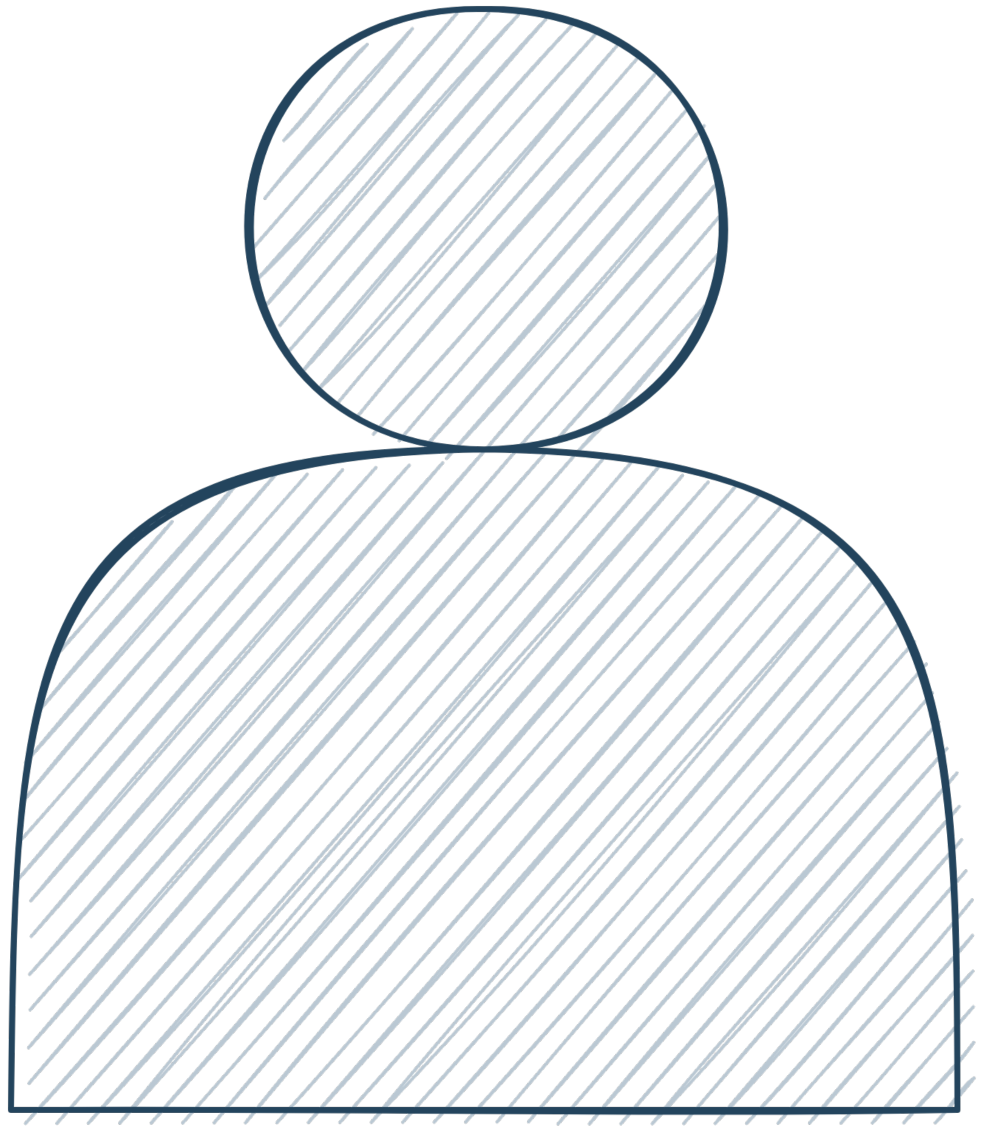
Developer Centered Security

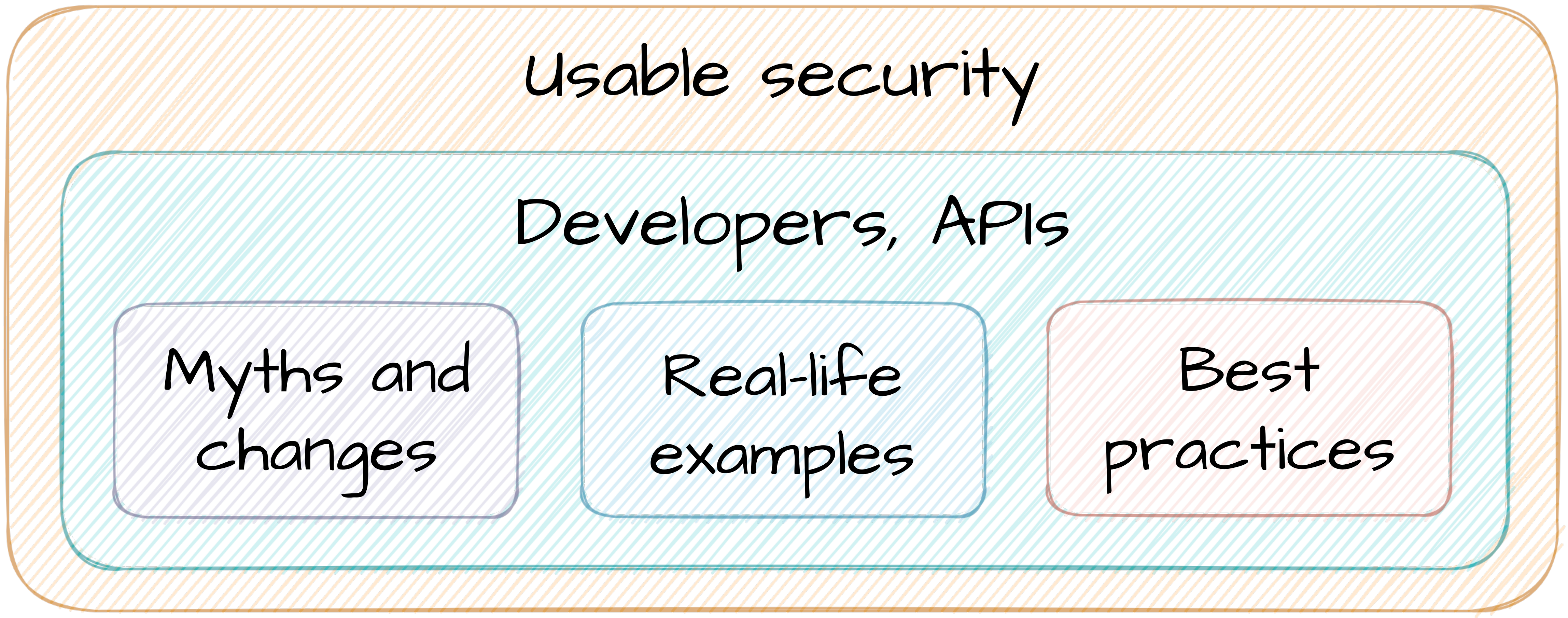
Olgierd Pieczul

Security Architecture

Oracle Cloud Infrastructure



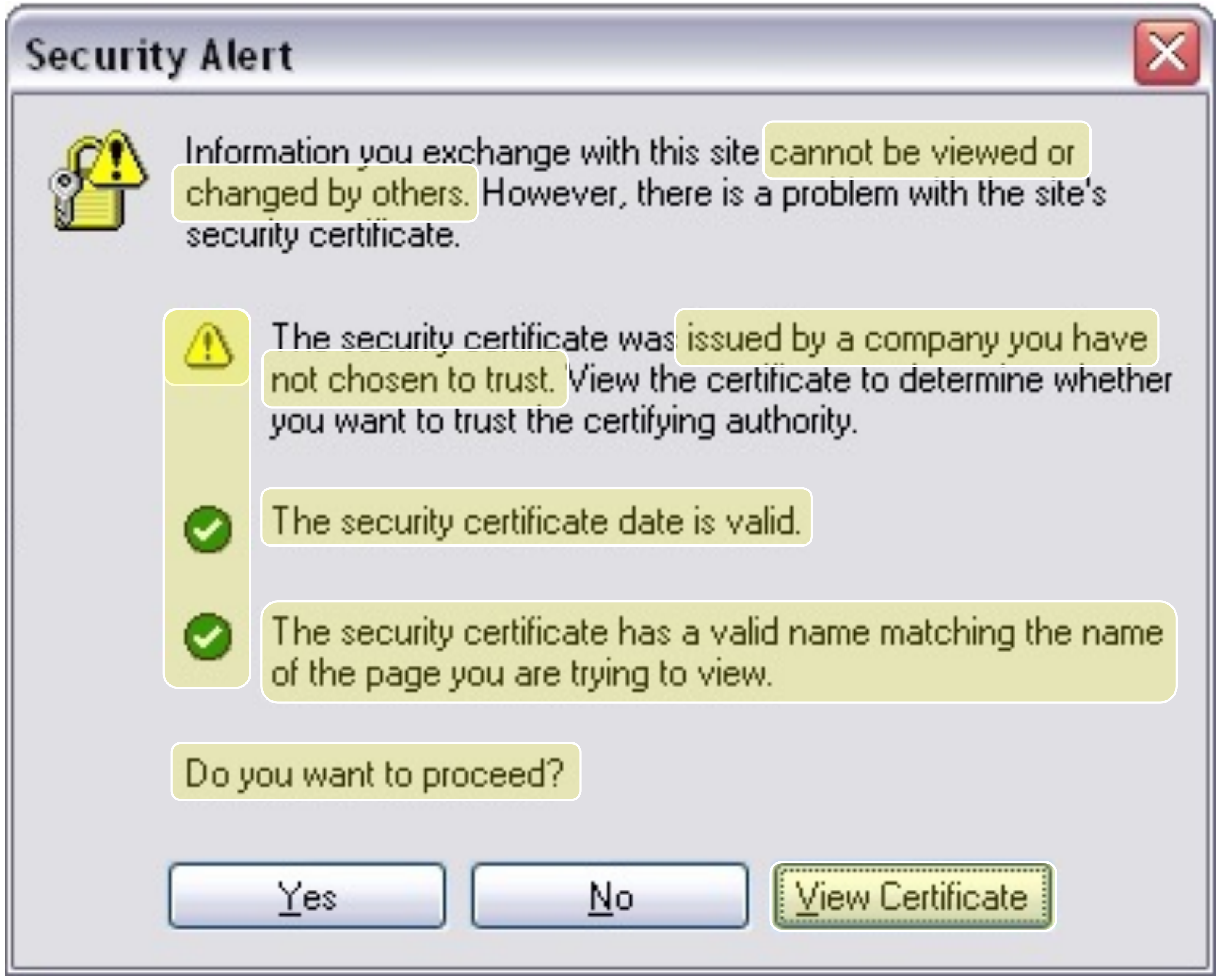
Agenda, layers



Usable Security Primer

The Good, the Bad and the Ugly

Ugly


Security Alert

Information you exchange with this site cannot be viewed or changed by others. However, there is a problem with the site's security certificate.

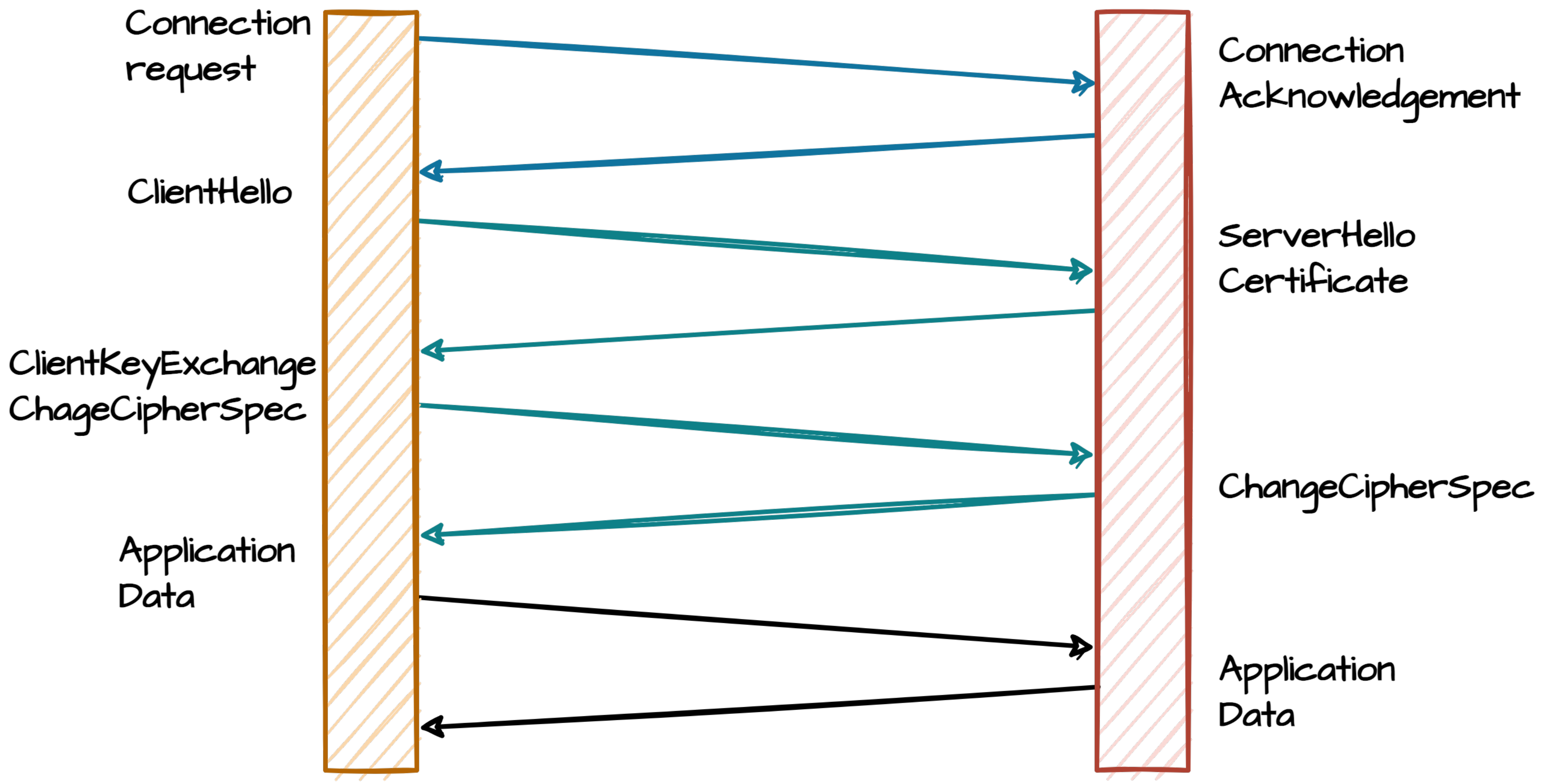
The security certificate was issued by a company you have not chosen to trust. View the certificate to determine whether you want to trust the certifying authority.

The security certificate date is valid.

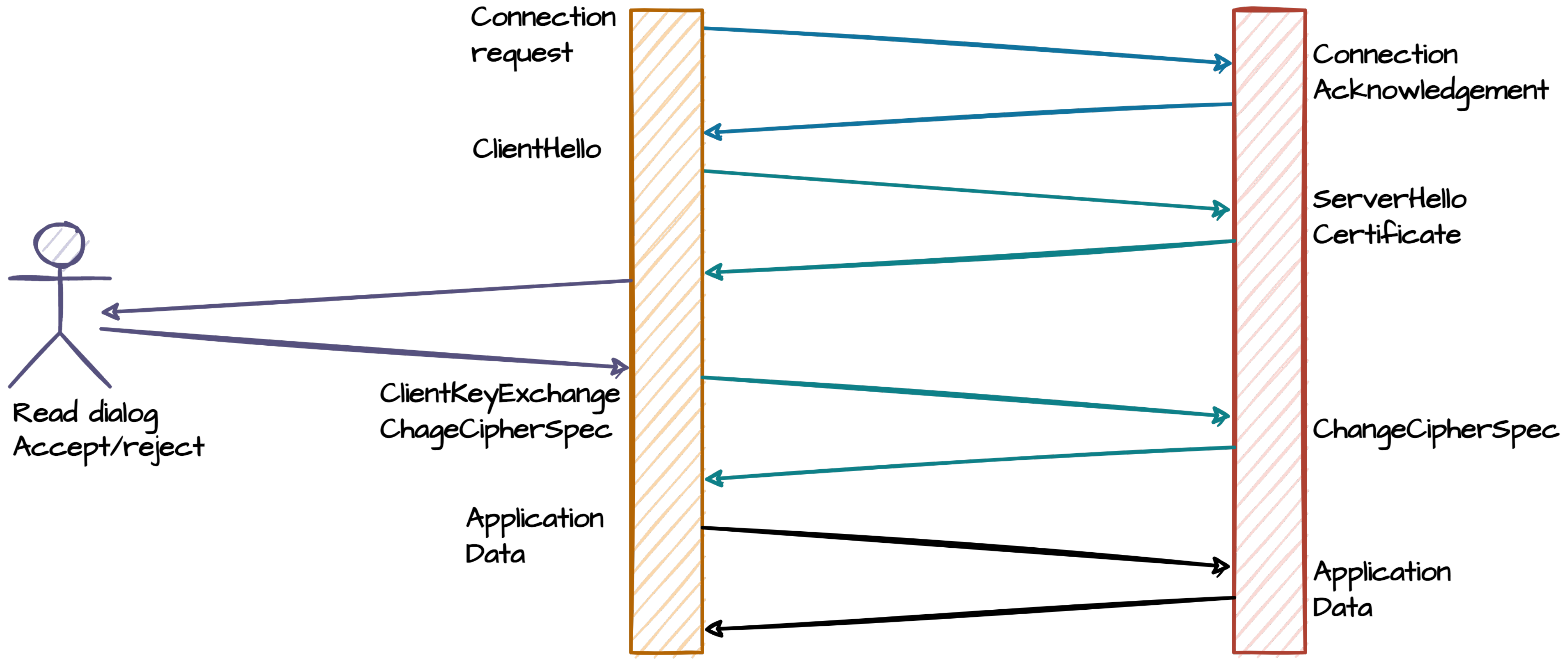
The security certificate has a valid name matching the name of the page you are trying to view.

Do you want to proceed?

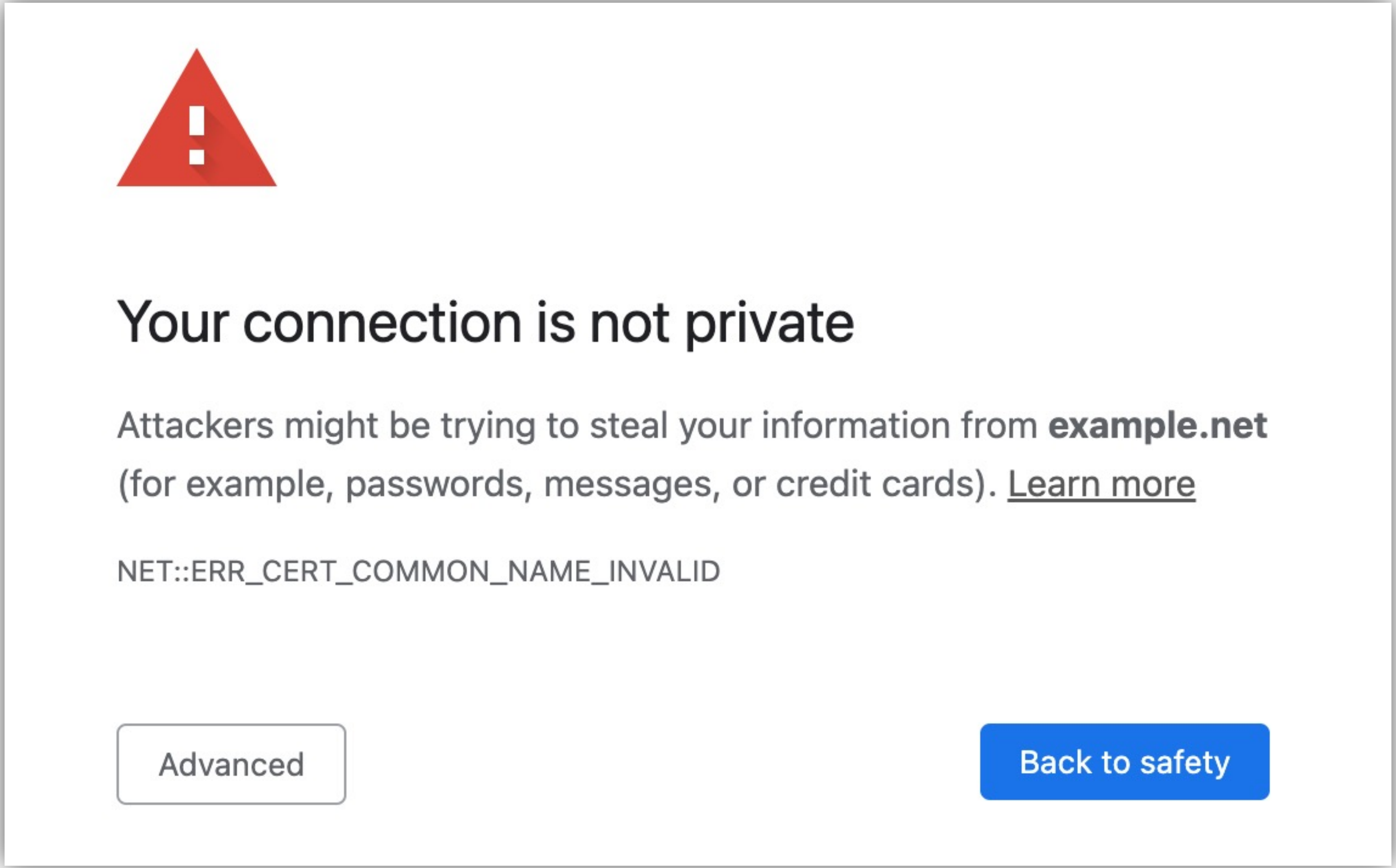
System in theory



System in practice



Good



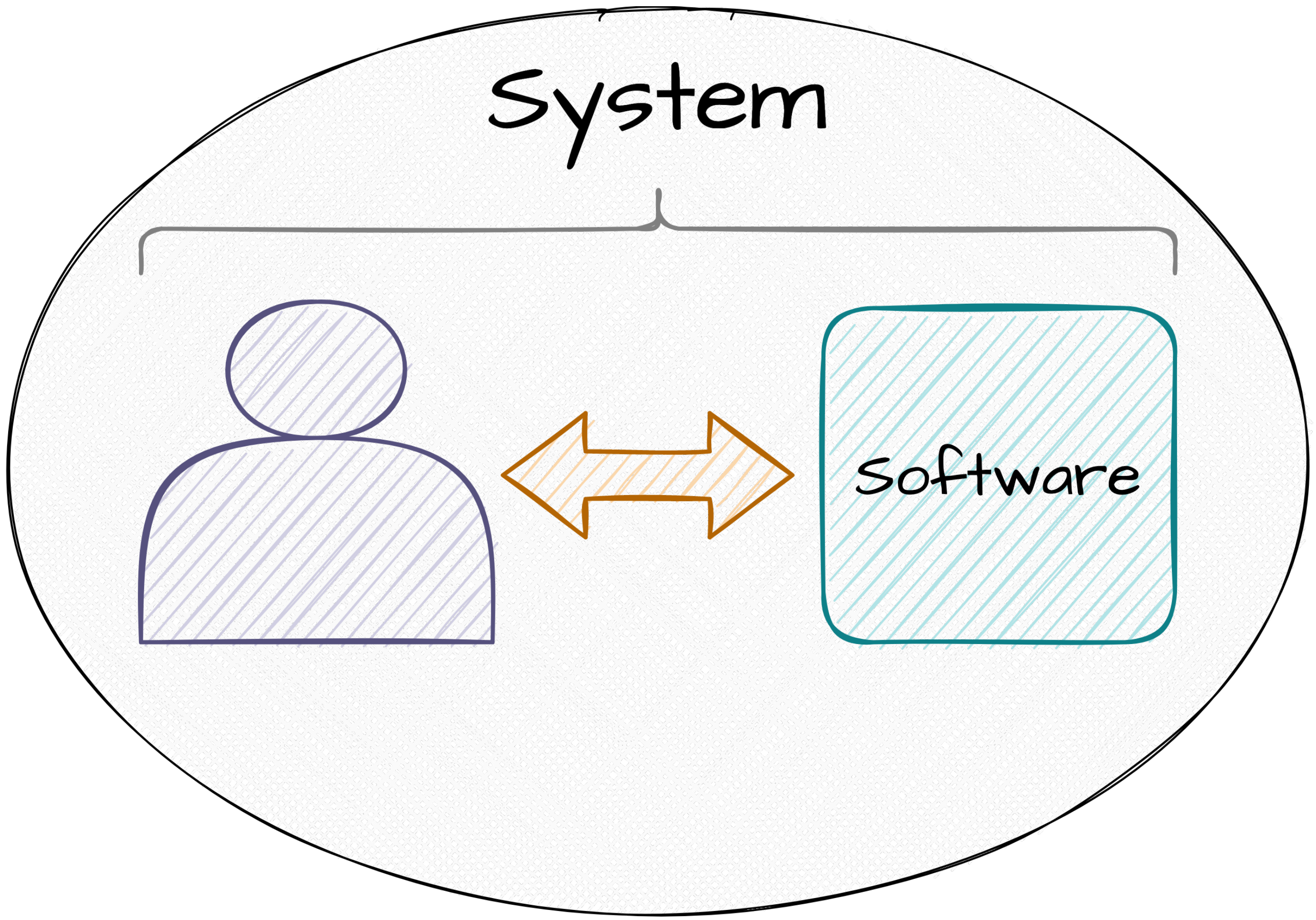
Your connection is not private

Attackers might be trying to steal your information from **example.net** (for example, passwords, messages, or credit cards). Learn more

NET::ERR_CERT_COMMON_NAME_INVALID

Summary

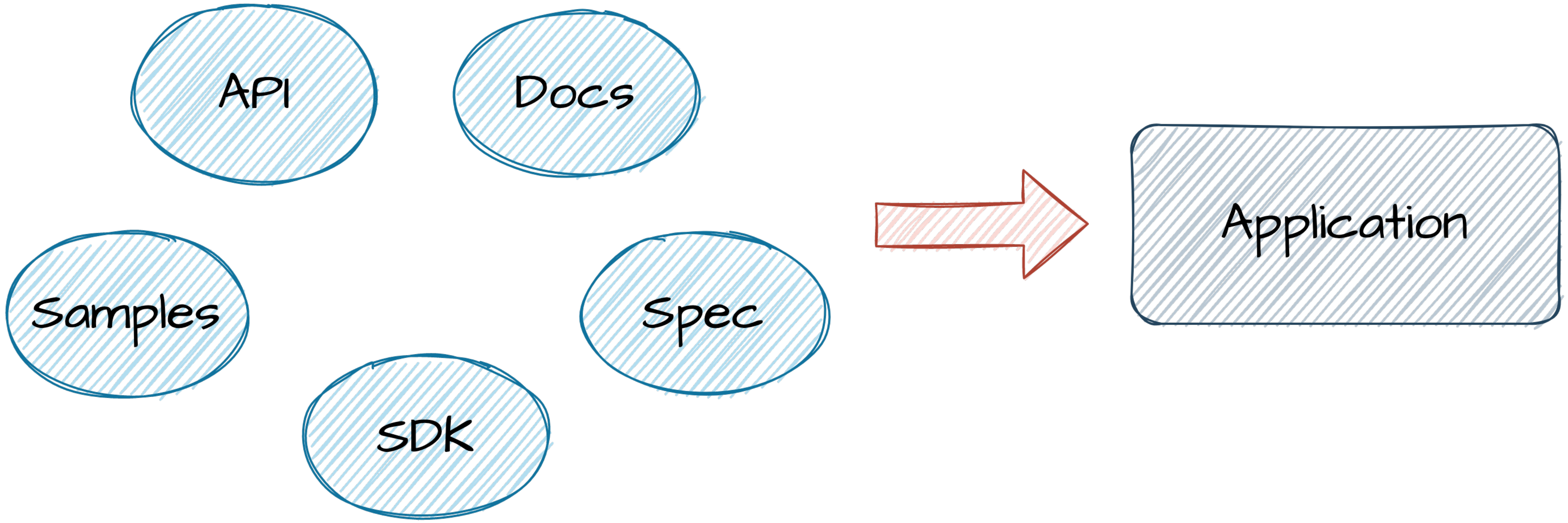
- Users are part of the system
- System design
  - Secure defaults
  - Not involving if possible
  - Clear and transparent information



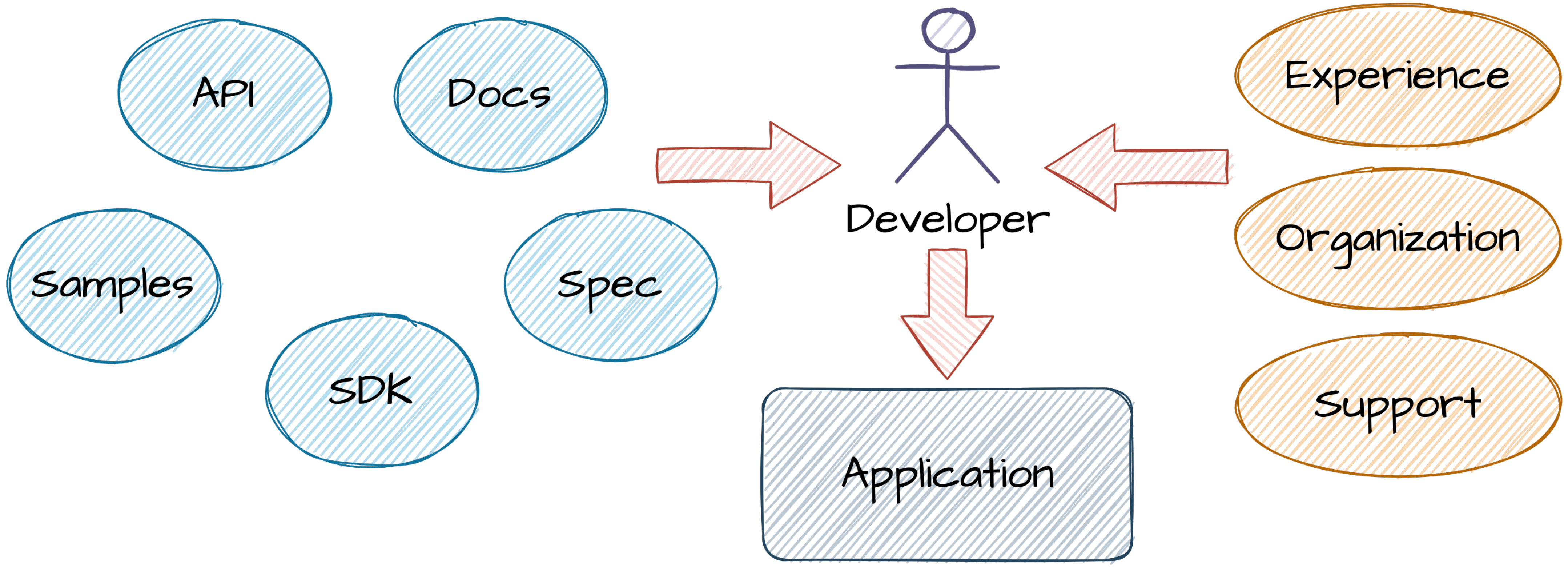
Developer Centered Security

Developers are humans too

Development stack



Development stack

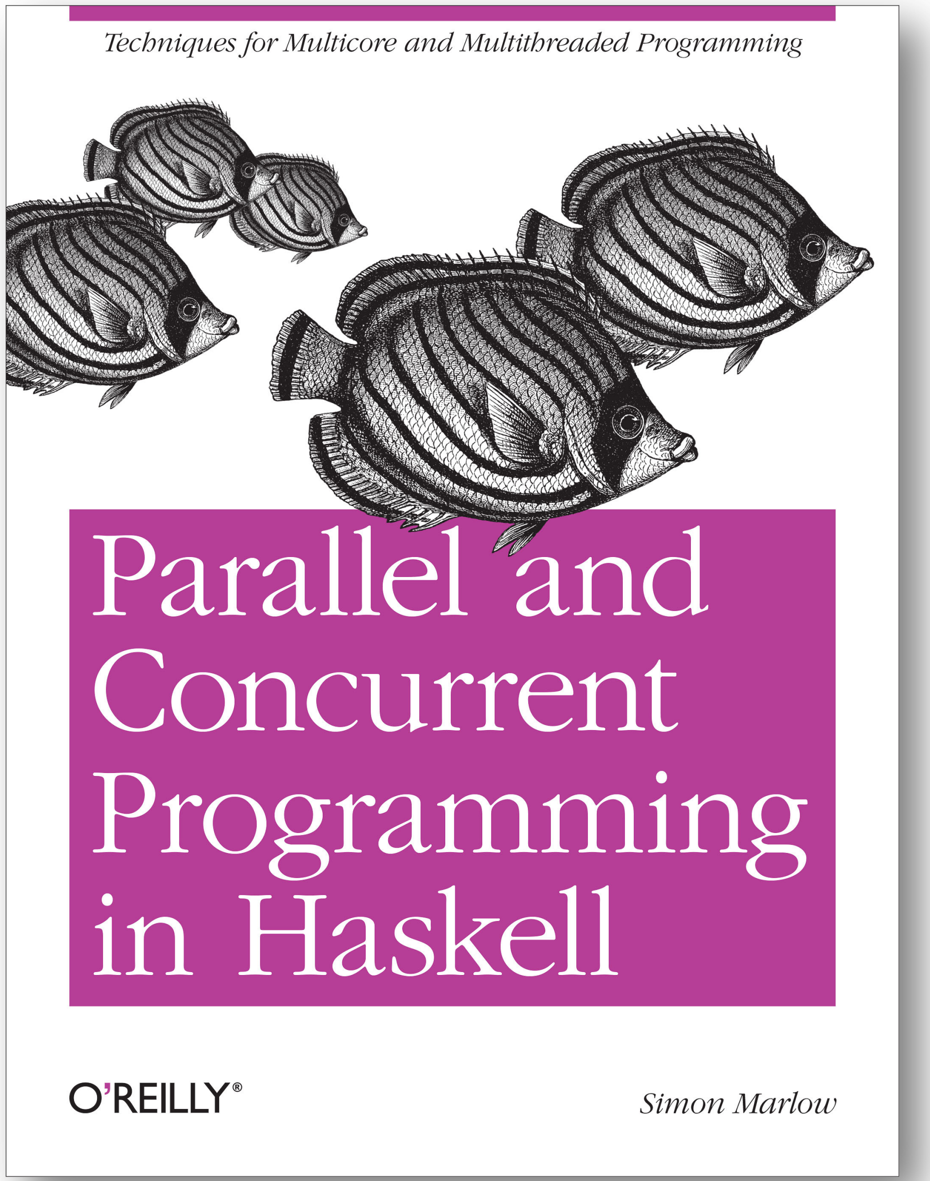


Who are developers?

And what do they do?

Accidental Developers

Source: O’Reilly Media



Techniques for Multicore and Multithreaded Programming

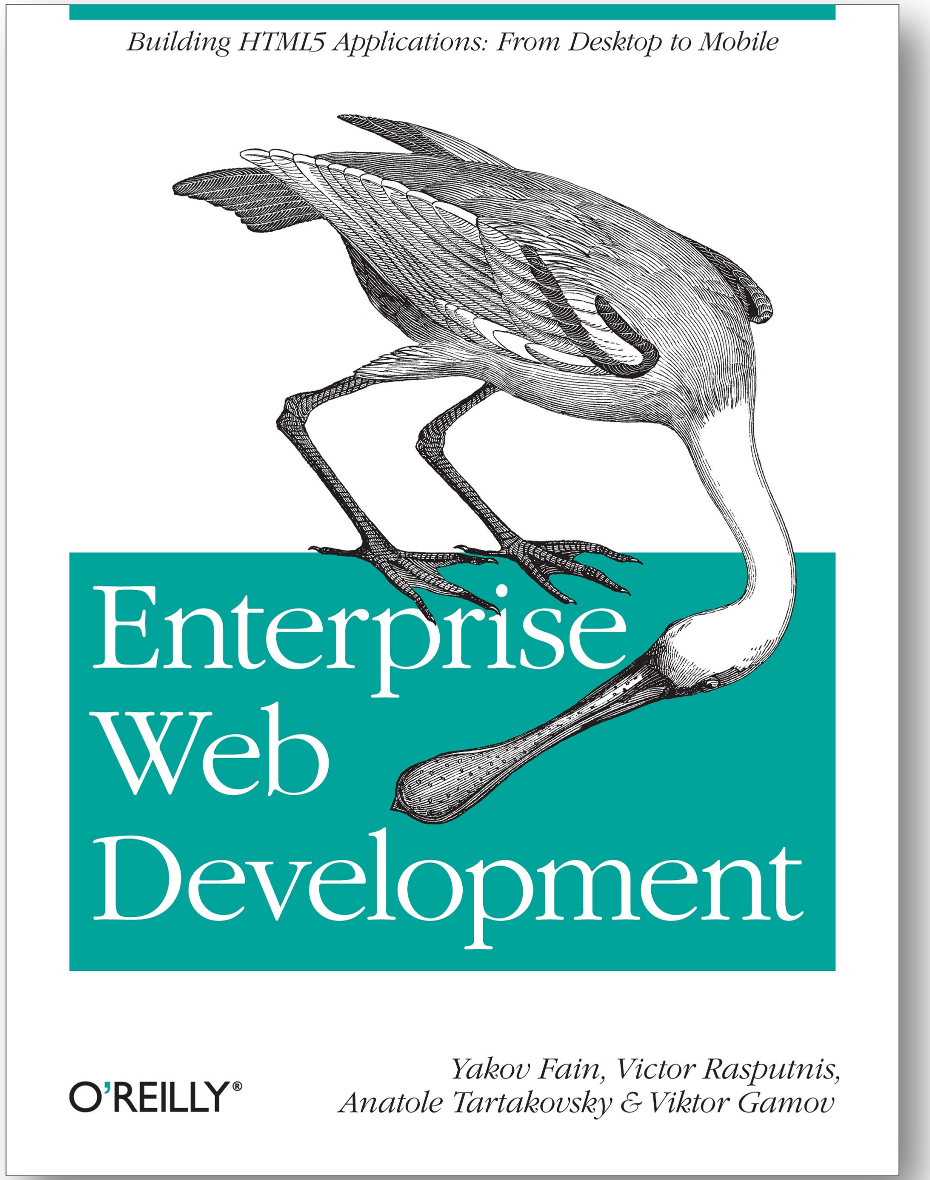
Parallel and Concurrent Programming in Haskell

O’REILLY®

Simon Marlow

Accidental Developers

Source: O’Reilly Media



Building HTML5 Applications: From Desktop to Mobile

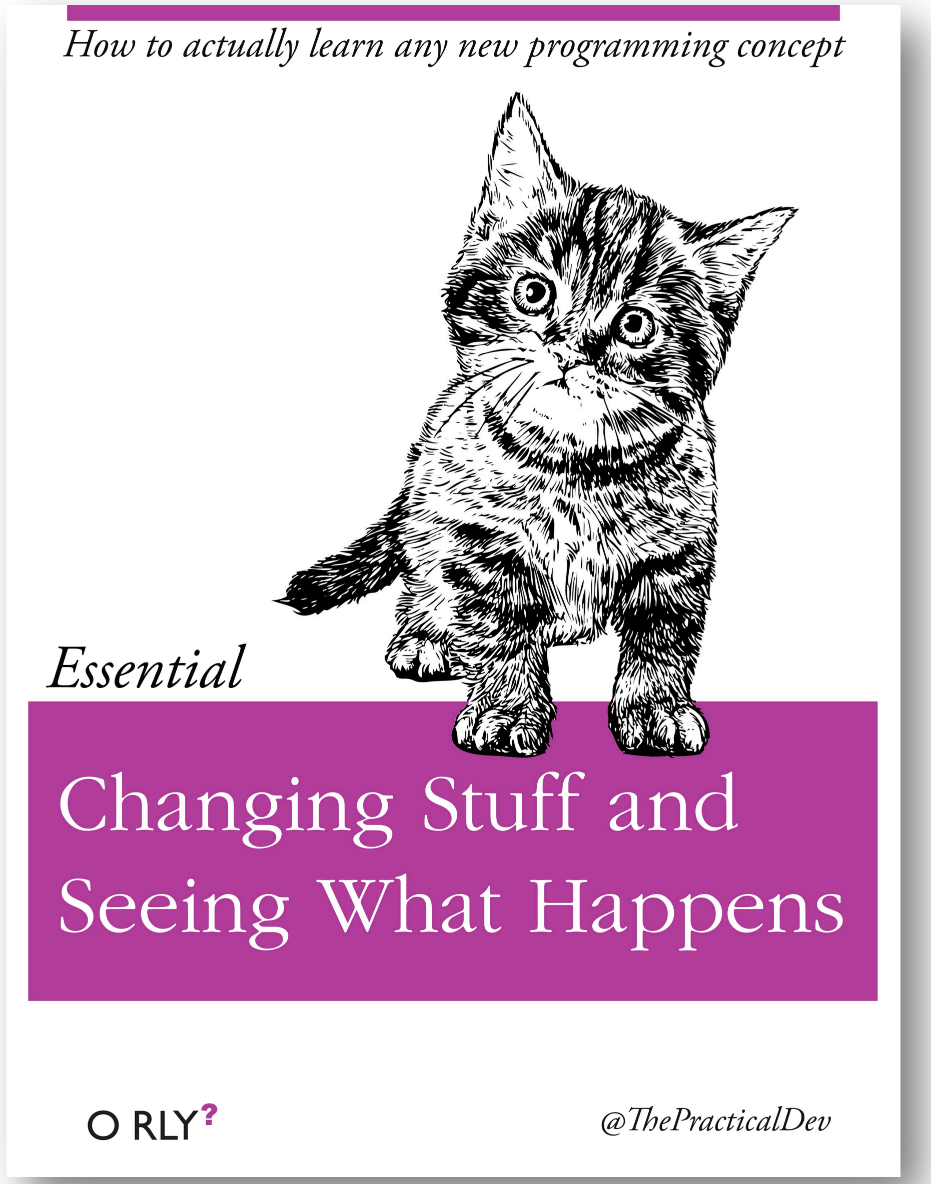
Enterprise Web Development

Yakov Fain, Victor Rasputnis, Anatole Tartakovsky & Viktor Gamov

O’REILLY®

Accidental Developers

Source: The Practical Dev


How to actually learn any new programming concept

Essential

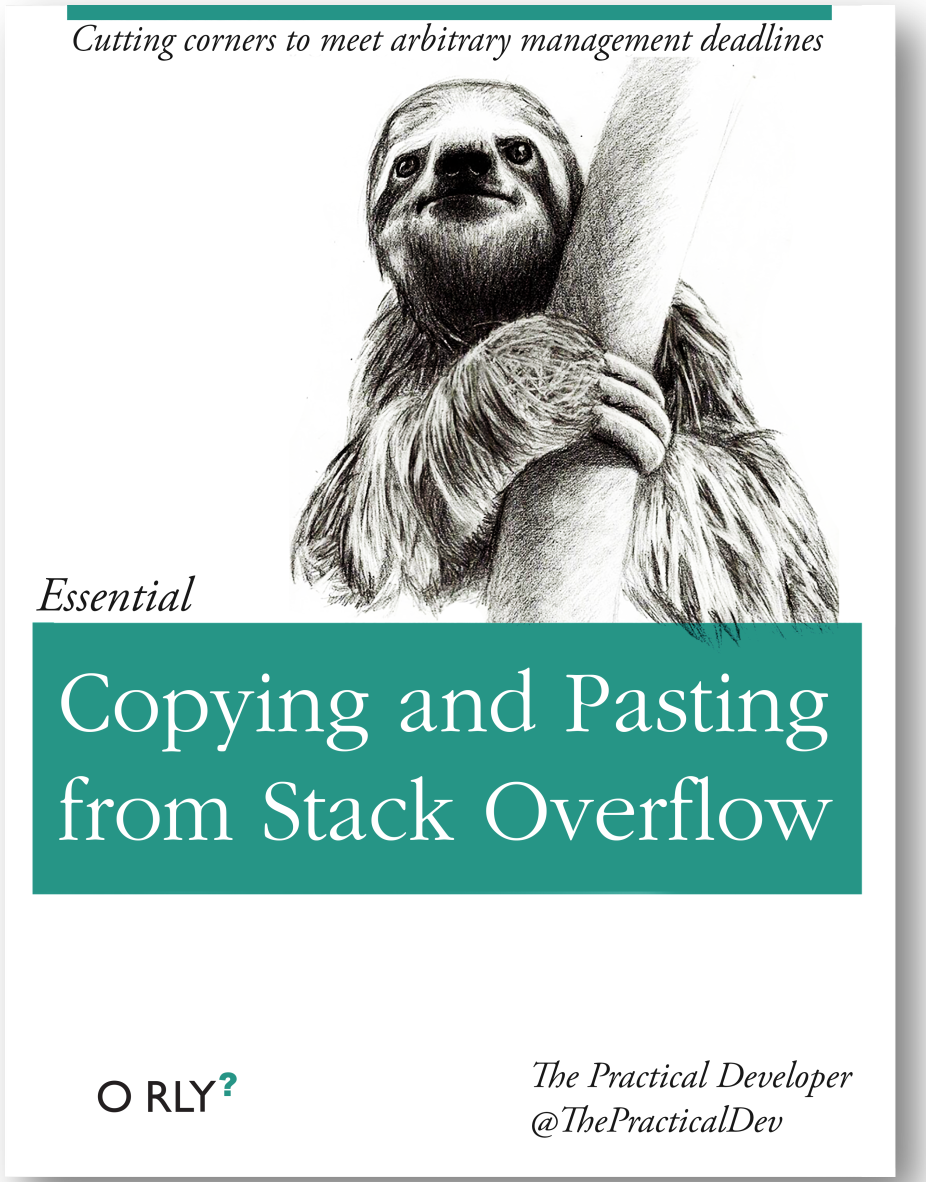
Changing Stuff and Seeing What Happens

O RLY?

@ThePracticalDev

Accidental Developers

Source: The Practical Dev



Cutting corners to meet arbitrary management deadlines

Essential

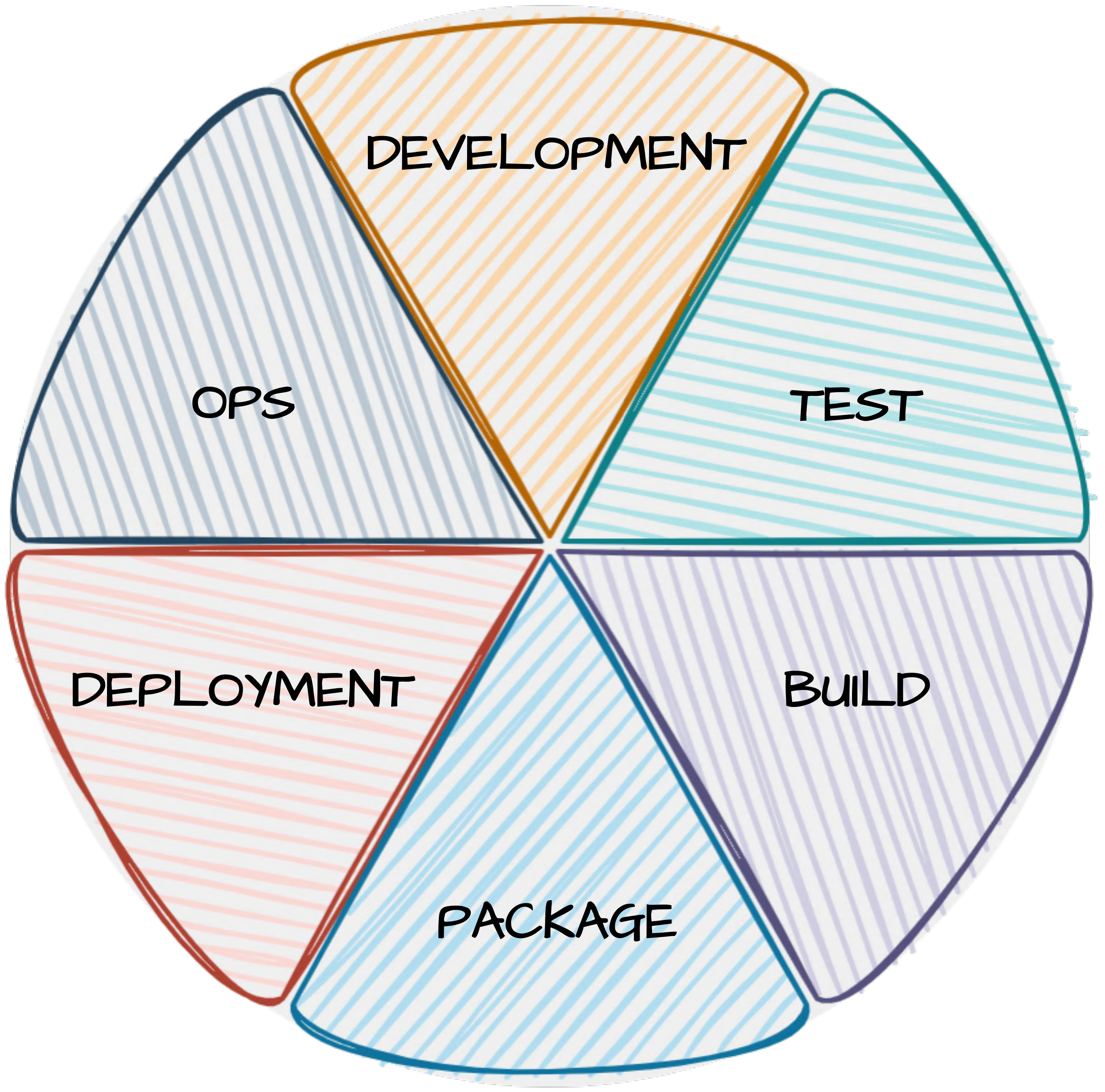
Copying and Pasting from Stack Overflow

O RLY?

The Practical Developer
@ThePracticalDev

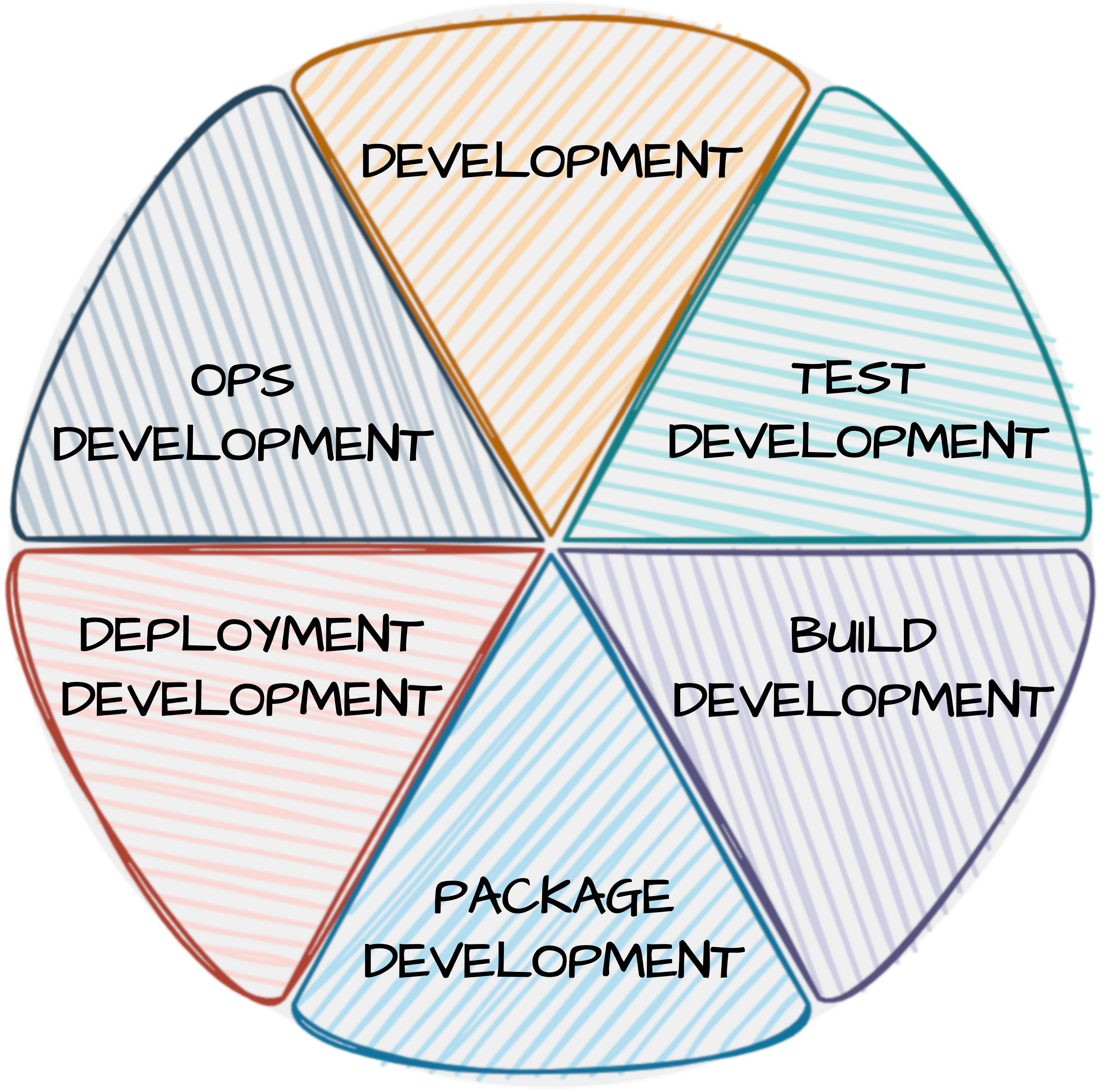
Everyday developers

- Coding as a specialized skill
- Separate from the rest of the lifecycle



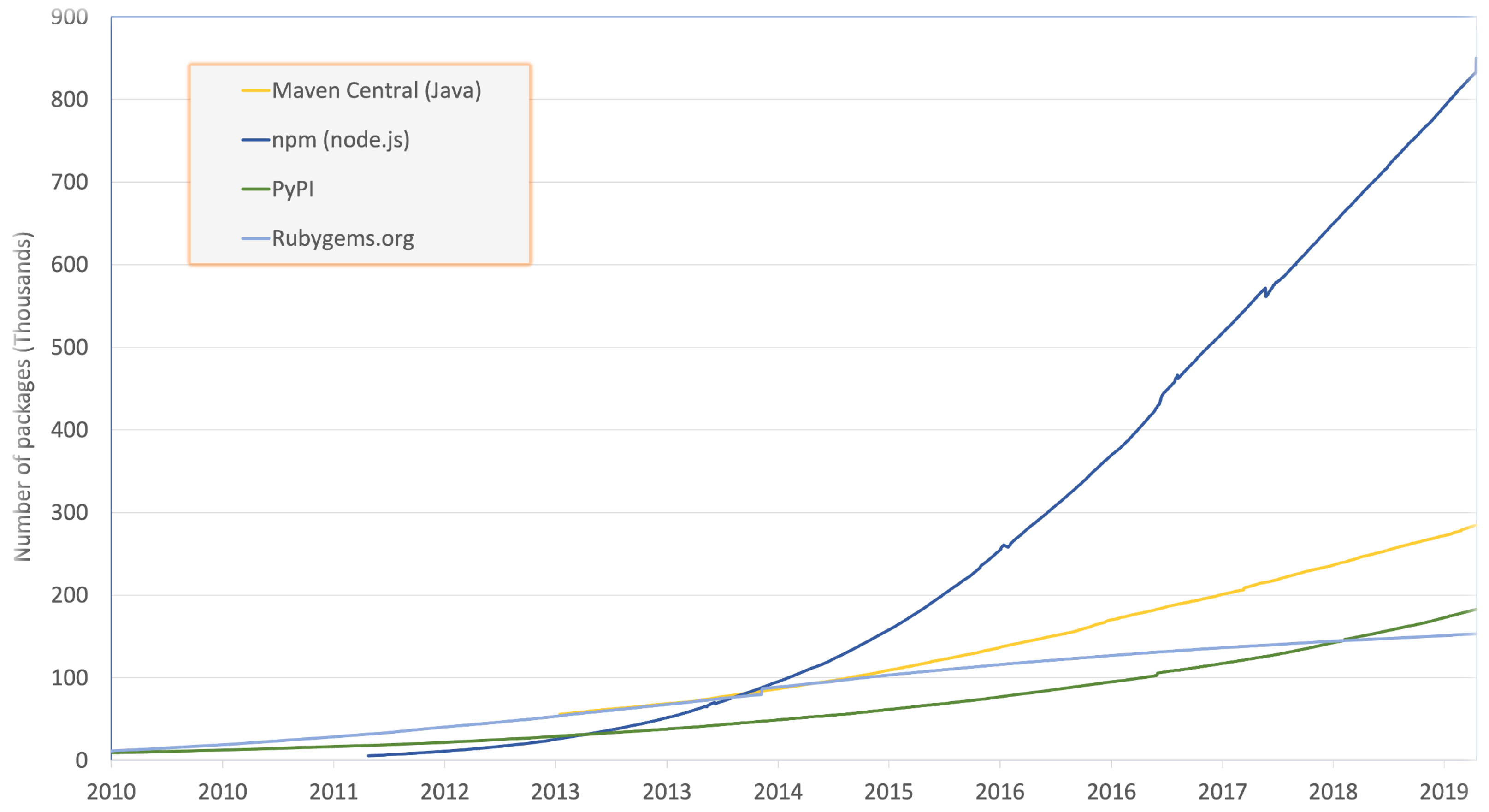
Everyday developers

It is all about coding now



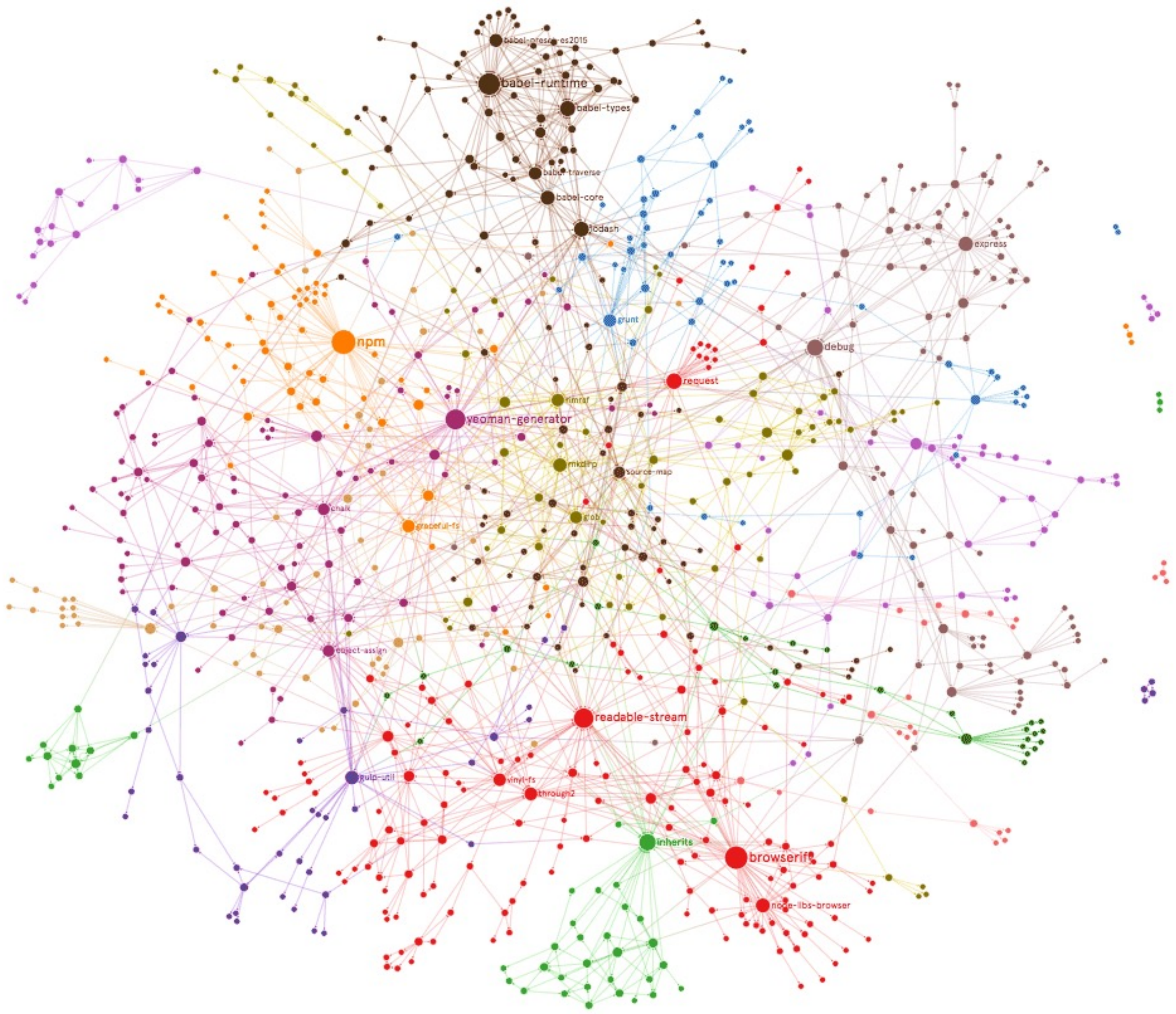
Third-party components

- Component integration
- Effortless, fast, cheap
- Amount of:
  - documentation
  - settings
  - options
  - defaults



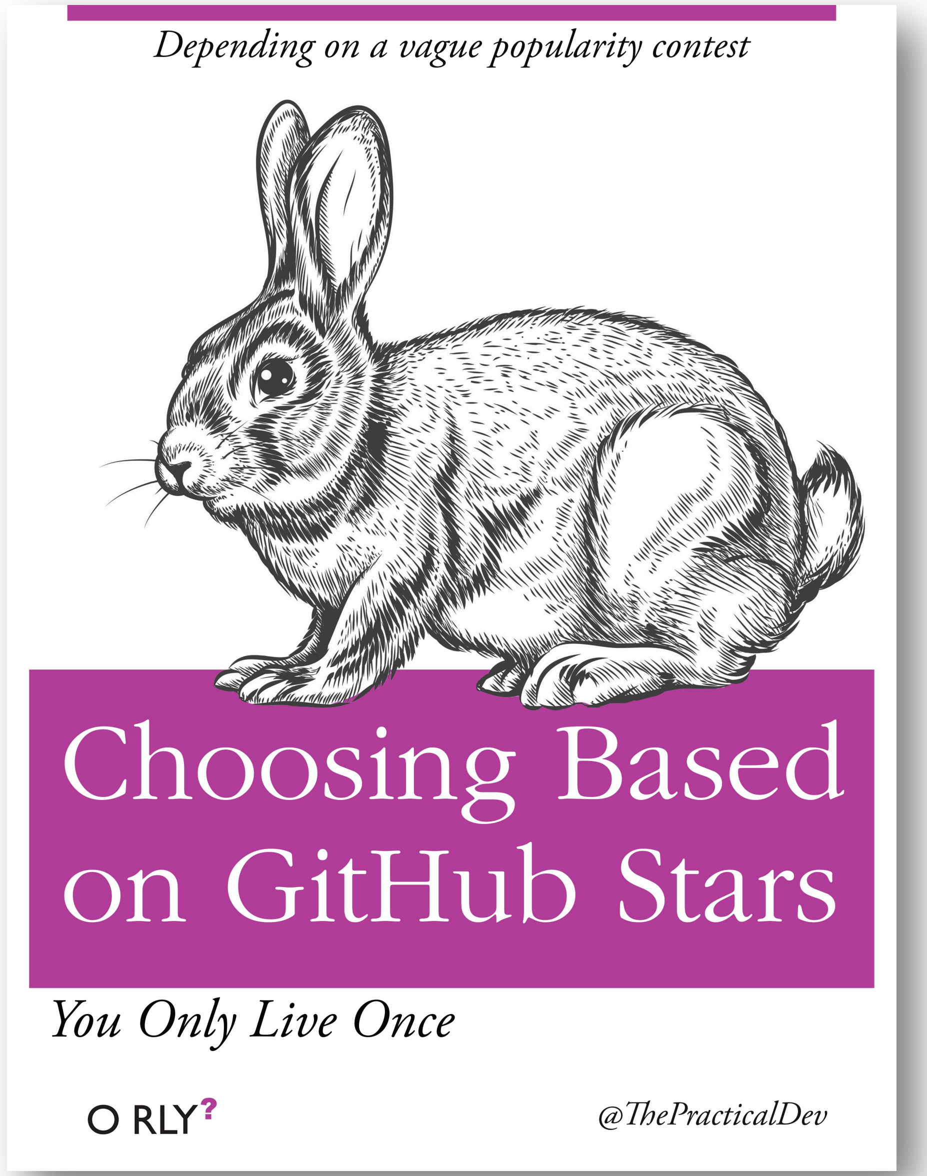
Third-party components

Direct and indirect usage



Third-party components

Source: The Practical Dev

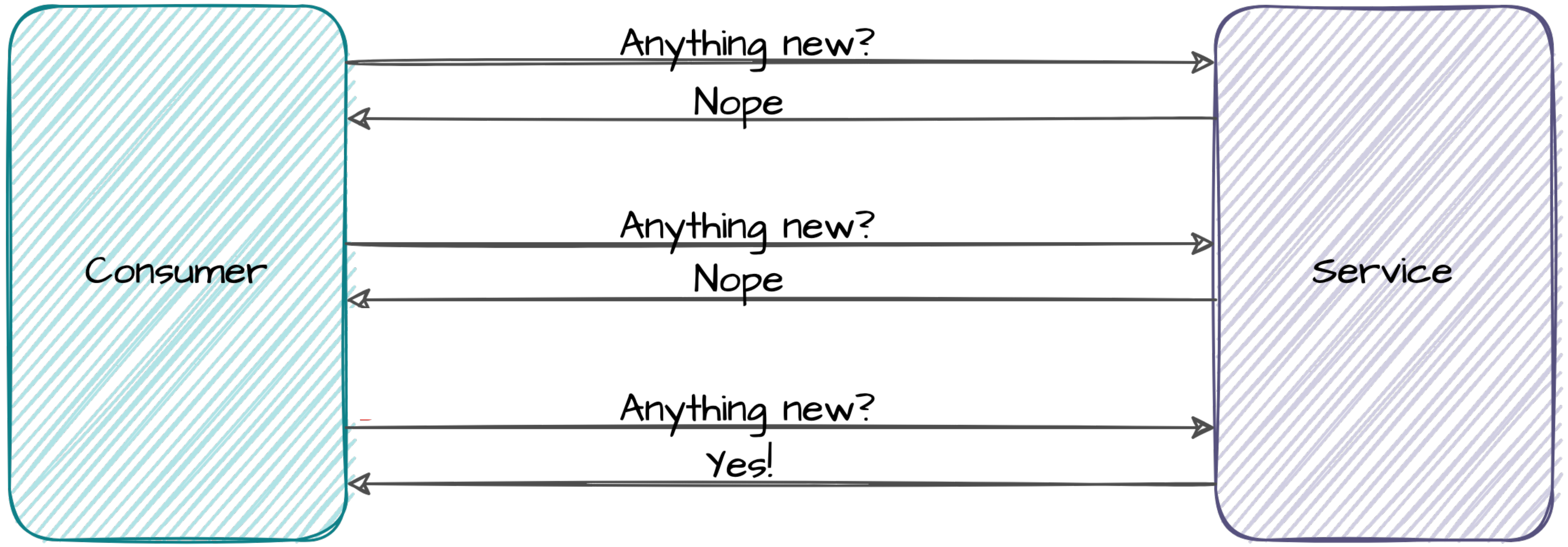


Depending on a vague popularity contest
Choosing Based on GitHub Stars
You Only Live Once
O RLY?
@ThePracticalDev

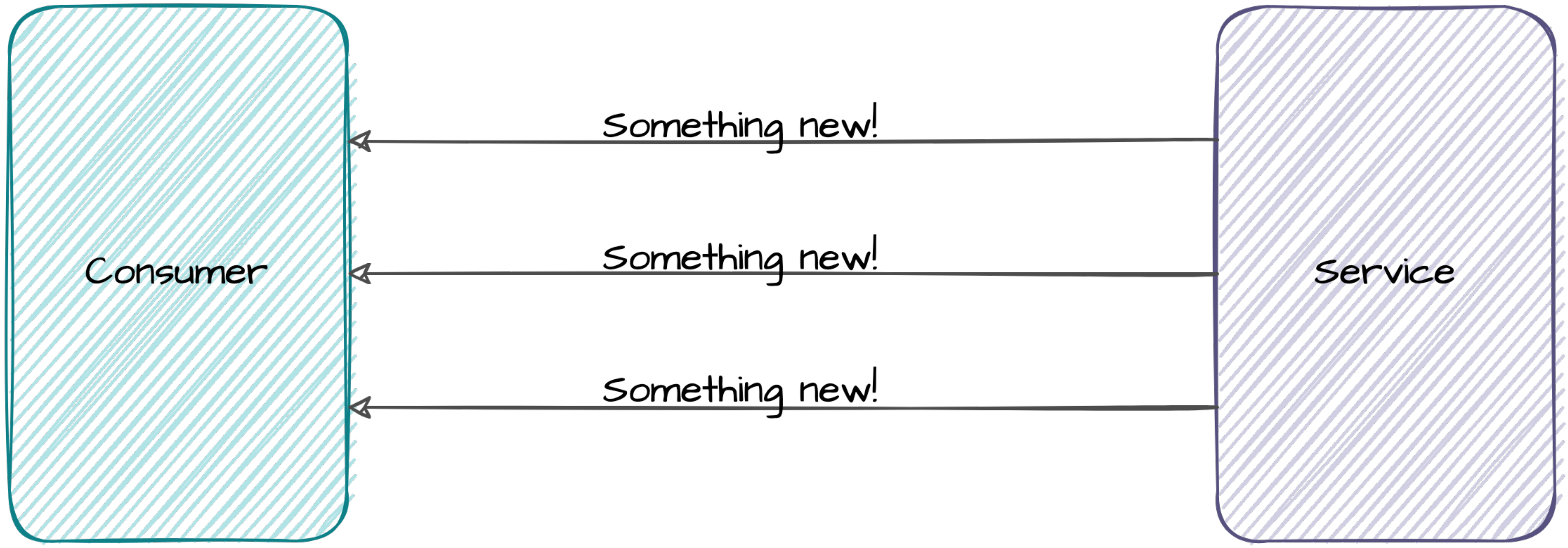
Case study

Webhooks

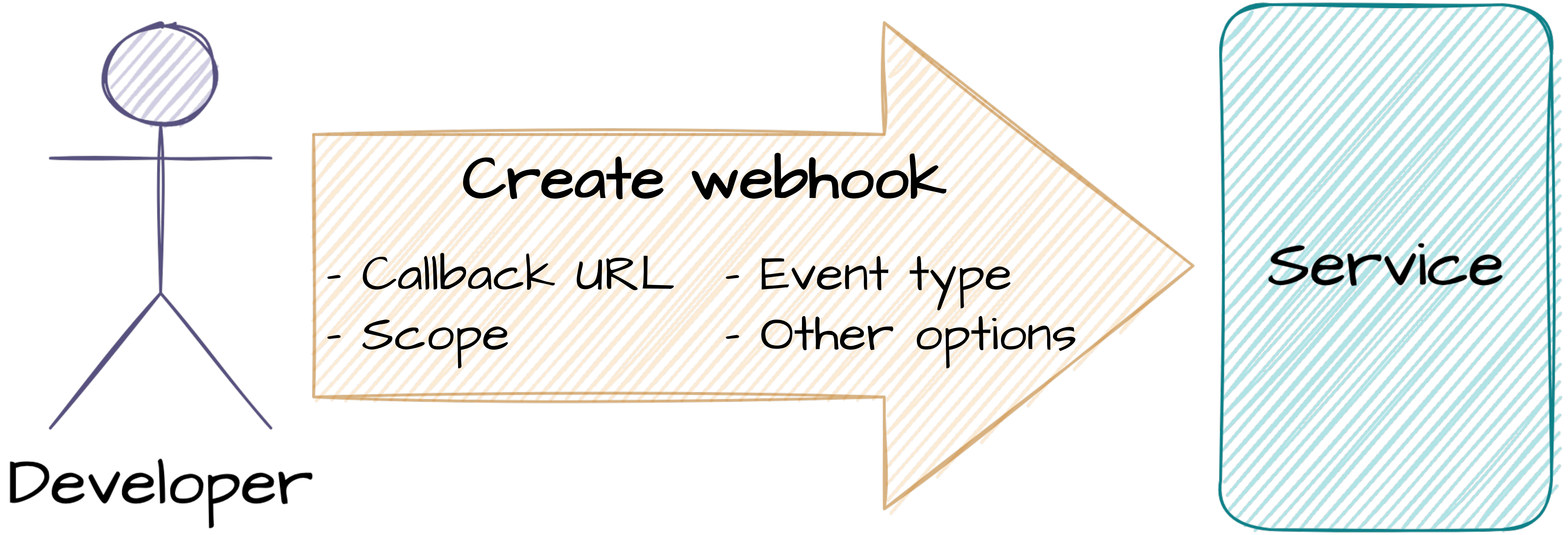
Before webhooks



With webhooks

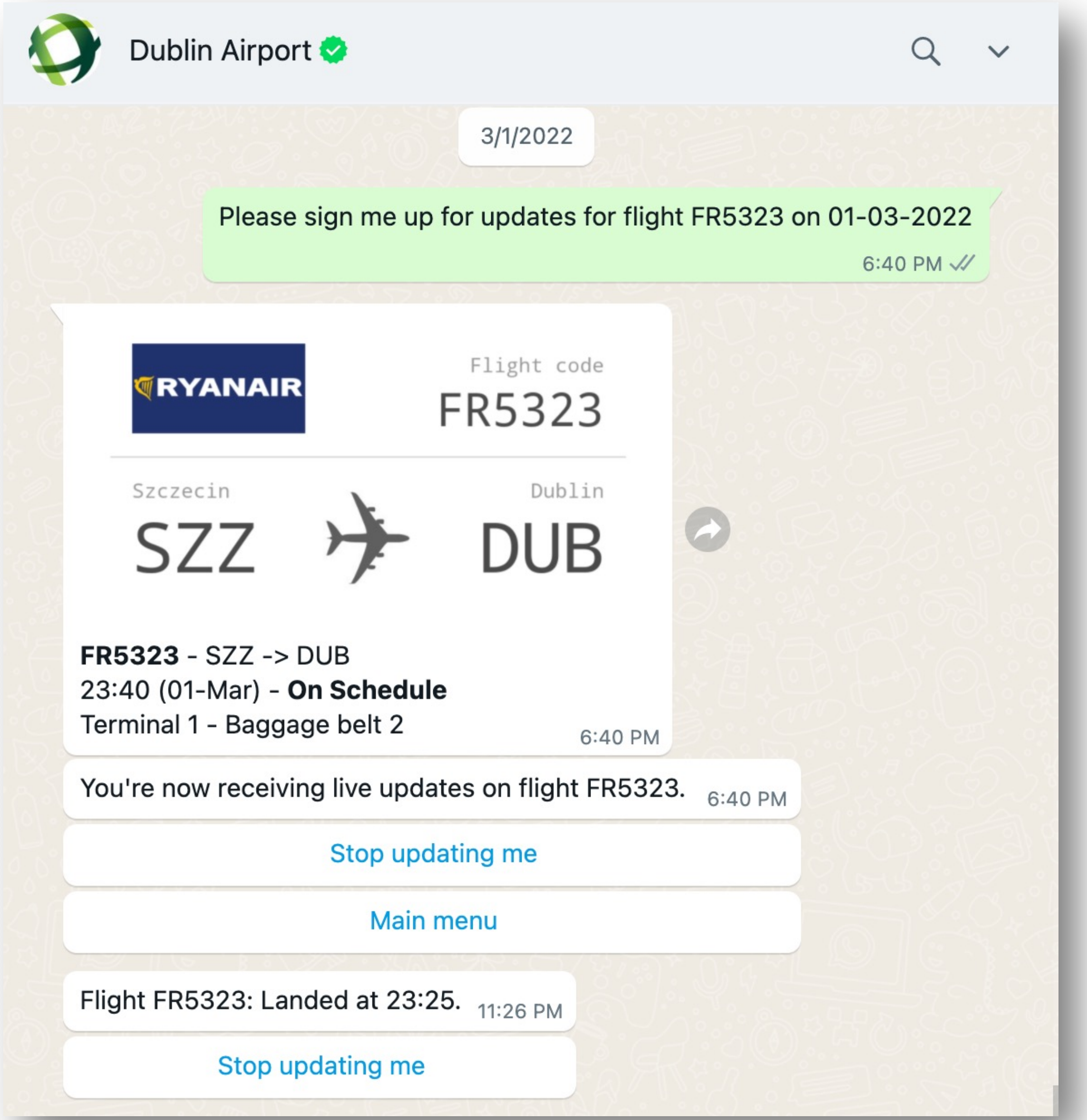


Setup


Create webhook
- Callback URL
- Scope
- Event type
- Other options
Service
Developer

Webhooks in action

```
{
  "contacts": [{
    "profile": {
      "name": "Olgierd Pieczul"
    },
    "wa_id": "16315551234"
  }],
  "messages": [{
    "timestamp": "1518694235",
    "text": {
      "body": "Please sign me up ..."
    },
    "type": "text"
  }]
}
```


Dublin Airport

3/1/2022

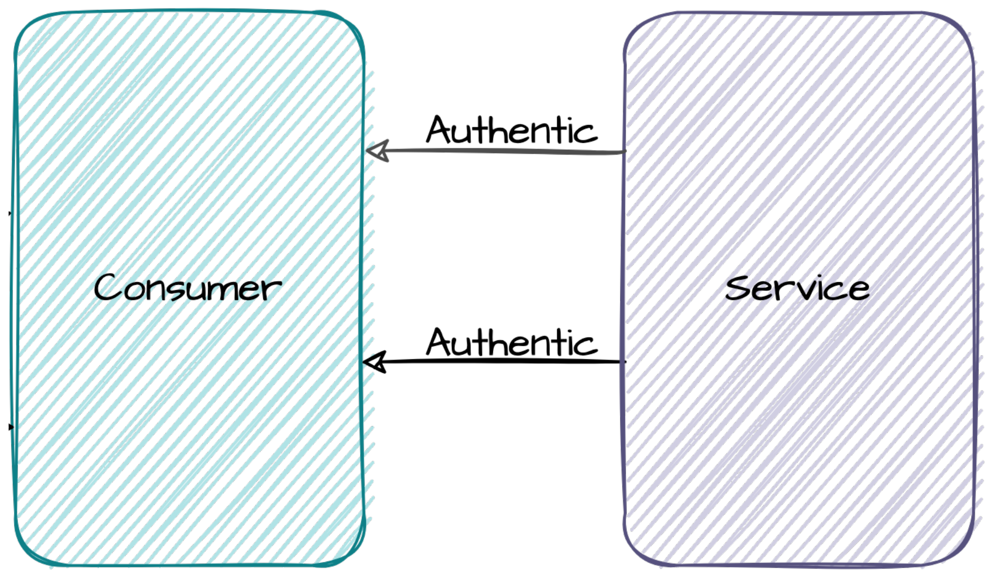
Please sign me up for updates for flight FR5323 on 01-03-2022
6:40 PM

FR5323 - SZZ -> DUB
23:40 (01-Mar) - **On Schedule**
Terminal 1 - Baggage belt 2
6:40 PM

You're now receiving live updates on flight FR5323. 6:40 PM

Flight FR5323: Landed at 23:25. 11:26 PM

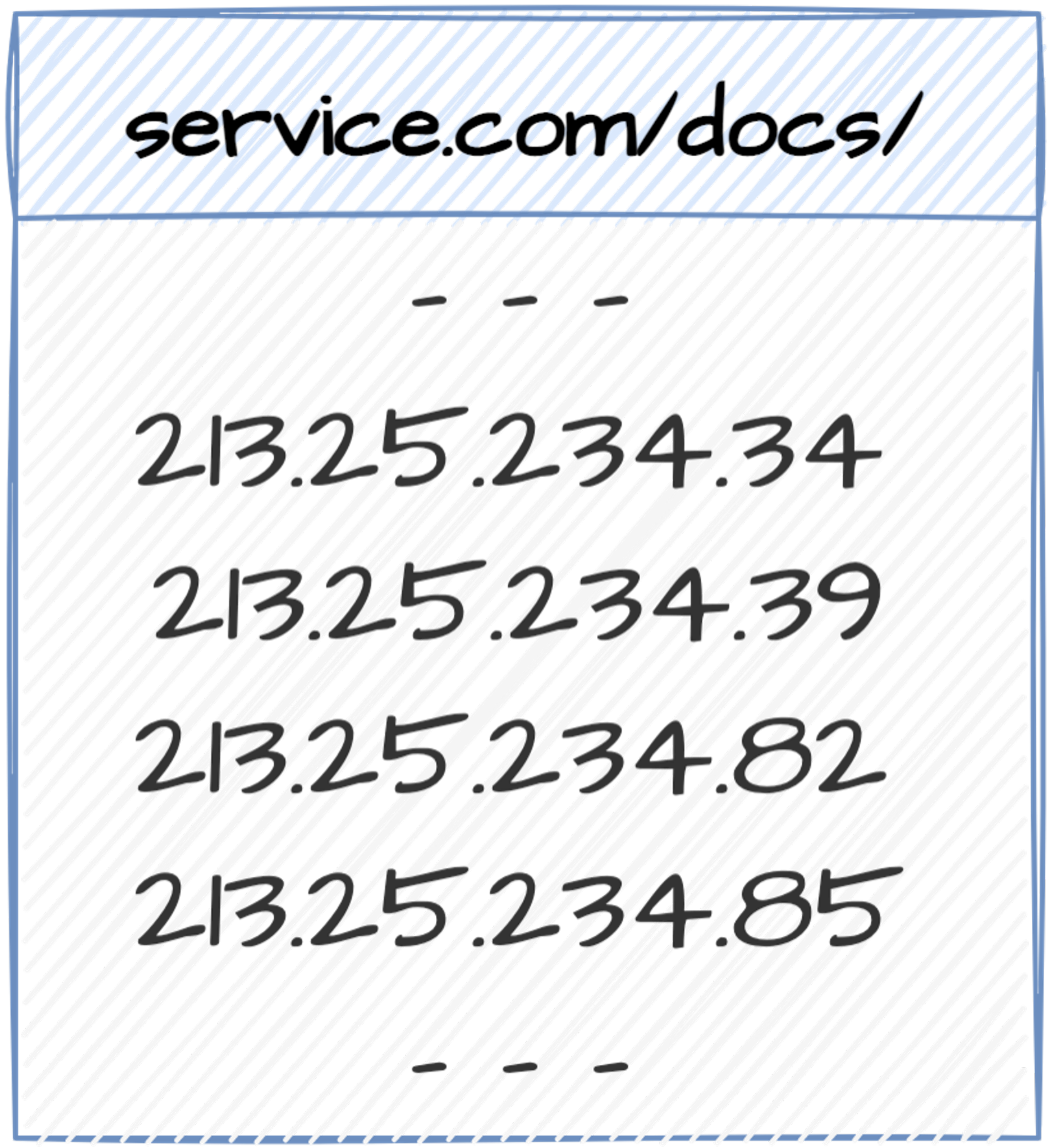
Why webhooks?



- Service *defines* controls
- Consumer *implements*
- No direct service impact
- Study of 10 services
  - API
  - Documentation
  - Code samples

Source address

- What this is for?
- Does it change?
- Poor mechanism
  - Layers, proxies
  - Shared infrastructure
  - Multiple services



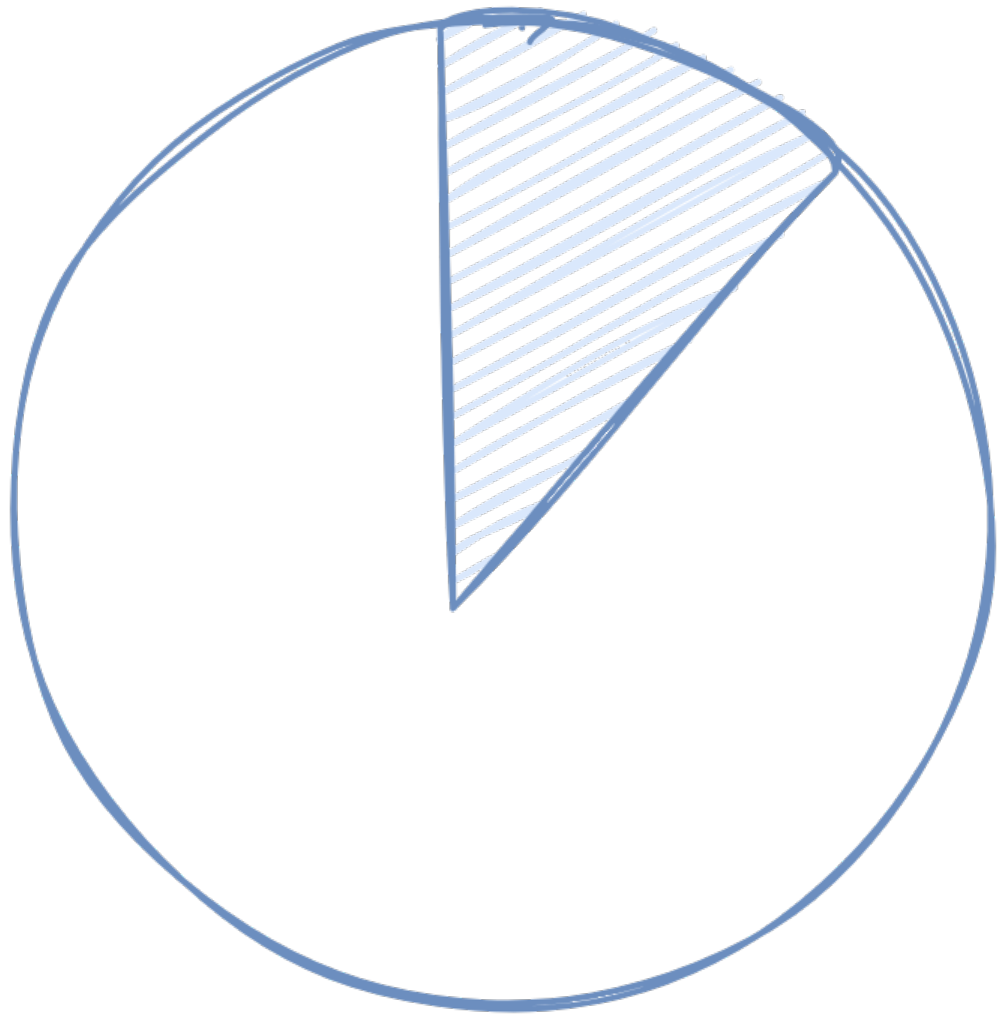
Source address with DNS

- How often?
- Integrated or manual?
- Plaintext

```
$ dig a +short service.com
213.25.234.34
213.25.234.39
213.25.234.82
213.25.234.85
```

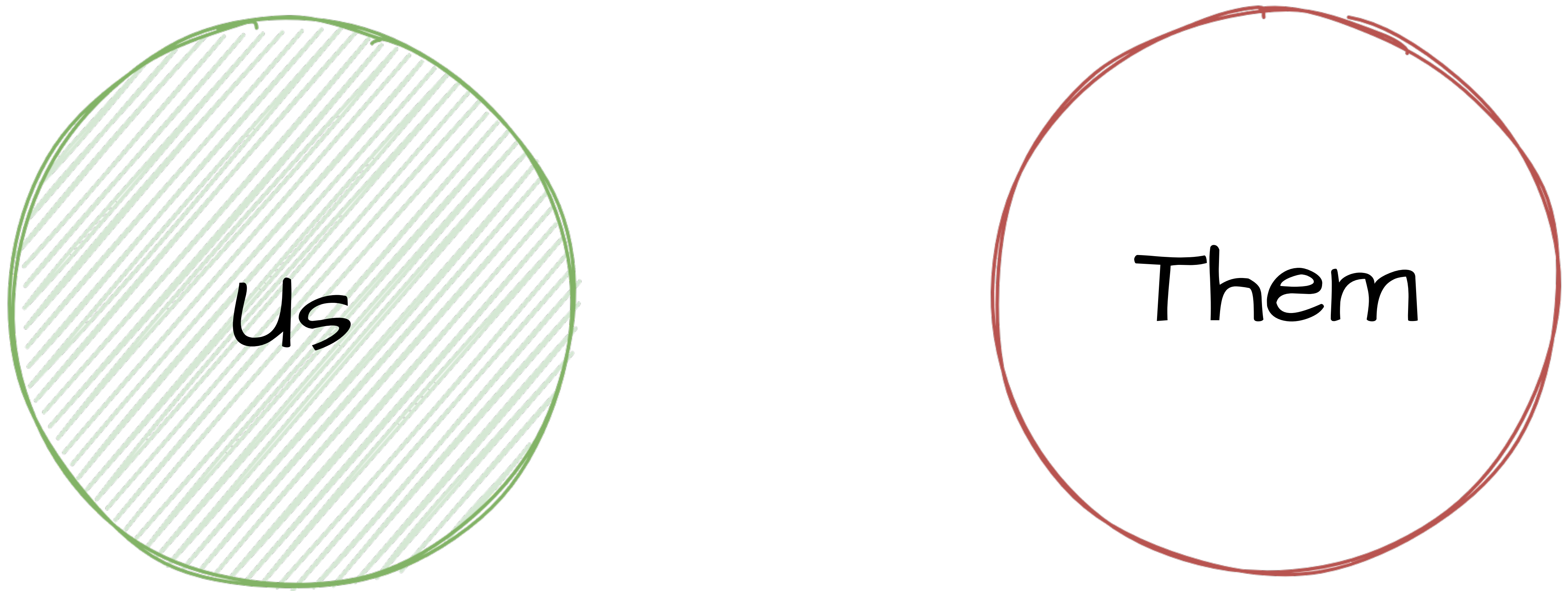
TLS

- 20% recommend using TLS
- Examples in documentation often use ‘http’ URLs
- Sample code uses plaintext endpoints



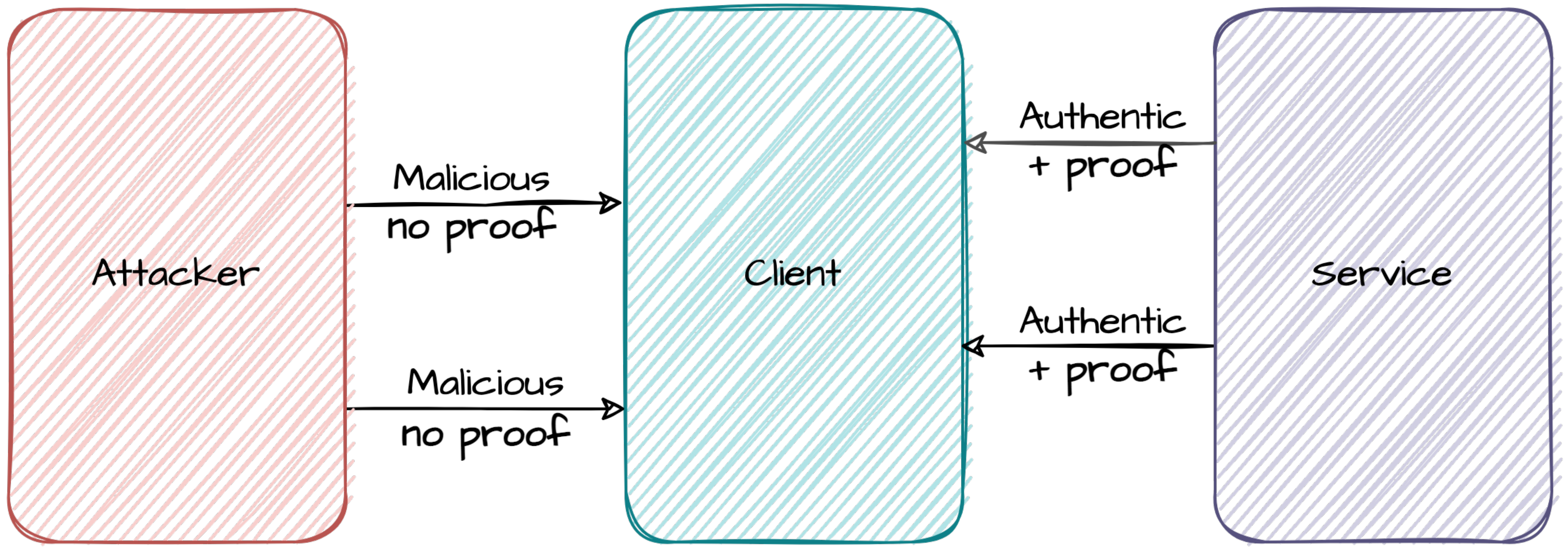
TLS

- 100% enforce TLS for incoming calls to APIs
- 0% enforce TLS for webhooks



Authentication

- 5 out of 10 authenticate outgoing calls
  - 1 does "secret URL", 1 BasicAuth, 3 HMAC



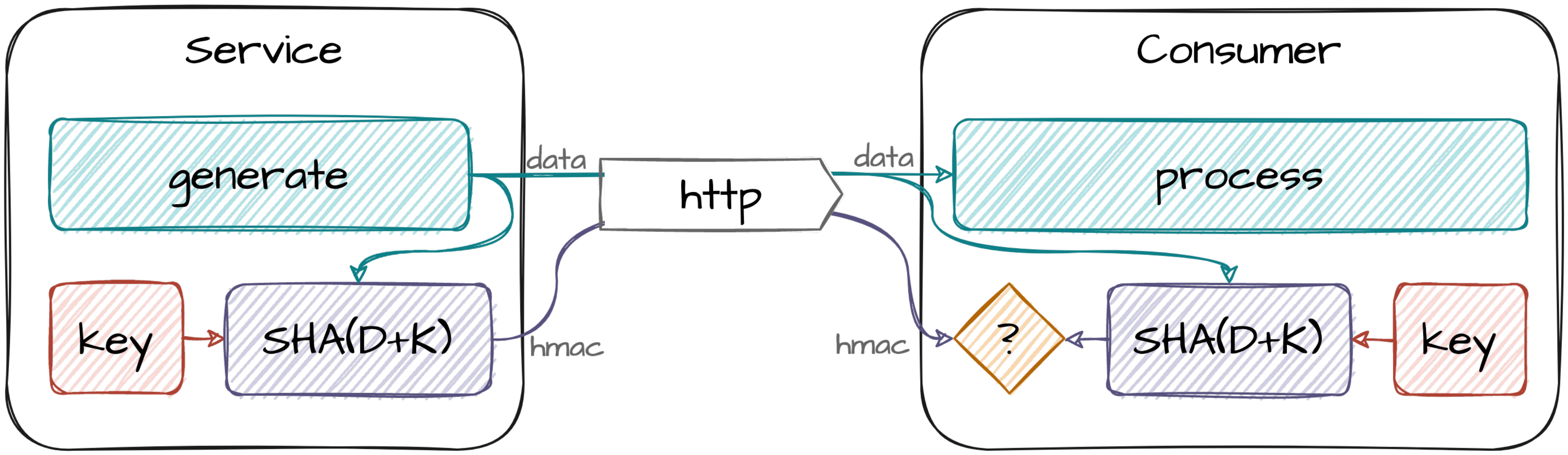
Authentication: Secret URL

"As a best practice, provide a callback URL that's not guessable and make sure you can easily change it."

— Service documentation

https://consumer.net/callback/foobar99

Authentication: HMAC



HMAC confusion

"For added security, webhooks sent to applications are signed so they can be verified as originating from Service and unaltered in transit."

"Service can optionally sign the webhook events it sends to your endpoints by including a signature in each event's header."

HMAC: keys, docs and samples

Key generation

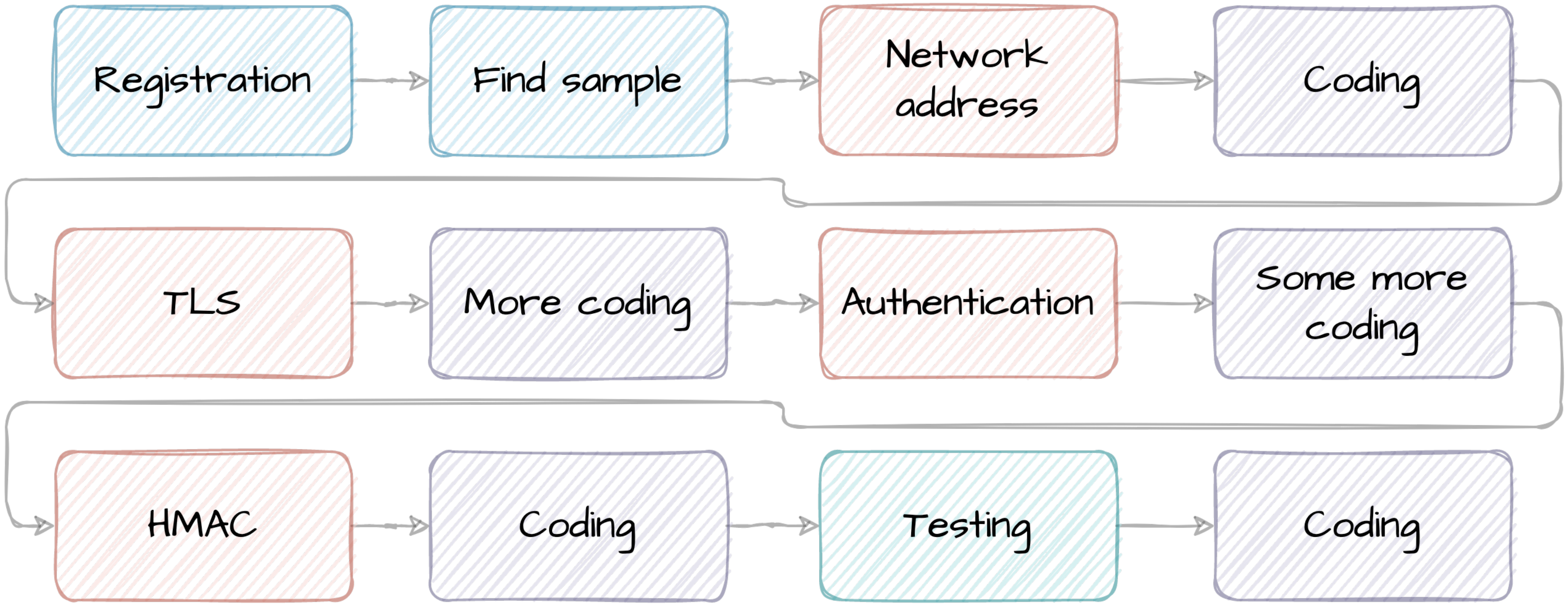
- Just one service provides sample
- None generates the key for the consumer
- Docs with web UI and trivial key
- Reuse of API key

No public key signatures

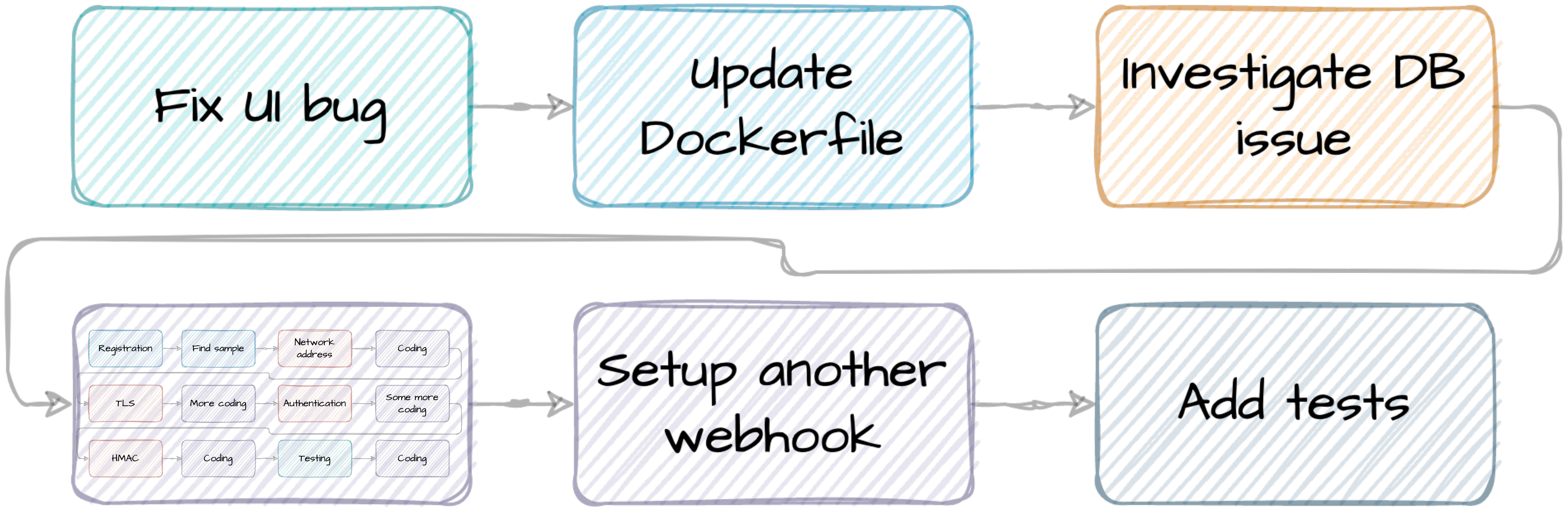
Tools and code samples

- Missing HMAC verification
- Hardcoded key
- Testing tool that requires turning off authentication

A Day in the Life



A Day in the Life



Best practices

Guidance

Avoid lazy security controls

Explain risks clearly

Do not provide insecure options

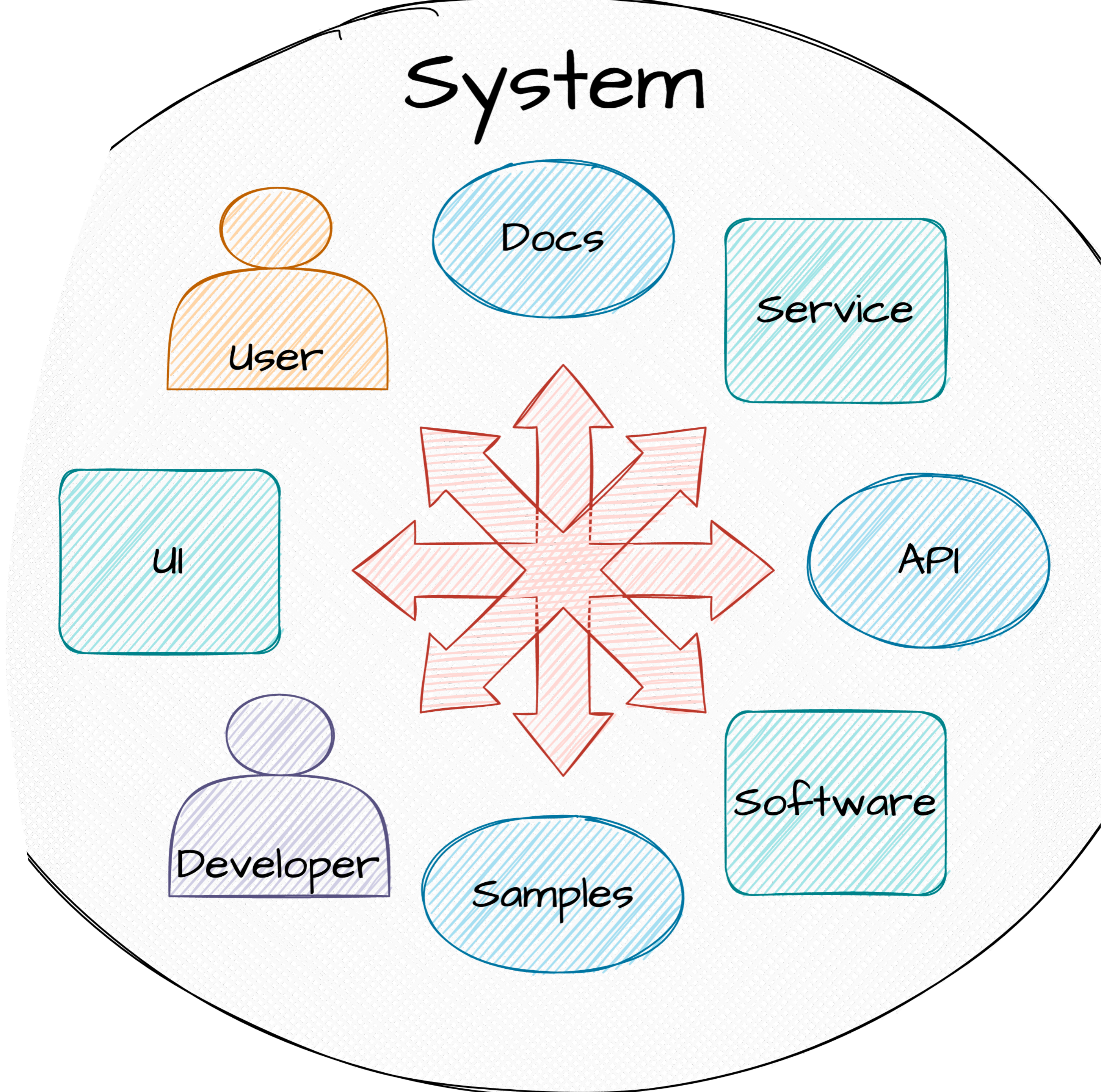
Do not delegate security tasks

Production quality code samples

Isolated, transparent debugging

Summary

- Humans are part of the system security
- Developers are new users
- APIs, docs, samples are developer interfaces
- Poor developer interfaces tricks them into security bugs
- Solutions are out there and easy



Thank you!

Questions?