USENIX SECURITY '26

SoK: DARPA's AI Cyber Challenge (AIxCC): Competition Design, Architectures, and Lessons Learned

Cen Zhang[1,5], Younggi Park, Fabian Fleischer[1], Yu-Fu Fu[1], **Jiho Kim**[1], Dongkwan Kim[5], Youngjoon Kim[1], Qingxiao Xu[2], Andrew Chin[1], Ze Sheng[2], Hanqing Zhao[1], Michael Pelican[3], David J. Musliner[3], Jeff Huang[2], Jon Silliman[4], Mikel Mcdaniel[4], Jefferson Casavant[4], Isaac Goldthwaite[4], Nicholas Vidovich[4], Matthew Lehman[4], Taesoo Kim[1,5]

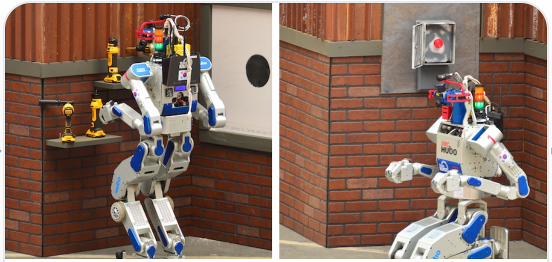
[1] Georgia Institute of Technology · [2] Texas A&M University · [3] Smart Information Flow Technologies
[4] Kudu Dynamics · [5] Microsoft
Younggi Park — Independent Researcher

Presented by **Jiho Kim**

DARPA Grand Challenges

A research competition becomes an early indicator of a new industry.

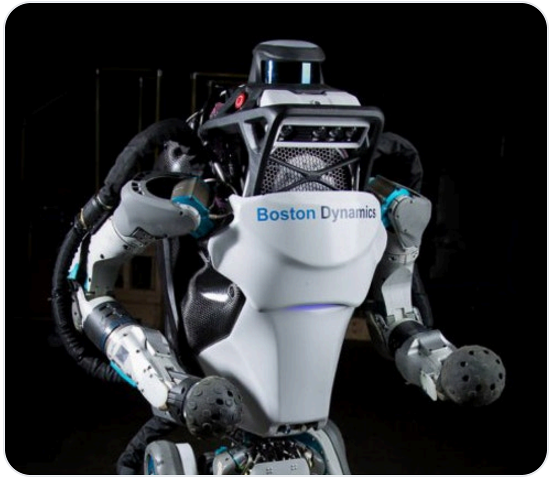
2004–2007
Autonomous Vehicle Challenge



2012–2015
Robotics Challenge

2014–2016
Cyber Grand Challenge

2023–2025
AI Cyber Challenge



Mayhem by ForAllSecure

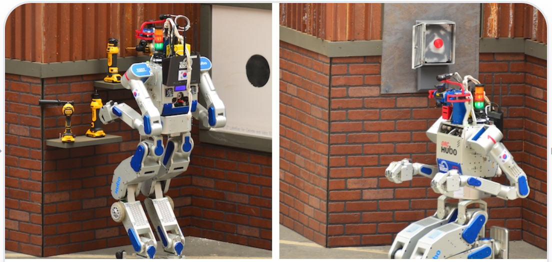
Microsoft

?

DARPA Grand Challenges

The newest signal is the largest autonomous-security evaluation run to date.

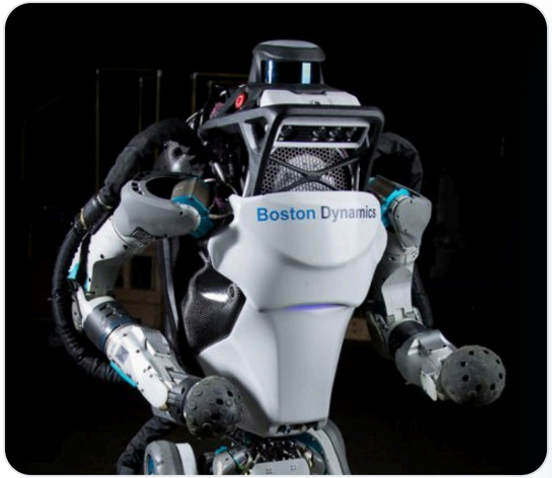
2004–2007
Autonomous Vehicle Challenge



2012–2015
Robotics Challenge

2014–2016
Cyber Grand Challenge

2023–2025
AI Cyber Challenge



AIxCC Final

DEF CON · AUGUST 2025

42 → 7
entrants narrowed to finalists

142.7 h
fully autonomous operation

$29.5M
competition prize purse

$135K
compute + LLM credits per team

$85K Azure compute + $50K LLM API credits — per finalist

Why an AIxCC SoK?

A milestone competition — but no systematic account of its **design, systems, or results.**

WHO NEEDS THE ANSWERS?

ORGANIZERS	Designing future competitions like AIxCC
RESEARCHERS	Building vulnerability discovery and repair techniques
PRACTITIONERS	Deploying autonomous security systems in practice

Why an AIxCC SoK?

A milestone competition — but no systematic account of its **design**, **systems**, or **results**.

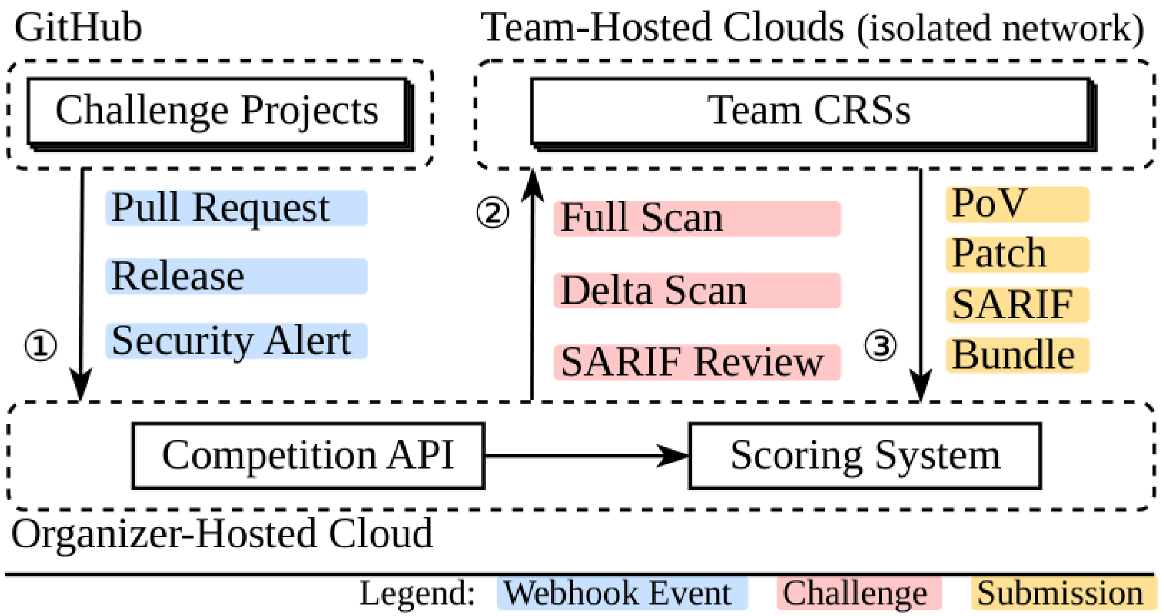
WHO NEEDS THE ANSWERS?

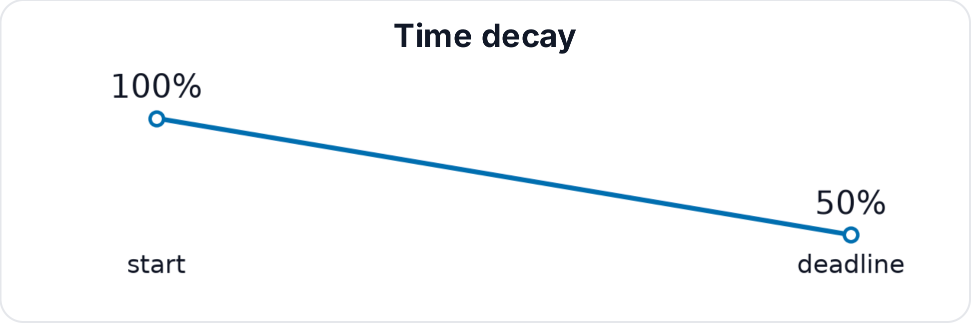
ORGANIZERS	Designing future competitions like AIxCC
RESEARCHERS	Building vulnerability discovery and repair techniques
PRACTITIONERS	Deploying autonomous security systems in practice

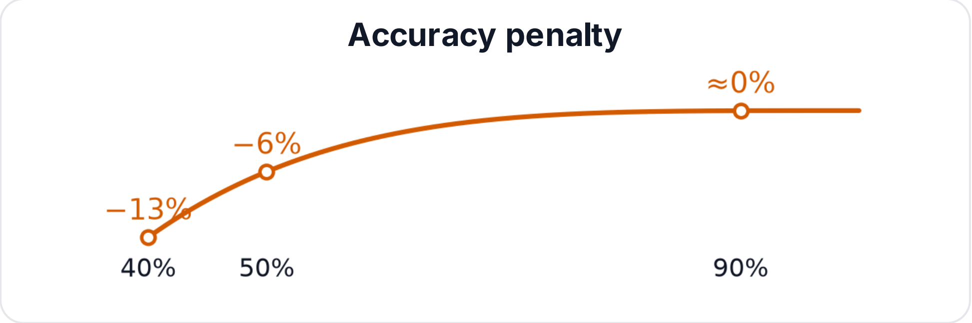
What evidence did we use?

1. **Full competition database**
   DARPA-approved access to challenges, logs, traces, and scores
2. **Regular discussions with organizers**
   Kudu Dynamics — competition design and organizer reflections
3. **Communication with all seven finalist teams**
   One-on-one meetings or a structured questionnaire
4. **Submission-version codebases**
   Code as the grounded fact · cross-validation by ≥2 authors

AIxCC Final Competition Design







Challenges (Synthetic Vulnerabilities)

24 OSS projects	**53** challenges	**63** injected vulns	**34** CWE types	**301** harnesses

CWE coverage CWE-20 · CWE-22 · CWE-28 · CWE-29 · CWE-35 · CWE-77 · CWE-120 · CWE-121 · CWE-122 · CWE-123 · CWE-125 · CWE-126 · CWE-129 · CWE-134 · CWE-190 · CWE-193 · CWE-382 · CWE-400 · CWE-407 · CWE-415 · CWE-416 · CWE-457 · CWE-476 · CWE-611 · CWE-680 · CWE-695 · CWE-770 · CWE-787 · CWE-789 · CWE-834 · CWE-835 · CWE-917 · CWE-918 · CWE-1333

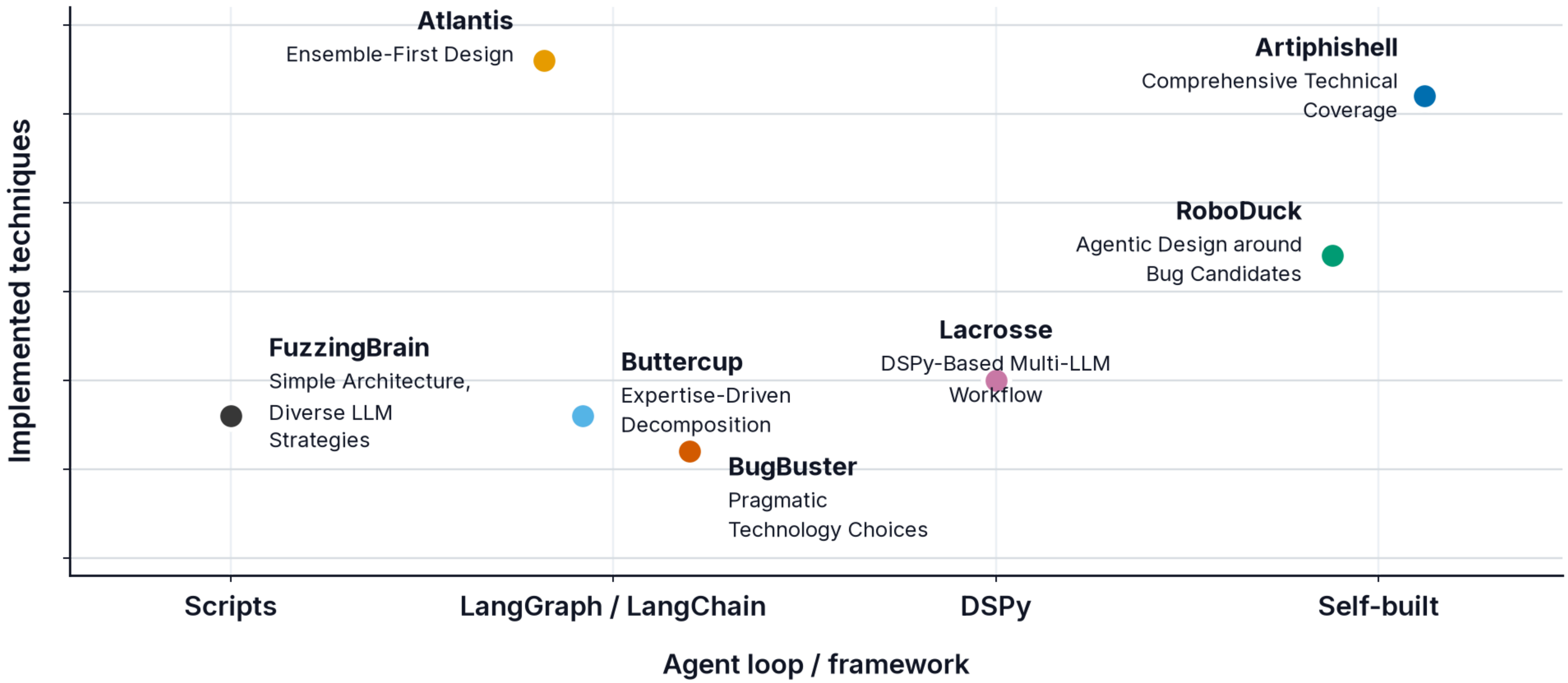
C 14 repositories

`curl` `dav1d` `freerdp` `little-cms` `libavif` `libexif` `libxml2` `mongoose`
`ndpi` `openssl` `shadowsocks-libev` `systemd` `wireshark` `xz`

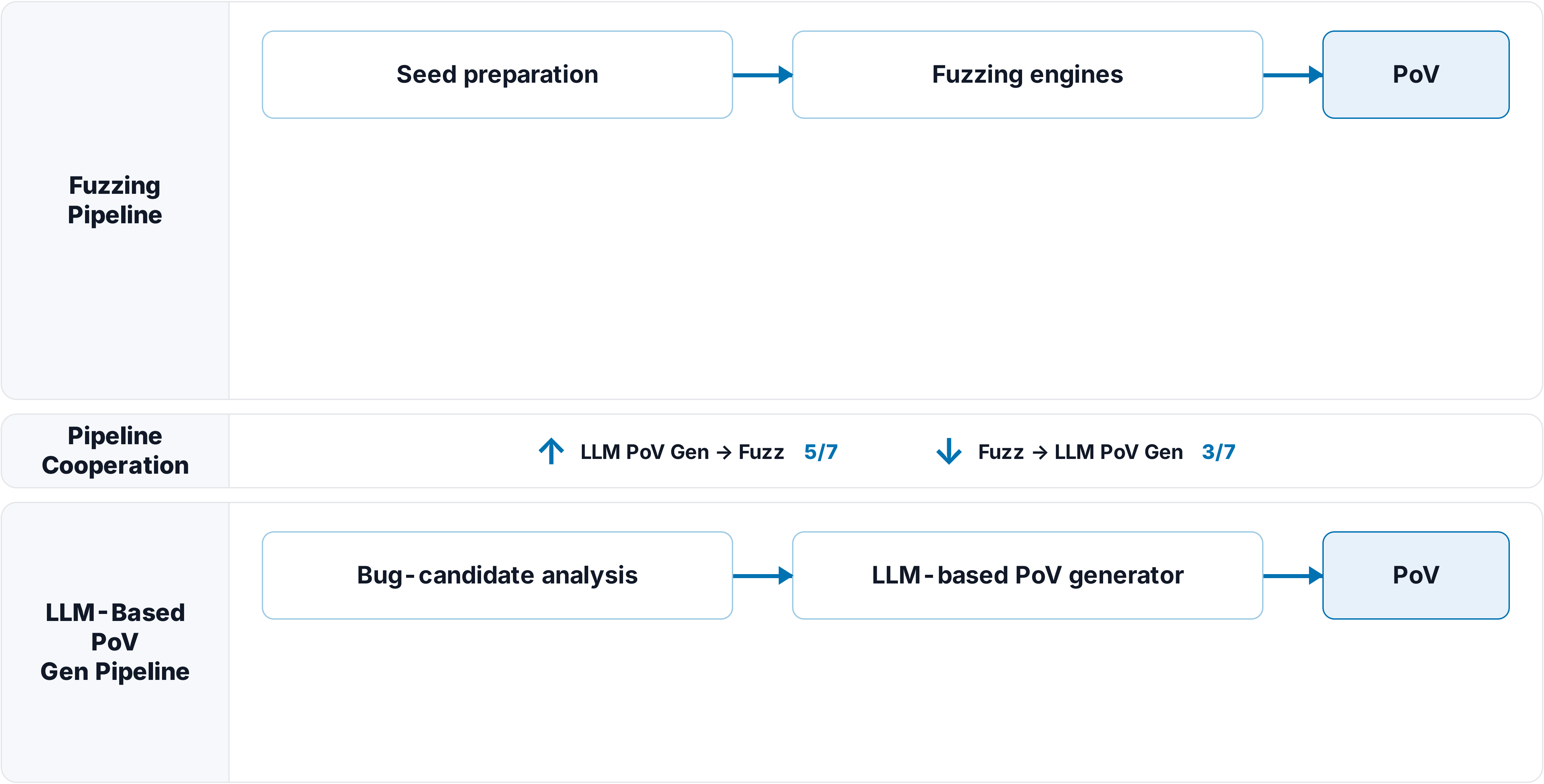
Java 10 repositories

`commons-compress` `dcm4che` `dicoogle` `healthcare-data-harmonization` `hertzbeat`
`jsoup` `logging-log4j2` `pdfbox` `poi` `tika`

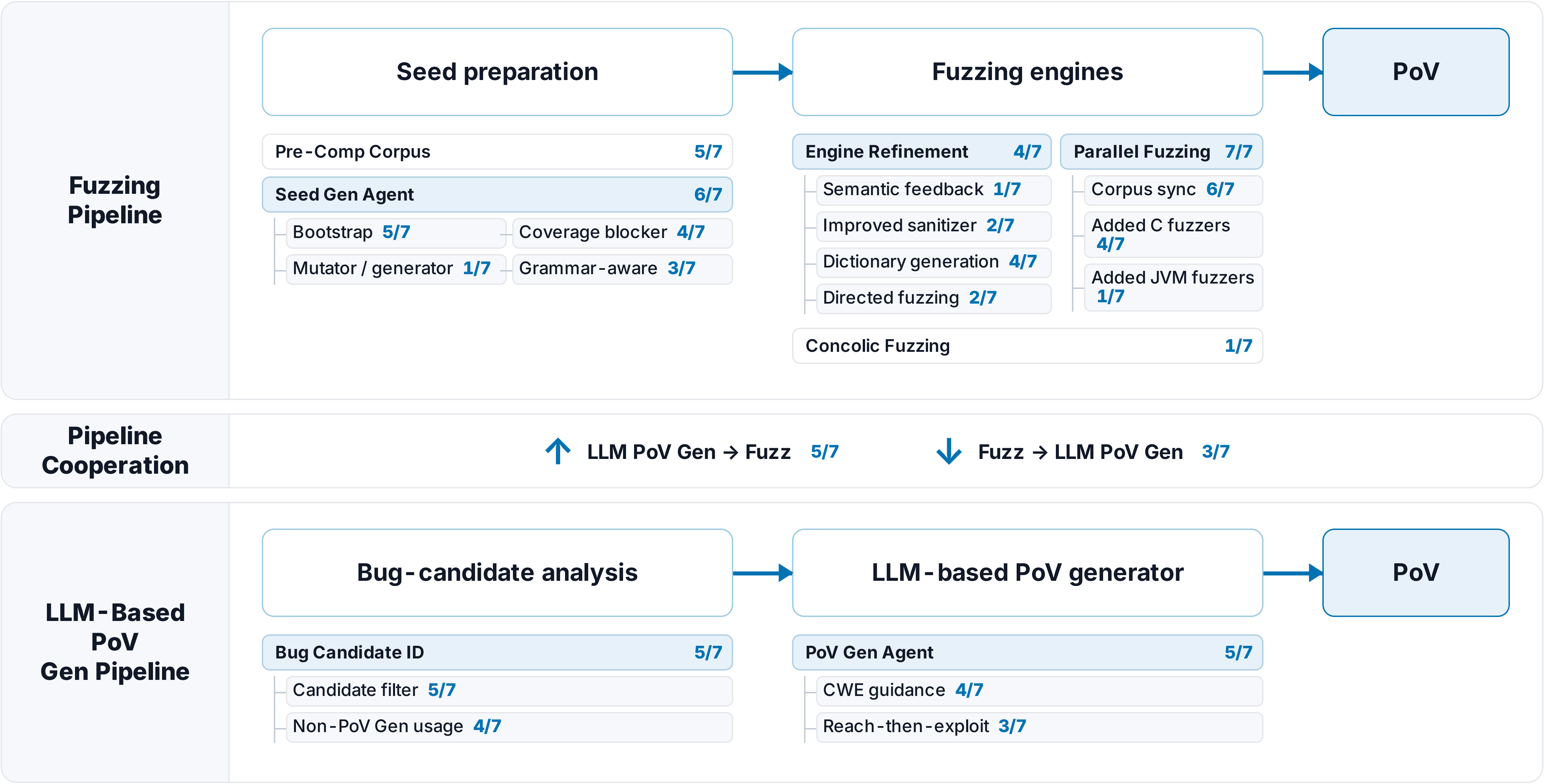
Seven Finalist Systems



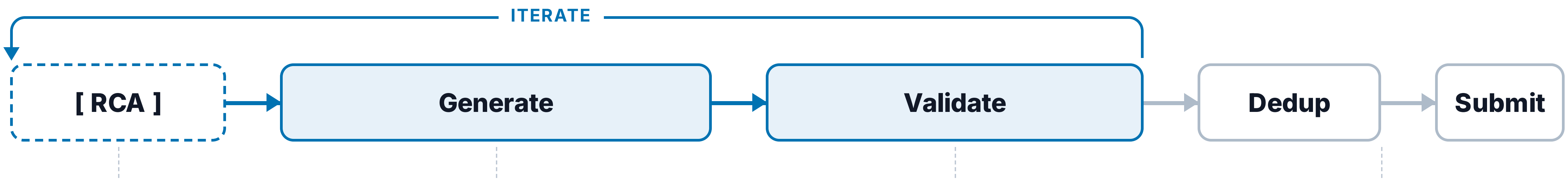
Two Cooperative Bug-Finding Pipelines



Two Cooperative Bug-Finding Pipelines

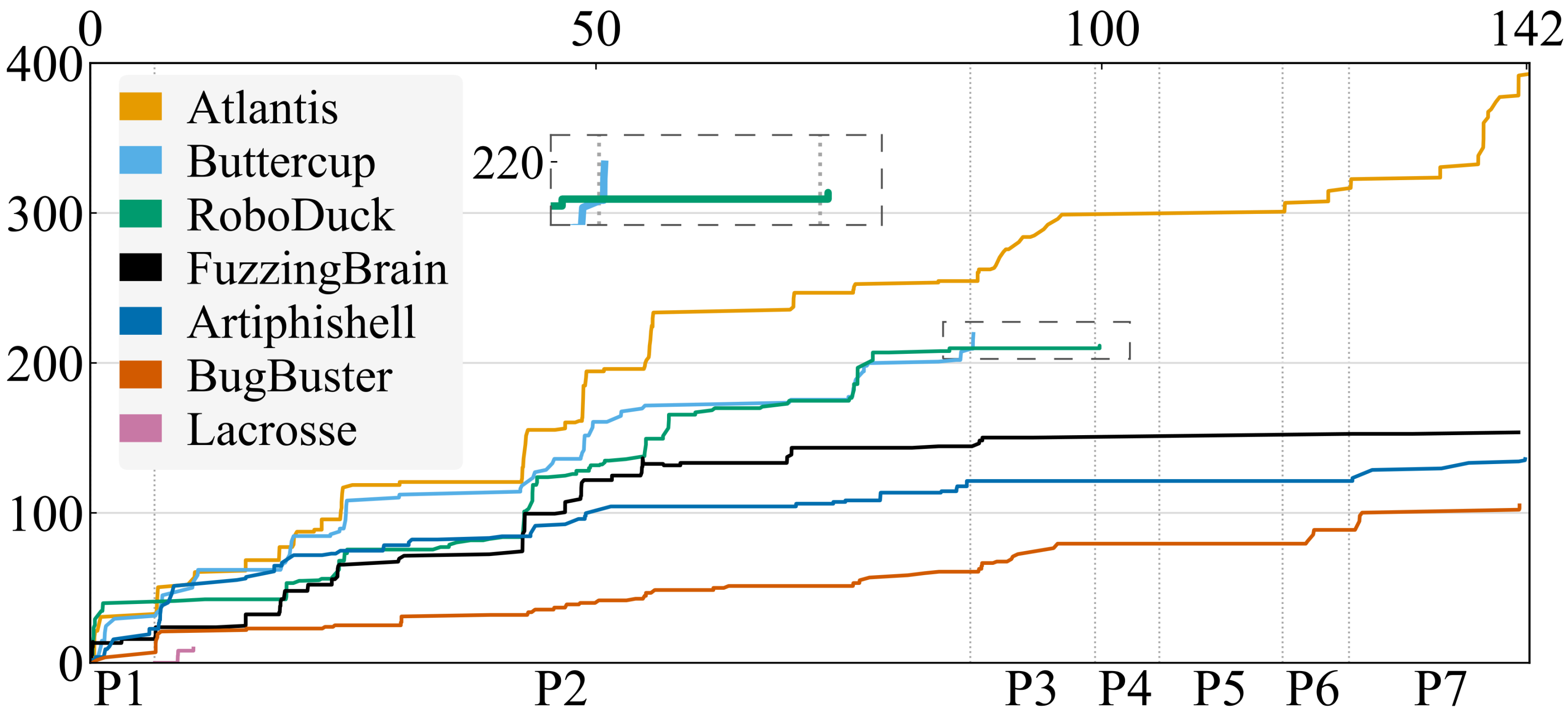


Patch Generation Pipeline



Patch Generation Pipeline

What the Scoreboard Reveals



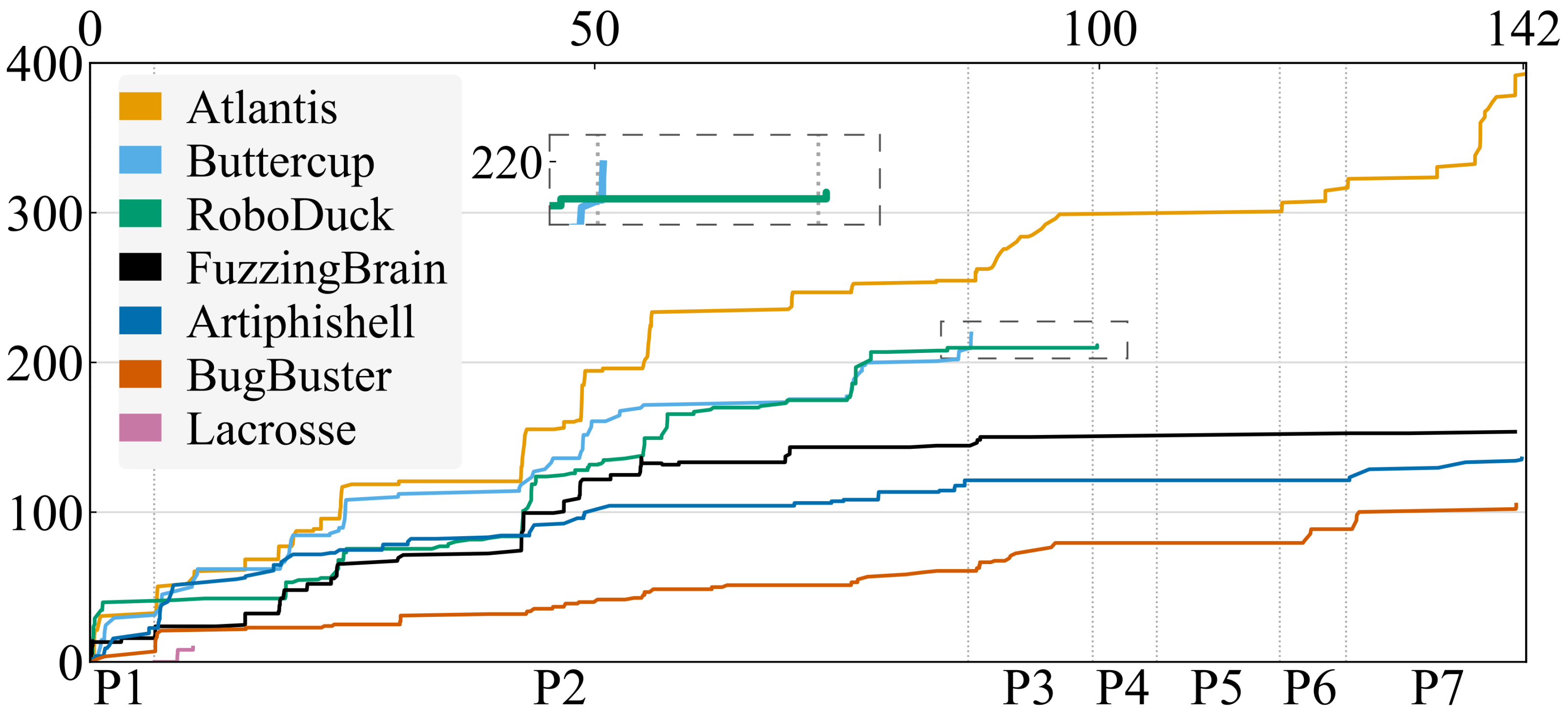
3 of 7

System stability
Stopped scoring before the final phases

−16.3

Submission accuracy
RoboDuck moved from second to third

What the Scoreboard Cannot Reveal



One score mixes three factors

- Technical capability
- Stability / engineering
- Strategy / accuracy

Direct attribution is unavailable

- ✗ **Ablation?** Prohibitive cost
- ✗ **Postmortem?** Incomplete telemetry

Per-Challenge Annotation

Representative techniques' performance as **analysis reference**

PoV annotation

Parallel Fuzzing (PF)

De facto non-LLM bug-finding approach

Final-round OSS-Fuzz config

16 cores / harness · 6 h × 3 runs

Patch annotation

MultiRetrieval (MR)

Minimal code-search + patch loop

Ground-truth PoV + crash evidence

Up to 3 patches × 3 runs

Correctness cross-validated by 2 experts

Patch annotation

Claude Code (CC)

General-purpose coding agent

Same evidence / budget as MR

Up to 3 patches × 3 runs

Correctness cross-validated by 2 experts

Ideal lab conditions
Competition-aligned budgets
LLM cutoff predates the final

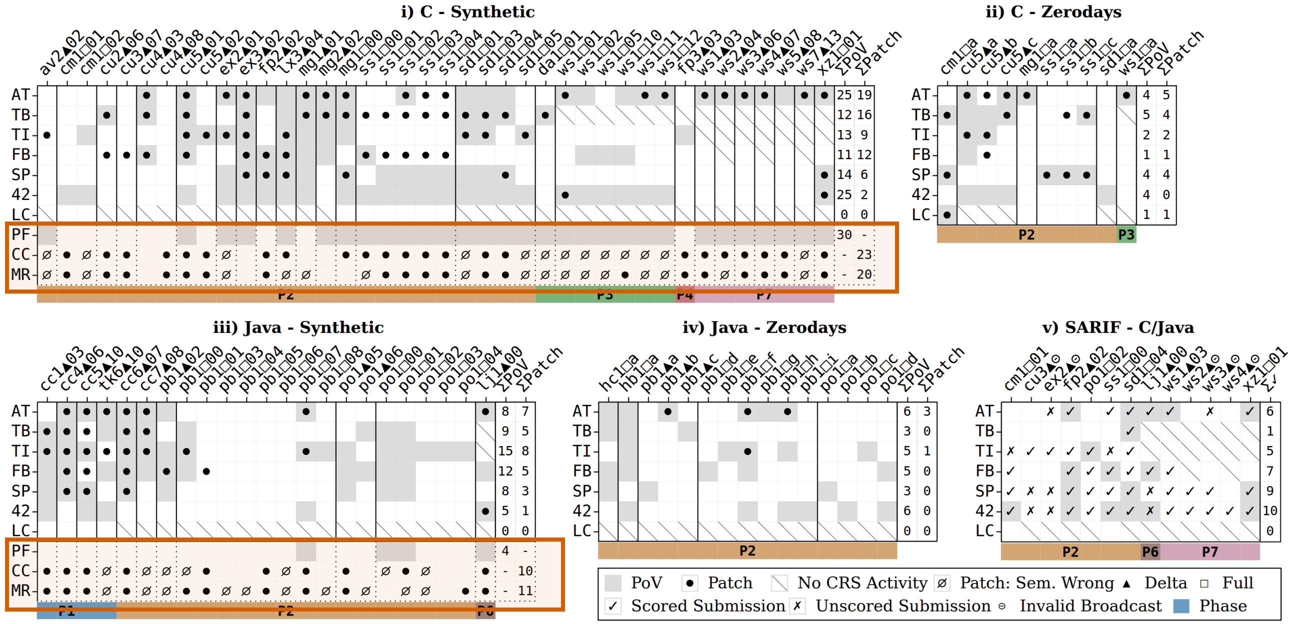
Per-Challenge Annotation

Representative techniques' performance as **analysis reference**

PoV annotation
Parallel Fuzzing (PF)
34 / 63

Patch annotation
MultiRetrieval (MR)
31 / 63

Patch annotation
Claude Code (CC)
33 / 63

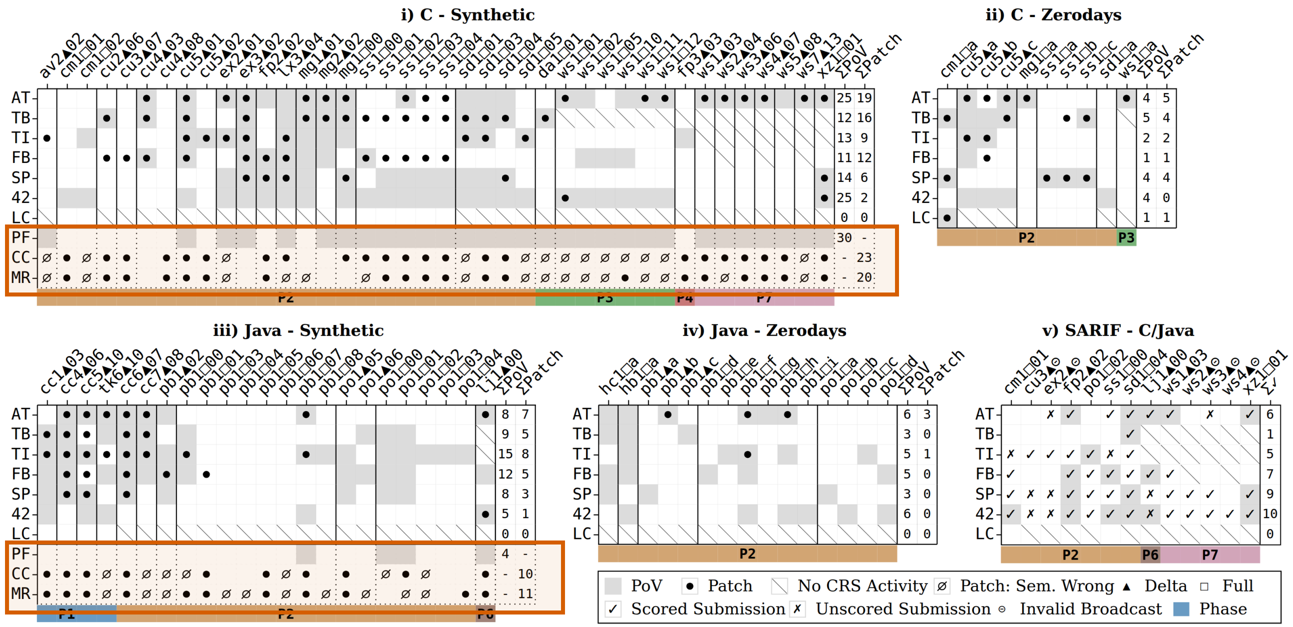


PF / CC / MR annotation rows

Revisit AIxCC Challenges with Annotations

AIxCC challenges emphasize real-world diversity—not extreme difficulty.

An autonomous system that reliably applies representative techniques would rank in the top three.



PF / CC / MR annotation rows

PoV Findings

PoV outcome	Finalist teams find	Finalist teams miss
PF finds	**PF coverage differs by language** • C: 30 / 40 (75%) • Java: 4 / 23 (17%)	**System robustness issues** • Broken dependencies and resource-heavy builds • Sanitizer or reproduction mismatch • Coarse crash deduplication
PF misses	**LLMs complement coverage-based fuzzing** • Targeted detection: diff-guided triggering inputs • Constraint-heavy inputs: structured formats and semantic guards	**Few unresolved cases** • Complex reasoning: cryptographic transformations • Fuzzing-unfriendly bugs: nested timeout/OOM cases

Patch Findings

Patch outcome	Finalist teams patch	Finalist teams miss
MR / CC patch	**Automatic checks do not establish semantic correctness** • Build, PoV reproduction, and project tests all pass • Yet CC: 37.7% wrong · MR: 45.6% wrong • High-accuracy finalist teams: still 16–21% wrong	**Patch capability can be lost before delivery** • 9 cases patchable by MR / CC, but missed by finalist teams • No PoV means no patch generation • Scheduling, outages, or per-case time limits stop the patch loop
MR / CC miss	**Finalist teams solve beyond MR / CC** • 16 additional vulnerabilities patched • Repair the underlying defect, not only the crashing path • Preserve functionality despite misleading diagnostics	**Seven Java vulnerabilities expose two limits** • Reasoning gaps: infinite loops, integer overflow, ReDoS • Rigid localization pipeline: timeout and JVM-crash signals reveal little root cause

Reflections

AIxCC is a trifecta of research, engineering, and strategy

Technique capability alone does not decide outcomes; stability and competition strategy determine what reaches the scoreboard.

As a competition, AIxCC emphasizes practicality—but translation to research and deployment is not automatic

Research needs analyzable telemetry and baselines; deployment needs efficient systems, standard interfaces, and OSS integration.

AIxCC-style competitions and AI-powered code scanning are mutually beneficial

Competitions foster the exploration and practical application of new techniques; new techniques, in turn, reshape future competitions.

Thank You

AIxCC Official Archive

https://archive.aicyberchallenge.com

Official finalist systems, challenge sets, and competition infrastructure

SoK Companion Website

https://occia.github.io/aixcc-sok-webpage

A collection of references, artifacts, supplemental analysis, and teams' public materials

OSS-CRS and CRS-Bench

https://oss-crs.openssf.org

OpenSSF CRS platform with a benchmark suite derived from AIxCC artifacts

SoK: DARPA's AI Cyber Challenge (AIxCC): Competition Design, Architectures, and Lessons Learned

Cen Zhang · Younggi Park · Fabian Fleischer · Yu-Fu Fu · **Jiho Kim** · Dongkwan Kim · Youngjoon Kim · Qingxiao Xu · Andrew Chin · Ze Sheng · Hanqing Zhao · Michael Pelican · David J. Musliner · Jeff Huang · Jon Silliman · Mikel Mcdaniel · Jefferson Casavant · Isaac Goldthwaite · Nicholas Vidovich · Matthew Lehman · Taesoo Kim