# Debugging with Event Replay

In Microservices

On a normal day, an engineer open a bug reported by customer care...

# A Normal Bug Fixing Routine

**Bug 10239:** User not receiving reset password email

User (userId: hans, email: me@hans.info) forgot his password, triggered reset password but haven't received any password reset email. He already tried 3 times and checked his spam and found nothing.

Kata.ai

# Step 1: Try to Reproduce

Engineer try to reset his own password and it works fine.

He ask his colleague to try out, works fine too.

# Step 2: Checking logs


```
{
    "timestamp": 1283758490099872
    "route": "/reset",
    "method": "POST",
    "body": {
        "email": "me@hans.info",
        "captcha_token": "msd9adun12khfagghni"
    },
    "response": {
        "status": "ok"
    }
}
```


**Comment:** everything looks fine. I see no error from mailer service as well

Kata.ai

# Step 3: Think hard and wondering

Checking the code... Nothing suspicious at first glance.

Replicate in staging? So time consuming to copy and replicate the user data from production.

Woke midnight to replicate and monitor the log real time when nobody is generating traffic: found no error and nothing suspicious.

Kata.ai

# Step 4: Priority changes and workaround

Hey Chris, since we have other priorities, we won't fix this bug this sprint. I changed his password to pass123, let him know that he should change this after logging in.

**Solution:** Bug needs more time to debug, priority set to low. No long term solution yet.

Kata.ai

# A Normal Bug Fixing Routine (5)

**Two weeks later...**

**Multiple reports on the same problem. Priority raised.**

Kata.ai

What if the developer uses **Event Replay** for debugging?

# Step 1: Check log and get Event ID


```
{
    "timestamp": 1283758490099872
    "route": "/reset",
    "method": "POST",
    "body": {
        "email": "me@hans.info",
        "captcha_token": "msd9adun12khfagghni"
    },
    "response": {
        "status": "ok"
    }
    "eventId": "a89ff028"
}
```

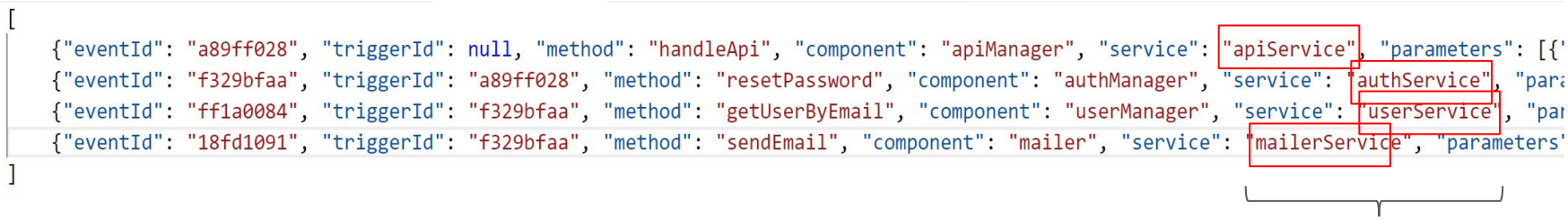

Kata.ai

# Step 2: Check trace in event store

```
$ devops-cli trace --event-id a89ff028 --out trace.json
```

Kata.ai

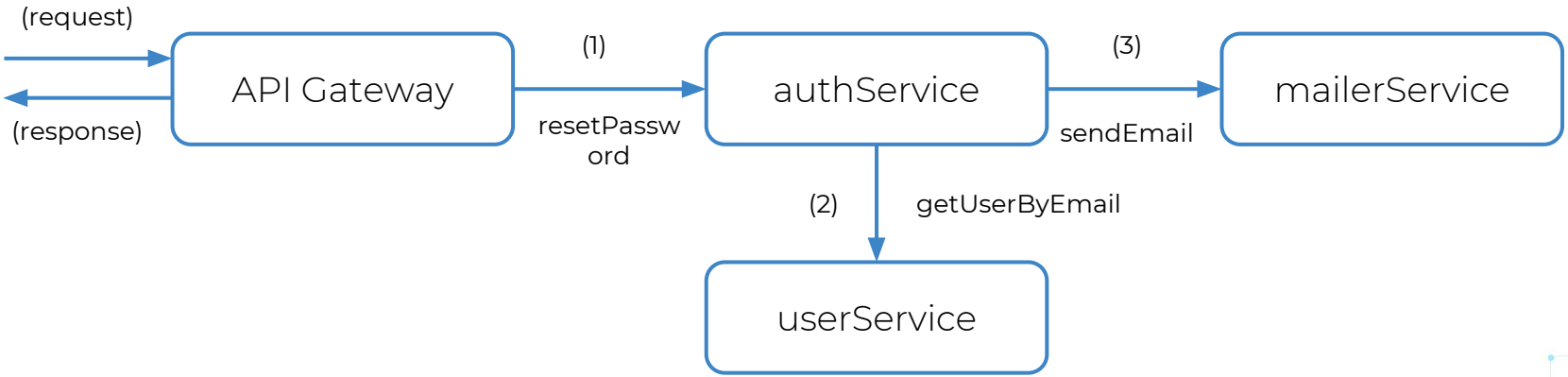
# Step 3: Analyze the trace.json




**eventId:** id of the created event
**triggerId:** eventId that triggers / precede current event
**method:** method that generated that event

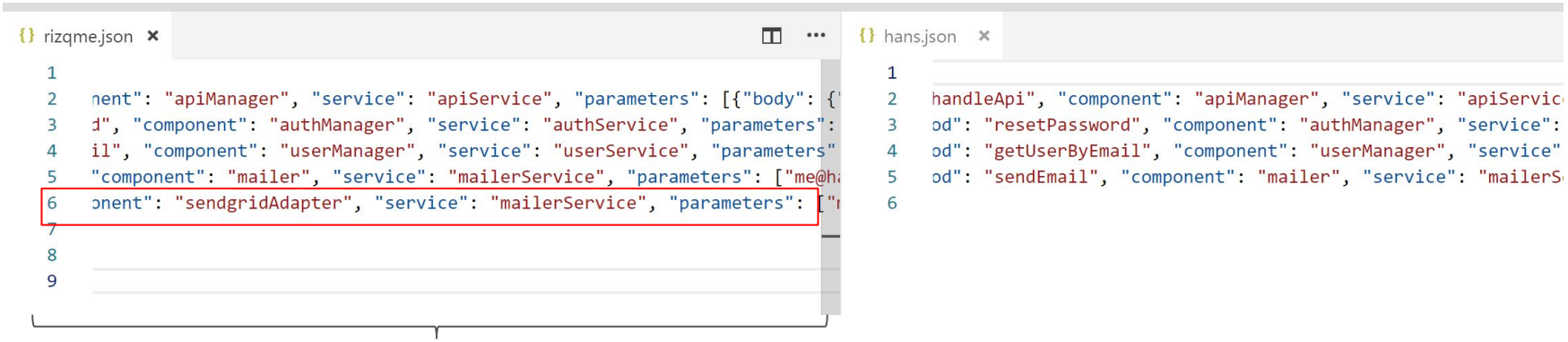
Kata.ai

# Step 4: Understand the flow




Kata.ai

# Step 5: Compare the trace with other trace



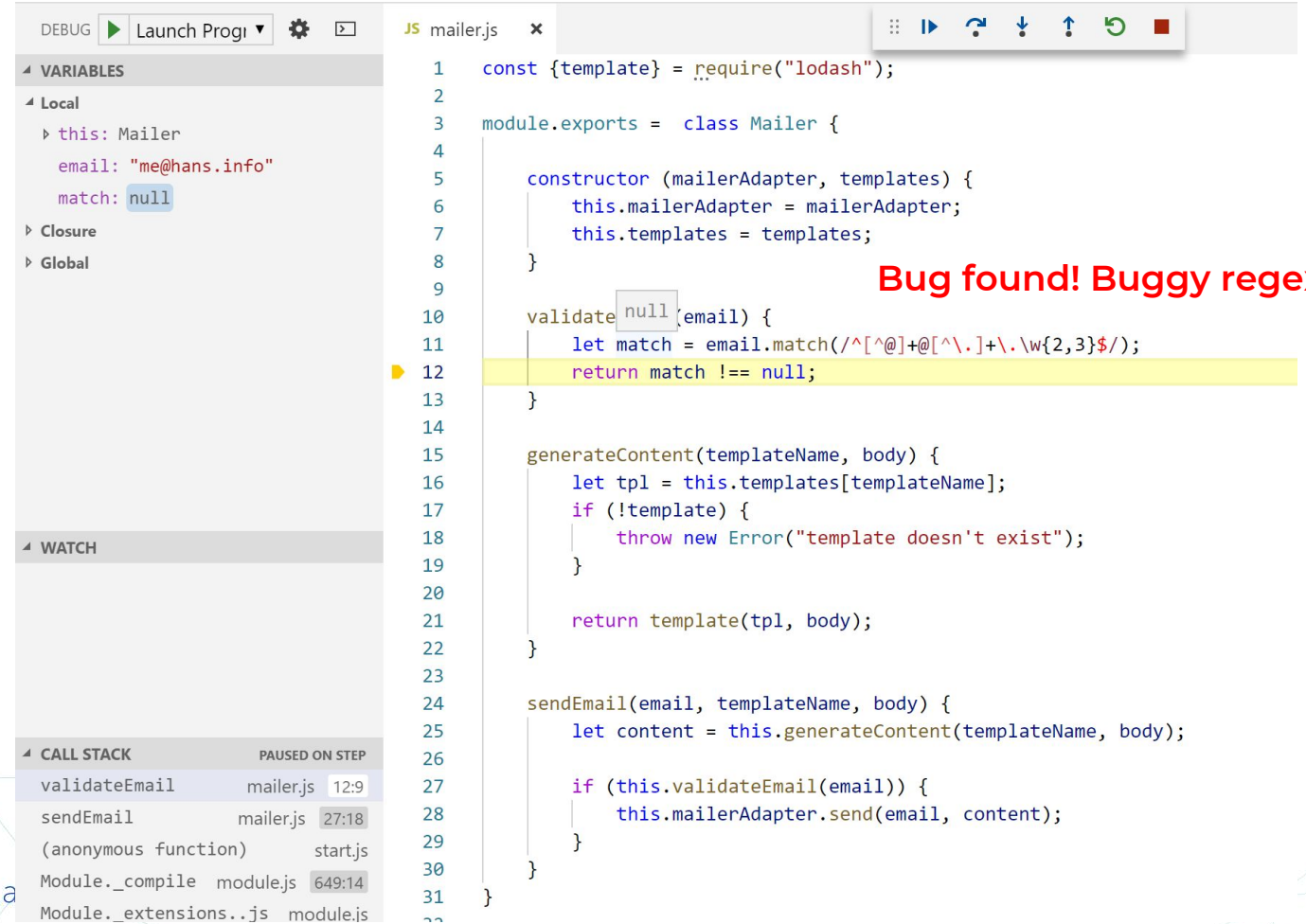

**Missing event**

**The one with buggy email address doesn't send email to sendgrid**

Kata.ai

# Step 6: Debug Mailer Service with Recorded Events

```
$ node --inspect-brk mailer-service --replay trace.json --start-event 18fd1091
```


DEBUG
Launch Progr
VARIABLES
Local
this: Mailer
email: "me@hans.info"
match: null
Closure
Global
WATCH
CALL STACK
PAUSED ON STEP
validateEmail mailer.js 12:9
sendEmail mailer.js 27:18
(anonymous function) start.js
Module._compile module.js 649:14
Module._extensions..js module.js
mailer.js
1 const {template} = require("lodash");
2
3 module.exports = class Mailer {
4
5 constructor (mailerAdapter, templates) {
6 this.mailerAdapter = mailerAdapter;
7 this.templates = templates;
8 }
9
10 validate null email) {
11 let match = email.match(/^[^@]+@[^\.]+\.\w{2,3}$/);
12 return match !== null;
13 }
14
15 generateContent(templateName, body) {
16 let tpl = this.templates[templateName];
17 if (!template) {
18 throw new Error("template doesn't exist");
19 }
20
21 return template(tpl, body);
22 }
23
24 sendEmail(email, templateName, body) {
25 let content = this.generateContent(templateName, body);
26
27 if (this.validateEmail(email)) {
28 this.mailerAdapter.send(email, content);
29 }
30 }
31 }
Bug found! Buggy regex

Kata.a

Only accept 2 or 3 letter domain!

```
validateEmail(email) {
    let match = email.match(/^[^@]+@[^\.]+\.\w{2,3}$/);
    return match !== null;
}
```

Kata.ai

Okay, how does it work?

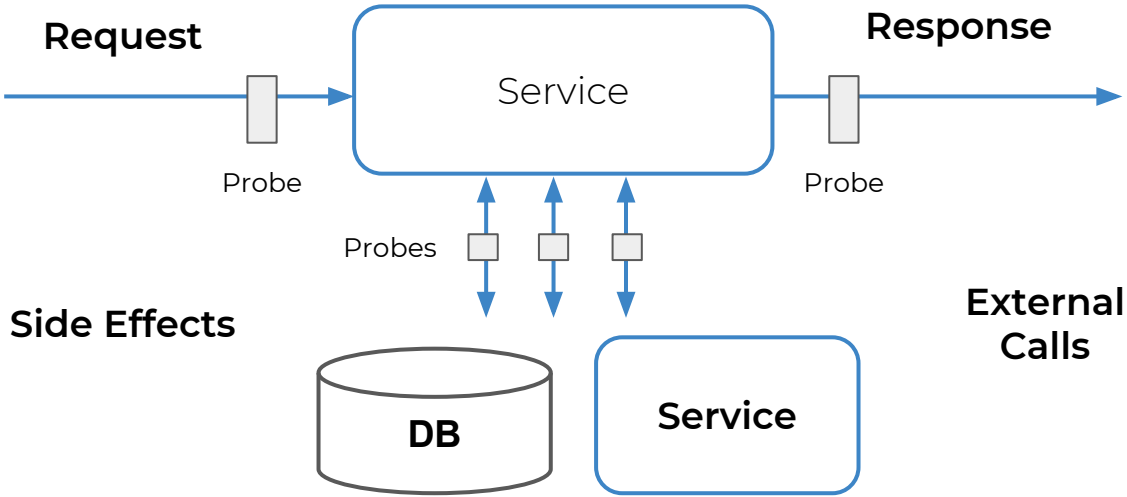
# Concept

**Probing / Instrumentation**
Intercept every methods that are interfacing to outside the service

**Tracing**
Log the events with information about the sequence of the flow

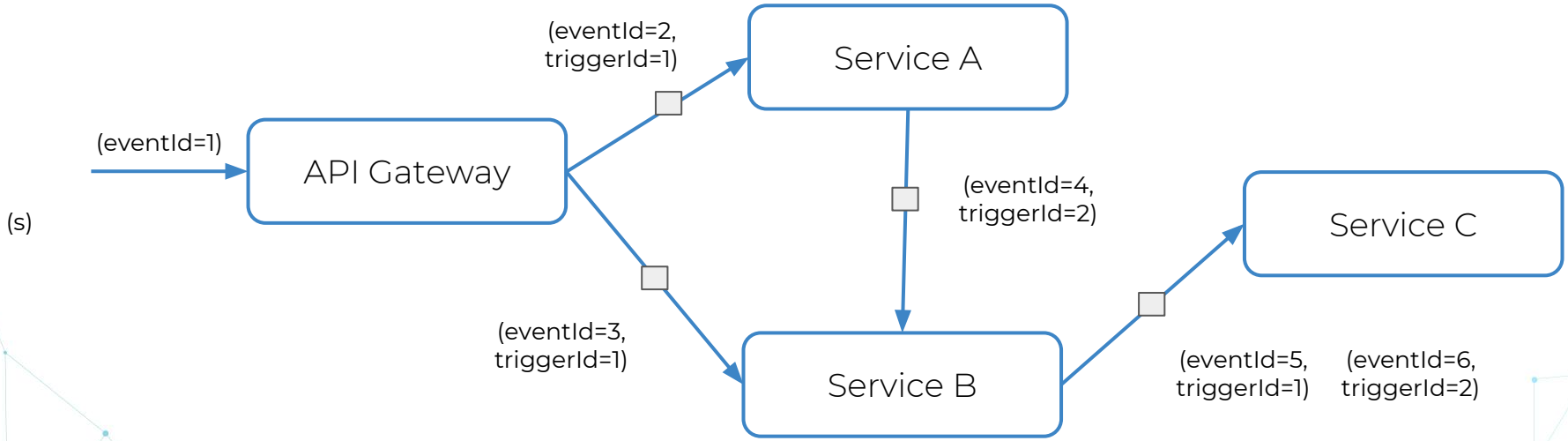
**Service Isolation**
The service to be debugged has to be able to run with all call to outside services replaced with replayed events

# Probing



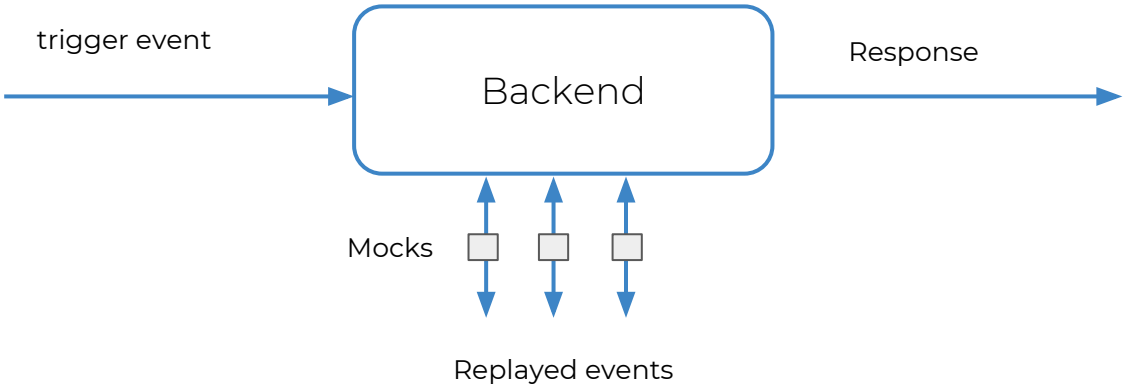

Kata.ai

# Tracing




Kata.ai

# Service Isolation




Kata.ai

# Event Replay @ Kata

Proof-of-concept (Not production yet)

Taylor made for in-house framework (merapi framework)

Goal: framework level implementation, no change of code

Event tracing using Async Hook

Custom-made event store

Kata.ai

# Summary

Preventive measure like TDD, Code Review and Code Coverage enforcement can reduce bugs

Developers are humans, bugs can sneak into production

Investing in instrumentation and debugging tools can save time and increase productivity