root@hacklab:~/parte-02/lezione-21

ETHICAL HACKING POTENZIATO DALL'IA // PARTE II

LEZIONE 21

DevSecOps e Secure SDLC

Integrare la sicurezza nel ciclo di sviluppo e nella pipeline, con approccio shift-left.

docente **Vincenzo Calabro** fase Specializzazione: DevSecOps e Secure SDLC

root@hacklab:~/lezione-21/agenda $

LEZIONE 21 / AGENDA

In questa lezione

// PERCORSO

> Secure SDLC e threat modeling
> SAST, DAST, SCA
> Sicurezza di CI/CD
> Infrastructure as Code
> Supply chain e SBOM

metodo obiettivi -> teoria -> strumenti -> laboratorio -> caso di studio -> etica -> verifica

LEZIONE 21 / OBIETTIVI

Obiettivi di apprendimento

// AL TERMINE SAPRAI (TASSONOMIA DI BLOOM)

> Comprendere — il Secure SDLC e il threat modeling
> Applicare — SAST/DAST/SCA in una pipeline
> Analizzare — sicurezza di CI/CD e IaC
> Valutare — l'integrazione shift-left
> Creare — una pipeline sicura con gate

LEZIONE 21 / INQUADRAMENTO

Inquadramento

// DI COSA PARLIAMO

Spostare la sicurezza a sinistra significa integrarla nel ciclo di sviluppo, non aggiungerla alla fine. Vediamo threat modeling, SAST/DAST/SCA e sicurezza di pipeline e supply chain.

// PERCHE CONTA

Integrare la sicurezza nel ciclo di sviluppo e nella pipeline, con approccio shift-left.

LEZIONE 21 / MAPPA // INSIEME

Mappa d'insieme del modulo

Attacco/tecnica · Difesa · Norme & metodo · IA

FONDAMENTI

- 01 Secure SDLC e threat modeling
- 02 Test di sicurezza automatizzati
- 03 Sicurezza di pipeline e supply chain
- 04 IaC e policy-as-code
- 05 Cultura e metriche DevSecOps

APPROFONDIMENTI

- 06 STRIDE e abuse case
- 07 Gate di qualita e falsi positivi
- 08 Segreti, dipendenze e artefatti
- 09 Hardening dell'infrastruttura come codice
- 10 Metriche e miglioramento continuo

LEZIONE 21 / FLUSSO // FONDAMENTI

Sequenza di fasi — Fondamenti

Attacco/tecnica · Difesa · Norme & metodo · IA

> Le fasi si susseguono: l'esito di ciascuna alimenta la successiva.

LEZIONE 21 / CONCETTO 1/10

[Difesa]

Secure SDLC e threat modeling

// SPIEGAZIONE

La sicurezza entra nel ciclo di sviluppo fin dal design, non alla fine: il Secure SDLC integra requisiti e verifiche a ogni fase. Il threat modeling con STRIDE anticipa le minacce ragionando su cosa puo andare storto prima di scrivere il codice.

// ESEMPIO

```
> STRIDE sul login evidenzia spoofing ed elevazione di privilegio da mitigare nel design.
```

LEZIONE 21 / CONCETTO 2/10

[Attacco/tecnica]

Test di sicurezza automatizzati

// SPIEGAZIONE

SAST analizza il codice sorgente, DAST l'applicazione in esecuzione, IAST dall'interno a runtime, SCA le dipendenze di terze parti. Integrati nella pipeline con gate di qualita, bloccano i problemi prima del rilascio.

// ESEMPIO

> La pipeline fallisce la build se SCA trova una dipendenza con vulnerabilita critica nota.

LEZIONE 21 / CONCETTO 3/10

[Norme & metodo]

Sicurezza di pipeline e supply chain

// SPIEGAZIONE

La pipeline stessa e un bersaglio: segreti esposti, dipendenze compromesse, artefatti non verificati. Un SBOM elenca cosa contiene il software, abilitando risposte rapide quando emerge una vulnerabilita in un componente.

// ESEMPIO

> Con un SBOM si individua in minuti quali build includono una libreria appena dichiarata vulnerabile.

LEZIONE 21 / CONCETTO 4/10

[Difesa]

IaC e policy-as-code

// SPIEGAZIONE

L'infrastruttura come codice con Terraform e Kubernetes va verificata come il software: una misconfigurazione si propaga su scala. Policy-as-code impone regole automatiche e un secrets manager evita credenziali sparse nel codice.

// ESEMPIO

> Una policy blocca il deploy di un bucket di storage configurato come pubblico.

Cultura e metriche DevSecOps

// SPIEGAZIONE

DevSecOps e prima di tutto cultura: la sicurezza e responsabilita di tutti, non di un reparto a valle. Metriche come tempo di correzione e densita di vulnerabilita misurano i progressi e orientano il miglioramento continuo.

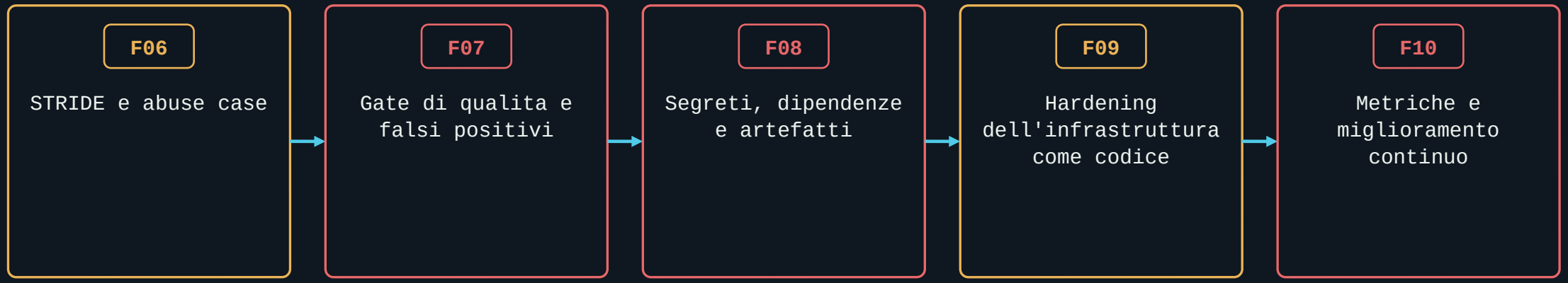
// ESEMPIO

> Si misura il tempo medio tra scoperta e fix di una vulnerabilita per vederlo calare nel tempo.

LEZIONE 21 / FLUSSO // APPROFONDIMENTI

Sequenza di fasi — Approfondimenti

Attacco/tecnica Difesa Norme & metodo IA



> Le fasi si susseguono: l'esito di ciascuna alimenta la successiva.

STRIDE e abuse case

// SPIEGAZIONE

STRIDE classifica le minacce in sei categorie: spoofing, tampering, repudiation, information disclosure, denial of service ed elevation. Gli abuse case descrivono come un attaccante userebbe male una funzione, guidando controlli mirati.

// ESEMPIO

```
> Abuse case 'l'utente altera il prezzo nel carrello' porta a validare il totale lato server.
```

LEZIONE 21 / CONCETTO 7/10

[Attacco/tecnica]

Gate di qualita e falsi positivi

// SPIEGAZIONE

I gate bloccano la pipeline oltre certe soglie di rischio, ma troppi falsi positivi spingono gli sviluppatori ad aggirarli. Senza una taratura attenta di regole e triage la sicurezza diventa un ostacolo da scavalcare, non un alleato.

// ESEMPIO

> Si marcano i falsi positivi noti perche non blocchino ogni build futura.

LEZIONE 21 / CONCETTO 8/10

[Attacco/tecnica]

Segreti, dipendenze e artefatti

// SPIEGAZIONE

I segreti non vanno nel codice ma in un vault; le dipendenze vanno fissate e verificate; gli artefatti firmati per garantirne l'origine. La supply chain e sicura solo se ogni anello e controllato e integro.

// ESEMPIO

> Firma degli artefatti e verifica in fase di deploy per impedire pacchetti manomessi.

Hardening dell'infrastruttura come codice

// SPIEGAZIONE

Scanner come Trivy o Checkov analizzano template Terraform e manifest Kubernetes prima del deploy; policy-as-code impone standard in modo automatico e ripetibile. Gli errori si correggono nel codice, non in produzione.

// ESEMPIO

```
$ checkov -d . (analizza i file IaC alla ricerca di misconfigurazioni)
```

LEZIONE 21 / CONCETTO 10/10

[Attacco/tecnica]

Metriche e miglioramento continuo

// SPIEGAZIONE

Senza misura non c'e miglioramento: tempo di remediation, percentuale di build che passano i gate, debito di sicurezza. Le metriche rendono visibili i progressi e sostengono la cultura oltre l'adozione degli strumenti.

// ESEMPIO

> Un cruscotto mostra il calo delle vulnerabilita aperte sprint dopo sprint.

LEZIONE 21 / STRUMENTI

Strumenti della lezione

// TOOLKIT

Semgrep | Dependency-Check | Trivy | OWASP ZAP | git-secrets | Checkov | SonarQube

// NOTA OPERATIVA

> Ogni strumento si usa in ambiente virtualizzato e isolato, solo su bersagli di proprieta o autorizzati. Output e artefatti vanno protetti e, a fine ingaggio, gestiti o distrutti.

root@hacklab:~/lezione-21/lab-1 $

LEZIONE 21 / LABORATORIO 1

Laboratorio 1

$ SAST/SCA in pipeline (gate + fix)

// PROCEDURA

[1] Integra SAST/SCA in una pipeline CI

[2] configura i gate e correggi i finding

[!] Verifica: Come si evita che i falsi positivi blocchino lo sviluppo?

root@hacklab:~/lezione-21/lab-2 $

LEZIONE 21 / LABORATORIO 2

Laboratorio 2

$ Segreti e IaC (Trivy/Checkov)

// PROCEDURA

[1] Scopri segreti committati e misconfigurazioni IaC (Trivy/Checkov) e rimedia

[!] Verifica: Perche la supply chain e un rischio sistemico?

LEZIONE 21 / CASO DI STUDIO

Caso di studio

// SCENARIO

> Log4Shell (2021) e SolarWinds (2020): lezioni di supply chain e dipendenze ubique.

// ANALISI RICHIESTA

> Ricostruisci la dinamica dei fatti con metodo e tracciabilita
> Individua errori, cause radice e punti di decisione critici
> Proponi contromisure, controlli e lezioni apprese

root@hacklab:~/lezione-21/etica $

LEZIONE 21 / ETICA & DIRITTO

Nota etico-legale

// PRINCIPI INDEROGABILI

> autorizzazione scritta scope definito ambiente isolato dati protetti responsible disclosure

// SPECIFICO DELLA LEZIONE

[!] Si opera sui propri repository e pipeline; i segreti scoperti vanno revocati e ruotati, mai divulgati.

LEZIONE 21 / RICAPITOLAZIONE

Ricapitolazione

// PUNTI CHIAVE

> Il Secure SDLC e il threat modeling
> SAST/DAST/SCA in una pipeline
> Sicurezza di CI/CD e IaC
> L'integrazione shift-left
> Una pipeline sicura con gate

LEZIONE 21 / AUTOVALUTAZIONE

Autovalutazione

// DOMANDE DI VERIFICA

? Come si evita che i falsi positivi blocchino lo sviluppo?

? Perche la supply chain e un rischio sistemico?

? A cosa serve un SBOM?

// VALUTAZIONE

Progetto di pipeline sicura con report e gate configurati.

// LETTURE

OWASP SAMM/ASVS; doc. Semgrep/Trivy.

lezione 21 completata

$ next

DevSecOps e Secure SDLC

[!] Le competenze di questa lezione si impiegano esclusivamente in contesti legali, autorizzati e deontologicamente corretti, entro lo scope e con autorizzazione scritta.

root@hacklab:~# _

root@hacklab:~/contatti $

CONTATTI / RISORSE

Restiamo in contatto

Vincenzo Calabro

cyber security / digital forensics /
docenza e divulgazione

// SCANSIONA O VISITA

> vincenzocalabro.it

Inquadra il QR con la fotocamera per aprire il sito.