# A05:2021 — Security Misconfiguration

## Introduction

La catégorie **A05:2021 – Security Misconfiguration** regroupe les failles liées à une mauvaise configuration des systèmes, serveurs, applications ou services cloud.
Elle se hisse à la **5e place du Top 10 OWASP 2021**, en hausse par rapport à la 6e position précédente, car **près de 90 % des applications testées présentent au moins une forme de mauvaise configuration**.
Avec un **taux d'incidence moyen de 4,51 %**, plus de **208 000 occurrences** et **789 CVE** répertoriées, cette catégorie illustre l'ampleur du problème.
Les causes typiques incluent l'activation de fonctionnalités inutiles, les comptes par défaut non changés, les messages d'erreur trop détaillés ou encore l'absence d'en-têtes de sécurité HTTP.
Cette montée dans le classement s'explique par la généralisation de logiciels et services hautement configurables : sans un processus de durcissement automatisé et reproductible, chaque composant devient une source potentielle d'exposition.

---

## Scénario 1 — Configuration / Backup File Disclosure

**Environnement**: bWAPP local (`http://127.0.0.1:8081`).
**But**: démontrer l'exposition de fichiers sensibles accessibles par HTTP conduisant à la divulgation d'identifiants.

### Étapes

1. Récupération d'une wordlist (SecLists `common.txt`)

```
curl -sS -o /home/env-admin/common.txt \
  https://raw.githubusercontent.com/danielmiessler/SecLists/master/Discovery/Web-
Content/common.txt
```

2. Scan du site web avec wordlist

```
gobuster dir -u http://127.0.0.1:8081/ -w /tmp/common.txt -x php,inc,bak,zip -t 40
-o /tmp/gobuster_simple.txt
```

3. Résultats pertinents (extraits)

```
/config.inc          (Status: 200) [Size: 774]          <-- **fichier de
configuration lisible**
/portal.zip          (Status: 200) [Size: 5396]         <-- **archive exposée**
/phpinfo.php         (Status: 200) [Size: 78557]        <-- **informations
système détaillées**
```

```
/server-status       (Status: 403) [Size: 291]          <-- accès refusé
(partiel)
```

> **Éléments clés**: `config.inc` accessible (200), artefacts de déploiement (`.zip`) exposés, page diagnostique `phpinfo.php` ouverte.

4. Téléchargement du fichier

```
curl -sS http://127.0.0.1:8081/config.inc -o /tmp/config.inc
```

5. Secrets extraits (extrait minimal)

```
$server   = "localhost";
$username = "bwapp";
$password = "bwApped";
$database = "bWAPP";
```

---

# Correction — Mesures de remédiation

## Objectif

Supprimer l'exposition des secrets via HTTP. Déplacer la configuration hors webroot. Bloquer l'accès direct. Nettoyer les artefacts. Appliquer le moindre privilège. Prévenir les régressions via CI/CD et scans réguliers.

## Principes

1. **Séparation stricte**: configuration et secrets **hors** webroot.
2. **Contrôle d'accès**: refus par défaut d'accès HTTP aux fichiers sensibles.
3. **Moindre privilège**: permissions minimales sur fichiers et processus.
4. **Hygiène de dépôt**: aucun secret en VCS.
5. **Automatisation**: durcissement reproductible à chaque déploiement.
6. **Surveillance**: DAST, forced-browse, inventaire continu.

## 1) Sortir les secrets du webroot

```
# Illustration (Linux + Apache)
sudo mkdir -p /var/www/secure
sudo mv /var/www/html/bWAPP/config.inc /var/www/secure/config.inc
sudo chown root:www-data /var/www/secure/config.inc
sudo chmod 640 /var/www/secure/config.inc
```

```
<?php
// Chargement de la configuration déplacée
```

```
require '/var/www/secure/config.inc';
```

**Impact attendu**: inaccessibilité HTTP même en cas de règle défaillante.

### 2) Variables d'environnement ou vault

```
# /var/www/secure/.env (hors webroot)
DB_HOST=localhost
DB_USER=bwapp
DB_PASS=bwApped
DB_NAME=bWAPP
```

```
<?php
$env = parse_ini_file('/var/www/secure/.env');
$db_host = $env['DB_HOST'] ?? 'localhost';
$db_user = $env['DB_USER'] ?? '';
$db_pass = $env['DB_PASS'] ?? '';
$db_name = $env['DB_NAME'] ?? '';
```

**Bénéfice**: rotation facilitée et absence de secrets en clair dans le code.

### 3) Bloquer l'accès HTTP aux fichiers sensibles

**Apache (vhost ou .htaccess)**

```
<FilesMatch "(?i)^(config\.inc|wp-config\.php|web\.config|.*\.bak|.*~|.*\.zip)$">
  Require all denied
</FilesMatch>
Options -Indexes
ServerSignature Off
ServerTokens Prod
<IfModule mod_headers.c>
  Header always set X-Content-Type-Options "nosniff"
  Header always set X-Frame-Options "SAMEORIGIN"
  Header always set Referrer-Policy "no-referrer-when-downgrade"
</IfModule>
```

**Nginx (server block)**

```
location ~* (^/config\.inc$|/wp-config\.php$|/web\.config$|\.bak$|~$|\.zip$) {
deny all; }
autoindex off;
server_tokens off;
add_header X-Content-Type-Options nosniff always;
```

```
add_header X-Frame-Options SAMEORIGIN always;
add_header Referrer-Policy "no-referrer-when-downgrade" always;
```

### 4) Nettoyer backups et artefacts

```
sudo find /var/www/html/bWAPP -maxdepth 1 -type f \( -name "*.bak" -o -name "*~" -
o -name "*.zip" -o -name "*.tar.gz" -o -name "*.sql" \) -print
sudo mkdir -p /var/backups/bWAPP
sudo mv /var/www/html/bWAPP/*.bak /var/backups/bWAPP/ 2>/dev/null || true
```

### 5) Permissions minimales

```
sudo chown -R www-data:www-data /var/www/html/bWAPP
sudo find /var/www/html/bWAPP -type d -exec chmod 750 {} \;
sudo find /var/www/html/bWAPP -type f -exec chmod 640 {} \;
sudo chown root:www-data /var/www/secure/config.inc
sudo chmod 640 /var/www/secure/config.inc
```

### 6) Hygiène Git et pipeline CI

```
# Secrets
config.inc
.env
*.bak
*~
*.zip
*.tar.gz
*.sql
```

```
name: security-checks
on: [push, pull_request]
jobs:
  secrets-scan:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - name: TruffleHog scan
        uses: trufflesecurity/trufflehog@v3
        with:
          path: .
          base: github
          extra_args: "--only-verified"
```

### 7) Durcissement cohérent (IaC)

```
- hosts: web
  become: yes
  tasks:
    - name: Ensure secure dir exists
      file: path=/var/www/secure state=directory owner=root group=www-data
mode=0750
    - name: Move config out of webroot
      command: mv /var/www/html/bWAPP/config.inc /var/www/secure/config.inc
      args: { removes: /var/www/html/bWAPP/config.inc }
    - name: Apache hardening
      copy:
        dest: /etc/apache2/conf-available/security-hardening.conf
        content: |
          ServerSignature Off
          ServerTokens Prod
          <Directory /var/www/html/bWAPP>
            Options -Indexes
          </Directory>
    - name: Enable conf and reload
      command: a2enconf security-hardening
    - name: Reload apache
      service: name=apache2 state=reloaded
```

---

## Vérifications post-correctifs

**HTTP**

```
curl -I http://127.0.0.1:8081/config.inc
```

**Attendu**: `403` ou `404`.

**Forced-browse**

```
gobuster dir -u http://127.0.0.1:8081/ -w /home/env-admin/common.txt -x
php,inc,bak,zip -t 40 -o /tmp/after_fix.txt
egrep " (200|301|302) " /tmp/after_fix.txt || echo "OK: aucun artefact
accessible."
```

**Permissions**

```
ls -l /var/www/secure/config.inc
ls -l /var/www/html/bWAPP | sed -n '1,100p'
```

## Risques résiduels et suivi

- **Risque humain**: réintroduction d'artefacts (`.bak`, archives). **Mesure**: hooks pré-commit, scans secrets CI, revue de code.
- **Divergence d'environnements**: Dev ≠ Prod. **Mesure**: IaC et exécution systématique des playbooks.
- **Régression**: changement de système ou de serveur. **Mesure**: checklists de durcissement et tests automatisés post-déploiement.

## Conclusion

Exposition de `config.inc`, d'archives de déploiement et de pages de diagnostic confirmée. Mesures proposées éliminent le vecteur principal (configuration hors webroot, interdictions serveur, nettoyage, permissions, CI/CD et IaC). Résultat attendu: **surface d'attaque réduite**, **secrets inaccessibles par HTTP**, et **contrôles automatisés** pour empêcher la réapparition de la vulnérabilité.