

{deepfactor}



Uncovering Hidden Security Risks in C++ Apps with LD_PRELOAD and API Interception

TUESDAY, APRIL 11TH

KIRAN KAMITY

CO-FOUNDER, DEEPFACTOR



// Evolution of software development over the last decade

Monoliths to Containers



currently using or planning to
use containers in production

Release Speed



release code weekly
or more frequently

Testing Automation



of functional testing
is automated

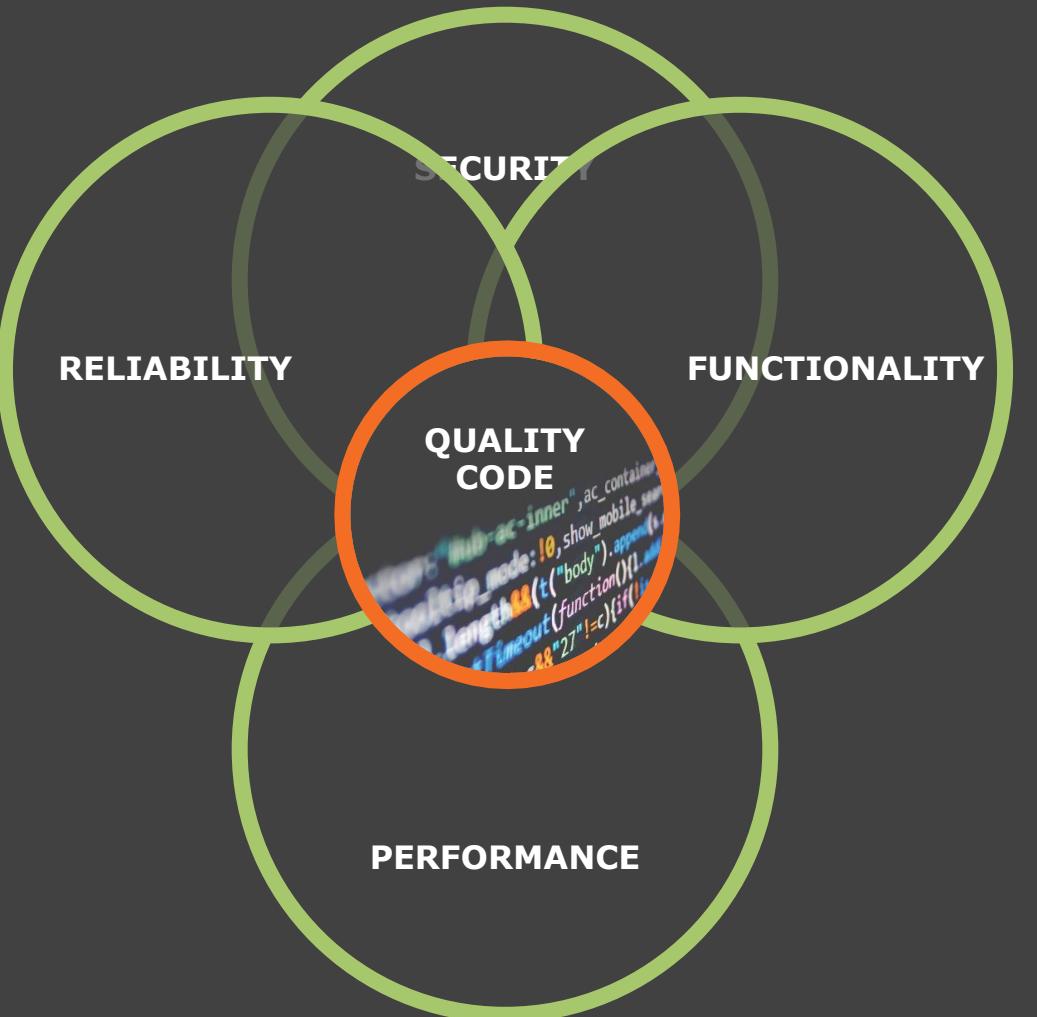
source: CNCF & The State of Testing Report 2022

{df}

{ “Quality” now entails user experience, quality of service, availability, performance, **security** and even business impact.

Improve Software Quality by Building Digital Immunity
Gartner

// Security is now part of quality code



// Security hasn't adapted to modern development



Web App

Vulnerability

Breach

\$2B loss

source: The Business Journals

{df}



Supply Chain
Attacks

\$90M loss

source: bitsight

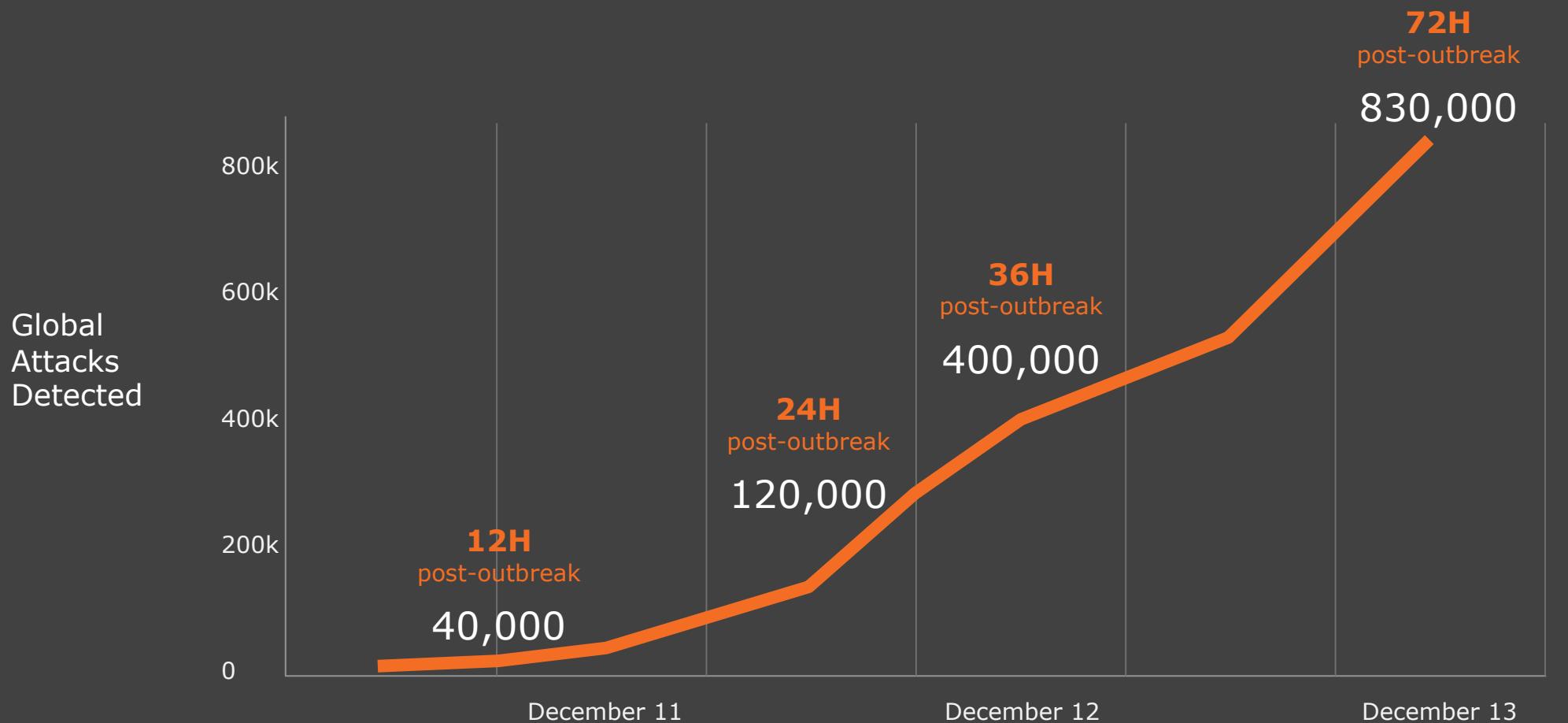


Zero Day Open
Source Vulns

830K attacks
in 72 hours

source: checkpoint software

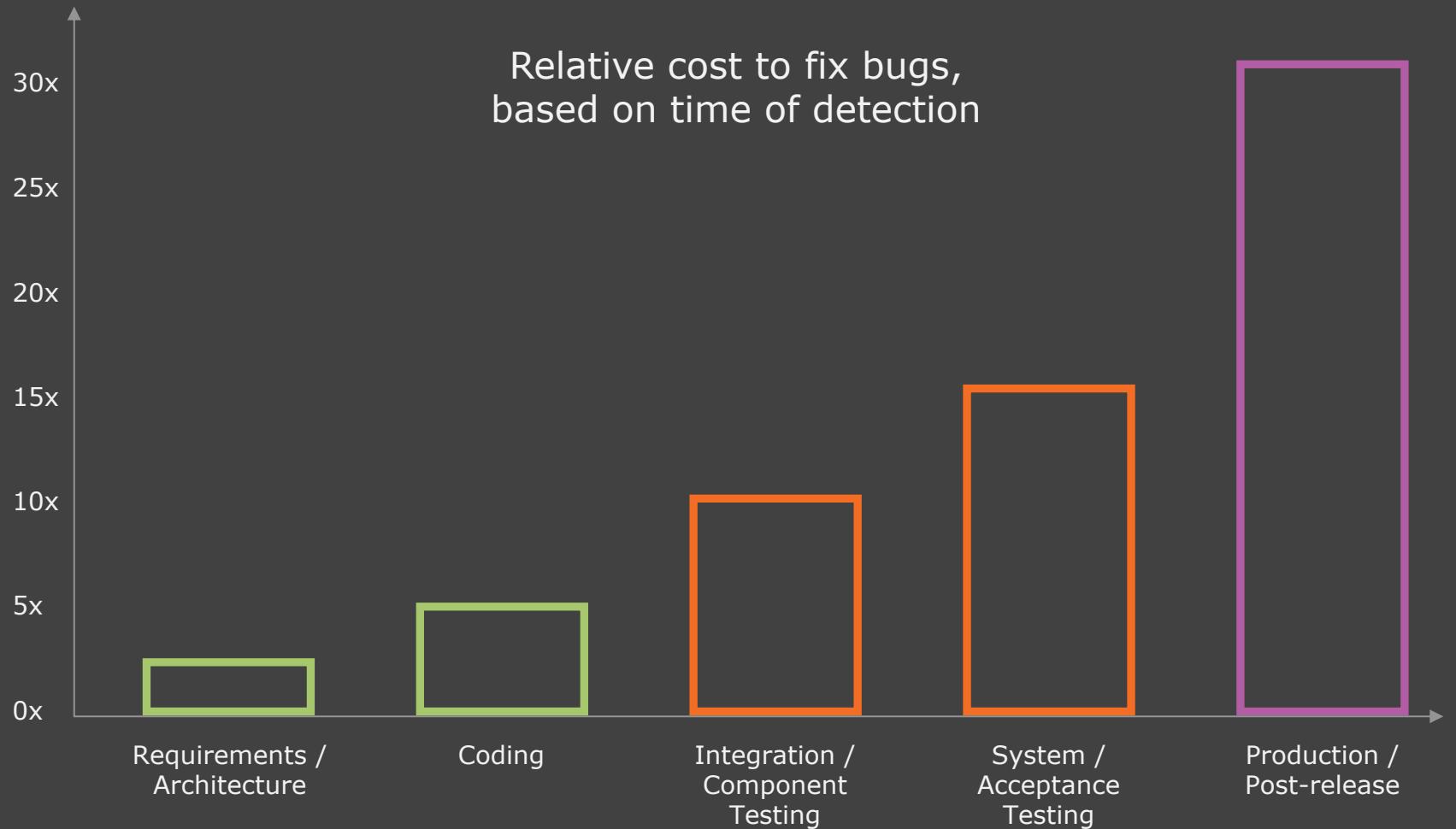
// Log4J Day Zero



source:checkpoint software

{df}

The cost of fixing vulnerabilities grows
// exponentially the later you fix it



source:NIST

{df}

// Different Approaches to Uncovering Security Risks



Static Application Security Testing (SAST)

Scan source code for known vulnerabilities.



Software Composition Analysis (SCA)

Scan for known vulnerabilities in OS packages, dependencies, and OSS



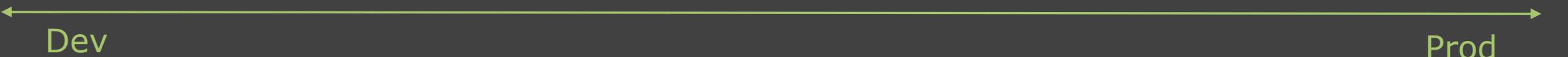
Runtime Security Dev & Test

Identify unknown vulnerabilities that can't be detected by SAST/SCA by analyzing running apps

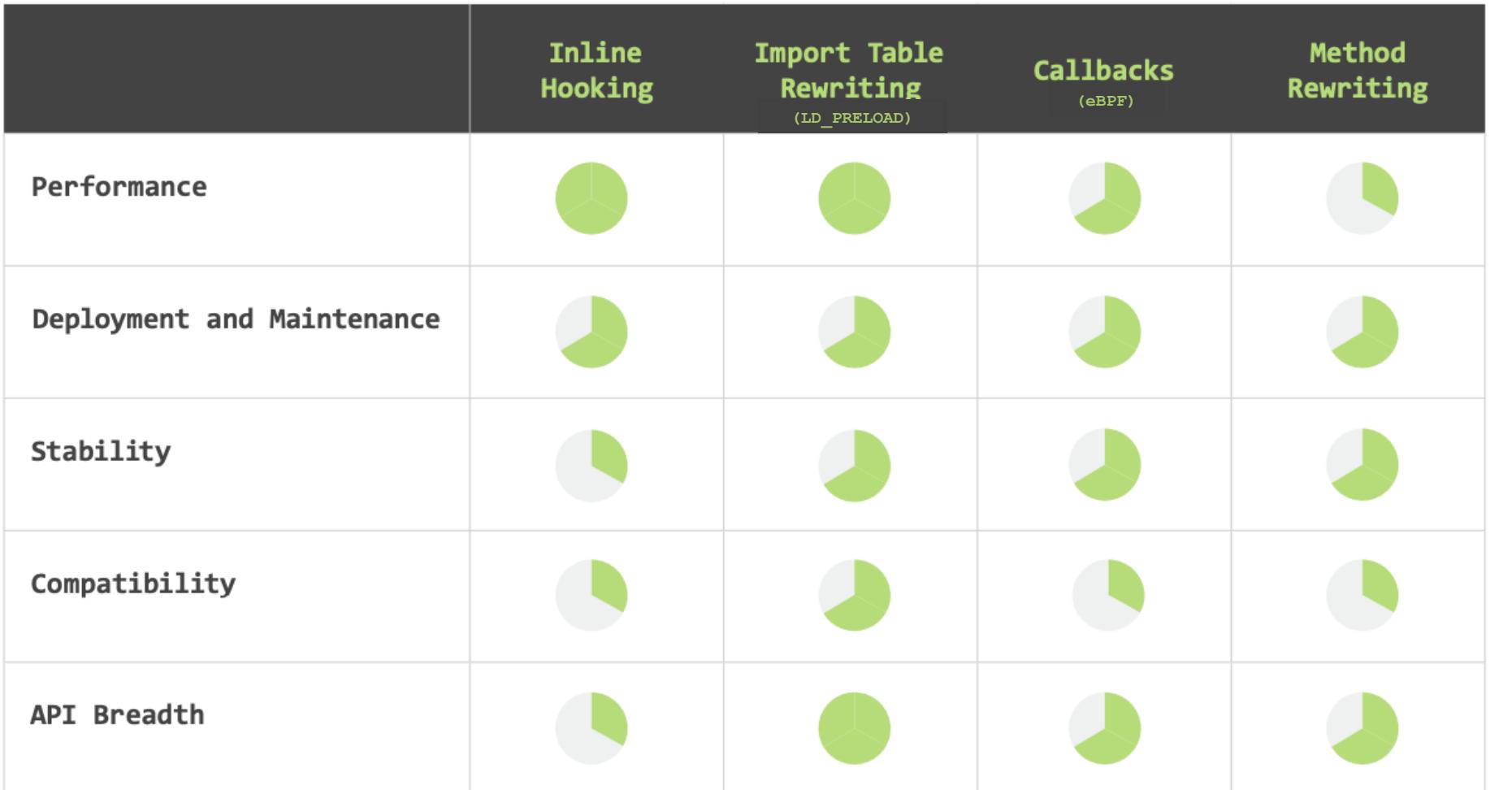


Runtime Security In Production

Detect insecure file, network & memory behaviors for security & compliance with PCI/SOC2/..



// Comparison of API Interception Techniques

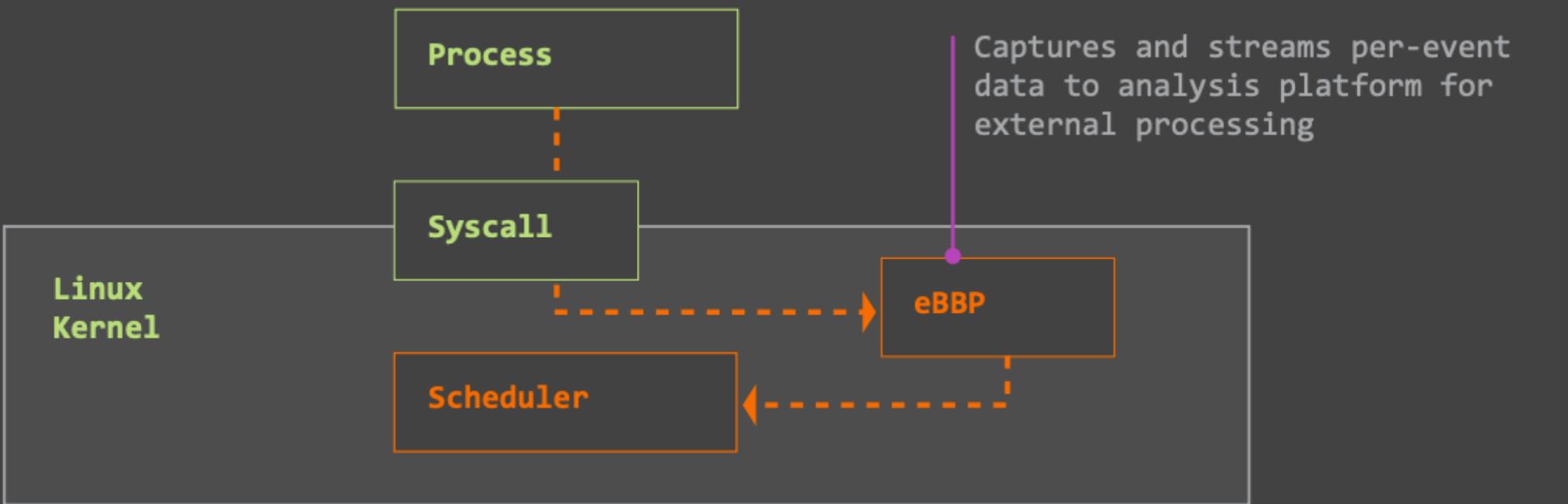


{df}

// Example of Import Table Rewriting with LD_PRELOAD



// Example of Callbacks with eBPF



// Runtime Analysis



- Execution
- File system behavior
- System calls
- Remote code execution
- Library behavior
- Memory behavior
- Network behavior
- Privilege escalation
- Buffer overflow
- Secrets

The screenshot shows the Deepfactor platform's alert policies interface. The top navigation bar displays the Deepfactor logo and the current section: Alert Policies / Deepfactor Demo Policy. The left sidebar contains icons for Home, Applications, OS Packages, API, and Settings. The main content area is divided into two columns. The left column lists application risks: Insecure Execution, Insecure File System Behavior, Improper Use Of System Calls, Remote Code Execution, Insecure Library Behavior, Insecure Memory Behavior, Insecure Network Behavior, and Privilege Escalation (which is highlighted with a light orange background). The right column displays a list of alert rules under the heading "Rules for Privilege Escalation". Each rule includes a status indicator (green circle), a priority level (P1 or P4), a description, and a "View Details" button. The rules are:

- P1 - Alert if process UID changes unexpectedly
- P1 - Alert if process GID changes unexpectedly
- P4 - Alert on use of setuid
- P4 - Alert on use of setgid
- P4 - Alert on use of setfsuid
- P4 - Alert on use of setreuid/setgroups
- P4 - Alert if application changes LD_LIBRARY_PATH after startup

Each rule also has an "Ignore if the following processes trigger this alert rule" link below it.

// Deployment Modes

- Monolithic applications
 - `dfctl run -a "my application" --cmd COMMAND_WITH_ARGUMENTS`
- Containers
 - `dfctl run -a "my application" --docker-run DOCKER_RUN_OPTIONS --image IMAGE`
- Kubernetes
 - `helm --install df-webhook-stable -n df-webhook deepfactor/webhook --set clusterName=CLUSTER_NAME_OF_YOUR_CHOICE --create-namespace -f webhook-override.yaml`
- Docker Swarm, AWS ECS, Lambda w/ container ...
 - `docker build -f Dockerfile.df --build-arg "APP_IMAGE=${APP_IMAGE}" --build-arg "APP_IMAGE_ID=`docker inspect ${APP_IMAGE} --format '{{.Id}}'`" --build-arg "DF_APP_NAME=my application" .`

// Demonstration

The screenshot shows the deepfactor application security dashboard for the 'bank-of-anthos' application. The top navigation bar includes tabs for Overview, Alerts (1742), Components, Runtime Security (selected), SCA, SBOM, DAST, Compliance, and Live Stream. The Runtime Security tab displays a list of 201 alerts, with the first 13 shown in the main table.

Alerts (201) Alert Rules Components (9)

Showing metrics for All Environment and All Namespaces and Latest Version

Showing 13 of 201

Severity	Type	Title	Component	Category	Last Observed	Alert	Status
P1	!	Process /usr/local/openjdk-11/bin/java running as root detected	balancereader	Runtime Security	Jan 30, 2023, 8:23:24 AM	BAE-4	Reported
P1	!	Process /bin/uname running as root detected	userservice	Runtime Security	Jan 30, 2023, 8:23:26 AM	BAE-24	Reported
P1	!	Process /env/bin/python3.7 running as root detected	userservice	Runtime Security	Jan 30, 2023, 8:23:25 AM	BAE-22	Reported
P1	!	Process /bin/dash running as root detected	userservice	Runtime Security	Jan 30, 2023, 8:23:25 AM	BAE-20	Reported
P2	!	Uncommon/risky environment variable PRIV_KEY_PATH detected	userservice	Runtime Security	Jan 30, 2023, 8:23:25 AM	BAE-13	Reported
P2	!	Uncommon/risky environment variable POSTGRES_PASSWORD detected	userservice	Runtime Security	Jan 30, 2023, 8:23:25 AM	BAE-12	Reported
P2	!	Attempt to connect to 10.43.226.17 without prior DNS resolution	balancereader	Runtime Security	Jan 30, 2023, 8:23:55 AM	BAE-180	Reported
P2	!	Uncommon/risky environment variable PUB_KEY_PATH detected	userservice	Runtime Security	Jan 30, 2023, 8:23:25 AM	BAE-19	Reported
P2	!	Uncommon/risky environment variable PUB_KEY_PATH detected	balancereader	Runtime Security	Jan 30, 2023, 8:23:24 AM	BAE-1	Reported
P2	!	Uncommon/risky environment variable POSTGRES_PASSWORD detected	balancereader	Runtime Security	Jan 30, 2023, 8:23:24 AM	BAE-2	Reported
P2	!	Uncommon/risky environment variable SPRING_DATASOURCE_PASSWORD detected	balancereader	Runtime Security	Jan 30, 2023, 8:23:24 AM	BAE-3	Reported
P3	!	Unsafe string API STRCPY used	userservice	Runtime Security	Jan 30, 2023, 8:23:25 AM	BAE-21	Reported
P4	!	Use of ioctl(FIONCLEX)	userservice	Runtime Security	Jan 30, 2023, 8:23:26 AM	BAE-31	Reported

20 rows 1-13 of 13

{df}

// Free 60-minute Risk Assessment Report



hello@deepfactor.io

www.deepfactor.io

The screenshot displays the Deepfactor application interface for the 'bank-of-anthos' environment. The top navigation bar includes tabs for Overview, Alerts (1742), Components, Runtime Security (selected), SCA, SBOM, DAST, Compliance, and Live Stream. A search bar at the top right allows filtering by Environment, Namespaces, and Version.

Correlated from Static image scans: This application uses 9 container images. The SCA section shows a donut chart with 3018 vulnerabilities categorized as follows: Critical (18), High (162), Medium (438), Low (1049), and Unknown (1351). A button 'Fix Available(1846)' is present.

Runtime Security: This application has 9 workloads. A donut chart shows 201 alerts categorized as: Critical (4), High (12), Medium (8), Low (51), and Unknown (123). A legend lists alert types: Insecure Execution, Remote Code Execution, Insecure Network Behavior, Insecure File System Behavior, Buffer Overflow, and Improper Use Of System Calls.

Software Bill of Materials: Languages: JAVA, C, PYTHON. Dependencies: 50 Vulnerable / 396 Total. OS Packages: 323 Vulnerable / 681 Total. A 'View all (1077)' link is provided.

DAST: A list of vulnerabilities: HTTP Request Spoofing, Insecure HTTP Cookie, Application Scanning, Path Traversal, Buffer Overflow, and Code Injection. A 'View all (14)' link is provided.

Compliance: PCI DSS v3.2, SOC 2 Type 2, NIST 800-53. Progress bars indicate compliance status: PCI DSS v3.2 (201), SOC 2 Type 2 (197), and NIST 800-53 (179).

Summary: 217.79k Events received from 9 Instances.

Inventory: Components: 9, Active Instances: 9.

Workload Alerts: Total Alerts: 215. A chart shows 24, 156, 24, and 11 alerts corresponding to workloads P1, P2, P3, and P4.

SCA & SBOM alerts: Total Alerts: 1527. A chart shows 222, 1305, 0, and 0 alerts corresponding to workloads P1, P2, P3, and P4.

{df}