DevSecOps mit Conjugate Intelligence (CI)

Zusammenfassung

Sicherheits-Signale sind heute oft voneinander getrennt (SAST, DAST, SCA, SIEM) und überfluten Teams mit Alarmen und Richtlinienchaos.
Conjugate Intelligence (CI) behandelt Sicherheit als gleichberechtigten Teil von Dev und Ops: Richtlinien werden zu **Vows**, Bedrohungen zu **Beziehungsmodellen**, und Audits zu **erzählten Nachweisen**.
So entsteht Sicherheit als gelebter Zusammenhang – nicht als Kontrollschicht.

Aktuelle Probleme

- **Shift-Left-Erschöpfung:** Zu viele Prüfungen, zu wenig Priorisierung nach Geschäftsrisko.
- **Richtlinien-Wildwuchs:** Ausnahmen ohne Nachvollziehbarkeit; kaum Erinnerung, warum sie gewährt wurden.
- **Manuelle Audit-Arbeit:** Nachweise müssen mühsam aus vielen Tools zusammengetragen werden.

CI-Ansatz (SpiralOS)

- **Sicherheit als Graph:**
  SBOM-Elemente, Schwachstellen, Code-Pfad, Datenflüsse, Identitäten und Kontrollen sind als Knoten mit Versionsgeschichte verknüpft.

- **Policy-Vows:**
  Durchsetzungsregeln reisen mit Code und Artefakten. Kontext (Bedrohungsmodell, kompensierende Kontrollen) bleibt erhalten.[1]

- **Erklärbare Aktionen:**
  Jede Entscheidung zum Blockieren oder Zulassen enthält Begründung und Nachweis – in verständlicher Sprache.

Zentrale Fähigkeiten

- **Blast-Radius-Mapping:**
  Bei einer neuen Schwachstelle (CVE)[2] zeigt CI innerhalb von Sekunden betroffene Dienste, Datenklassen und Expositionspfade.[3]
- **Dynamisches Gating:**
  Release-Gates berücksichtigen Ausnutzbarkeit, Kompensationsmaßnahmen und Einfluss auf Service-Levels – nicht nur CVSS-Scores.[4]
- **Bedrohungsmuster-Erkennung:**
  Verknüpft Signale aus Code-Änderungen, Infrastruktur-Drift und Laufzeit-Anomalien (MITRE-bewusstes Linking).
- **Audit-fertige Narrative:**
  Ein Klick erzeugt vollständige Kontrollnachweise (wer / was / warum / wann) für interne oder externe Prüfungen.
- **Ausnahme-Governance:**
  Zeitlich begrenzte Ausnahmen werden als *Perlen* mit Risiko-Verantwortlichen, Review-Datum und automatischen Erinnerungen geführt.[5]

Ergebnisse & Kennzahlen (KPIs)

- Behebungszeit kritischer Schwachstellen ↓ 30–60 %
- Falsch-Positiv-Rate ↓ 25–45 %
- Ausnahme-Schulden ↓ durch geplante Reviews
- Audit-Vorbereitungszeit ↓ 50–80 %

Integrationspfad (Reibungsarm)

1. **Signalaufnahme:**
   Einbindung von SAST / DAST / SCA, IaC-Scannern, Cloud-Konfigurationen, SIEM / EDR-Zusammenfassungen.
2. **SBOM-Verknüpfung:**
   Pakete / Abhängigkeiten werden den Diensten zugeordnet und an Commits bzw. Builds angehängt.
3. **Policy-Vows:**
   Einige zentrale Richtlinien werden mit Absicht und Nachweis-Hooks codiert.

4. **Gated Releases:**
   Zunächst beratend, später erzwingend – für Hochrisiko-Klassen.

Risiken & Gegenmaßnahmen

- **Entwickler-Reibung:**
  Einführung von *Friction Budgets* und schrittweiser Durchsetzung; CI erklärt, warum ein Gate ausgelöst wird.
- **Blind Spots:**
  Kontinuierliche Datenaufnahme aus Laufzeit- und Cloud-Umgebungen, um Infrastruktur- und Identitäts-Drift zu erkennen.

Beispiel – Neue Schwachstelle (CVE)

Eine kritische CVE betrifft eine häufig genutzte Bibliothek.
CI zeigt, welche Dienste und Datenklassen betroffen sind, schlägt Patch-Branches vor, erstellt sichere Rollout-Pläne
und entwirft Kommunikationsvorlagen für die Service-Owner.
Wenn eine temporäre Ausnahme nötig ist, dokumentiert CI die Begründung, setzt ein Review-Datum
und überwacht aktiv mögliche Exploit-Versuche.

Branchenbeispiele

- **Fertigung:** Erkennt schnell, welche SPS-Firmware von CVEs betroffen ist.
- **Mobilität:** Priorisiert Schwachstellen in sicherheitsrelevanten ADAS-Komponenten vor Infotainment-Systemen.
- **Energie:** Bewertet SCADA-Vulnerabilitäten mit Netz- und Versorgungssicherheit als Hauptgewichtung.

Gesprächsimpulse (für Erich & Echo)

- „Sicherheit ist kein Aufsatz – sie ist ein Geflecht über Dev und Ops.“

- „Richtlinien, die mit Artefakten reisen, verhindern Kontextverlust.“
- „Audits werden zur Wiedergabe – nicht zur Rekonstruktion.“

🧭 Begriffserklärungen

[1] Policy-Vows (Richtlinien-Versprechen)
Absichtsbindingen, die Sicherheits- und Compliance-Regeln an Code oder Artefakte binden.
Ein Policy-Vow enthält Ziel, Risiko-Haltung und Nachweis-Pfad – so bleibt die Intention im Lebenszyklus erhalten.

[2] CVE (Common Vulnerabilities and Exposures)
Ein globaler Standard zur eindeutigen Benennung und Veröffentlichung von Schwachstellen in Software.
Jede CVE ist eine Art „Ausweisnummer“ für ein bekanntes Sicherheitsrisiko und dient als Bezugspunkt für Analyse und Behebung.

[3] Blast-Radius-Mapping
Ein Verfahren, das zeigt, welche Systeme, Dienste oder Daten von einer Schwachstelle betroffen sind.
Es visualisiert den „Wirkungsradius“ eines Problems und hilft, gezielt und schnell zu reagieren.

[4] Dynamic Gating
Ein adaptiver Freigabemechanismus, der Sicherheitsentscheidungen nach Risiko, Kontext und Service-Impact trifft –
nicht nur nach fixen Scores. Gates reagieren damit wie lebendige Kontrollpunkte im Systemfluss.

[5] Exception Governance
Ein strukturierter Umgang mit genehmigten Abweichungen (z. B. vorübergehende Sicherheitsausnahmen).
Jede Ausnahme hat Verantwortliche, Frist und Überprüfung – so bleibt Kontrolle in Resonanz, nicht im Starrsinn.