

TESTPASSPORT

練習題



質 量 更 高

服 務 更 好

<https://www.testpassport.net>

免費一年更新服務

Exam : CAPen

**Title : Certified AppSec Pentester
(CAPen)**

Version : DEMO

1. Use a Google dork to identify login pages of vulnerable sites running PHP.

A. See the Explanation.

Answer: A

Explanation:

1. Open Google and use the dork: `inurl:login.php intitle:"Login"`
2. This query targets pages with "login.php" in the URL and "Login" in the title.
3. Review results and pick non-government, non-critical sites for testing only.
4. Combine with `site:` to target specific domains, e.g., `site:.edu`.
5. Validate findings using a test environment to ensure ethical usage.

2. Find public documents on a government site that may contain sensitive metadata using Google dorks.

A. See the Explanation.

Answer: A

Explanation:

1. Use the dork: `site:gov filetype:pdf`
2. To find specific content, extend it: `site:gov filetype:pdf confidential`
3. Download a few sample PDFs (legally and ethically).
4. Use `exiftool filename.pdf` or `pdftinfo filename.pdf` to extract metadata.
5. Check for usernames, software versions, or timestamps that leak OSINT data.

3. Discover open directories with potentially sensitive files using Google dorks.

A. See the Explanation.

Answer: A

Explanation:

1. Use: `intitle:"index of" "parent directory" +passwd`
2. This searches for open directories with file listings containing "passwd".
3. Replace `passwd` with other keywords like `.sql`, `.bak`, `.env`, etc.
4. Confirm findings are in non-sensitive environments or demos.
5. Access is legal only if no authentication is bypassed or required.

4. Identify email addresses belonging to a company using OSINT techniques.

A. See the Explanation.

Answer: A

Explanation:

1. Use Hunter.io or theHarvester tool: `theHarvester -d target.com -b google`
2. Search Google with: `@target.com`
3. Use LinkedIn or GitHub advanced search: `site:github.com "@target.com"`
4. Collect emails for phishing simulation (if permitted).
5. Verify email validity using tools like EmailHippo or Debounce.io.

5. Find GitHub repositories accidentally exposing sensitive credentials.

A. See the Explanation.

Answer: A

Explanation:

1. Use GitHub search with: filename:.env SMTP_PASSWORD
2. Try: filename:.git-credentials or AWS_SECRET_ACCESS_KEY
3. Use Google dork: site:github.com "AWS_SECRET_ACCESS_KEY"
4. Examine the commit history to verify it's not a dummy key.
5. Report using GitHub's responsible disclosure if real data is found.