

New issue



Security Risk: 4 Secrets Key Detector in 4 files #110

Open

Labels

automated high-priority prosecurelab secrets security



secrets-key-detector bot opened now

Contributor



Executive Security Report: Secret Detection Summary



Repository: meganathan44/demo_repo



Branch Scanned: main



Total Secrets Detected: 4



Files Affected: 4



Scan Date: 2025-09-23 11:44:10 UTC



Scanner Used: ProSecureLab Shield vunknown



Summary of Findings

File	Total Secrets
README.md	1
exapmle.py	1
main.py	1
workflows/dependancy-check.yml	1



Detailed Findings by File



README.md (Total Secrets: 1)

Secret Type	Line	Status	Excerpt
Generic Token	5	Unknown	ghp_...z123

exapmle.py (Total Secrets: 1)


Secret Type	Line	Status	Excerpt
Generic Token	12	Unknown	sk-l...CD78

main.py (Total Secrets: 1)

Secret Type	Line	Status	Excerpt
Generic Token	1692	Unknown	Acce...ials

workflows/dependancy-check.yml (Total Secrets: 1)

Secret Type	Line	Status	Excerpt
Generic Token	164	Unknown	cont...mber

 Although the secrets are flagged as invalid, their exposure indicates poor credential management practices and requires remediation.

Immediate Actions Required

To minimize risk and ensure secure operations, the following actions should be prioritized:

- **Rotate any exposed secrets immediately**
- **Purge secrets from codebase history**
- **Update all systems or environments using these credentials**
- **Audit access logs for suspicious or unauthorized activity**

Recommended Remediation Steps

To strengthen overall secret management and avoid recurrence:

- Conduct a full review of the affected files and adjacent modules
- Migrate all secrets to environment variables or secure vaults
- Establish secret management using tools like:
 - AWS Secrets Manager
 - HashiCorp Vault
 - Azure Key Vault
- Enforce pre-commit hooks with integrated secret scanning

Prevention Best Practices

Do's

- Store secrets outside the codebase using environment variables
- Apply routine secret rotation and access review policies
- Use GitHub's built-in secret scanning for early detection
- Conduct regular internal security audits and penetration testing

❌ Don'ts

- Never commit credentials or API keys to version control
- Avoid hardcoding secrets in source files or config files
- Do not reuse the same secret across different environments

This issue has been raised by **ProSecureLab**'s automated security compliance framework. Please treat this issue as urgent. Any delay in addressing exposed credentials may result in critical security compromise.

📖 Reference Resources

Helpful links to deepen your understanding of secure secret management:

- [GitHub Secret Scanning](#)
- [GitGuardian Documentation](#)
- [OWASP Security Guidelines](#)

🐙 GitHub Docs

🔗 [Keeping secrets secure with secret scanning - GitHub Docs](#)

Let GitHub do the hard work of ensuring that tokens, private keys, and other code secrets are not exposed in your repository.

🌐 owasp.org

🔗 [OWASP Foundation, the Open Source Foundation for Application Security](#)

OWASP Foundation, the Open Source Foundation for Application Security on the main website for The OWASP Foundation. OWASP is a nonprofit foundation that works to improve the security of software.



secrets-key-detector added **security** **secrets** **automated** **high-priority**
prosecurelab now

[Sign up for free](#) to join this conversation on GitHub. Already have an account? [Sign in to comment](#)

Metadata

Assignees

No one assigned

Labels

automated **high-priority** **prosecurelab** **secrets** **security**

Projects

No projects


Milestone

No milestone

Relationships

None yet

Development

 Code with agent mode

▼

No branches or pull requests

Participants

No participants