New issue

[Bug]: Missing `root_api_key` enables anonymous ROOT access (broken access control) #302

Closed #310

Assignees

Labels bug

13ernkastel opened on Feb 26 Contributor

Bug Description

Bug Description

When OpenViking server starts without `server.root_api_key` configured, authentication is effectively disabled and requests can be resolved as `Role.ROOT` without any API key. This allows anonymous callers to access protected endpoints and perform administrative actions.

This is a broken access control issue caused by configuration fallback behavior.

Affected Surface

When `root_api_key` is not configured, all endpoints that depend on `get_request_context` / `require_role` are effectively impacted because requests are resolved as `Role.ROOT`.

Examples of affected routes:

- `/api/v1/admin/*` (account/user lifecycle, key rotation)
- `/api/v1/resources/*`
- `/api/v1/fs/*`
- `/api/v1/content/*`
- `/api/v1/search/*`

- `/api/v1/relations/*`
- `/api/v1/sessions/*`
- `/api/v1/pack/*`
- `/api/v1/debug/*`
- `/api/v1/observer/*`
- `/api/v1/system/status` and `/api/v1/system/wait`

Code-Level Explanation

1. **Startup behavior in `create_app()`**

- In `openviking/server/app.py`, the lifespan hook checks `if config.root_api_key`.
- If set, it initializes `APIKeyManager` and stores it in `app.state.api_key_manager`.
- If unset, it explicitly sets `app.state.api_key_manager = None` and logs that authentication is disabled.

2. **Identity resolution in `resolve_identity()`**

- In `openviking/server/auth.py`, every protected route eventually depends on `get_request_context`, which depends on `resolve_identity`.
- `resolve_identity()` first reads `api_key_manager = getattr(request.app.state, "api_key_manager", None)`.
- If that manager is `None`, the function returns `ResolvedIdentity(role=Role.ROOT, ...)` **without requiring any API key**.

3. **Authorization effect in `require_role()`**

- `require_role()` checks whether the resolved context role is in allowed roles.
- Because the fallback role is ROOT, checks like `require_role(Role.ROOT)` and `require_role(Role.ROOT, Role.ADMIN)` pass immediately.

4. **Resulting trust model break**

- Authentication state (configured vs not configured) changes authorization semantics globally.
- A deployment mistake (missing `root_api_key`) becomes an automatic privilege grant.

Affected Code Blocks (Broken Access Control)

1) Auth initialization fallback (`openviking/server/app.py`)

```
# Initialize APIKeyManager after service (needs AGFS)
if config.root_api_key:
api_key_manager = APIKeyManager(
root_key=config.root_api_key,
agfs_url=service._agfs_url,
)
await api_key_manager.load()
app.state.api_key_manager = api_key_manager
logger.info("APIKeyManager initialized")
else:
app.state.api_key_manager = None
logger.info("Dev mode: no root_api_key configured, authentication disabled")
```

2) Identity resolver ROOT fallback (`openviking/server/auth.py`)

```
api_key_manager = getattr(request.app.state, "api_key_manager", None)

if api_key_manager is None:
return ResolvedIdentity(
role=Role.ROOT,
account_id=x_openviking_account or "default",
user_id=x_openviking_user or "default",
agent_id=x_openviking_agent or "default",
)
```

3) Role gate that trusts resolved context (`openviking/server/auth.py`)

```
def require_role(*allowed_roles: Role):
async def _check(ctx: RequestContext = Depends(get_request_context)):
if ctx.role not in allowed_roles:
raise PermissionDeniedError(
f"Requires role: {', '.join(r.value for r in allowed_roles)}"
)
return ctx

return Depends(_check)
```

4) Broken Access Control impact across routers

The following route modules enforce access via `Depends(get_request_context)` or `require_role(...)`, so they are affected when identity resolution returns anonymous ROOT:

- `openviking/server/routers/admin.py` (ROOT/ADMIN sensitive operations)
- `openviking/server/routers/resources.py`
- `openviking/server/routers/filesystem.py`
- `openviking/server/routers/content.py`
- `openviking/server/routers/search.py`
- `openviking/server/routers/relations.py`
- `openviking/server/routers/sessions.py`
- `openviking/server/routers/pack.py`
- `openviking/server/routers/debug.py`
- `openviking/server/routers/observer.py`
- `openviking/server/routers/system.py` (notably `/api/v1/system/status` and `/api/v1/system/wait`; `/health` and `/ready` are intentionally public)

Steps to Reproduce

Steps to Reproduce

1. Configure OpenViking server with a `server` section that omits `root_api_key`.
2. Start the server normally.
3. Call a protected endpoint **without** `X-API-Key` or `Authorization`:
   `GET /api/v1/system/status`
4. Call a ROOT-only endpoint **without credentials**:
   `GET /api/v1/admin/accounts`
5. (Optional) Attempt admin mutation without credentials:
   `POST /api/v1/admin/accounts` with valid JSON payload.

Expected Behavior

Protected endpoints should reject unauthenticated requests with `401` (or `403` where applicable). ROOT-only endpoints must never succeed without a valid root API key.

Actual Behavior

Requests without authentication can succeed when `root_api_key` is absent. Access context may be resolved as ROOT, allowing protected and admin operations to proceed.

Minimal Reproducible Example

```
# No API key headers provided
curl -i http://127.0.0.1:1933/api/v1/system/status
curl -i http://127.0.0.1:1933/api/v1/admin/accounts

# Optional admin write path
curl -i -X POST http://127.0.0.1:1933/api/v1/admin/accounts \
  -H 'Content-Type: application/json' \
  -d '{"account_id":"poc-account","admin_user_id":"poc-admin"}'
```

Error Logs

```
No explicit error is required to trigger this bug. Relevant startup log may indicate de
```

OpenViking Version

v0.1.18

Python Version

3.10+

Operating System

Linux

Model Backend

OpenAI

Additional Context

Affected code path

- `openviking/server/app.py`: sets `app.state.api_key_manager = None` if `config.root_api_key` is missing.
- `openviking/server/auth.py`: `resolve_identity()` returns `ResolvedIdentity(role=Role.ROOT, ...)` when `api_key_manager is None`.
- `openviking/server/auth.py`: `require_role(...)` then accepts the ROOT role for protected endpoints

Security impact

- Category: Broken Access Control / Security Misconfiguration

- Risk: High (anonymous privilege escalation to administrative scope)

Suggested fix

- Fail closed when `root_api_key` is missing (except explicit local-only insecure mode).
- Disallow insecure mode unless bound to localhost.
- Add startup validation and strong warning/error for unsafe config.

13ernkastel added bug on Feb 26

github-project-automation added this to OpenViking project on Feb 26

github-project-automation moved this to Backlog in OpenViking project on Feb 26

qin-ctx assigned qin-ctx and unassigned qin-ctx on Feb 26

qin-ctx added a commit that references this issue on Feb 26

`fix(server): 未配置 root_api_key 时仅允许 localhost 绑定` ··· 0251c70

qin-ctx mentioned this on Feb 26

fix(server): 未配置 root_api_key 时仅允许 localhost 绑定 #310

github-project-automation moved this from Backlog to In progress in OpenViking project on Feb 26

qin-ctx closed this as completed in #310 on Feb 26

github-project-automation moved this from In progress to Done in OpenViking project on Feb 26

ZaynJarvis added a commit that references this issue on Feb 26

`fix(server): 未配置 root_api_key 时仅允许 localhost 绑定 (volcengine#310` ···

Verified 9e69113

Metadata

Assignees

qin-ctx

Labels

bug

Type

No type

Projects

OpenViking project

Status Done

Milestone

No milestone

Relationships

None yet

Development

fix(server): 未配置 root_api_key 时仅允许 localhost 绑定
volcengine/OpenViking

cli@0.2.0

Participants