SUMMARY CYBERSECURITY

Kubernetes Security with Cilium and Tetragon

Indledning

Indlægget centreres om en konkret sikkerhedsincident, hvor et Kubernetes cluster blev kompromitteret på få minutter. Casen bruges aktivt til at forklare, hvorfor manglende visibilitet, isolation og runtime kontrol gør Kubernetes-miljøer sårbare, og hvordan specifikke tekniske løsninger adresserer disse svagheder.

Case: Et kompromis på tre minutter

Fadi beskriver en reel hændelse, hvor et Kubernetes cluster blev kompromitteret via adgang til en enkelt pod. Her fik angriberen:

- Adgang til pods med overprivilegerede identiteter ("high value pods").
- Bevægede sig lateralt i clustret og opsamlede netværkstrafik.
- Indførte malware, eksfiltrerede data og nåede ressourcer uden for clustret.
- Slettede eller overdængede logs med falske positiver, så efterforskning blev vanskelig.

Angrebet var overstået, før et SOC manuelt kunne reagere. Pointen er, at manuel respons ikke er realistisk, når kompromittering sker så hurtigt.

Hvorfor Kubernetes er anderledes – og sværere at sikre

Anders rammesætter de strukturelle udfordringer ved Kubernetes:

- **Manglende visibilitet:** Sikkerhedsteams mangler indsigt i, hvilke pods der kommunikerer med hinanden.
- **Dynamiske miljøer:** Pods opstår og forsvinder konstant, hvilket gør IP-baserede regler uegnede.
- **Namespace begrænsninger:** Namespaces er logisk organisering, ikke reel sikker isolation, især i et multitenant-setup.
- **Efterfølgende sikring er risikabelt:** At tilføje netværkspolitikker uden overblik kan bryde applikationer.

Konsekvensen er, at traditionelle netværksmodeller ikke matcher Kubernetes' dynamik.

Kubernetes sikkerhed: centrale tekniske elementer

Anders og Fadi viser herefter, hvordan identitet, visibilitet og runtime kontrol kan etableres i Kubernetes uden klassisk per pod-service mesh. Det her er de centrale tekniske byggeklodser:

- Service Mesh bruges som reference: giver isolation og observability via proxies i hver pod, men med høj kompleksitet, latency og ressourceforbrug.
- Cilium baseret på eBPF er hoved-mekanismen. Ved at arbejde direkte i Linux-kernelen muliggør Cilium identitetsbaserede netværkspolitikker fra L3–L7 uden per pod proxy og med væsentligt lavere overhead. Løsningen er bredt understøttet af cloud providers.
- Hubble giver realtime visibilitet i pod-til-pod trafik og anvendes til at opdage og afgrænse kompromittering, mens den sker.
- Tetragon leverer runtime-sikkerhed og håndhæver, hvilke systemkald og handlinger pods og containere må udføre, med høj granularitet.
- GitOps fungerer som clusterets autoritative sandhed og ruller uautoriserede ændringer tilbage automatisk.
- SPIFFE/SPIRE anvendes til kryptografisk workload identitet og danner grundlag for Zero Trust-politikker i clustret.

Handlingsrettede konsekvenser for virksomheder

- Design Kubernetes sikkerhed til at modstå kompromis, ikke kun forhindre det.
- Etabler visibilitet, isolation og runtime kontrol fra start – ikke bagefter.
- Brug GitOps og eksterne log systemer til hurtig gendannelse og analyse.
- Implementér Zero Trust-principper konsekvent i cluster arkitekturen.

Samlede sikkerhedsprincipper:

- Identitet erstatter IP-adresser som styrende sikkerhedsprimitiv
- Mindst mulig privilegering reducérer blast radius
- Automation er nødvendig – kompromittering sker på minutter
- Runtime kontrol supplerer statiske policies
- Logs skal ligge uden for clustret

Key take aways

- Et Kubernetes cluster kan kompromitteres på få minutter, hvis én pod får for brede rettigheder.
- Visibilitet via eBPF-baserede værktøjer som Cilium og Hubble er afgørende for at opdage og begrænse angreb.
- GitOps og ekstern logning sikrer, at kompromitteringer ikke kan skjules eller fastholdes.
- Effektiv Kubernetes-sikkerhed kræver identitetsbaseret isolation, runtime kontrol og automatiseret respons – ikke manuelle indgreb.

Vil du vide mere?

Kontakt vores specialister

Anders og Fadi er vores specialister
Kontakt dem til en snak om dine virksomhedsbehov.

Anders Pedersen
Senior Cloud Security Architect, Conscia
anp@conscia.com

Fadi Dasus
Senior Cloud Security Architect, Conscia
fad@conscia.com

Conscia
Conscia A/S
Østbanegade 135,
2100 København Ø
Denmark

Kontakt
+45 70 20 77 80
conscia.com

Conscia
Secure progress