ZABALA GAILETAK, S.A.

EKOIZPEN SEGURUAN JARTZEA

SSDLC eta DevSecOps — Garapen Segurua

Software Garapen Bizi-ziklo Seguruaren Inplementazioa

Dokumentu Kodea:	EKOK-ZG-001
Bertsioa:	1.0
Data:	2026-02-19
Ikasturtea:	2026
Sailkapena:	Heziketa — Barne Erabilera
Egilea:	Zabala Gailetak Zibersegurtasun Taldea

1. Sarrera — DevSecOps eta SSDLC

Software Garapen Bizi-ziklo Segurua (SSDLC) segurtasuna diseinutik kodearen hedapena arte integratzen dituen metodologia da. Zabala Gailetak-en DevSecOps kultura ezartzea da helburu nagusia: segurtasuna 'shift left' eginez, akatsak goizago eta merkeago konpontzeko.

1.1 Formalki ezarritako faseak

Fasea	Ekintza Nagusia	Tresnak
Plangintza	Arriskuen analisia, eskakizun seguroak	Threat Modeling, STRIDE
Diseinua	Arkitektura segurua, Defense in Depth	OWASP TM / LINDDUN
Garapena	Coding seguruak, ez hardcoded credentials	SonarLint, ESLint, PHPStan
Testing	SAST, DAST, SCA, Pentest	SonarQube, OWASP ZAP, Snyk
Berrikuspena	Pull Request code review segurtasun-fokua	GitHub / GitLab PR checks
Hedapena	IaC, konfigurazio gogortu, sekretu kudeaketa	Ansible, Vault, Docker Bench
Eragiketak	SIEM, monitoreo, adabaki kudeaketa	ELK + Wazuh, Dependabot

2. Threat Modeling — STRIDE Metodologia

Aplikazio berrirako STRIDE mehatxu-modelizazioa egin da diseinu fasean:

Mehatxu Mota	Deskribapena	Kontrol Neurria
S — Spoofing (Iruzurra)	Besteen identitatea hartu	MFA + JWT sinadura
T — Tampering (Manipulazioa)	Datuak modifikatu	HTTPS + HMAC
R — Repudiation (Ukatzea)	Ekintzak ukatu	Audit log sinatu
I — Info Disclosure	Datu sentikorrak filtratu	Enkriptazioa AES-256
D — Denial of Service	Zerbitzu etena	Rate limiting + CDN
E — Elevation of Privilege	Baimenik gabeko pribilegioaren eskuratzea	RBAC + Least Privilege

3. CI/CD Pipeline Segurua

Zabala Gailetak-en CI/CD pipeline segurua GitHub Actions eta Docker-ekin ezarri da:

```
# .github/workflows/security.yml (laburpena) jobs: sast: runs-on: ubuntu-latest steps:
- uses: actions/checkout@v4 - name: SonarQube SAST Analisia uses:
SonarSource/sonarqube-scan-action@master sca: steps: - name: Snyk SCA Dependentzia
Azterketa run: snyk test --severity-threshold=high dast: steps: - name: OWASP ZAP DAST
Eskaneatzea uses: zaproxy/action-full-scan@v0.9.0
```

4. Docker Kontainer Segurtasuna

4.1 Dockerfile Zergatik Printzipioak

- Oinarri-irudi minimoak erabili: alpine edo distroless.
- Root izateko EZ exekutatu: USER appuser ezarri.
- Gehitu .dockerignore fitxategia kode sentikorra kanpoan uzteko.
- Irudian sekreturik EZ gorde — erabili Docker Secrets edo Vault.
- Irudi guztiak Trivy-z eskaneatu push egin aurretik.

```
# Dockerfile segurua — adibidea FROM php:8.4-fpm-alpine # Oinarri txikia RUN addgroup -S appgroup && adduser -S appuser -G appgroup COPY --chown=appuser:appgroup . /app USER appuser # Ez erabili root HEALTHCHECK CMD curl -f http://localhost/ || exit 1
```

5. Ekoizpeneko Konfigurazio Segurua

5.1 HTTPS eta TLS 1.3

```
# Nginx TLS konfigurazioa — zati segurua ssl_protocols TLSv1.3; # TLS 1.0/1.1/1.2 desgaitu ssl_ciphers 'TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256'; ssl_prefer_server_ciphers off; add_header Strict-Transport-Security 'max-age=63072000; includeSubDomains; preload'; add_header X-Content-Type-Options nosniff; add_header X-Frame-Options DENY; add_header Content-Security-Policy "default-src 'self'";
```

5.2 Sekretu Kudeaketa

- HashiCorp Vault erabili sekretu guztiak (DB pasahitzak, API gakoak) gordetzeko.
- .env fitxategiak **INOIZ EZ** commit egin Git-era.
- CI/CD-n GitHub Secrets erabili ingurune-aldagaietarako.
- Sekretuak errotatu aldian-aldian (90 egunero gutxienez).