







# Awaria Cloudflare 2025

Jak jeden plik "położył" 16% internetu

Geeks Club2

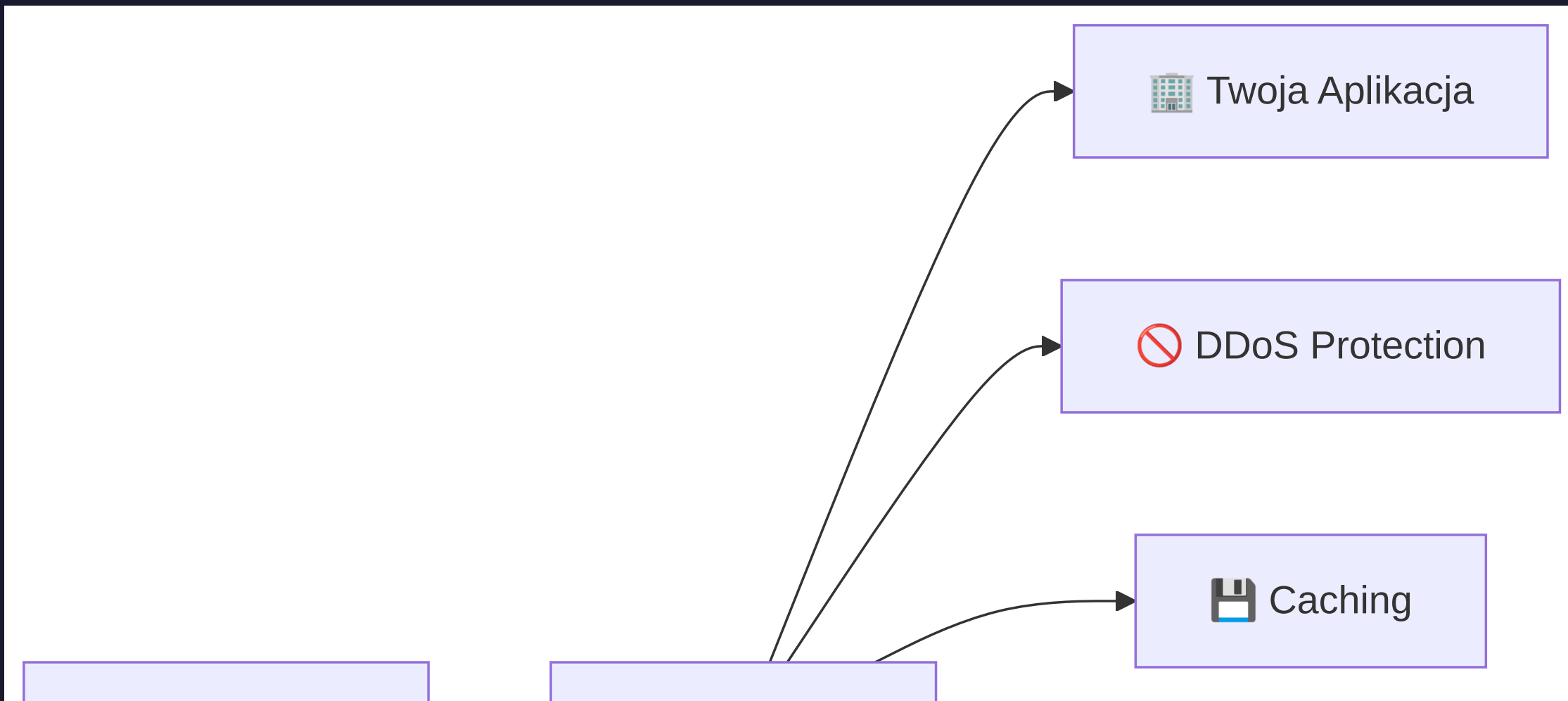
 10 grudnia 2025 r.

# Agenda

1.  **Dlaczego Cloudflare jest ważny?**
2.  **Co się stało?** - Timeline awarii
3.  **Analiza techniczna** - ClickHouse, Rust, unwrap()
4.  **Czynniki mylące** - Dlaczego myśleli, że to atak DDoS
5.  **Wnioski i działania naprawcze**
6.  **Komentarz** - Co my z tego wyciągamy?

# Co to jest Cloudflare?

Middleware między klientem a Twoją aplikacją



# Skala Cloudflare

~16% całego ruchu internetowego 

Każdy co szósty request w internecie przechodzi przez Cloudflare

## Znani użytkownicy:

Kategoria	Firmy
 Technologia	Mozilla, Microsoft Azure, Office 365, IBM
 E-commerce	Nike, H&M, Shopify
 Social	Reddit, Digital Ocean

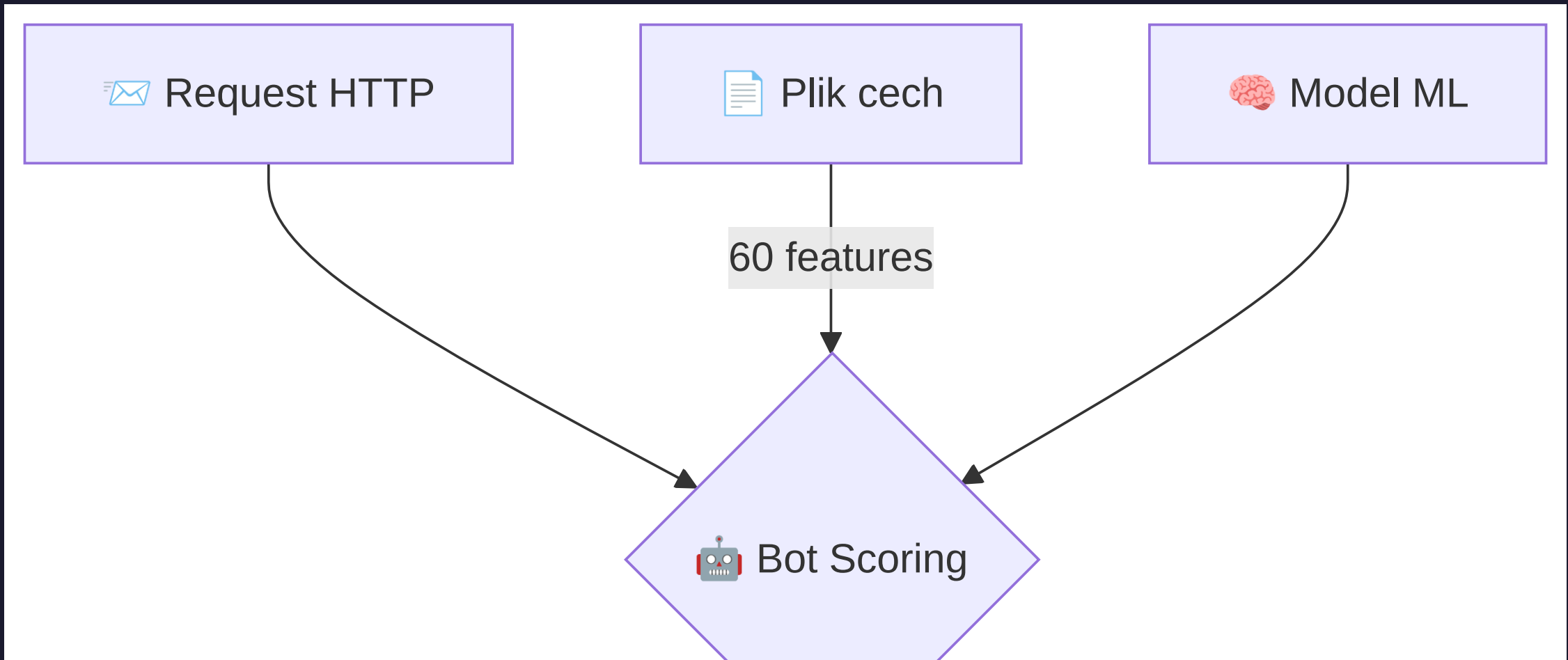
## Timeline awarii

```
title 18 listopada 2025 Awaria Cloudflare
"11:05": "Wdrożono zmianę uprawnień w ClickHouse"
"11:20": "● Początek problemów - błędy 5xx"
"11:28": "Wdrożenie dociera do produkcji"
"11:32": "Analiza - początkowo podejrzenie DDoS"
"13:05": "Obejście dla Workers KV i Access"
"14:24": "Identyfikacja przyczyny - plik bot managementu"
"14:30": "● Wdrożenie poprawnego pliku"
"17:06": "● Pełna normalizacja"
```



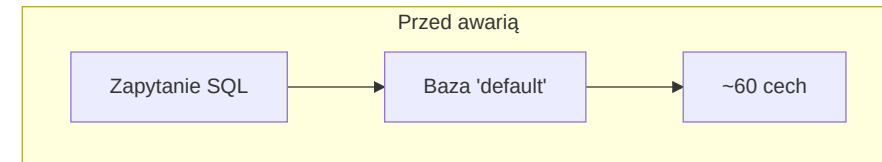
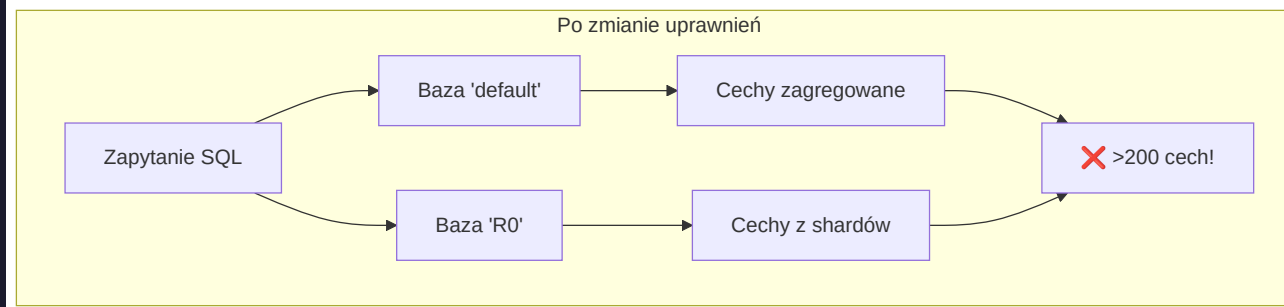
# Bot Management - Źródło problemu

Jak działa ocena botów?



# Architektura ClickHouse

## Bazy danych i shardy



# Zapytanie bez dyskryminatora bazy

```
SELECT
  name,
  type
FROM system.columns
WHERE
  table = 'http_requests_features'
ORDER BY name;
```

## Problem:

- Brak `WHERE database = 'default'`
- Po zmianie uprawnień → widoczne obie bazy
- **60 cech × 2 = 120+** cech





# Rust i fatalne `unwrap()`

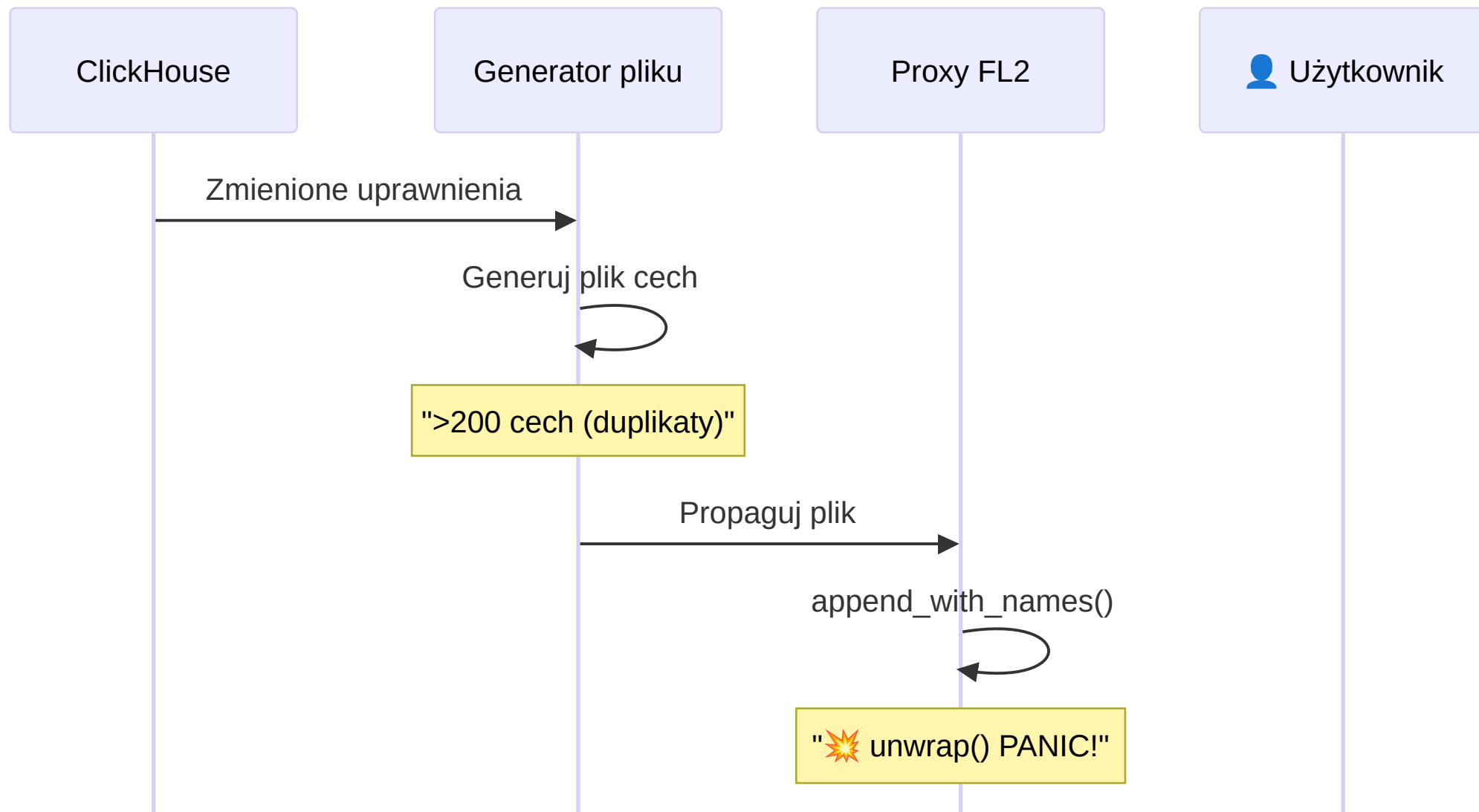
```
// Uproszczony kod który spowodował panikę
fn load_features(config: &Config) -> Features {
    let features = append_with_names(&config)
        .unwrap(); // 💣 BOOM!

    features
}
```

## Problem z prealokacją pamięci:

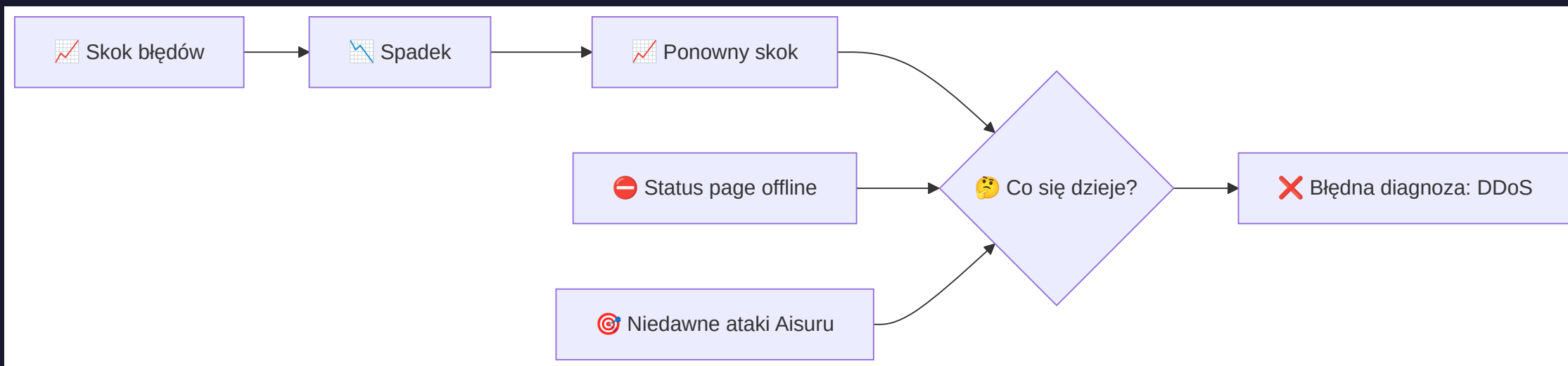
- **Limit:** 200 cech (bufor bezpieczeństwa)
- **Oczekiwane:** ~60 cech
- **Otrzymane:** >200 cech (duplikaty)
- **Rezultat:** `Result::unwrap()` on `Err` → **PANIKA** 💀

# 🌟 Mechanizm awarii



# Czynniki mylące

## Dlaczego myśleli o ataku DDoS?



### Nietypowe zachowanie:

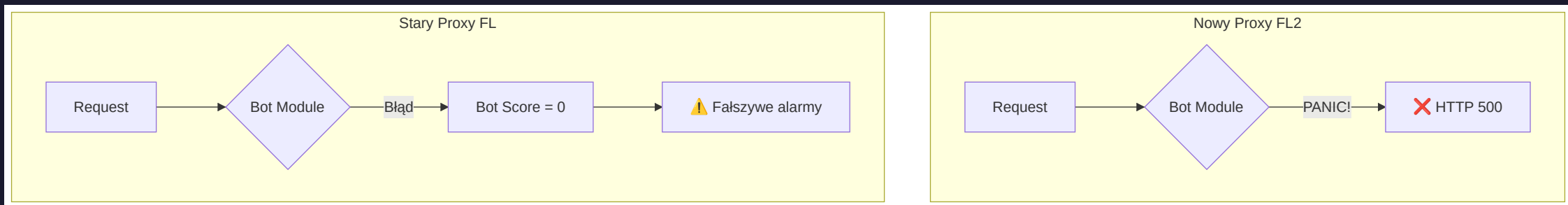
- Fluktuacje: stare nody miały poprawny cache
- Status page (niezależna infra) też offline → **zbieg okoliczności!**



## Wpływ na usługi

Usługa	Wpływ
 <b>CDN / Bezpieczeństwo</b>	HTTP 5xx dla wszystkich klientów
 <b>Turnstile</b>	Całkowity brak działania
 <b>Workers KV</b>	Podwyższony poziom błędów
 <b>Dashboard</b>	Brak możliwości logowania
 <b>Access</b>	Błędy uwierzytelniania
 <b>Email Security</b>	Obniżone wykrywanie spamu

## FL vs FL2 - Różny wpływ







**FL2:** Twarde błędy 500

**FL:** Wszystko = "nie-bot" → problemy z regułami blokowania



# Działania naprawcze Cloudflare

## Oficjalna lista:

1.  **Hardening** konfiguracji wewnętrznej (jak dane od użytkowników)
2.  **Kill-switches** - globalne wyłączniki funkcji
3.  **Core dumps** - nie mogą przeciążać systemu
4.  **Przegląd trybów awarii** wszystkich modułów proxy

*"Dzisiejsza awaria była najpoważniejszym incydentem od 2019 roku"*

— Matthew Prince, CEO

## 💡 Nasze wnioski techniczne

Co można było zrobić lepiej?

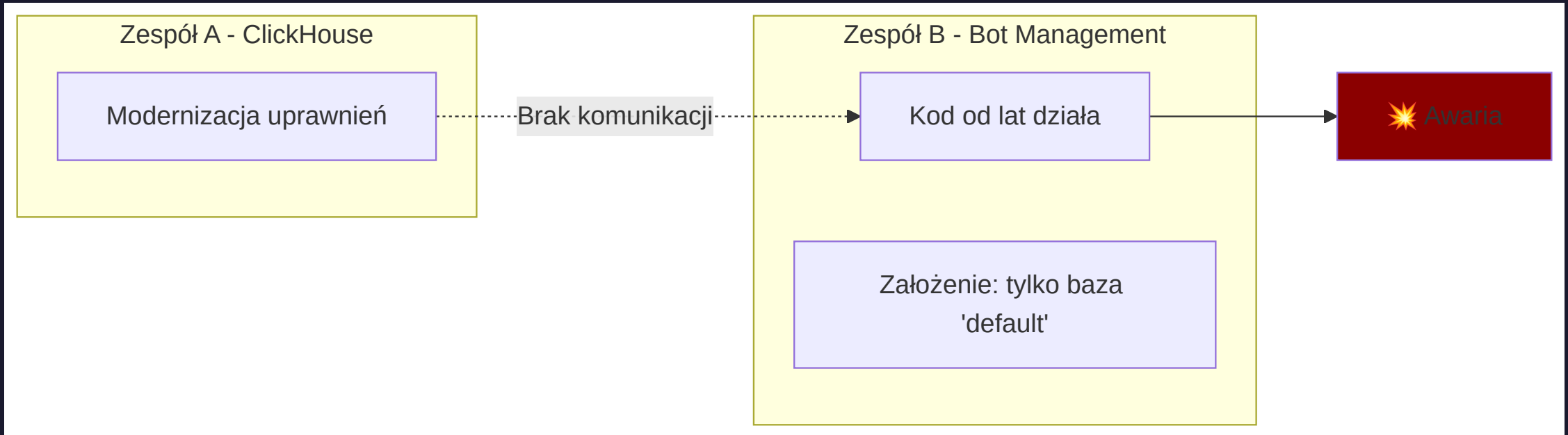
Pobrano  $>200$  cech



```
graph TD; A[Pobrano >200 cech] --> B[ ];
```



# Problem organizacyjny



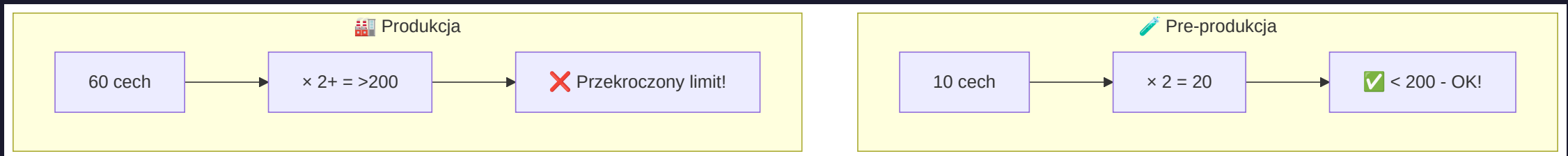
## Kluczowy problem:

**Zmiana w jednym miejscu → eksplozja w innym**



# A co ze środowiskiem testowym?

Możliwe wyjaśnienie:



**Skala produkcji  $\neq$  Skala testów**

# Kluczowe lekcje

## 1 Defensywne programowanie

Nigdy nie ufaj, że dane wejściowe będą poprawne

## 2 Graceful degradation

System powinien działać ograniczenie, nie crashować

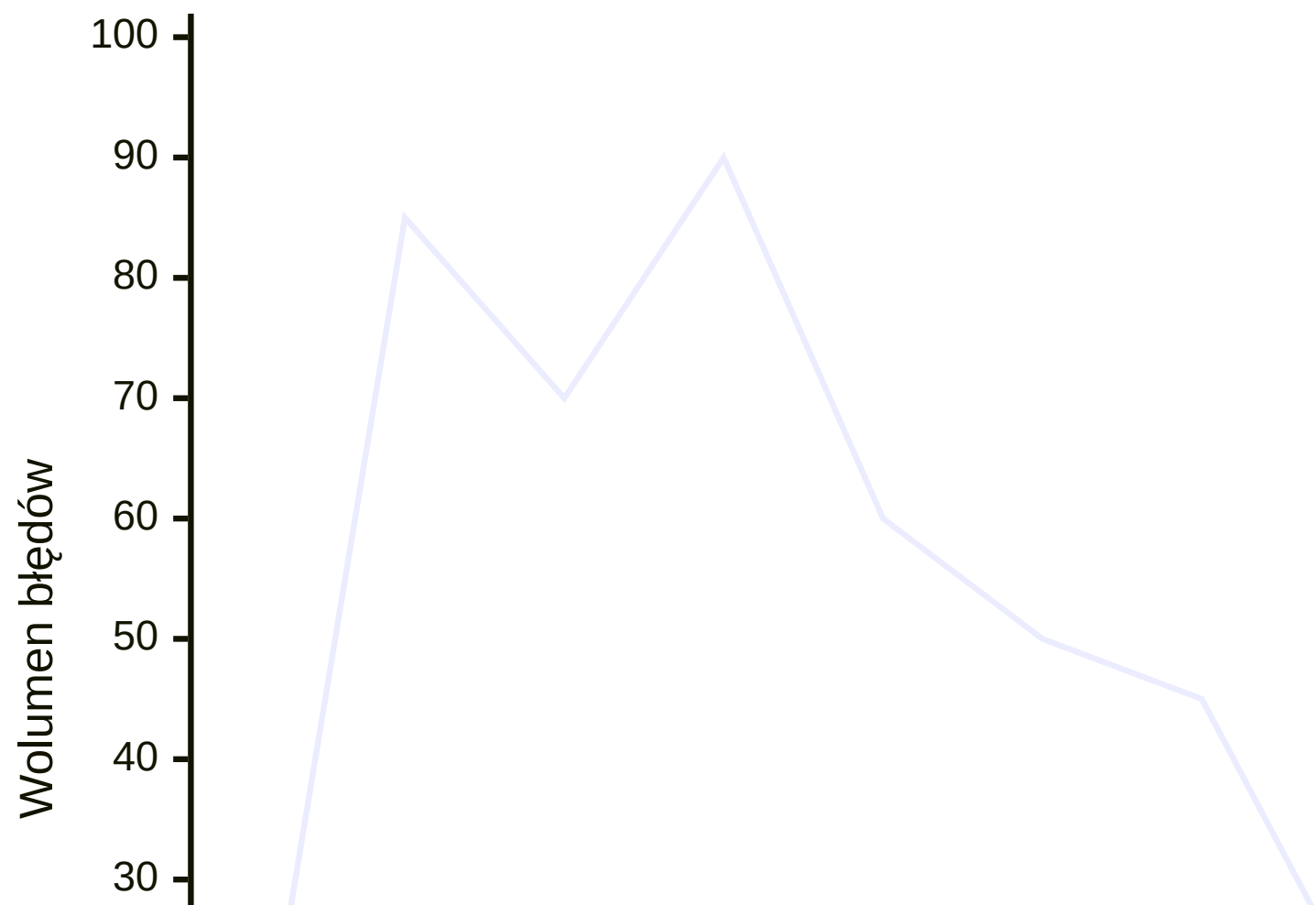
## 3 Komunikacja między zespołami

Zmiany w jednym systemie mogą wpłynąć na inne



# Wizualizacja awarii

Błędy HTTP 5xx podczas incydentu



# Do dyskusji

## Pytania dla zespołu:


1. 🔍 **Czy mamy podobne "ukryte zależności"** w naszych systemach?
2. 🦀 **Jak obsługujemy błędy** w krytycznych ścieżkach kodu?
3. 📊 **Czy nasze środowiska testowe** odzwierciedlają skalę produkcji?
4. 🔔 **Jak szybko wykryjemy** awarię przed użytkownikami?
5. 📝 **Czy robimy post-mortemy** i czy są publiczne?

# Podsumowanie



## Źródła

### Oficjalne Post-Mortem:

 [blog.cloudflare.com/pl-pl/18-november-2025-outage](https://blog.cloudflare.com/pl-pl/18-november-2025-outage)

### Video:

 [IT News #25 - DevMentors](#)



# Dziękuję!

# Pytania?

Post-Mortem 18.11.2025

**Kontakt:** [granica.lukasz@gmail.com](mailto:granica.lukasz@gmail.com)

