University of Stuttgart

MONASH University

Baden-Württemberg Stiftung

rcas

# Efficient Resilience Benchmarking of Microservice Architectures

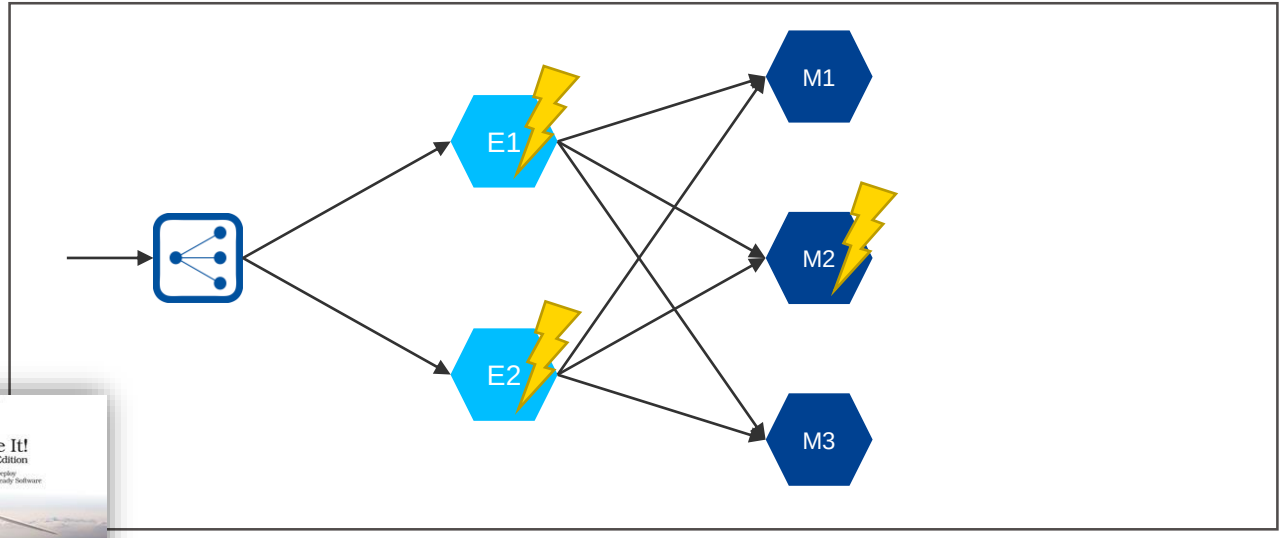
André van Hoorn
Aldeida Aleti
Thomas F. Düllmann
Teerat Pitakrat
*et al.*

@andrevanhoorn

2nd Vienna Software Seminar (VSS) on DevOps and Microservice APIs
August 29, 2019

# Resilience Antipattern

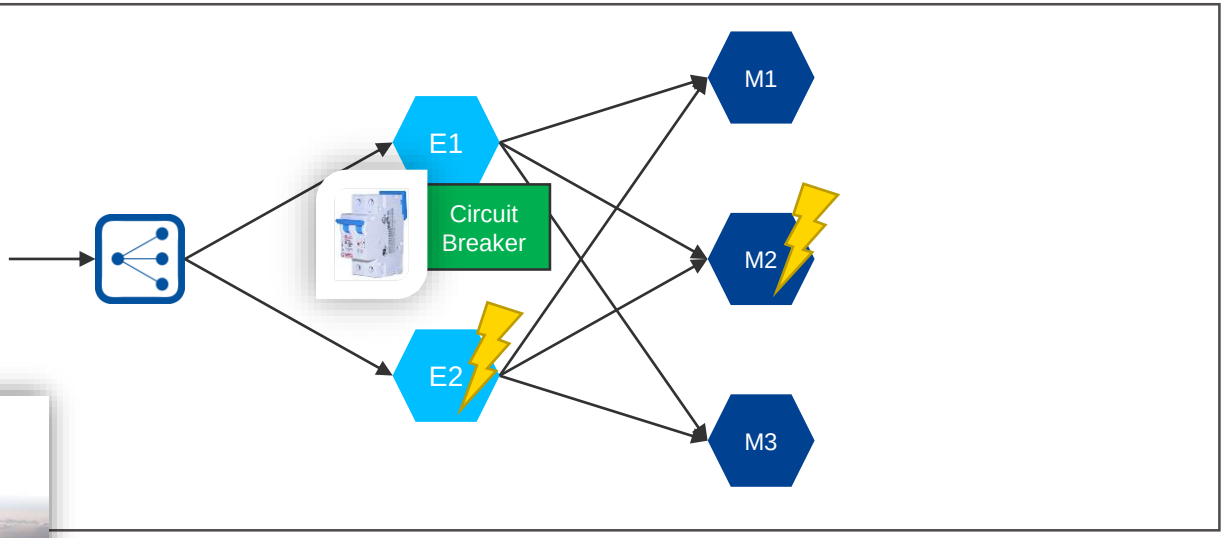
> “Recurring solution to common problem with negative consequences for the system”
>
> (Brown et al. Antipatterns: Refactoring Software, Architectures, and Projects in Crisis. John Wiley & Sons, Inc., 1998)





Cascading Failures

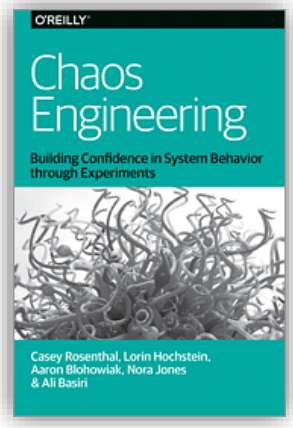
# Resilience Pattern Example: Circuit Breaker




# Resilience Benchmarking – aka Chaos Engineering

- How to accept failures? – Learning by doing:
  Intentionally inject failures into the production system

> "Chaos Engineering is the discipline of experimenting on a distributed system in order to build confidence in the system's capability to withstand turbulent conditions in production." — *Principles of Chaos*

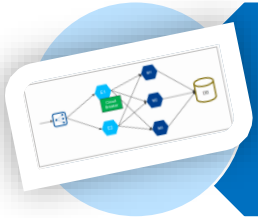


**“Current resilience benchmarking practice is inefficient.”**

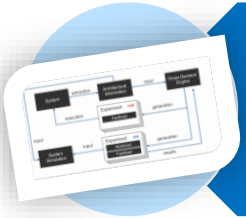
**André et al.**

Goal: **Make it more efficient!**

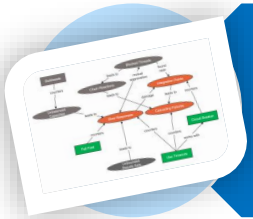
# Idea of the rcas Project



Consider software architectural knowledge to generate experiments



Combine model-based (simulations) and measurement-based („real“) resilience experiments



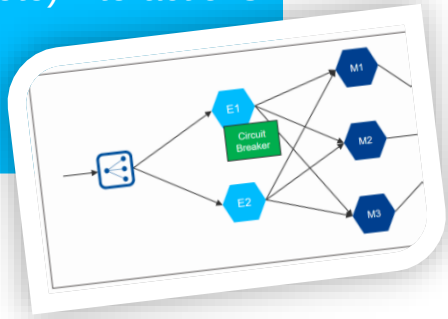
Leverage relationship between resilience patterns, antipatterns, and fault injections

# Envisioned Framework

Static and dynamic analysis + manual enrichment

System → extraction → Architectural Information

- Services, deployment, (remote) interactions
- Patterns and anti-patterns
- Criticality of services
- Steady-state metrics



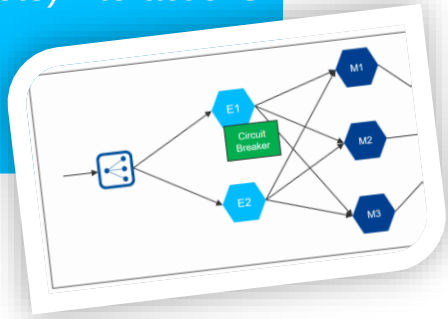

A. van Hoorn, A. Aleti, T. F. Düllmann, T. Pitakrat:
*ORCAS: Efficient Resilience Benchmarking of Microservice Architectures.*
ISSRE 2018

# Envisioned Framework

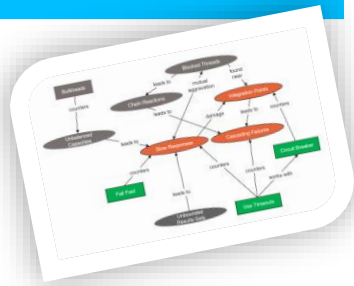
Static and dynamic analysis + manual enrichment

System → extraction → Architectural Information → input → Orcas Decision Engine

- Services, deployment, (remote) interactions
- Patterns and anti-patterns
- Criticality of services
- Steady-state metrics

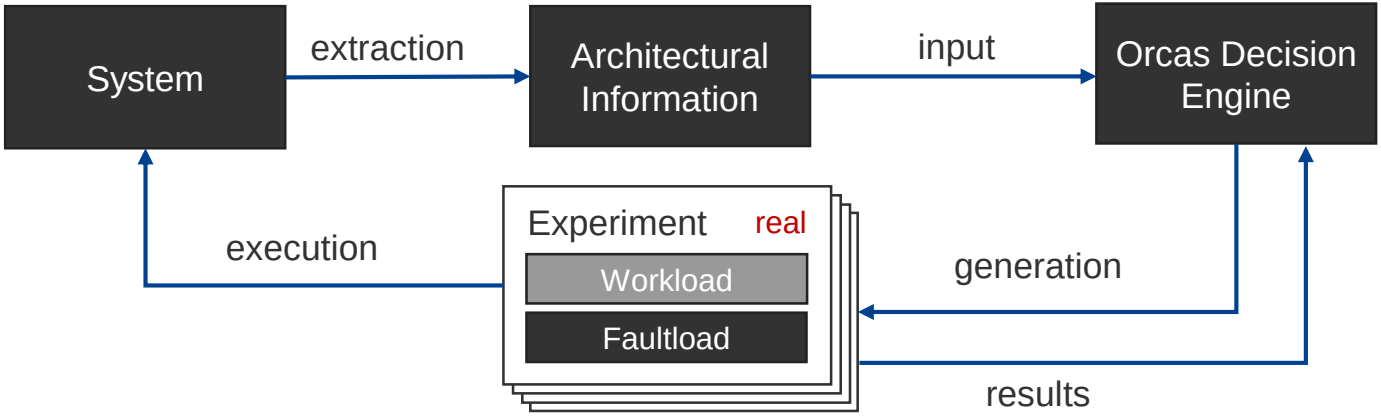



Knowledge and algorithms – „the magic“



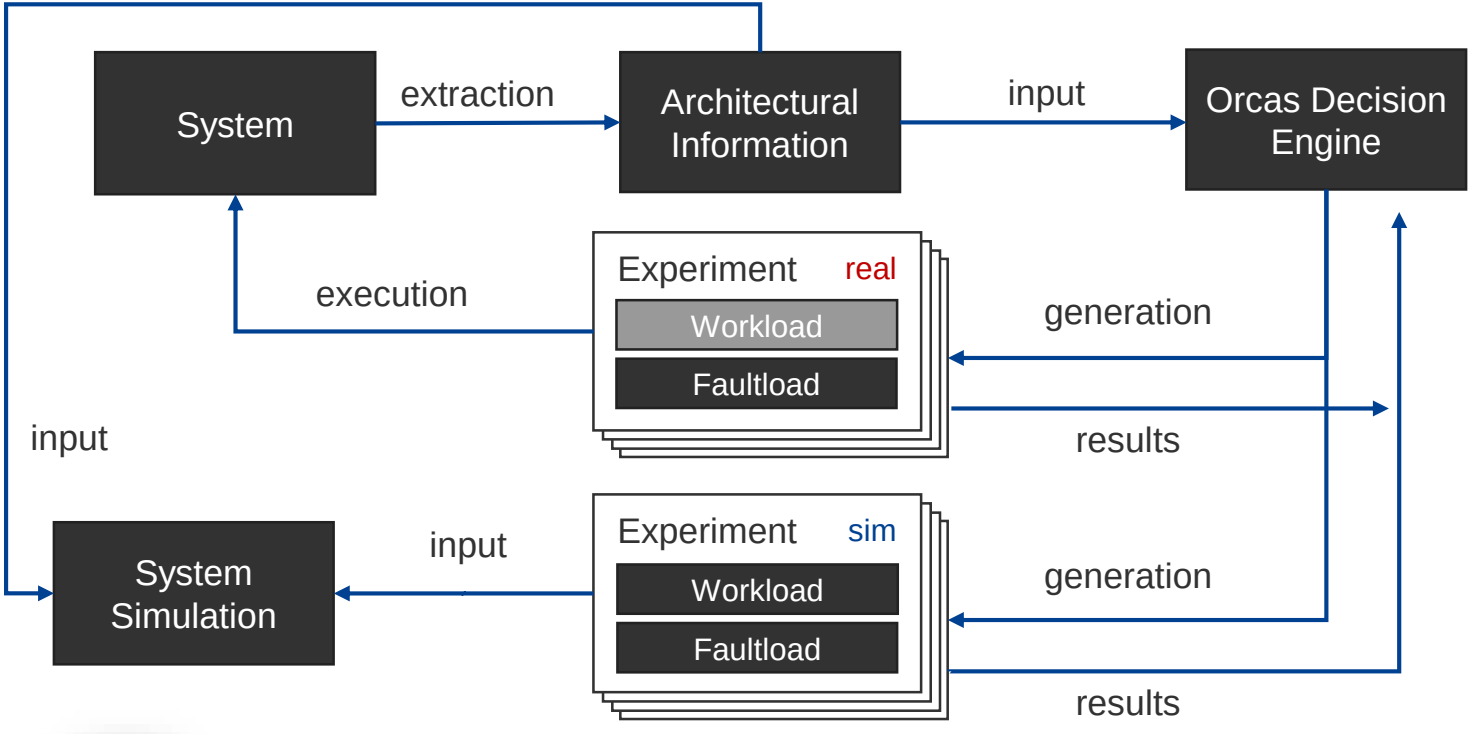
A. van Hoorn, A. Aleti, T. F. Düllmann, T. Pitakrat:
*ORCAS: Efficient Resilience Benchmarking of Microservice Architectures.*
ISSRE 2018

# Envisioned Framework



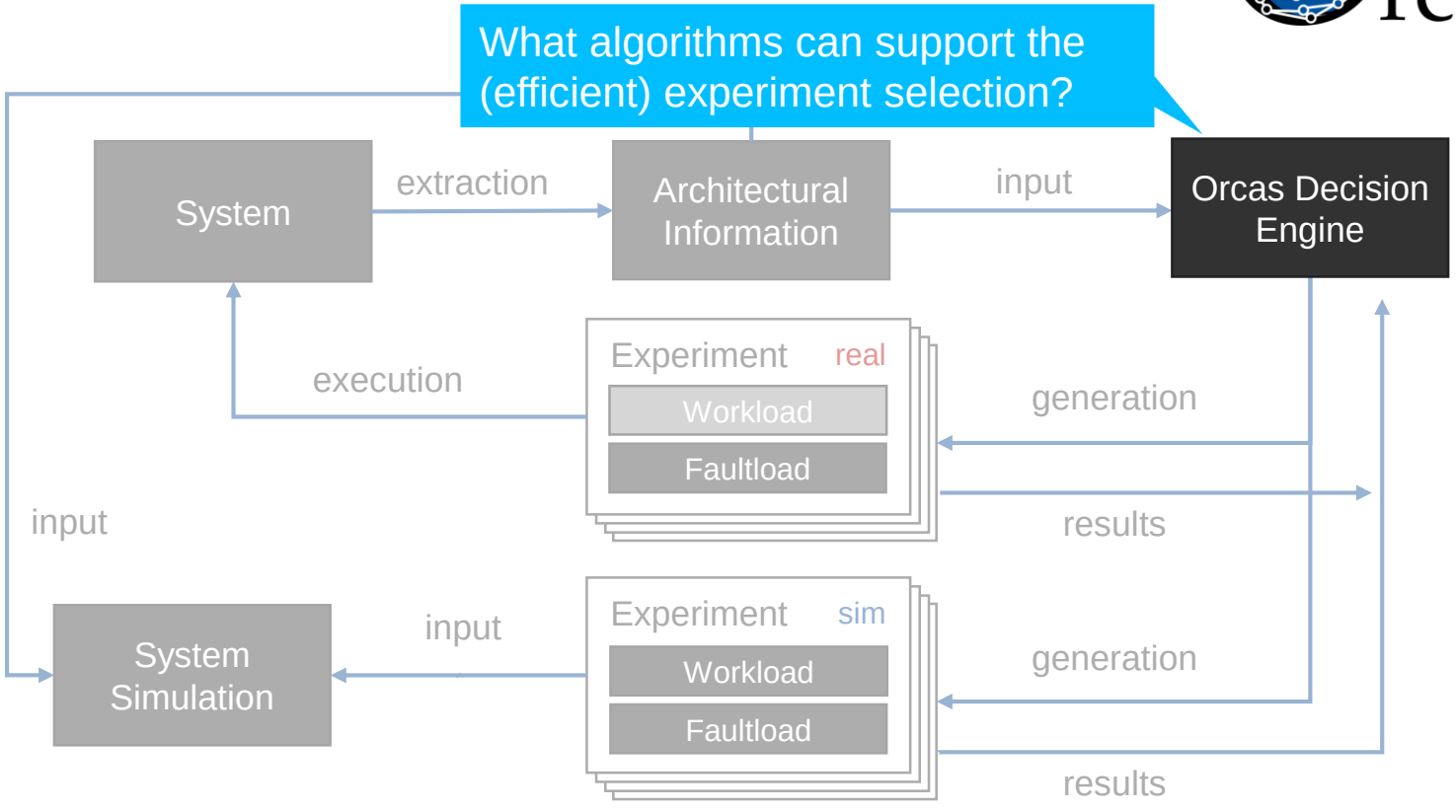

A. van Hoorn, A. Aleti, T. F. Düllmann, T. Pitakrat:
*ORCAS: Efficient Resilience Benchmarking of Microservice Architectures.*
ISSRE 2018

# Envisioned Framework

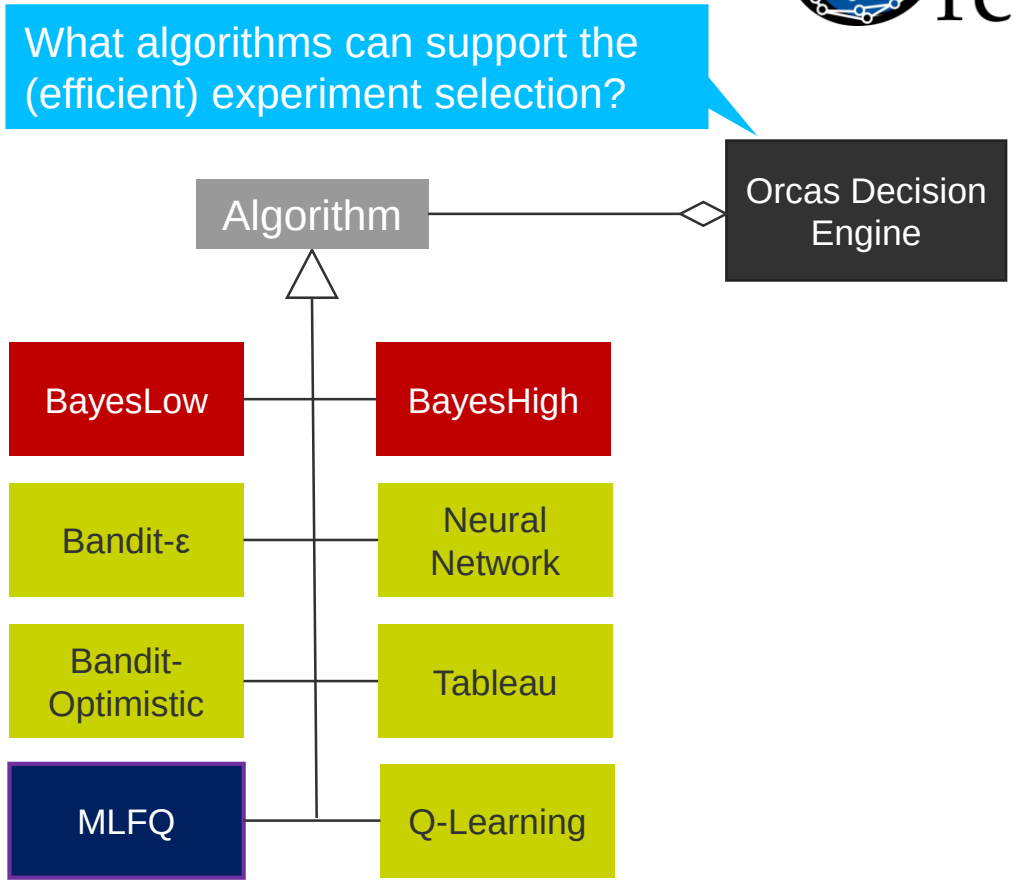



A. van Hoorn, A. Aleti, T. F. Düllmann, T. Pitakrat:
*ORCAS: Efficient Resilience Benchmarking of Microservice Architectures.*
ISSRE 2018

# Envisioned Framework – Decision Engine



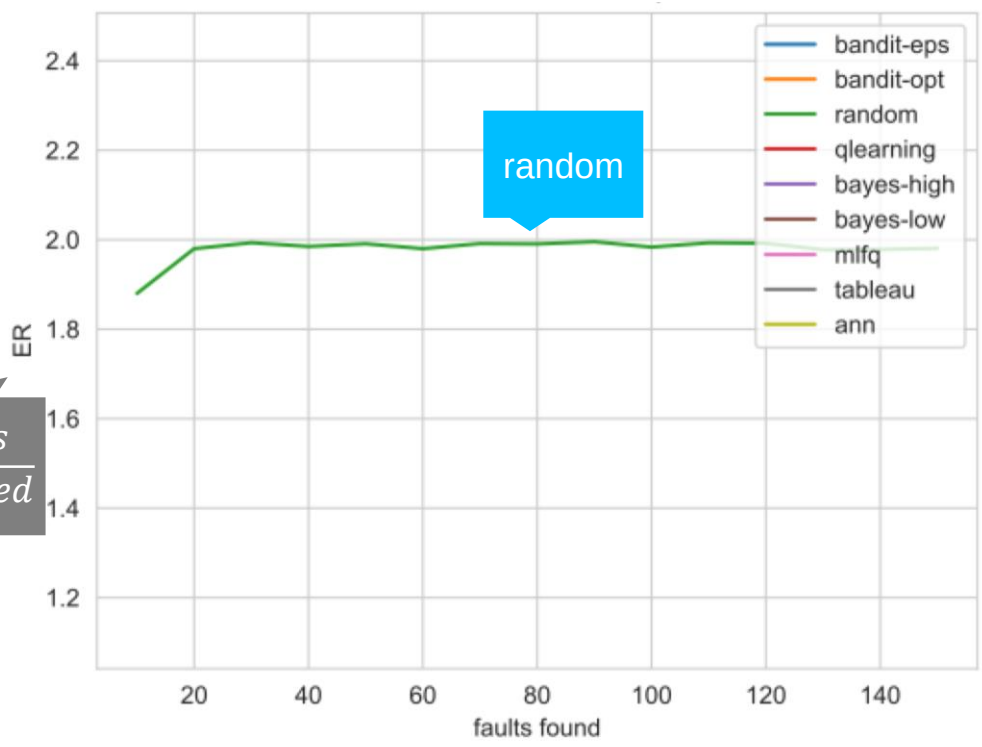
# Decision Engine




*Acknowledgment:*
Niklas Kammhoff. Algorithms for Efficient Chaos Experiment Selection in Microservice Architectures. July 2019. Bachelor's Thesis, University of Stuttgart

# Initial Results – Efficiency Rate (ER)

How do the algorithms compare with respect to the (fault detection) efficiency of the generated experiments?



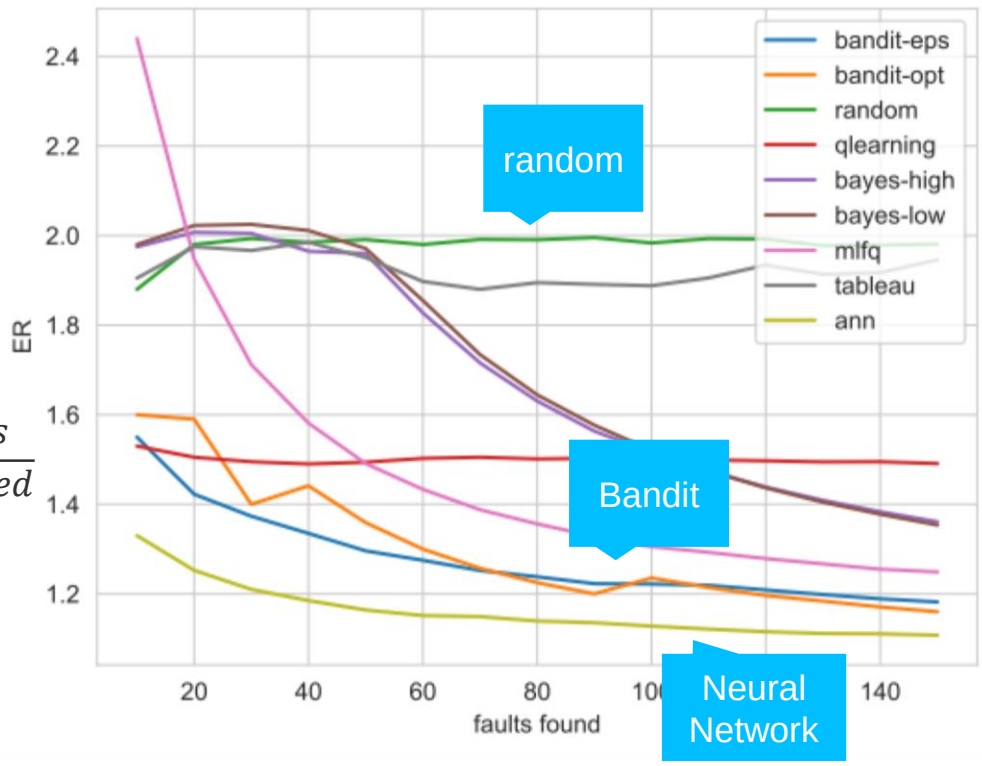

$$\frac{\#experiments}{\#faultsRevealed}$$

*Acknowledgment:*
Niklas Kammhoff. Algorithms for Efficient Chaos Experiment Selection in Microservice Architectures. July 2019. Bachelor's Thesis, University of Stuttgart

# Initial Results – Efficiency Rate (ER)

How do the algorithms compare with respect to the (fault detection) efficiency of the generated experiments?

$$\frac{\#experiments}{\#faultsRevealed}$$



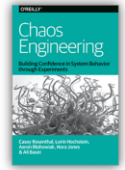

*Acknowledgment:*
Niklas Kammhoff. Algorithms for Efficient Chaos Experiment Selection in Microservice Architectures. July 2019. Bachelor's Thesis, University of Stuttgart

## Resilience Benchmarking – aka Chaos Engineering

- How to accept failures? – Learning by doing:
  Intentionally inject failures into the production system

> "Chaos Engineering is the discipline of experimenting on a distributed system in order to build confidence in the system's capability to withstand turbulent conditions in production." — *Principles of Chaos*

- Who is doing this?
  - Game Day exercises
  - Simian Army for AWS
  - Error Monkey for Node.js




## Idea of the rcas Project

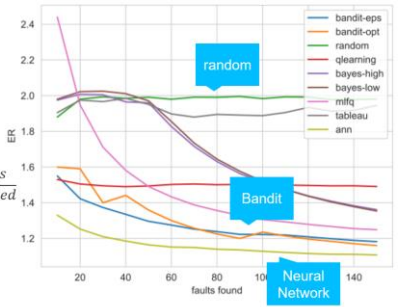
- Leverage relationship between resilience patterns, antipatterns, and fault injections
- Consider software architectural knowledge to generate experiments
- Combine model-based (simulations) and measurement-based („real") resilience experiments

## Envisioned Framework

rcas

## Initial Results – Efficiency Rate (ER)

How do the algorithms compare with respect to the (fault detection) efficiency of the generated experiments?




van Hoorn et al.: Efficient Resilience Benchmarking of Microservice Architectures

## Current and Next Steps

- Extended experimental evaluation
- Simulator extension and validation
- Combination with simulations
- + (anti-)pattern/injection knowledge
- Industry case study

A. van Hoorn, A. Aleti, T. F. Düllmann, T. Pitakrat: *ORCAS: Efficient Resilience Benchmarking of Microservice Architectures.* ISSRE 2018

https://github.com/orcas-elite/