Secrets Scanning: Hands-On, Recap & Q&A

# 1. What is Secrets Scanning?

Secrets Scanning is a security technique used to detect sensitive information (like API keys, passwords, tokens, etc.) accidentally exposed in source code, logs, or other text-based content.

Why it matters:

- Exposed secrets can lead to unauthorized access.  
- Automated scanners prevent breaches before code is pushed or deployed.  
- It ensures compliance and protects infrastructure.

# 2. Hands-On: Performing Secrets Scanning

Tools commonly used:

- GitHub Secret Scanning  
- GitLeaks  
- TruffleHog  
- Talisman  
- Detect Secrets (by Yelp)

Steps: Using GitLeaks (example)

Installation:

brew install gitleaks # Mac

Scan a local repo:

gitleaks detect --source .

Scan a remote GitHub repo:

gitleaks detect --source=https://github.com/org/repo

Example Output:

{  
 "RuleID": "Generic API Key",  
 "Secret": "AKIAIOSFODNN7EXAMPLE",  
 "File": "config.py",  
 "StartLine": 23  
}

Action on detection:

- Revoke the secret immediately  
- Rotate keys  
- Purge the secret from Git history using tools like BFG Repo-Cleaner

# 3. Real-Time Example Scenario

Scenario:  
A developer accidentally commits an AWS key in settings.py

AWS\_ACCESS\_KEY = "AKIAIOSFODNN7EXAMPLE"

Detection with Gitleaks: Gitleaks flags this line based on regex patterns.

Remediation:

1. Remove the key.  
2. Push a fix commit.  
3. Invalidate the compromised key in the AWS console.  
4. Add the file to .gitleaksignore or use environment variables instead.

# 4. Recap: Key Takeaways

|  |  |
| --- | --- |
| Feature | Description |
| Purpose | Identify hardcoded secrets before they reach production |
| Tools | GitLeaks, TruffleHog, GitHub Secret Scanning |
| Best Practices | Use .env files, CI/CD scanning, secrets management tools |
| Remediation | Revoke, rotate, revalidate secrets quickly |

# 5. Q&A Session Summary

Q1. Can secrets scanning prevent secrets from being pushed?

A: Yes, pre-commit hooks and CI/CD integration can block pushes with secrets.

Q2. What files are commonly ignored during scans?

A: Binary files, .gitignore items, or those listed in .gitleaksignore.

Q3. What are alternatives to storing secrets in code?

A: Use: AWS Secrets Manager, Azure Key Vault, HashiCorp Vault, GitHub Encrypted Secrets (for Actions)

# 6. Bonus: CI/CD Integration

GitHub Actions Example:

name: Secrets Scan  
  
on: [push]  
  
jobs:  
 gitleaks:  
 runs-on: ubuntu-latest  
 steps:  
 - uses: actions/checkout@v3  
 - name: Run Gitleaks  
 uses: zricethezav/gitleaks-action@v2