url = str(request.url)
544 +        assert "foo%23fragment" in url
545 +
546 +    def test_normal_values_still_work(self, director, path_route):
547 +        request = director.build(
548 +            path_route, {"id": "user-123"}, "https://api.example.com"
549 +        )
550 +        assert (
551 +            str(request.url) == "https://api.example.com/api/v1/users/user-
    123/profile"
552 +        )
553 +
554 +    def test_dotted_values_encode_dots(self, director, path_route):
555 +        """Dots are encoded to prevent path normalization by urljoin."""
556 +        request = director.build(
557 +            path_route, {"id": "v1.2.3"}, "https://api.example.com"
558 +        )
559 +        url = str(request.url)
560 +        assert "v1%2E2%2E3" in url
561 +        assert url.startswith("https://api.example.com/api/v1/users/")
562 +
563 +    def test_numeric_values_still_work(self, director, path_route):
564 +        request = director.build(path_route, {"id": 42},
    "https://api.example.com")
565 +        assert str(request.url) ==
    "https://api.example.com/api/v1/users/42/profile"
566 +
567 +    def test_bare_single_dot_encoded(self, director, path_route):
568 +        """Bare '.' must be encoded so urljoin doesn't normalize it away."""
```

```
569 +         request = director.build(path_route, {"id": "."},
    "https://api.example.com")
570 +         url = str(request.url)
571 +         assert "%2E" in url
572 +         assert url.startswith("https://api.example.com/api/v1/users/")
573 +
574 +     def test_bare_dotdot_encoded(self, director, path_route):
575 +         """Bare '..' must be encoded so urljoin doesn't resolve it as
    traversal."""
576 +         request = director.build(path_route, {"id": ".."},
    "https://api.example.com")
577 +         url = str(request.url)
578 +         assert "%2E%2E" in url or "%2e%2e" in url
579 +         assert url.startswith("https://api.example.com/api/v1/users/")
580 +
581 +     def test_double_encoded_traversal(self, director, path_route):
582 +         request = director.build(
583 +             path_route,
584 +             {"id": "..%2F..%2Fadmin"},
585 +             "https://api.example.com",
586 +         )
587 +         url = str(request.url)
588 +         decoded = unquote(unquote(url))
589 +         # Verify traversal didn't escape the users/ prefix
590 +         assert decoded.startswith("https://api.example.com/api/v1/users/")
591 +         assert url.startswith("https://api.example.com/api/v1/users/")
```

Comments 0

Please sign in to comment.