SYSTEM CALL CHAIN PROFILING OF CONTAINERIZED APPLICATIONS FOR SECURITY REGRESSION TESTING IN CI/CD PIPELINES

Konstantin Chebunin · ITMO University

kochebunin@itmo.ru

Why this work is relevant

Software supply-chain risk reaches developers and CI before production.

1.23M

malicious open-source packages logged by Sonatype since 2019

2026 State of the Software Supply Chain

53%

of malicious packages analyzed in 2025 targeted developer machines at install time

Sonatype, Attacking the Assembly Line

Why existing CI/CD controls can still miss the change

STATIC SCAN

Packages, versions, known vulnerabilities

→

FUNCTIONAL TESTS

Expected inputs and outputs

→

KERNEL BEHAVIOUR

Files, processes, network and privileges

Need: behavioural regression testing against an approved baseline.

Threat Model and Research Goal

The candidate build may change; the CI runner, tracer and approved baseline are trusted.

Candidate build can be influenced through

- DEPENDENCY
- BASE IMAGE
- BUILD LAYER
- RUNTIME CONFIG

ATTACKER OBJECTIVE

Add exfiltration, covert communication, unauthorized execution or privilege use while the reference suite still passes.

TRUSTED SECURITY BOUNDARY

- CI runner
- eBPF tracer
- baseline registry

Research Question: Does candidate behaviour deviate from an approved build in a security-relevant way?

Behavioural Model B = (A, M, W)

The baseline captures admissible call context, stable resource domains and expected behaviour frequency.

A - STRUCTURE

DFA over contexts of the last k-1 calls. A transition exists when the corresponding k-chain was observed in approved runs.

M - ARGUMENTS + PRIVILEGES

Stable domains D(c,i) for selected paths, endpoints and flags, plus the exercised capability footprint.

W - FREQUENCY

Per-chain occurrence counts across R approved runs define expected frequency and significant shifts.

Candidate conforms if

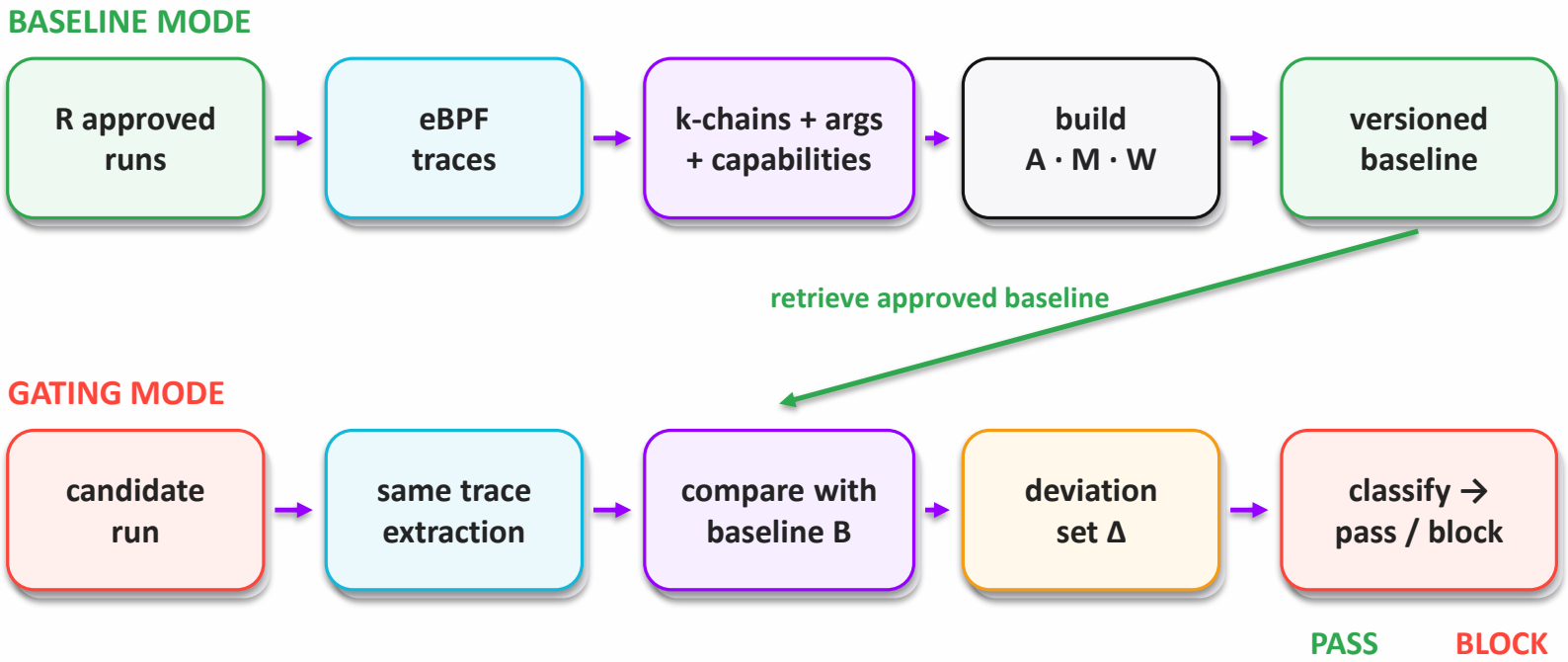
- **k-chains ∈ A**
- **stable args ∈ M**
- **capabilities ⊆ footprint**
- **frequency within threshold**

Δ(candidate, B) = ∅ → PASS

Δ(candidate, B) ≠ ∅ → classify concrete deviations

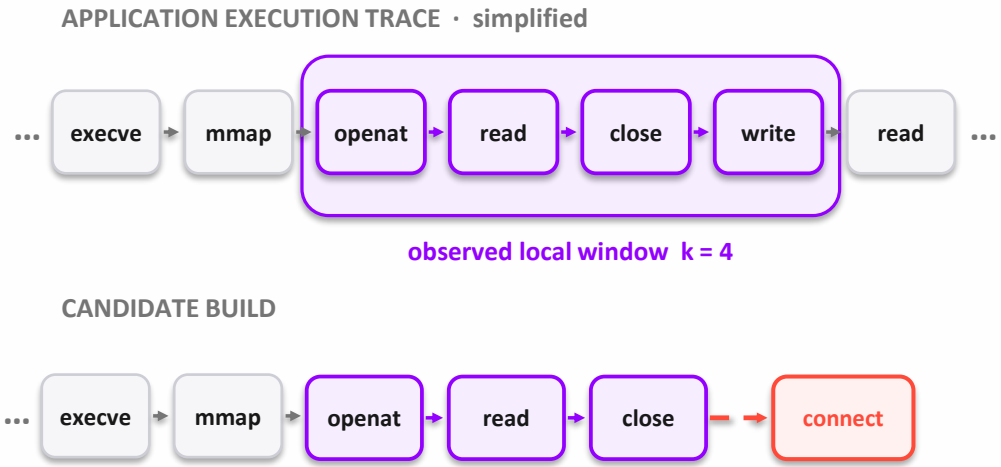
Baseline Construction and Candidate Comparison

Both paths use identical trace extraction; only approved runs update the baseline.



What counts as a deviation?

A k-chain is a short sliding window inside the application's complete system-call trace.



DEFAULT ACTION - BLOCK

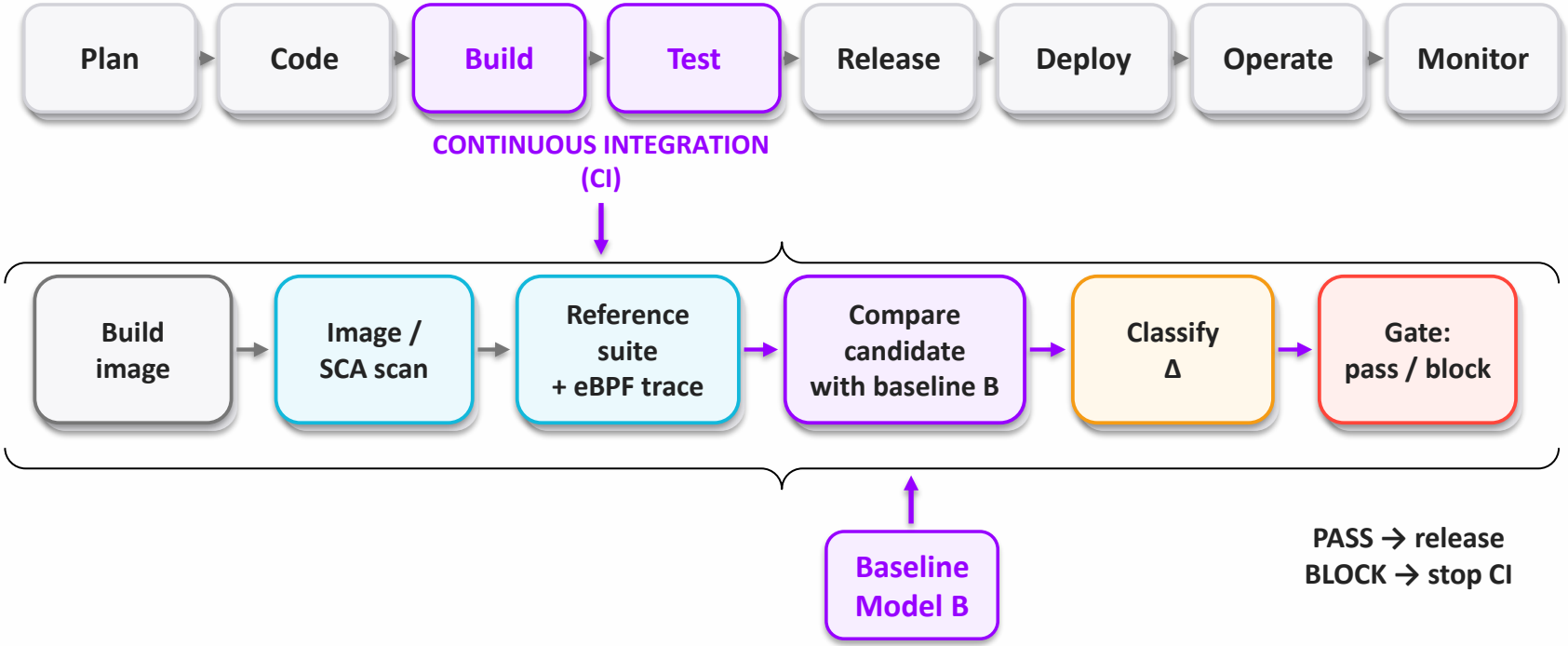
- new system call
- new k-chain
- sensitive argument
- capability escalation

WARNING FIRST

- other arguments
- frequency

DevOps Application Lifecycle

Industry-standard lifecycle stages show where CI runs the behavioural regression gate.



Evaluation Setup

Containerized Python REST service exercised by a pytest integration suite.

- **Python REST**
- **Gunicorn**
- **PostgreSQL + Redis**
- **Debian-slim**
- **containerd / runc**

k = 4

system-call chain length

R = 30

approved reference runs

100

clean rebuilds

12

injected modifications

30 call types · 229 chains · 123 states · 527 argument values

Injected Modifications

All modifications preserve the functional behaviour checked by the suite.

7 × SUPPLY — Reverse shell · credential exfiltration · DNS exfiltration · backdoor staging · call-home · download-execute · privilege probe

4 × CONFIG — Bind-address change · encryption/port change · extra capability · host socket mounted and opened

1 × FREQ. — In-profile beacon → significant frequency shift in an approved call chain

Detection Results

Strict build-blocking evaluation over 120 modified runs and 100 clean rebuilds.

11 / 12
modifications blocked on every run

1.00
precision

0.92
recall

0.96
F1 score

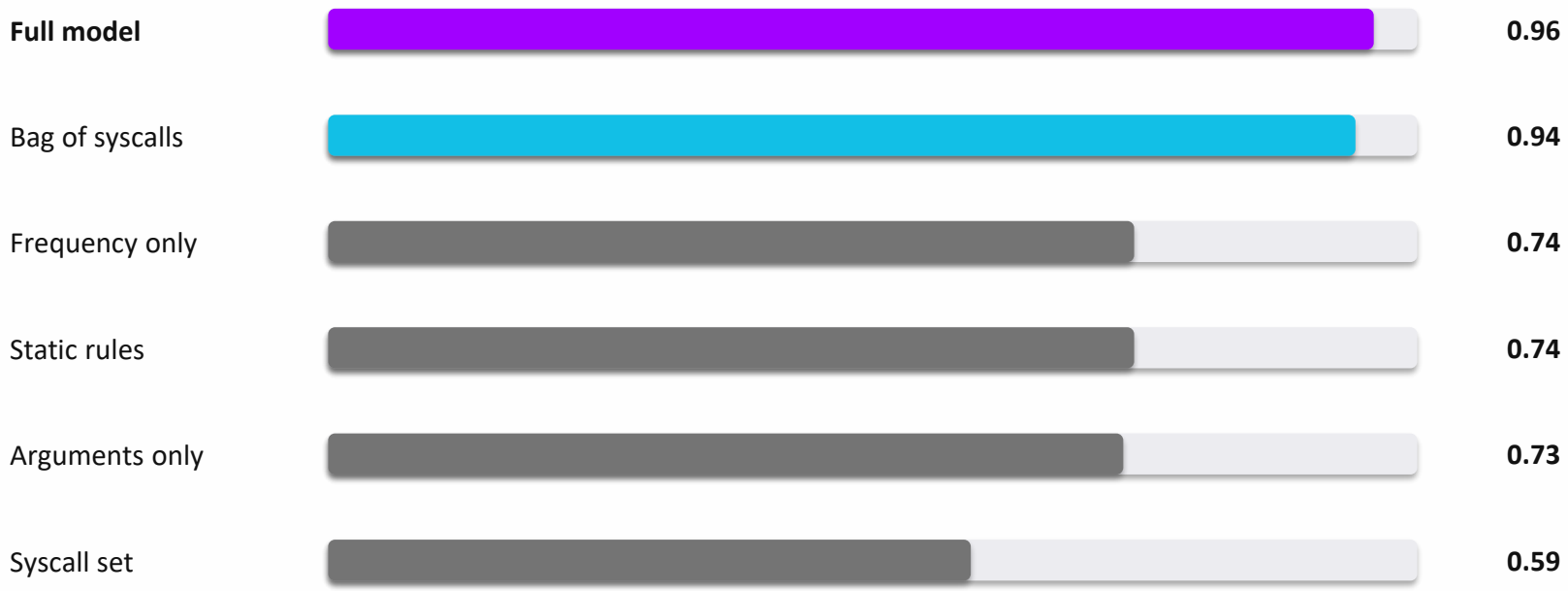
0.00
clean-block rate

5–8 ms
candidate comparison latency

C2 (port change) produced a warning rather than a block — by design.

Comparison with Simpler Detectors

Strict F1 on the same traces. The combined model preserves a zero clean-block rate.



All 12 modifications produced a finding; 11 were build-blocking.

Limitations and Next Steps

ITMO

LIMITATIONS

1. **Detection is bounded by reference-suite coverage.**
2. Runner, tracer and baseline registry are trusted.
3. Current model covers one cgroup / application object.

NEXT STEPS

1. Model relations across arguments and calls.
2. Extend profiling to interacting containers.
3. **Compile approved behaviour into seccomp/capability policies.**

THANK YOU

Behavioural regression testing turns system-call traces into an explicit delivery-time security gate.

Konstantin Chebunin · ITMO University

kochebunin@itmo.ru