CYBERAID

AI-Driven Cybersecurity for Financial Service Providers

Module 3

The Kernel-Level Monitor and Financial Threats

CyberAId Learning Centre – eBPF Tutorial Course

October 2026

Co-funded by the European Union

ECCC EUROPEAN CYBERSECURITY COMPETENCE CENTRE

The project funded under Grant Agreement No 101249596 is supported by the European Cybersecurity Competence Centre

CYBERAID

The proof-of-concept architecture

5 **Wazuh / OpenSearch integration**
Custom decoders, financial rules, graded events, indexing and dashboards

4 **Structured output**
JSON events: timestamp, process, PID/UID, src/dst IP and port, rule, severity

3 **Detection rules**
Suspicious ports, connection bursts, suspicious process execution: INFO to CRITICAL

2 **Kernel hooks (BCC / eBPF)**
TCP connection tracer, process monitor, file-access monitor

1 **Simulated financial environment**
Payment, core banking and authentication services (Docker) with normal and attack traffic generators

Design principles

No changes to monitored applications

Probes attach from outside: no modification, recompilation or restart

Metadata, not payloads

No transaction content is processed; supports data minimisation

Minimal performance impact

Filtering in the kernel; overhead benchmarked under load

Open-source stack

BCC, Wazuh, OpenSearch and Cilium: inspectable and adaptable

Three kernel sensors

TCP connection tracer

Connect and accept events

Source and destination IP and port

Owning process, PID and UID

Process monitor

Process execution events

Parent and child relationships

User IDs, e.g. UID 0 from a non-root parent

File-access monitor

File-open events

Path watch-list per monitoring profile

Keys, database files, credentials

The BCC sensors illustrate the PoC; production agents are expected to use CO-RE.

CYBERAID

From kernel event to SOC dashboard

1
eBPF capture
Kernel events with process, user and connection context

2
JSON event
Structured record, one per observation

3
Forward
Events sent to Wazuh

4
Decode
Wazuh decoders parse the fields

5
Detection rules
Financial-domain rules grade severity

6
Dashboard
OpenSearch: severity, trends, critical detections

Dashboard figures in the PoC are based on simulated traffic.

CYBERAID

Mapping eBPF signals to financial threats

Threat scenario	eBPF signal	Example detection logic	ATT&CK
Brute force on authentication	Bursts of TCP accept/connect to the auth port	More than N attempts from one source in T seconds	T1110
Internal reconnaissance	Many short-lived connections to distinct ports	Distinct-port count per source above threshold	T1046
Data exfiltration	Outbound connect to a destination outside the profile	Per-service destination allow-list	T1041
Reverse shell / C2	execve of shell or nc + outbound connect, same PID tree	Combined process + network rule	T1059, T1071
Privilege escalation	Child process with UID 0 from non-root parent	Parent/child UID mismatch	T1068
Access to keys or credentials	File opens on keys, DB files, credentials	Path watch-list per profile	T1552

Example logic only; thresholds are placeholders to be calibrated.

Monitoring profiles per system type

Payment processors

Predictable outbound destinations: alert on any connection outside the allow-list

Core banking

Few, stable internal peers: watch privilege changes and database file access

Authentication

Natural brute-force target: watch connection bursts on the auth port

Trading systems

Latency-sensitive: minimal overhead; watch unexpected processes and outbound links

Illustrative profile logic; profiles and thresholds are defined with each institution.

CYBERAID

Example: reverse shell on a payment host

Signals

execve of a shell from the payment service

Outbound connect from the same process tree

Rule and result

Combined process + network rule

CRITICAL, T1059 and T1071


```
{ "rule": "reverse_shell_suspected",
  "severity": "CRITICAL",
  "attack": ["T1059", "T1071"],
  "host": "payments-api-01",
  "process": "nc", "pid": 4182, "uid": 0,
  "parent": "payment-api",
  "dst_ip": "203.0.113.25",
  "dst_port": 4444,
  "timestamp": "2026-10-12T09:14:03Z" }
```


Illustrative event only; values are examples, not measured output.

Key takeaways

- Five-layer PoC: from a simulated financial stack to Wazuh and OpenSearch
- Design principles: no application changes, metadata only, low overhead, open stack
- Three sensors: connections, processes, files
- Rules are graded INFO to CRITICAL and mapped to MITRE ATT&CK
- Value comes from context: process + network + identity, per monitoring profile



Thank you!

Connect With CyberAID:
LinkedIn: CyberAID

Next: Module 4 – Integration into CyberAId and its Agents

CyberAID has received funding from the DIGITAL EUROPE Programme under grant agreement No. 101249596