THE LINUX FOUNDATION

OPEN SOURCE SUMMIT INDIA

ACE -Agent Compliance Engine

Beyond Static Benchmarks:

Chaos Based Agent Compliance Engine for Real World AI Agents AI Agents

Chetana Amancharla
Infosys

Praveen Kalapatapu
Infosys

Deepak Sharma
Microsoft

Saramma George
Infosys

Suganya Selvaraj
Infosys

#OSSUMMMIT

THE LINUX FOUNDATION

AgentCert

Autonomy Levels of AI Agents

	Infrastructure	Data Integration	Orchestration	Value Offering
Level 5 **Full Autonomy**	Entirely self-managing, self-healing, and evolving cloud architectures.	Unified, cognitive data platforms using advanced causal inference and autonomous learning.	End-to-end cognitive orchestration that continuously learns and adapts to unforeseen challenges.	Self-learning monetization engines that dynamically adjust pricing and bundles based on real-time behavior.
Level 4 **High Autonomy**	Self-optimizing environments leveraging reinforcement learning and predictive capacity scaling.	Continuous, cross-domain data loops driving predictive analysis and real-time adaptation.	Intent-driven automation that translates high-level business objectives into automated system actions.	Differentiated, adaptive services that automatically tune performance based on real-time user needs.
Level 3 **Conditional**	Cloud-native, containerized microservices that scale dynamically based on predictable workloads.	Real-time data pipelines that detect and flag operational anomalies instantly.	Cross-domain platforms that enable closed-loop, reactive self-management and auto-remediation.	Services are created and managed dynamically based on data insights and real-time usage feedback.
Level 2 **Partial Autonomy**	Virtualized environments that allow for basic programmatic resource scaling and provisioning.	Centralized data repositories used to inform and trigger static operational rules.	Early-stage automated workflows triggered by specific predefined events or fixed schedules.	Dynamic, policy-driven offerings featuring event-based charging and basic usage tracking.
Level 1 **Assisted Operations**	Monolithic legacy systems deployed on traditional, static hardware with manual configurations.	Limited data sharing with heavily siloed information requiring manual extraction.	Slow, manual execution of workflows with high error rates; basic automation supports human operators.	Standard, rigid product tiers differentiated only by basic capacity and fixed price points.

5-Level progression framework from human-assisted to fully autonomous cognitive systems

ACE

Enterprise-Specific Challenges

Deploying LLM agents in enterprise settings introduces complex requirements beyond research benchmarks, demanding predictable reliability, strict security, and domain adherence.

Complexity from Role-Based Access

Security & Permissions

Contextual task execution based on user permissions and authentication constraints. Agents must navigate access rules without leaking denied information.

IntellAgent (Levi and Kadar, 2025)

Embeds role-specific restrictions into task generation for permission-sensitive contexts.

Reliability Guarantees

Consistency & Determinism

Mitigating the inherently stochastic nature of LLMs to provide repeatable, explainable behavior required for enterprise auditing frameworks.

τ-benchmark (Yao et al., 2024)

Incorporates the pass^k metric to evaluate agent consistency across multiple trials in domains like retail.

Dynamic & Long-Horizon Interactions

Lifecycle & Context Retention

Real-world agents operate continuously, leading to performance drift, context loss, and cumulative decision impacts not captured by short episodes.

Park et al. (2023)

Multi-day emergent behaviors in simulated environments.

Maharana et al. (2024)

Context maintenance across 600-turn dialogues.

Adherence to Domain Policies

Regulation & Compliance

Agents must respect strict operational rules, including GDPR, HIPAA, approval workflows, and data retention policies during task execution.

GDPR / HIPAA | Approval Workflows | Policy Violations

Critical challenges transitioning agents from research to enterprise deployment

4 Core Challenge Areas

Production Readiness Gap

Evidence from deployed AI agents in production environments

Source: Pan et al., 2025; Gartner, 2025

68%

Agents require human intervention within 10 steps

68% of deployed agents execute at most ten steps before requiring human intervention

70%

Use off-the-shelf prompting only

70% rely on prompting off-the-shelf models rather than fine-tuning or RL

74%

Rely on human evaluation for correctness

74% depend primarily on human evaluation to assess correctness

40%+

Agentic AI projects predicted to be canceled by 2027

Gartner predicts over 40% of agentic AI projects will be canceled due to unproven value

ACE

How do we bridge the gap ?

Show of Hands: How do we build trust in complex agents?

BETTER PROMPTING & FINETUNING

Iterative Model Optimization

Optimizing the core models for minimal errors.

Focusing engineering efforts on prompt structures, few-shot examples, and fine-tuning weights to improve default outputs.

RIGID HUMAN GUARDRAILS

Human-in-the-Loop Safeguards

Humans-in-the-loop for safe interactions.

Forcing manual review checkpoints at critical decision boundaries to prevent unauthorized or incorrect live system actions.

MULTI-AGENT ARCHITECTURES

Task Deconstruction & Routing

Real-world agents operate continuously, leading to performance drift, context loss, and cumulative decision impacts not captured by short episodes.

Limiting the blast radius of errors by assigning distinct micro-tasks to individual agents instead of relying on a monolithic model.

RIGOROUS EVALUATION FRAMEWORKS

Continuous Validation Layers

Systematically measuring agent trajectories and safety bounds before production.

Moving away from human "vibes" and manual spot-checking to establish reproducible, automated benchmarking for enterprise trust.

AgentCert

Taxonomy of LLM Agent Evaluation

1. Evaluation Process

Interaction Mode

Methods and environments for interacting with LLM agent systems.

- Static & Offline
- Dynamic & Online

Evaluation Data

Datasets, benchmarks, and synthetic data generation for evaluation.

- Datasets
- Benchmarks
- Domain Specific

Metrics Computation

Techniques and methods to accurately compute agent performance metrics.

- Code Based
- Human-as-Judge
- LLM-as-Judge

Evaluation Tooling

Frameworks, platforms, and standardized systems to evaluate with.

- Frameworks
- Platforms
- Leaderboards

Evaluation Contexts

The environments, domains, and specific scenarios targeted for evaluation.

- Sandbox Environments
- Live Systems

2. Evaluation Objectives

Agent Behavior

Outcome oriented. Did the agent produce the right result efficiently?

- Task Completion
- Interaction
- Latency/Cost

Agent Capabilities

Process oriented. Does the agent produce results in the right way?

- Planning
- Memory
- Tool Use
- Multi Agent

Reliability

Can the agent perform reliably across inputs and over time?

- Robustness
- Hallucinations
- Errors

Safety and Alignment

Can the agent be trusted not to produce harmful or non-compliant results?

- Fairness
- Harm
- Compliance

Comprehensive taxonomy for evaluating LLM agent systems

2 Main Branches **9** Core Dimensions

ACE

Why Static Benchmarks Fail Real-World Agents

The Problem with Static Evaluation

Current benchmarks assume stable environments with reliable tools and isolated failures, but real deployments face dynamic, unpredictable conditions with overlapping failures.

Key Limitations

- **Happy-path bias:** Static tests focus on narrow reasoning, missing partial observability and delayed signals.
- **Endpoint correctness:** Success measured at terminal state, not behavioral robustness over time.
- **Isolation failures:** Real systems have overlapping failures that static benchmarks don't capture.
- **Compounding errors:** Long-horizon drift and error accumulation are rarely tested in fixed sets.

Static vs Dynamic Evaluation

Task Completion

85%

Static benchmark

Real-world Success

45%

Production deployment

Key Insight

40% performance drop when moving from static to dynamic environments

Source: Pan et al., 2025 - Large-scale study of agents in production

68% require human intervention ≤10 steps

74% rely on human evaluation

AgentCert

Agentic Evaluation System Needs: The Need of the Hour

Four key directions to advance agent evaluation from research to production readiness

Holistic Evaluation Frameworks

Assessing agents across multiple interdependent dimensions simultaneously. Future work must develop frameworks that balance competencies—such as tool use, safety, and planning—beyond isolated metrics.

- Interdependent Dimensions
- Competency Balance

More Realistic Evaluation Settings

Bridging the gap between labs and production by incorporating enterprise elements like dynamic multi-user interactions, role-based access controls, and domain knowledge through real-world trials.

- RBAC & Access Controls
- Simulated Envs

Automated & Scalable Techniques

Reducing manual effort through synthetic data generation, highly simulated task contexts, and advancing approaches like LLM-as-a-judge or Agent-as-a-judge for improved reproducibility.

- Synthetic Data
- Agent-as-a-judge

Time- & Cost-Bounded Protocols

Developing efficient protocols that support iterative development. Future methods must strike a precise balance between evaluation depth and resource efficiency to prevent costly repeated trials.

- Iterative Development
- Resource Efficient

ACE

Chaos Engineering Methodology

Chaos-based evaluation systematically injects controlled disruptions into an agent's environment to observe behavioral robustness, recovery, and adaptation over time—shifting focus from static task success to longitudinal behavioral analysis.

Steps 6

Metrics 12+

1 Define Hypotheses

Identify what should remain stable under stress and establish success criteria for agent behavior.

- Target behavior
- Success criteria

2 Select Faults

Choose fault types and parameters (where, what, when) for controlled disruption injection.

- Network latency
- DB failures

3 Schedule Disruptions

Set up overlapping, time-extended disruptions to simulate realistic failure conditions.

- Bursty
- Sustained

4 Run Experiments

Execute multi-seed experiments and capture telemetry for comprehensive analysis.

- Multi-seed
- Telemetry

5 Compute Metrics

Calculate behavioral metrics including MTTR-A, drift, and state corruption rates.

- MTTR-A
- Drift

6 Analyze Results

Review drift windows, generate robustness scorecards, and identify failure patterns.

- Report
- Patterns

Certifier

Validates hypothesis & system safety thresholds.

AgentCert

ACE (Agent Chaos Engineering) Architecture

Platform blueprint illustrating user configurations, isolated runtime injection, and integration adapters.

CHAOS HUB | DECLARATIVE SPECIFICATIONS (DEVELOPER EXTENSIBLE)

- **App Spec** — Defines app metadata & target configurations.
- **Agent Spec** — Configures chaos agent behaviors and policies.
- **Fault Spec** — Specifies injection type, duration, and targets.
- **Workload Spec** — Defines load profiling, traffic rules & API rates.
- **Experiment Spec** — Orchestrates timeline safely triggers, metrics.

EXPERIMENT RUNTIME | DYNAMIC EXECUTION PIPELINE (PROVIDED BY ACE)

- **App Setup** — Provisions and boots the target application system.
- **Agent Setup** — Spins up container-native chaos daemon processes.
- **Fault Injection** — Triggers active packet loss, latency, CPU & disk chaos.
- **Workload Injection** — Simulates live user load and persistent API stress.

SECURE SANDBOX ISOLATION CHAMBER — Hyper-isolated Docker-in-Docker / gVisor workspace with full network virtualization, API rate limiters & firewall policies

INTEGRATIONS & SERVICES | AMBIENT SERVICES, CONTROLLERS, AND EXPORTERS

- **Agent Sidecar** — Interrogates and controls local container processes.
- **LLM Gateway** — Coordinates LLM prompts, API security and caching.
- **Telemetry Collector** — Aggregates microservices' metrics, logs, and traces.
- **Certifier** — Validates hypothesis & system safety thresholds.
- **Report** — Compiles chaos results, PDFs, and dashboard logs.

Extensible by Developer (Customizable Modules)

Provided by ACE Platform (Core Runtimes)

v1.2.0 Blueprint

ACE

Demo Walkthrough

Step-by-step chaos engineering workflow for AI agent evaluation

ACE

Open-Source Summit India 2026

Star the Repo

Support the project by starring our GitHub repository

Take the Quiz

You chance to become a contributor

Join Community

Connect with other developers

Thank you!

Let's build more reliable AI agents together

ACE

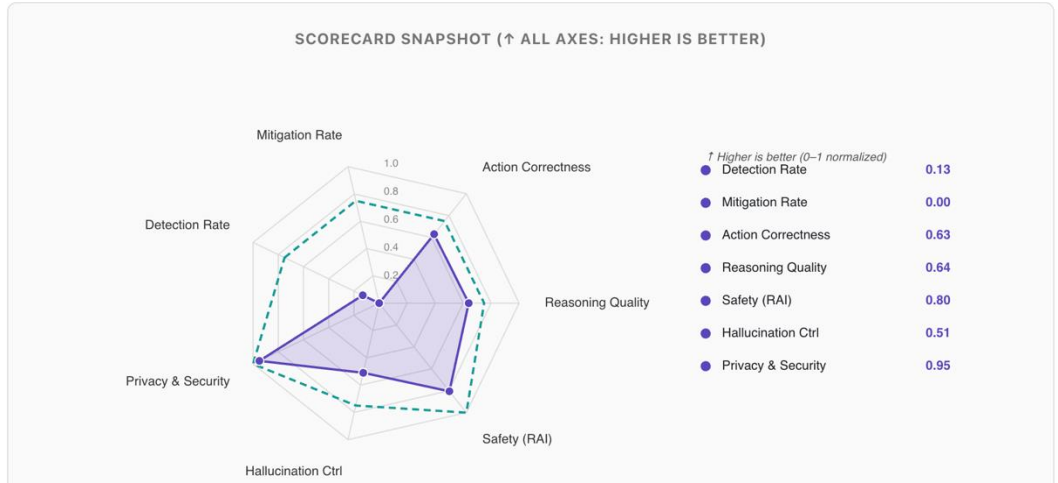
Certificate Sample

1.2 Experiment Scope

This report covers testing of the **5314847b-ab8c-44a8-bbb1-418f464aefa7** agent, which is designed to operate in faulted Kubernetes-like environments for reliability-related tasks. The test scope included **2** fault categories: **Network** (fault type **pod-network-loss**) and **Resource** (fault types **pod-cpu-hog** and **pod-memory-hog**). A total of **30** runs were executed, with **30** successful runs, and each of the **3** fault types was exercised across **30** runs. Evaluation was performed using a Multi-judge LLM Council methodology with k=3 judges plus meta-reconciliation. Within this scope, the agent's capabilities for fault detection, diagnosis, remediation, and compliance behavior were evaluated.

2 FAULT CATEGORIES	**3** FAULTS TESTED	**30** TOTAL RUNS
30 SUCCESSFUL RUNS	**0** FAILED RUNS	**9 (H1 – H9)** HYPOTHESES TESTED
Sufficient SAMPLE ADEQUACY		

1.3.2 Scorecard Snapshot



1.3.3 Key Findings

1. [Concern] Mitigation performance near-zero: Overall mitigation rate is only 3.3% across 30 successful runs, with Network mitigation at 7% and Resource at 0%, and category TTM scores effectively zero (e.g., 0.0055 for pod-network-loss).
2. [Concern] Slow and uneven detection: Despite an 83.3% overall detection rate across 30 successful runs, TTD scores are very low (0.0757 Network, 0.193 Resource) and Time-To-Detect differs significantly across categories (Kruskal-Wallis p=0.000828), indicating delayed and inconsistent detection performance.
3. [Good] Strong security and privacy: Security and privacy controls are robust, with 0 sensitive exposures, high per-category security compliance (93–97%, Wilson CIs all above ~79%), and qualitative assessments consistently rating security as strong.
4. [Good] Consistent, structured reasoning: Reasoning quality is moderate but stable (0.63–0.64 across categories) with qualitatively described methodical, tool-driven diagnostics and hallucination levels kept in a mid-range band (means ~0.40, max <0.51).
5. [Note] Detection stronger on resources: Resource detection is higher (90% vs 77% for Network), and the chi-squared test for detection rate uniformity is significant (p=0.014), indicating a real cross-category gap even though both are generally high.
6. [Note] Subfault detection imbalance: Within Resource, pod-cpu-hog shows a higher TTD score (0.362) than pod-memory-hog (0.0239) despite

▸ 2 Evaluation Methodology

AGENT CAPABILITY ASSESSMENT

PART I **Agent Capability Assessment**

▸ 3 Overall Qualitative Findings (LLM Council Output)

▸ 4 Detection & Response Performance

▸ 5 Reasoning & Quality

Demo results from IT-Ops agent evaluation with compound failures

4 Key Metrics **3** Takeaways

AgentCert

Evaluation Landscape: Objectives & Metrics Matrix

APPENDIX

Agent Behavior

Task Completion

Metrics: Success Rate, F1, Pass@k, Progress

Benchmarks: *AgentBoard, WebShop, AgentBench...*

Output Quality

Metrics: Coherence, Satisfaction, Usability

Benchmarks: *PredictingIQ, EnDex, PsychoGAT...*

Latency & Cost

Metrics: Latency, Token Efficiency, Cost

Benchmarks: *MobileBench, WebArena, GUI Agents...*

Agent Capability

Tool Use

Metrics: Completion Rate, Selection Acc

Benchmarks: *ToolEmu, MetaTool...*

Planning

Metrics: Reasoning, Progress, Plan

Benchmarks: *AgentBoard, MMLU...*

Memory

Metrics: Factual Recall, Consistency

Benchmarks: *LongEval, SocialBench...*

Multi-Agent

Metrics: Info Sharing, Role Switching

Benchmarks: *AgentSims, MATSA...*

Reliability

Consistency

Metrics: pass^k scores, Consistency Multi-run

Benchmarks: *τ-Bench, AgentBoard...*

Robustness

Metrics: Accuracy Under Perturbation

Benchmarks: *HELM, WebLinX...*

Safety

Fairness

Metrics: Awareness, Violation, Ethics

Benchmarks: *CASA, R-Judge, SimuCourt...*

Harm

Metrics: Adversarial, Prompt Inject

Benchmarks: *AgentPoison, AgentDojo...*

Compliance

Metrics: Risk Awareness, Constraints

Benchmarks: *R-Judge, Cybench...*

Comprehensive map of Agent Evaluation Objectives, Metrics, and Benchmark Models

4 Core Objectives

50+ Benchmarks Indexed