Multitenant Cybersecurity Operations Center as an Open Cloud Service with Machine Learning Elements

Ivana Gabrišová, Gabriel Koman
University of Žilina



Why this matters

Massive, heterogeneous telemetry

Rules alone miss novel, low-and-slow attacks

Need transparency across multiple tenants

Open stacks enable scale + explainability



Goal & method

Goal

Which algorithms and open architectures actually work for a multitenant SOC?

Method

Web of Science search

- “Elastic/ELK” or “OpenSearch” + AI/ML/SOC/Anomaly…
- 33 hits → 13 full-text, English, on-topic case studies
- Synthesis across architecture, methods, data, operations



Open SOC blueprint

Telemetry plane	Orchestration	Streaming option	Edge/OT	Tenant-scoped alerting & cases
Elastic Stack (Beats → Logstash → Elasticsearch → Kibana)	Containers/Kubernetes, per-tenant indices & RBAC	Kafka → Spark ML → ELK for bursty, low-latency needs	Suricata inline; honeypots for threat intel	Rules/ connectors, separate spaces

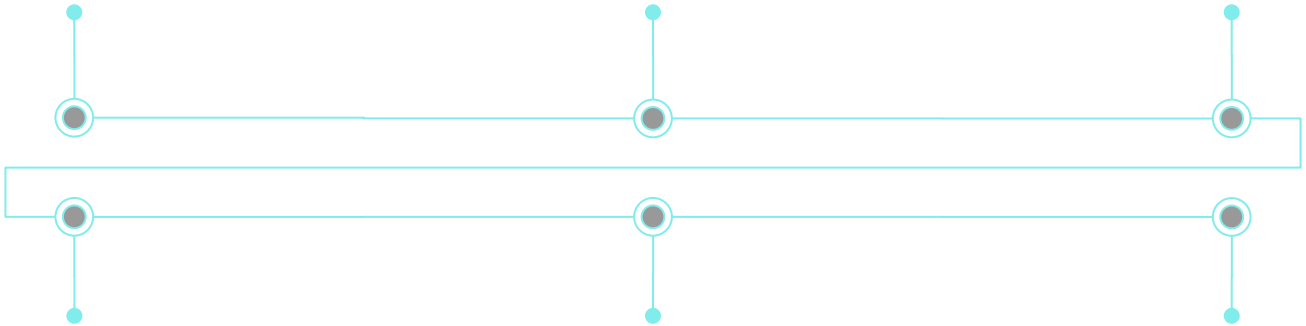


Methods that showed up

Gradient-boosted trees
(labeled flows)

Deep models
(sequence patterns)

PCA + LOF
(unlabelled logs)



k-Means + k-NN
(streaming hybrid)

Evolutionary ensembles
(behaviour analytics)

Rule engines for self-healing
(with safety rails)

Telemetry that feeds the SOC

Network flows	System/app logs	IoT/OT & IDS	Honeypots	Microservices
ports, bytes, packets, duration, protocol	e.g., Postfix program ratios in short windows	Suricata alerts + sensor context	credentials, commands, malware with enrichment	logs, metrics, traces for root-cause and SLOs



Comparison I: placement

Where the methods live

Trees batch/near-real-time, results to ELK	**Deep models** sidecar service parallel to ingest	**PCA/LOF** analytics job inside ELK loop
k-Means + k-NN Kafka → Spark → ELK	**Evolutionary UBA** ELK on Kubernetes, distributed learners	**Rules** ELK analytics → action engine



Comparison II: trade-offs

Explainability for tenants trees, PCA/LOF	**Low labels** PCA/LOF, streaming clustering	**Sequence patterns** deep models
Behaviour baselining evolutionary ensembles	**MTTR** rules + playbooks with safety rails	**Caveats** drift per tenant, cross-tenant isolation, streaming SRE



Operational patterns

1. Stream-before-index for bursts

2. Enrichment at ingest (geo, asset, VirusTotal)

3. ATT&CK-mapped validation and observability scoring

4. Tenant-scoped spaces for alerts and cases

5. Honeypots: isolate and egress allow-list



Operational takeaways

1. Choose models by data reality and explainability
2. Place compute to meet latency and cost constraints
3. Treat anomalies as candidates; triage and correlate
4. Monitor drift; retrain and audit per tenant



Future research

Standardised multitenant benchmarks

OpenSearch vs. Elastic parity for ML

Edge/IoT placements and governance

Safety rails for automation (canary, rollback, blast-radius)



Acknowledgment

Funded by the EU NextGenerationEU through the Recovery and Resilience Plan for Slovakia under the project No. 09I05-03-V02-00010.



Thank you

