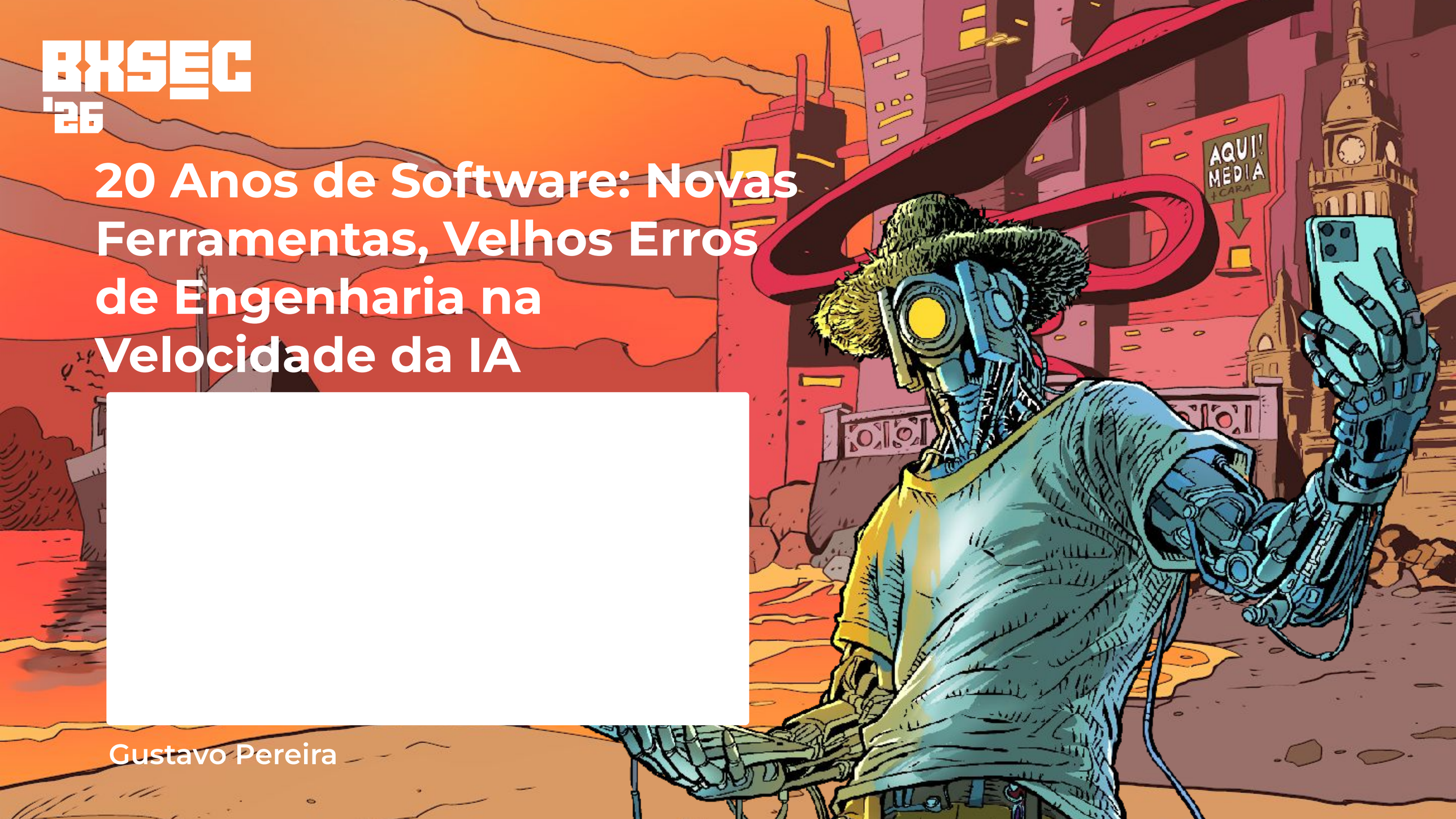

AQUI!
MEDIA
+CARA

BXSEC '26

20 Anos de Software: Novas Ferramentas, Velhos Erros de Engenharia na Velocidade da IA

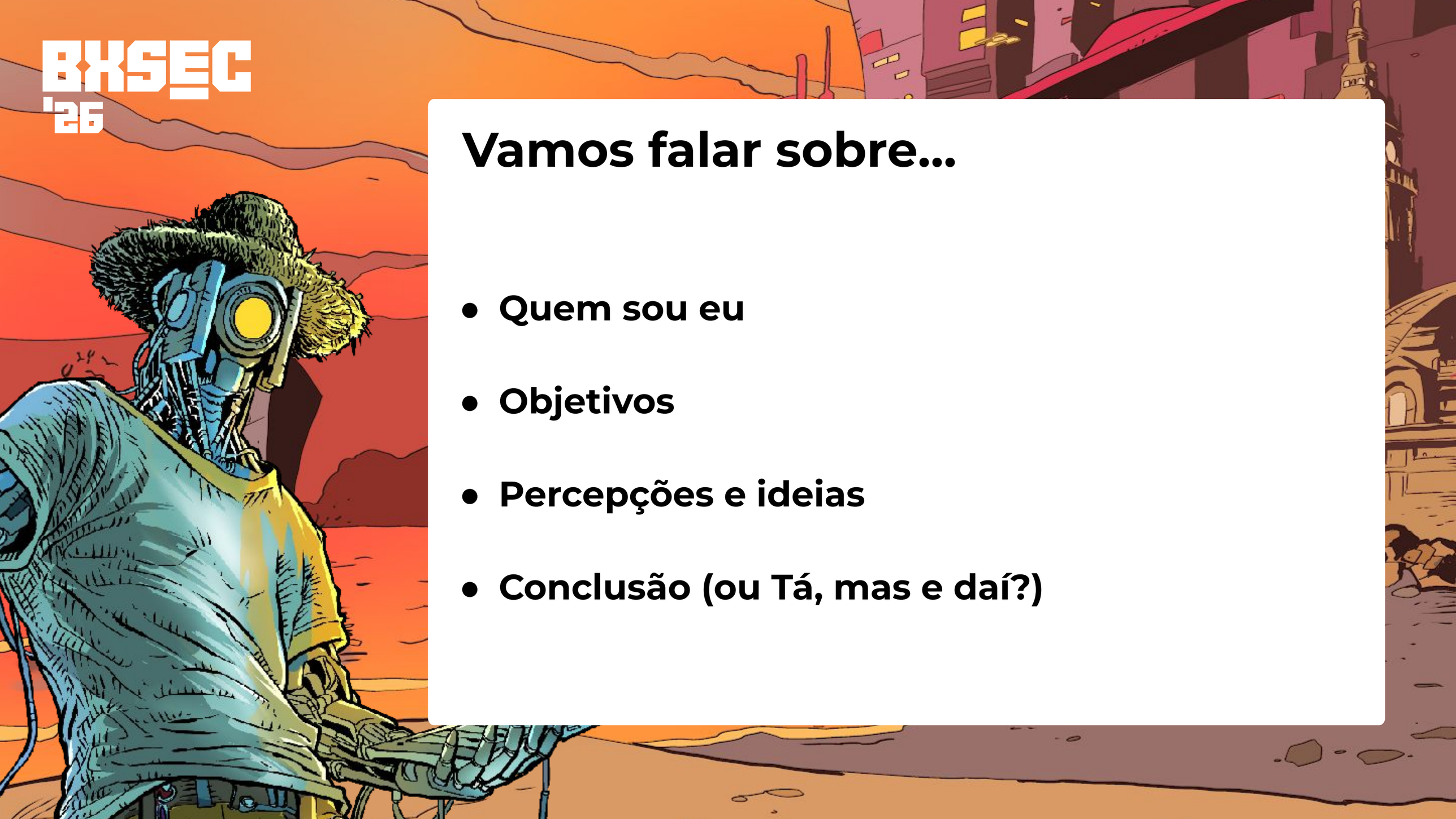


Gustavo Pereira

BXSEC '26

20 Anos de Software: Novas Ferramentas, Velhos Erros de Engenharia na Velocidade da IA

Gustavo Pereira



Vamos falar sobre...

- Quem sou eu
- Objetivos
- Percepções e ideias
- Conclusão (ou Tá, mas e daí?)

BXSEC '26

Oi, eu sou o Gustavo!

- **Comecei lá em 2001 criando site de RPG em HTML!**
- **Sempre muito envolvido em comunidades**

BXSEC '26

Oi, eu sou o Gustavo!

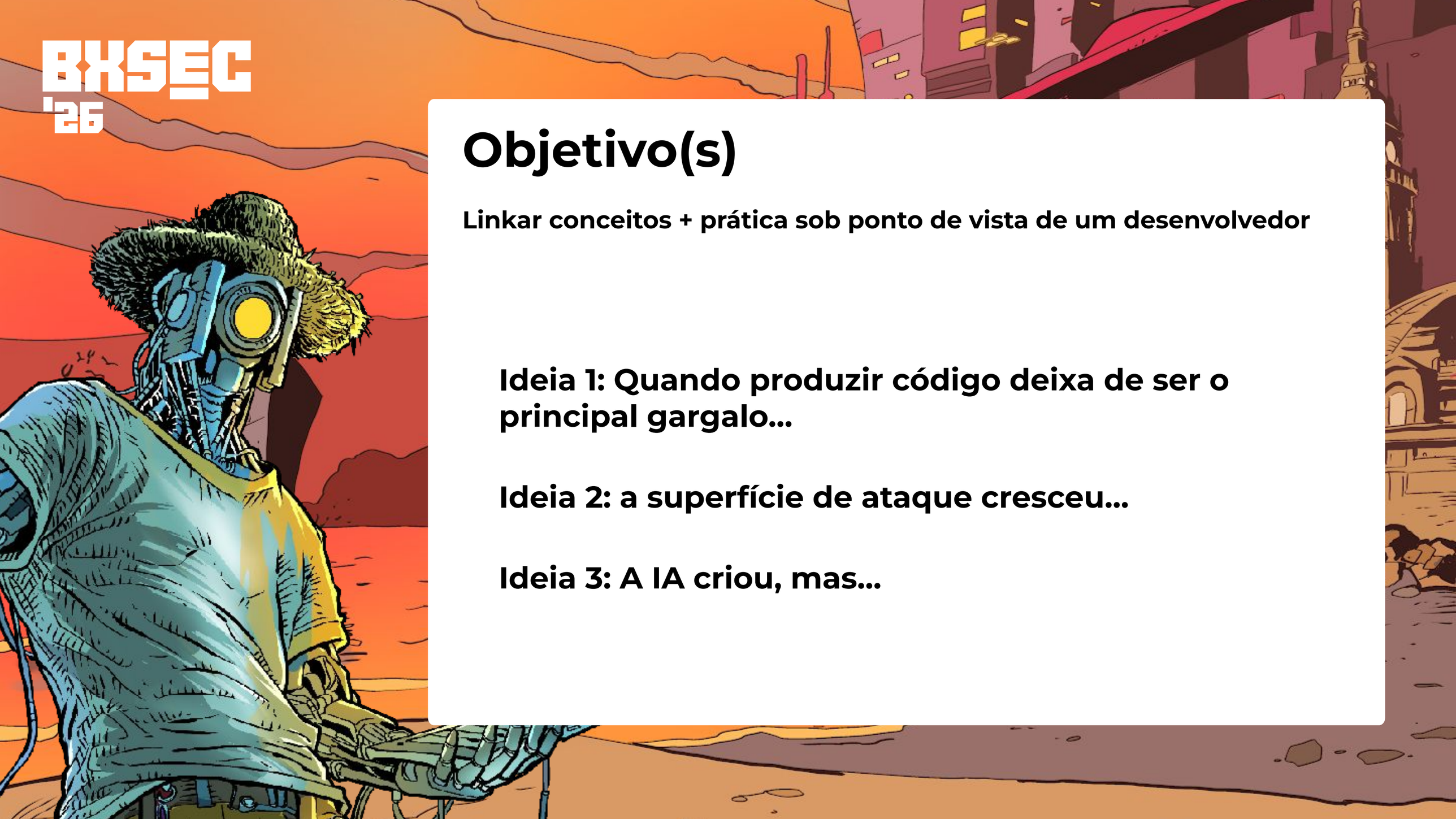
- **Bacharel em Ciência da Computação - UNISANTA**
- **Dinossauro da turma de Processamento de Dados - Fatec/Santos**
- **Pós - Arquitetura de Software Distribuído - PUC MINAS**

BXSEC '26

Oi, eu sou o Gustavo!

- **Engenheiro de Software Sr - Itaú (2020 - atual)**

Só tenho linkedin: linkedin.com/in/gustavopereira2



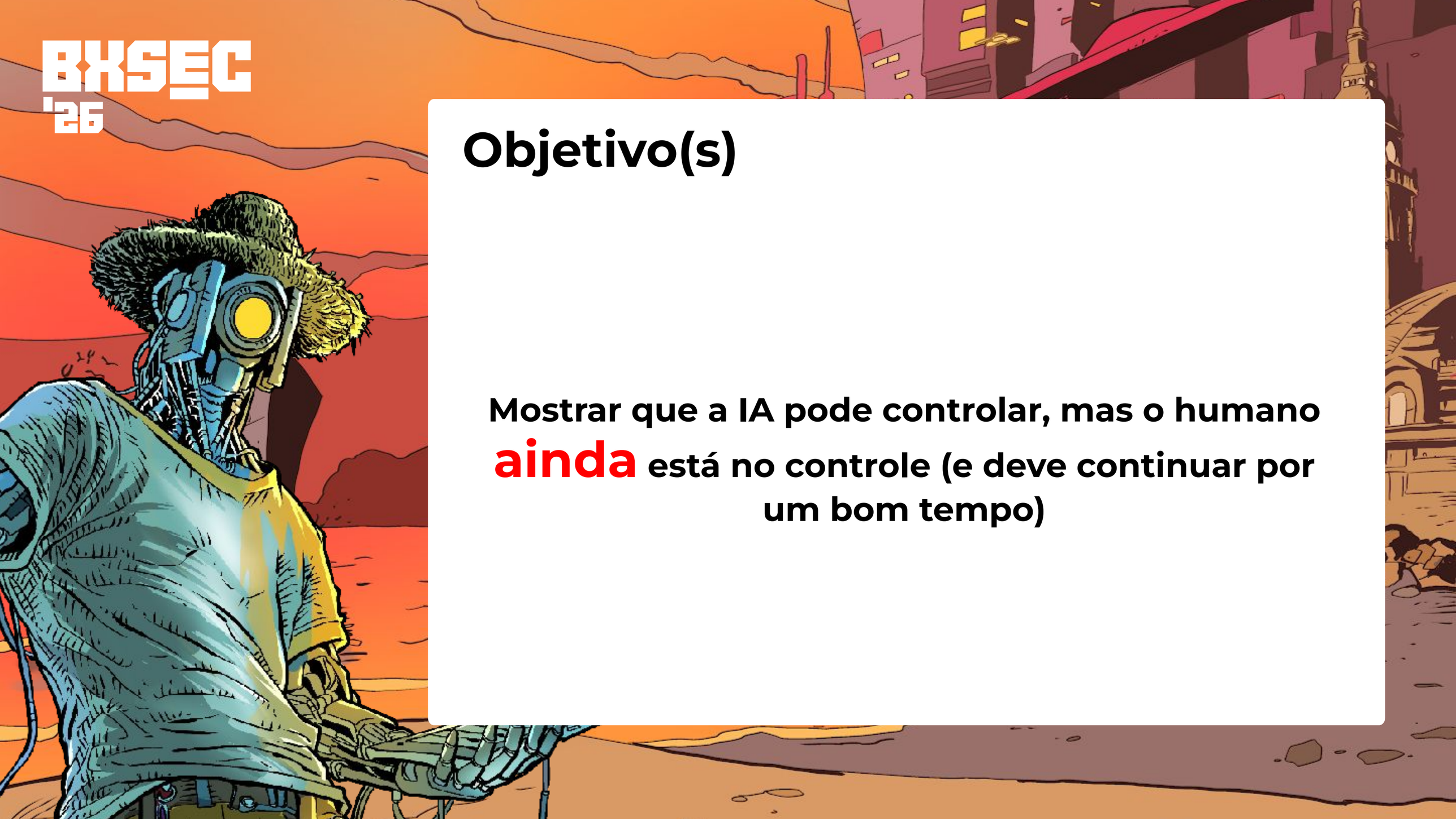
Objetivo(s)

Linkar conceitos + prática sob ponto de vista de um desenvolvedor

Ideia 1: Quando produzir código deixa de ser o principal gargalo...

Ideia 2: a superfície de ataque cresceu...

Ideia 3: A IA criou, mas...



Objetivo(s)

Mostrar que a IA pode controlar, mas o humano ainda está no controle (e deve continuar por um bom tempo)

BXSEC '26

II. Percepções e ideias

Era 1:

“Nós escrevíamos o código”

Gargalos:

- **Produção de código**
- **Processos lentos e burocráticos para publicação de features**
- **Conhecimento altamente especializado**

Era 2:

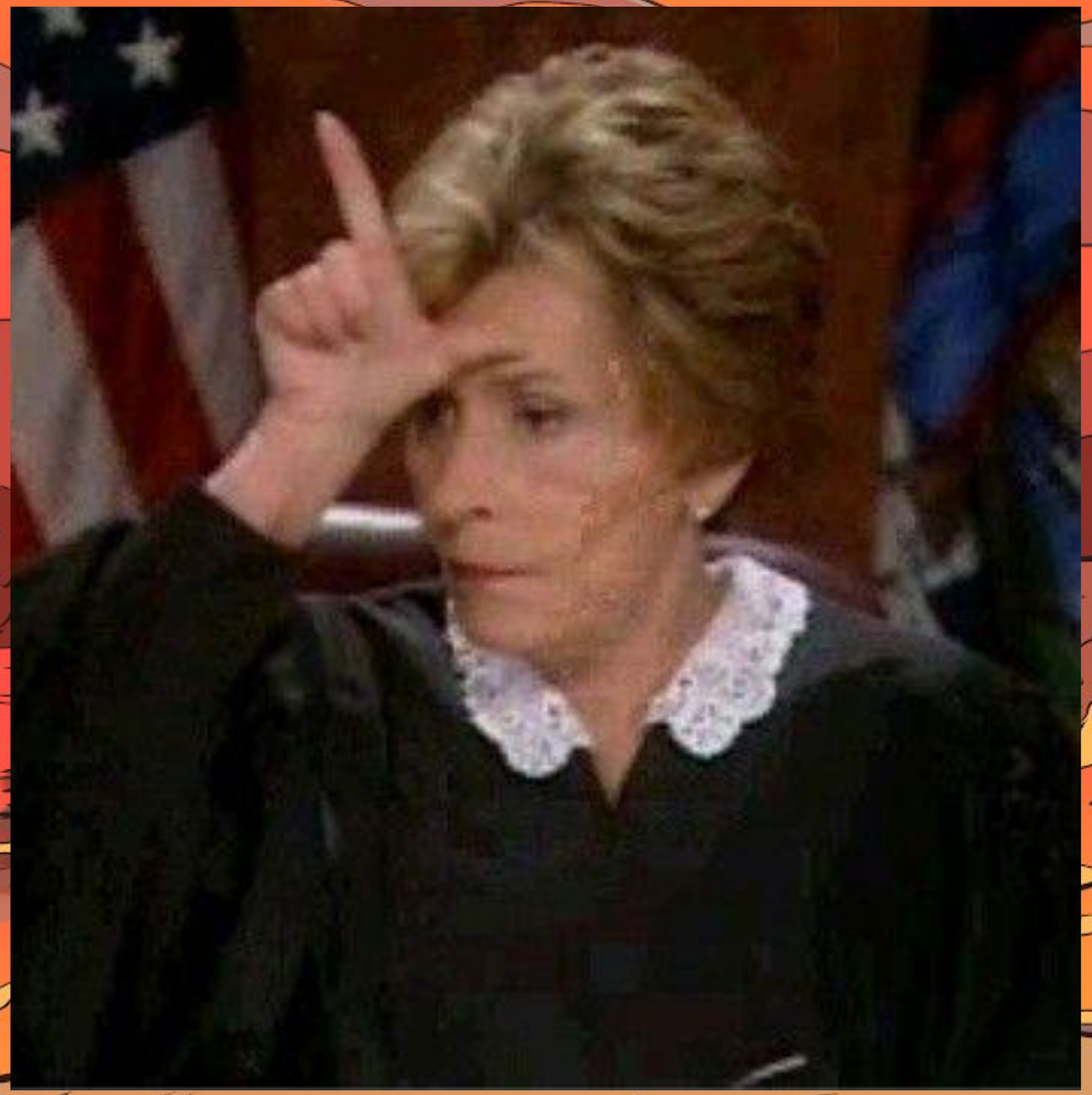
Frameworks aceleram código

Frameworks e seus boilerplates

O gargalo era arquitetura e operação.

Entregar ficou “menos oneroso”.

BXSEC '26

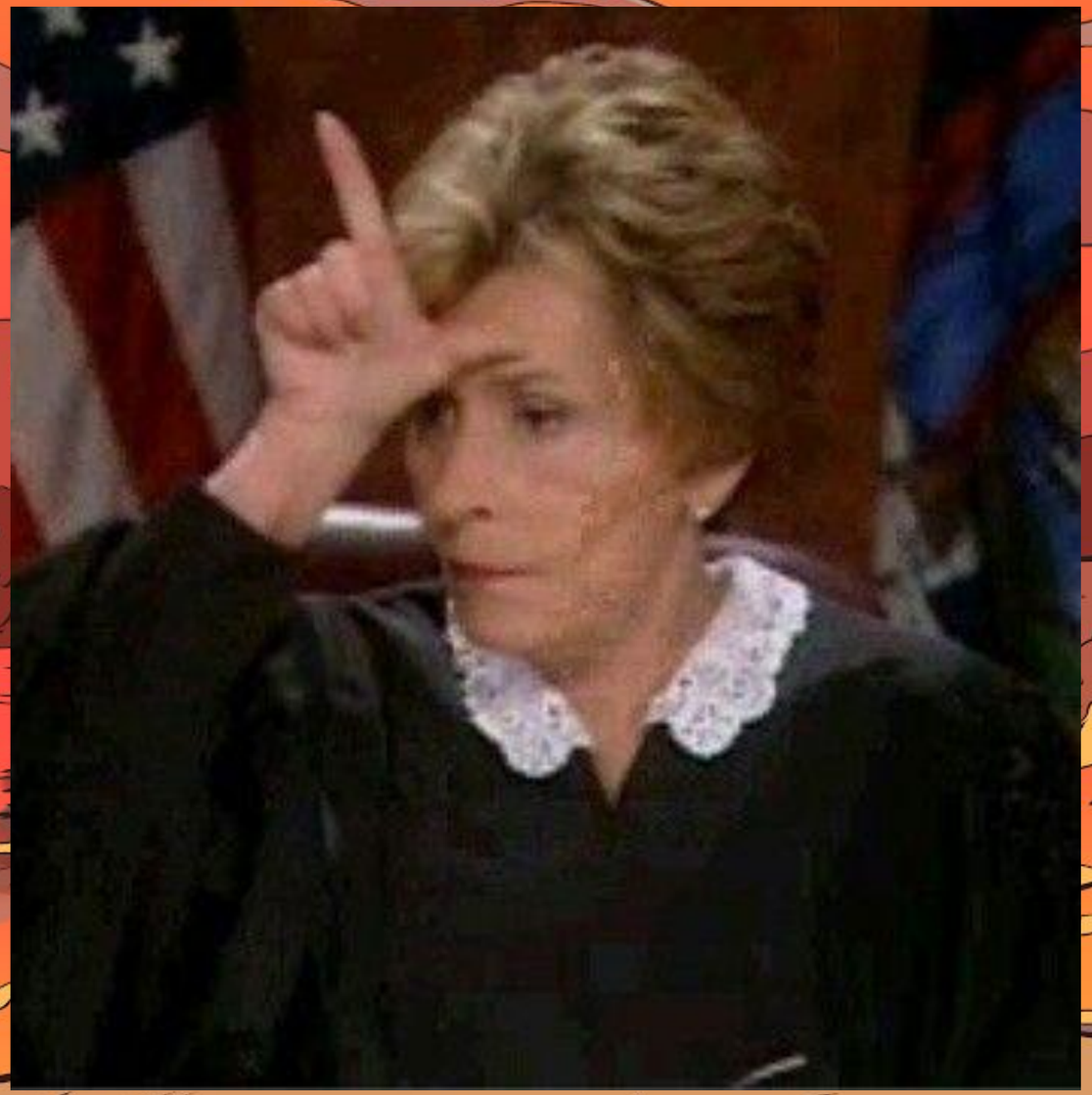


Era 3: A IA escreve

A IA escreve, você JULGA

AGENTS/SKILLS
Progressive Disclosure
Janelas de contexto
mecanismos agênticos (autonomia IA)
ETC

BXSEC '26



Era 3: A IA escreve

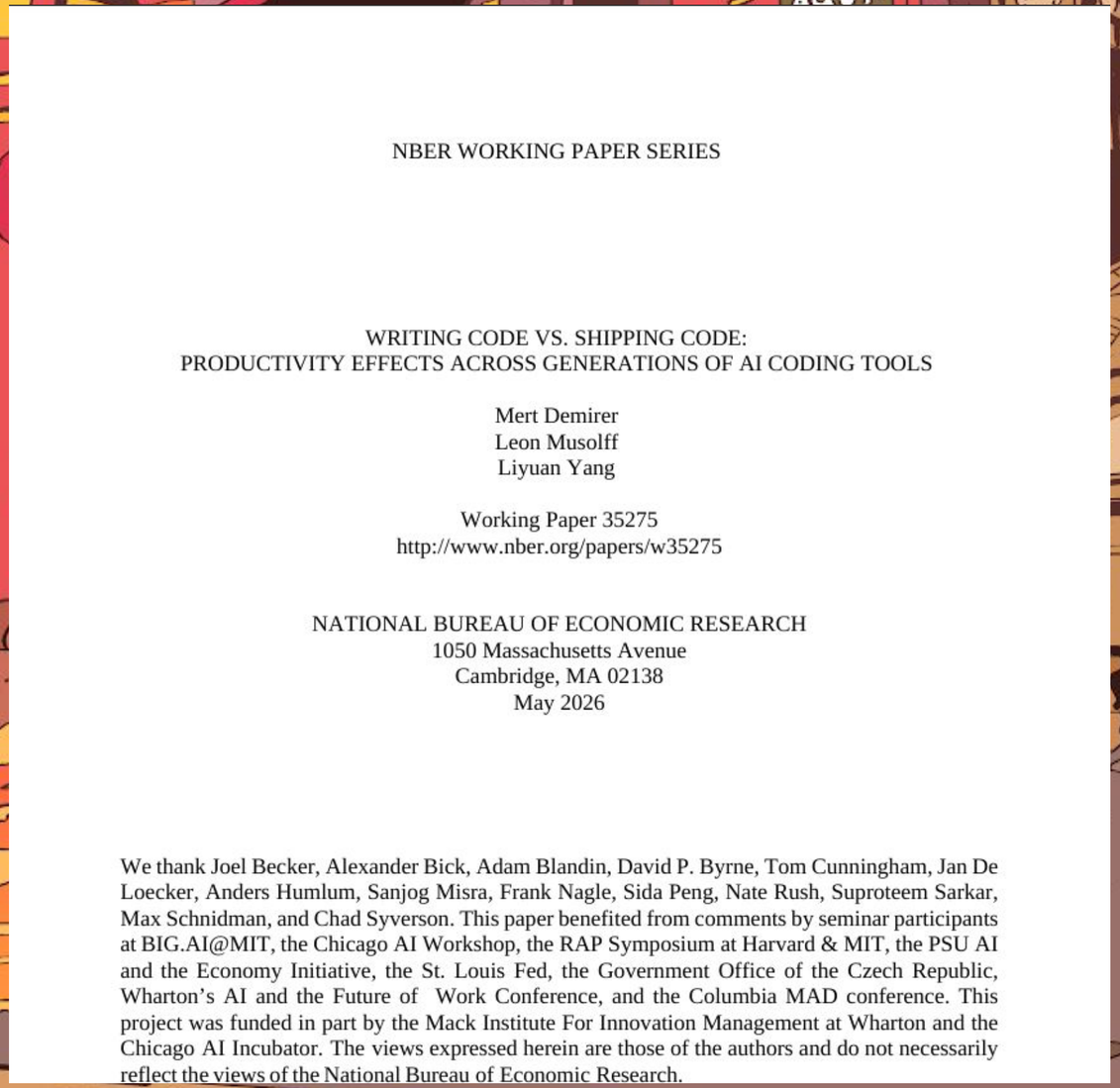
A Supply Chain aumentou consideravelmente seu nível de complexidade e importância...

MAS E A RELEVANCIA?

BXSEC '26

https://www.nber.org/papers/w35275

IA aumenta a velocidade, mas o resultado ainda não está sendo refletido na entrega final



NBER WORKING PAPER SERIES

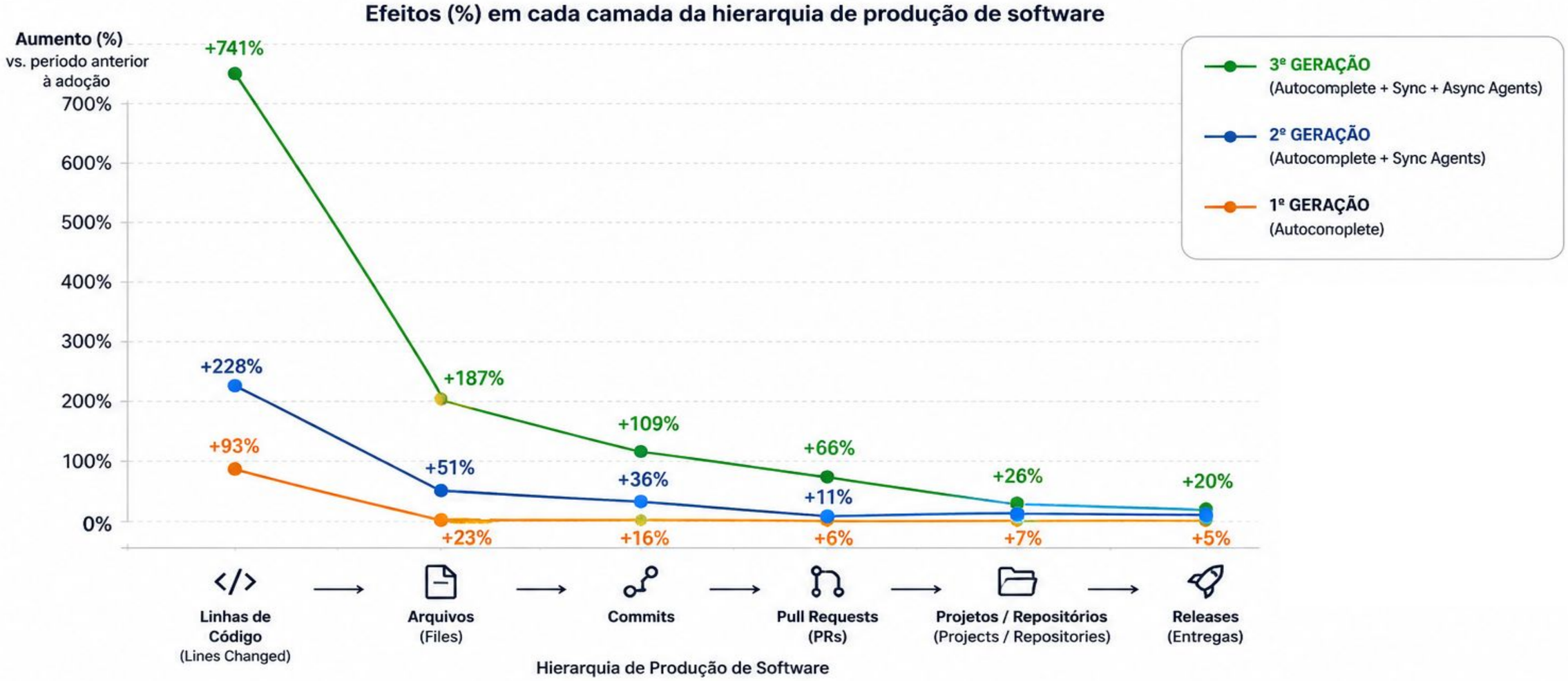
WRITING CODE VS. SHIPPING CODE: PRODUCTIVITY EFFECTS ACROSS GENERATIONS OF AI CODING TOOLS

Mert Demirer
Leon Musolff
Liyuan Yang

Working Paper 35275
http://www.nber.org/papers/w35275

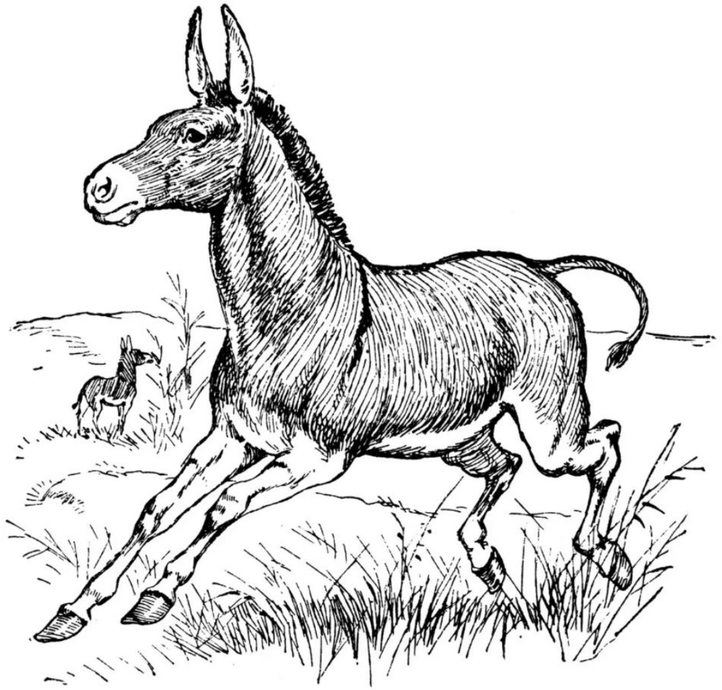
NATIONAL BUREAU OF ECONOMIC RESEARCH
1050 Massachusetts Avenue
Cambridge, MA 02138
May 2026

We thank Joel Becker, Alexander Bick, Adam Blandin, David P. Byrne, Tom Cunningham, Jan De Loecker, Anders Humlum, Sanjog Misra, Frank Nagle, Sida Peng, Nate Rush, Suproteem Sarkar, Max Schnidman, and Chad Syverson. This paper benefited from comments by seminar participants at BIG.AI@MIT, the Chicago AI Workshop, the RAP Symposium at Harvard & MIT, the PSU AI and the Economy Initiative, the St. Louis Fed, the Government Office of the Czech Republic, Wharton's AI and the Future of Work Conference, and the Columbia MAD conference. This project was funded in part by the Mack Institute For Innovation Management at Wharton and the Chicago AI Incubator. The views expressed herein are those of the authors and do not necessarily reflect the views of the National Bureau of Economic Research.



Fonte: Demirer, M., Darouich, M., & Schwaab, B. (2025). Writing vs. Shipping Code: The Productivity Effects of Generative AI in Software Development. NBER Working Paper No. 35275.

Living the extraortinary in the AI era



Coding with no fear of being hacked

because Shift Left and Threat Modeling are hype

Idéia 1:

Quando produzir código deixa de ser o principal gargalo, decidir o que deve ser construído passa a ser ainda mais importante.

BXSEC '26

E a superfície de ataque?

Anos 70/80/90

Quase nula!

Anos 2000

Monolitos

Servidores físicos, superfície de ataque mais restrita

Equipes bem-definidas, mas restritas

2010: Tudo na cloud

- DevOps
- Mobile First
- SOA
- Serverless
- Cloud Providers

Saiu do físico e tomou proporções maiores

O modelo era focado no perímetro, com duas “verdades”

Internet: má / VPN: Boa

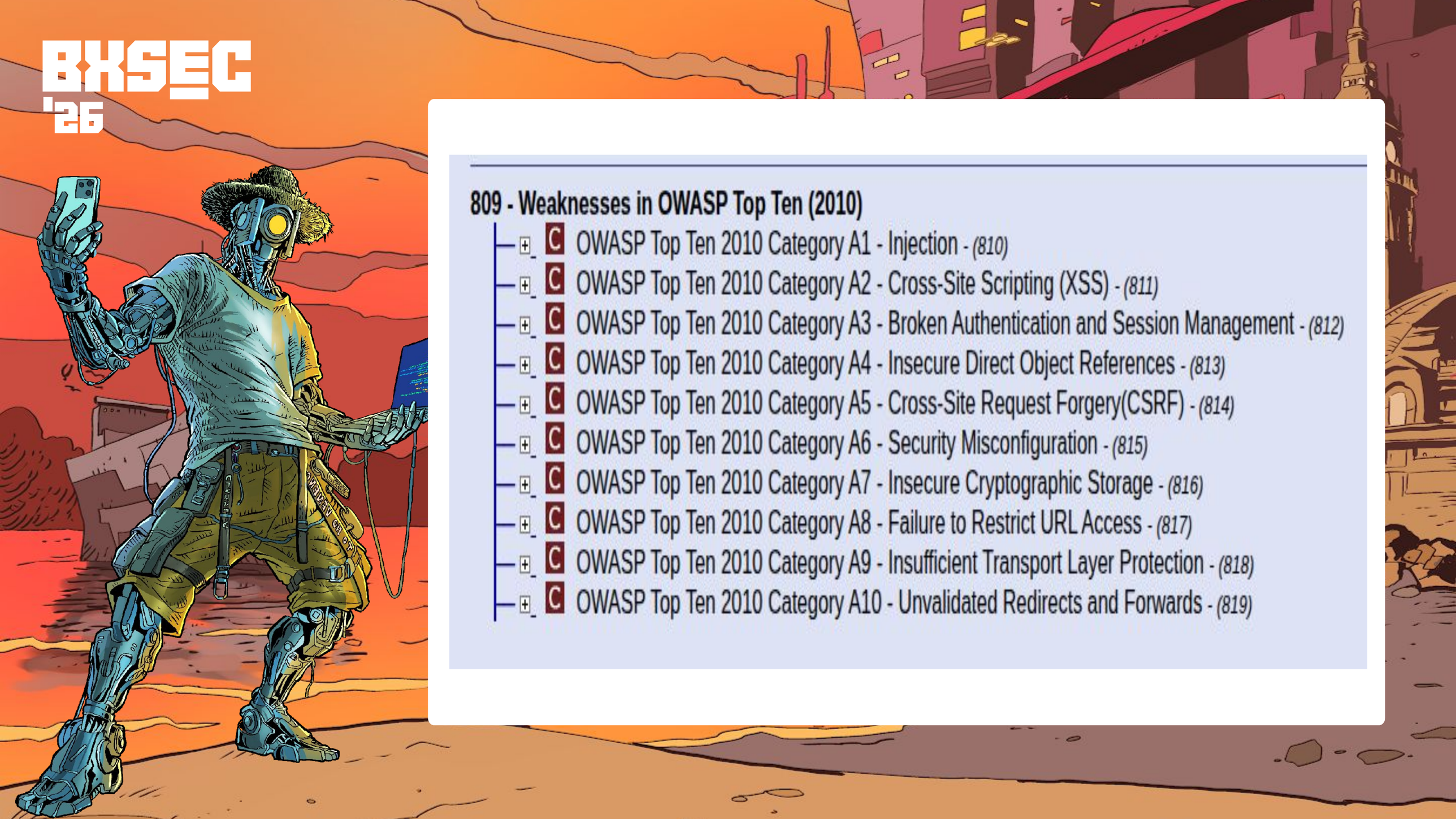
2010: Tudo na cloud

Superfície associada a:

- IPs públicos
- Portas abertas
- webservers
- endpoints
- Banco de Dados
- Active Directory

Dissolução do perímetro

OWASP TOP 10 (2010)



809 - Weaknesses in OWASP Top Ten (2010)

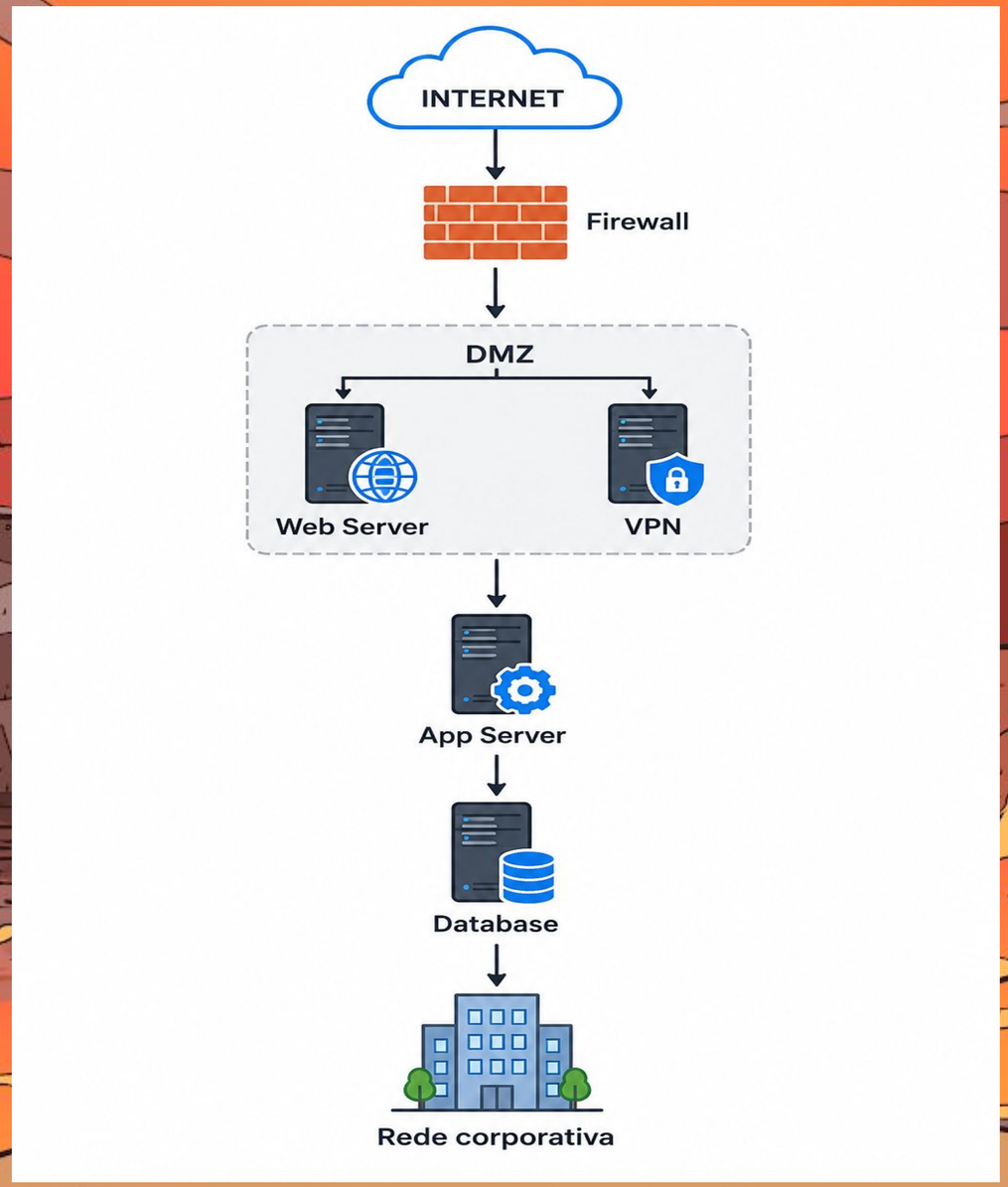
- C OWASP Top Ten 2010 Category A1 - Injection - *(810)*
- C OWASP Top Ten 2010 Category A2 - Cross-Site Scripting (XSS) - *(811)*
- C OWASP Top Ten 2010 Category A3 - Broken Authentication and Session Management - *(812)*
- C OWASP Top Ten 2010 Category A4 - Insecure Direct Object References - *(813)*
- C OWASP Top Ten 2010 Category A5 - Cross-Site Request Forgery(CSRF) - *(814)*
- C OWASP Top Ten 2010 Category A6 - Security Misconfiguration - *(815)*
- C OWASP Top Ten 2010 Category A7 - Insecure Cryptographic Storage - *(816)*
- C OWASP Top Ten 2010 Category A8 - Failure to Restrict URL Access - *(817)*
- C OWASP Top Ten 2010 Category A9 - Insufficient Transport Layer Protection - *(818)*
- C OWASP Top Ten 2010 Category A10 - Unvalidated Redirects and Forwards - *(819)*

2020-Atual: Tudo ao mesmo tempo agora

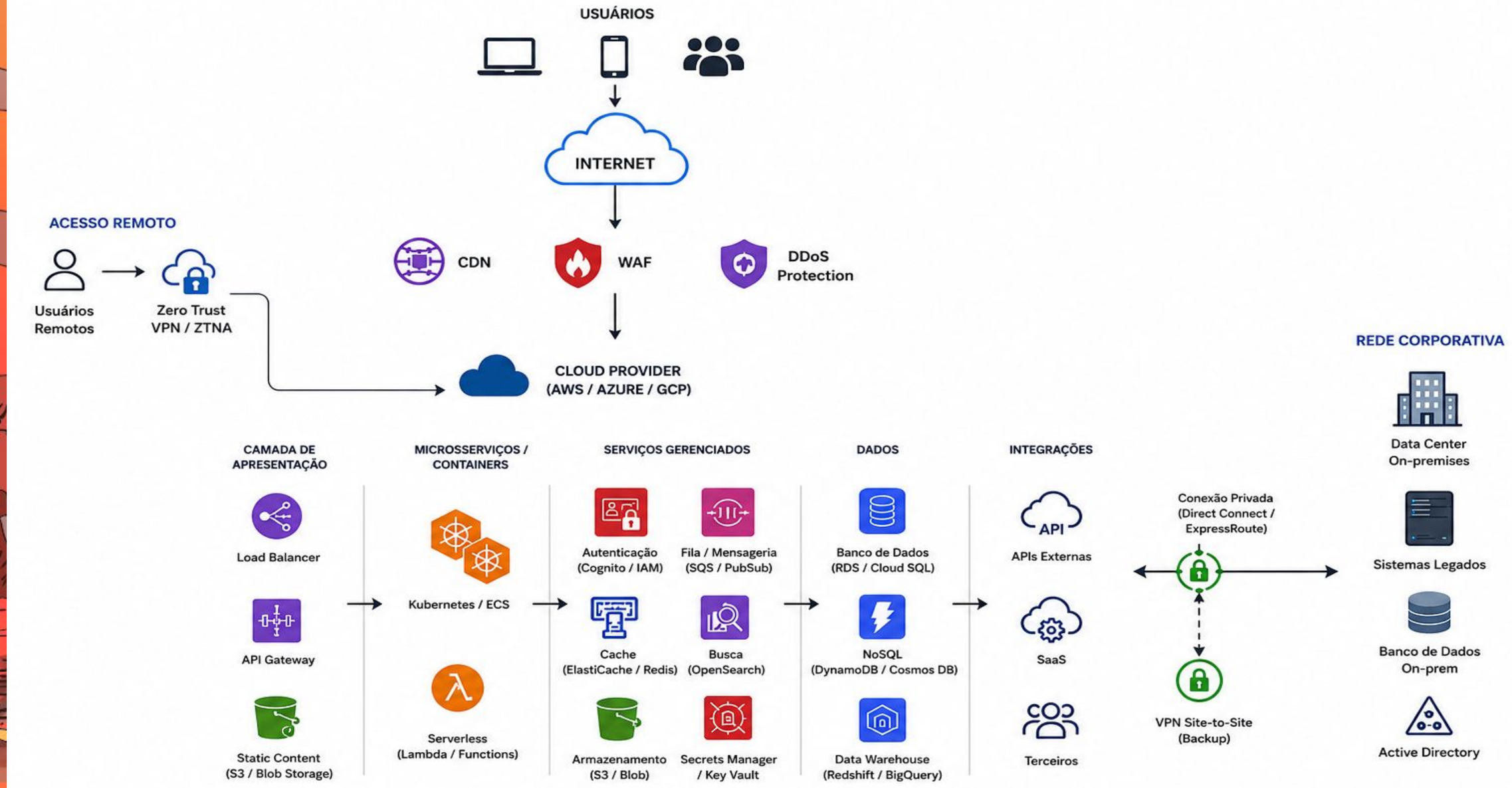
- Containers
- Kubernetes
- Multi-cloud
- Resiliência (99,9999% de uptime)

- IA Generativa
- agentes autônomos

Environment mais complexo



Nós saímos desse modelo...

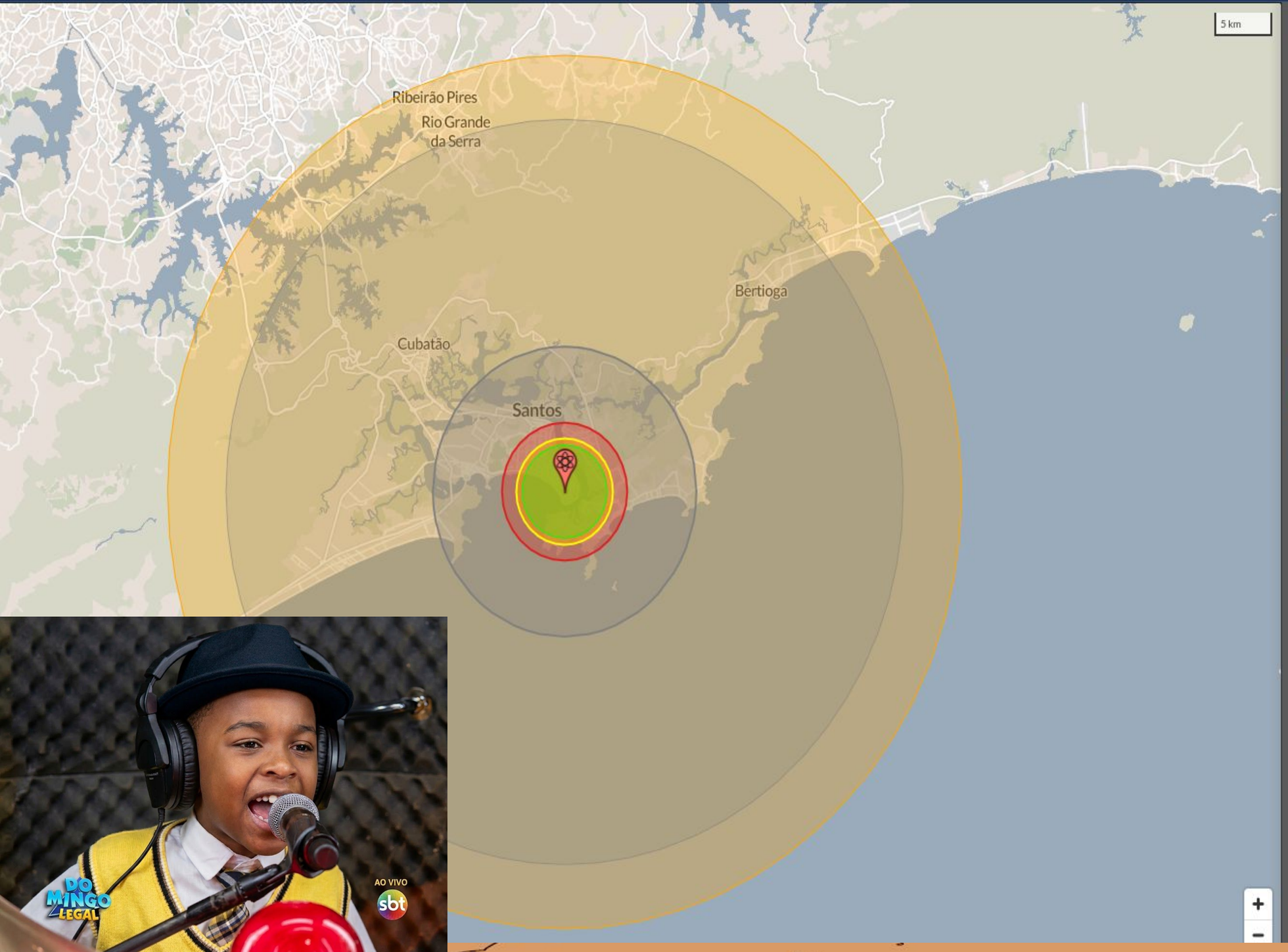

USUÁRIOS
INTERNET
ACESSO REMOTO
Usuários Remotos
Zero Trust VPN / ZTNA
CDN
WAF
DDoS Protection
CLOUD PROVIDER (AWS / AZURE / GCP)
REDE CORPORATIVA
CAMADA DE APRESENTAÇÃO
Load Balancer
API Gateway
Static Content (S3 / Blob Storage)
MICROSSERVIÇOS / CONTAINERS
Kubernetes / ECS
Serverless (Lambda / Functions)
SERVIÇOS GERENCIADOS
Autenticação (Cognito / IAM)
Fila / Mensageria (SQS / PubSub)
Cache (ElastiCache / Redis)
Busca (OpenSearch)
Armazenamento (S3 / Blob)
Secrets Manager / Key Vault
DADOS
Banco de Dados (RDS / Cloud SQL)
NoSQL (DynamoDB / Cosmos DB)
Data Warehouse (Redshift / BigQuery)
INTEGRAÇÕES
API
APIs Externas
SaaS
Terceiros
Conexão Privada (Direct Connect / ExpressRoute)
VPN Site-to-Site (Backup)
Data Center On-premises
Sistemas Legados
Banco de Dados On-prem
Active Directory

Antes: quais APIs estão expostas?

Hoje: quais CREDENCIAIS estão expostas?

BXSEC '26

Você troca uma API comprometida por uma credencial AWS comprometida?



NUKEMAP 2.76 : FAQ

You might also try: MISSILEMAP

1. **Drag** the marker to wherever you'd like to target.

Or you can select a preset...

Or type in the name of a city: Go

2. **Choose a warhead yield:** [?] 15000 kilotons [?]

Credencial AWS na mão dos cria

3. **Basic options:** *Height of burst:* [?] Airburst Surface
Other effects: Casualties Radioactive fallout

Advanced options: ▶

4. **Click** the "Detonate" button below.

Detonate | Clear all effects | Add new detonation | Center ground zero | Inspect location

Note that you can drag the target marker after you have detonated the nuke.

Effect distances for a 15 megaton surface burst: ▼

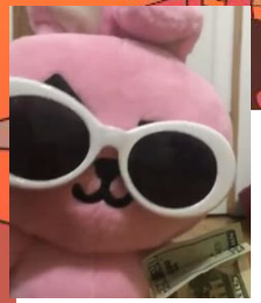
Radiation radius (500 rem): 3.63 km (41.4 km²)
500 rem ionizing radiation dose; likely fatal, in about 1 month; 15% of survivors will eventually die of cancer as a result of exposure.

Fireball radius: 4.14 km (53.8 km²)
Maximum size of the nuclear fireball; relevance to damage on the ground depends on the height of detonation. If it touches the ground, the amount of radioactive fallout is significantly increased. Anything inside the fireball is effectively vaporized.

Heavy blast damage radius (20 psi): 5.37 km (90.5 km²)
At 20 psi overpressure, heavily built concrete buildings are severely damaged or demolished; fatalities approach 100%. Often used as a benchmark for **heavy** damage in cities.

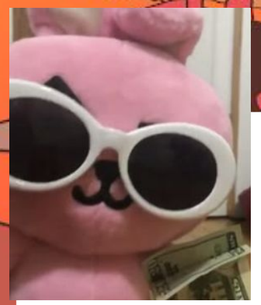
Moderate blast damage radius (5 psi): 11.3 km (400 km²)
At 5 psi overpressure, most residential buildings collapse, injuries are universal, fatalities are widespread. The chances of a fire starting in commercial and residential damage are high, and buildings so damaged are at high risk of spreading fire. Often used as a benchmark for **moderate** damage in cities.

Light blast damage radius (1 psi): 29 km (2,640 km²)
At around 1 psi overpressure, glass windows can be expected to break. This can cause many injuries in a surrounding population who comes to a window after seeing the flash of a nuclear explosion (which travels faster than the pressure wave). Often used as a benchmark for **light** damage in cities.



Ideia 2:

A superfície de ataque cresceu porque a complexidade dos sistemas cresceu



Ideia 2:

A IA chegou quando a superfície de ataque já era enorme e aumentou a velocidade com que conseguimos modificá-la.

BXSEC '26

“IA, crie uma API SpringBoot com JWT”

(O perigo do vibecoder)

BXSEC

Search

Home Comments Post Messages Notifications Lobs Download

Gabriel Silva
Junior Software Devel...
Age 24

Cednew

Share

Featured
Change Develojzs
Contacts
Entrise met

Comments (12)
Junior Software I
Permuatrce new hires
Drous rorem
Operators
Graces
Achievements

Gabriel Silva
Junior Software Developer
24 ·

Super excited to start my journey as a Junior Software Developer at Fictitious Dynamics today!

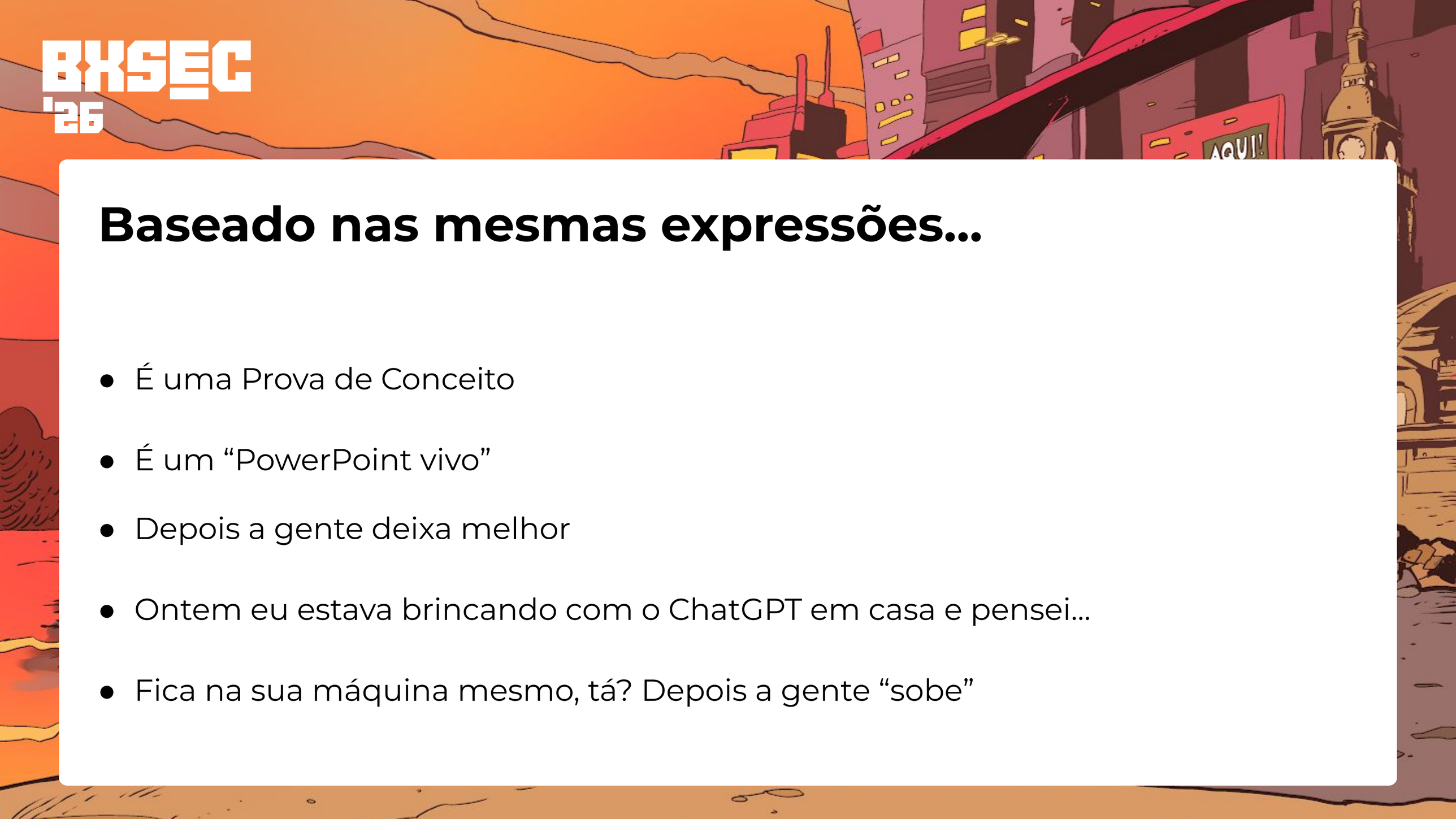
Ready to dive into the code!

#NewBeginnings #FirstDay #TechJobs

38 12 Comments

Likes Comment Shard

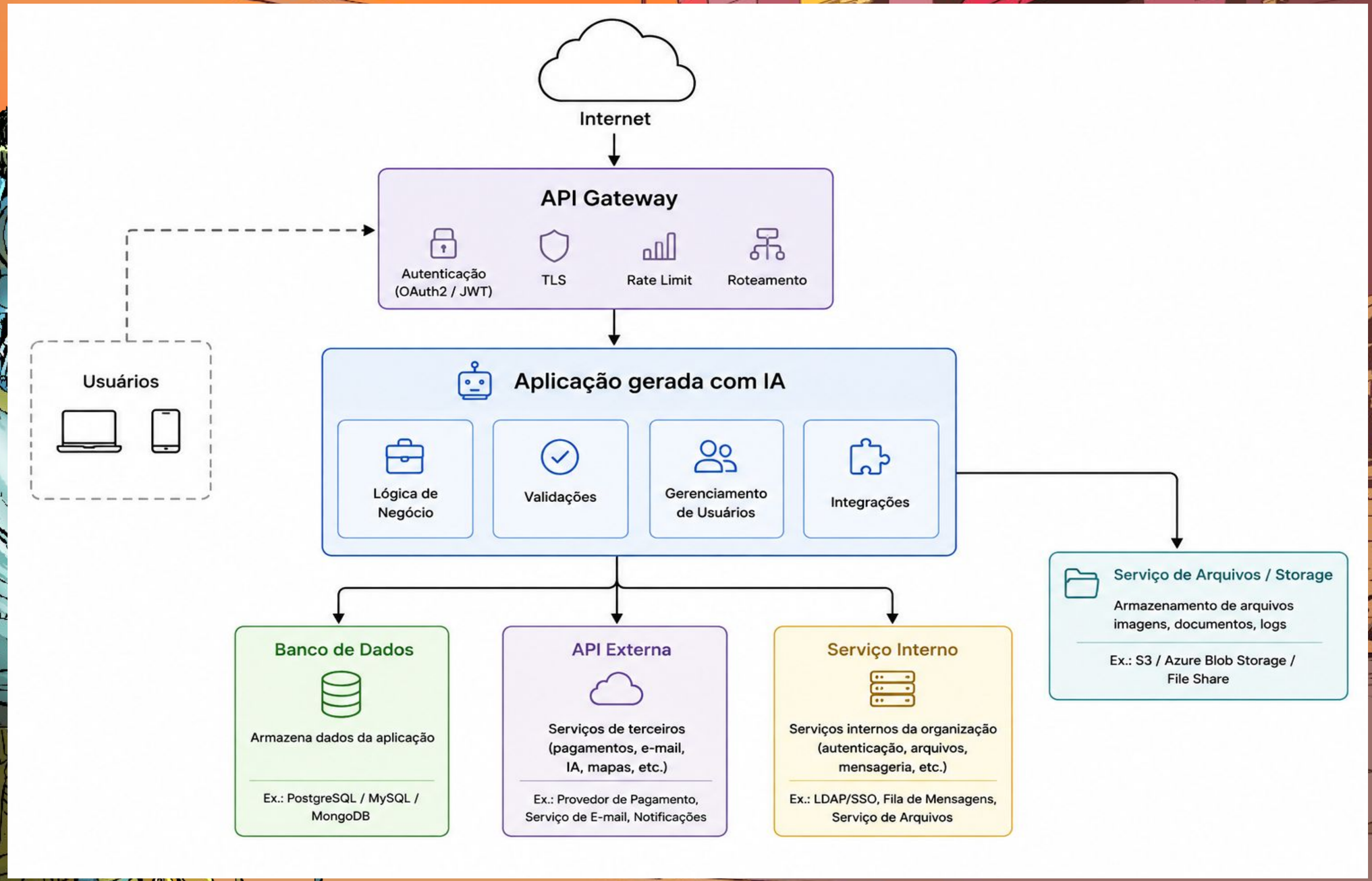
Parabéns, G · 2ad
Parabéns, Gabriel! Cuidado com o que mostra na tela 👀!
Like · 1 · Responder



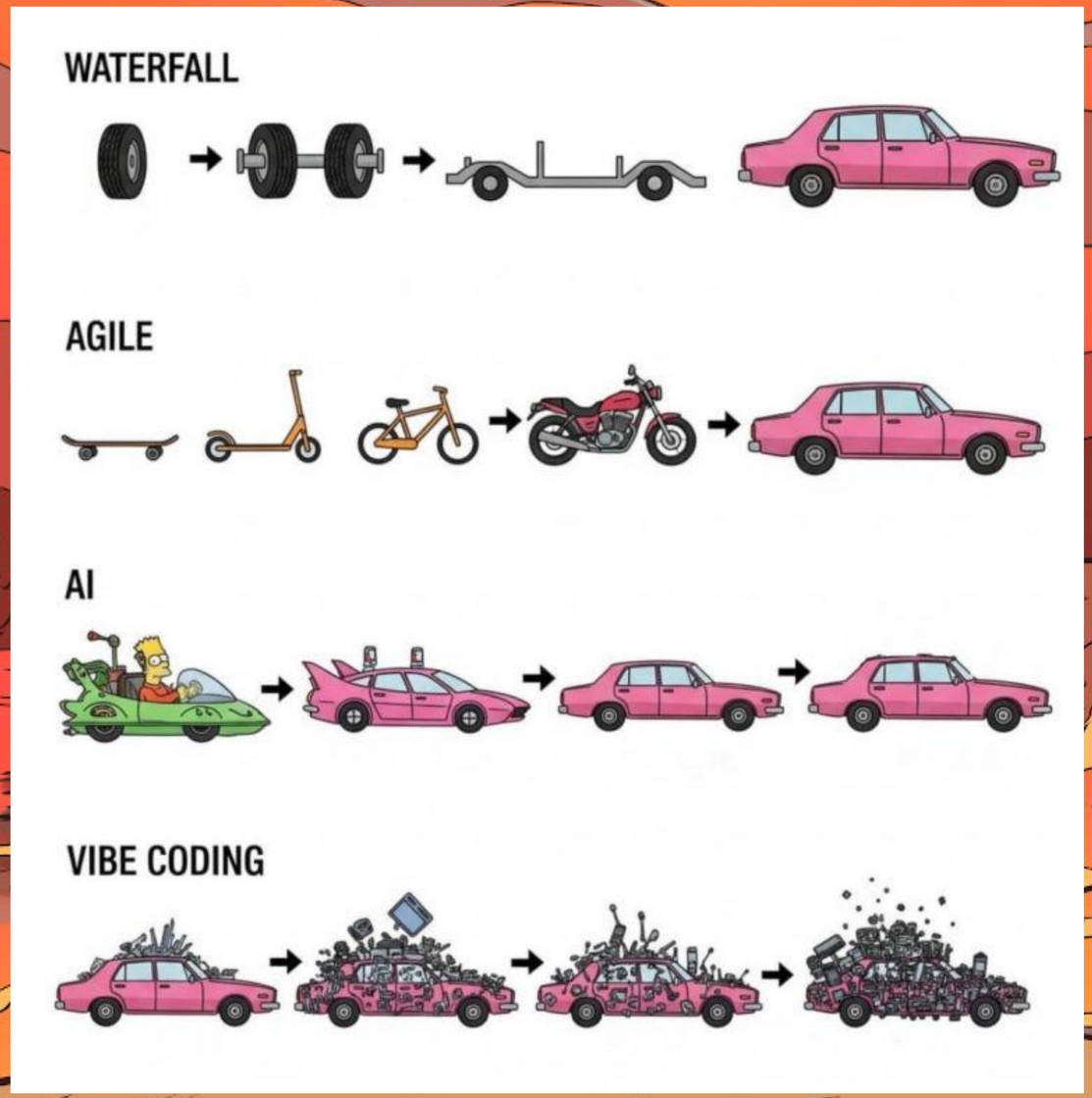
Baseado nas mesmas expressões...

- É uma Prova de Conceito
- É um “PowerPoint vivo”
- Depois a gente deixa melhor
- Ontem eu estava brincando com o ChatGPT em casa e pensei...
- Fica na sua máquina mesmo, tá? Depois a gente “sobe”

BXSEC '26


Internet
API Gateway
Autenticação (OAuth2 / JWT)
TLS
Rate Limit
Roteamento
Usuários
Aplicação gerada com IA
Lógica de Negócio
Validações
Gerenciamento de Usuários
Integrações
Serviço de Arquivos / Storage
Armazenamento de arquivos imagens, documentos, logs
Ex.: S3 / Azure Blob Storage / File Share
Banco de Dados
Armazena dados da aplicação
Ex.: PostgreSQL / MySQL / MongoDB
API Externa
Serviços de terceiros (pagamentos, e-mail, IA, mapas, etc.)
Ex.: Provedor de Pagamento, Serviço de E-mail, Notificações
Serviço Interno
Serviços internos da organização (autenticação, arquivos, mensageria, etc.)
Ex.: LDAP/SSO, Fila de Mensagens, Serviço de Arquivos

BXSEC '26



A prova de Conceito virou produto!

(em menos de 1 ~~semana~~ ~~dia~~ hora)

Um produto que...

... confia demais (ou de menos) no perímetro de rede

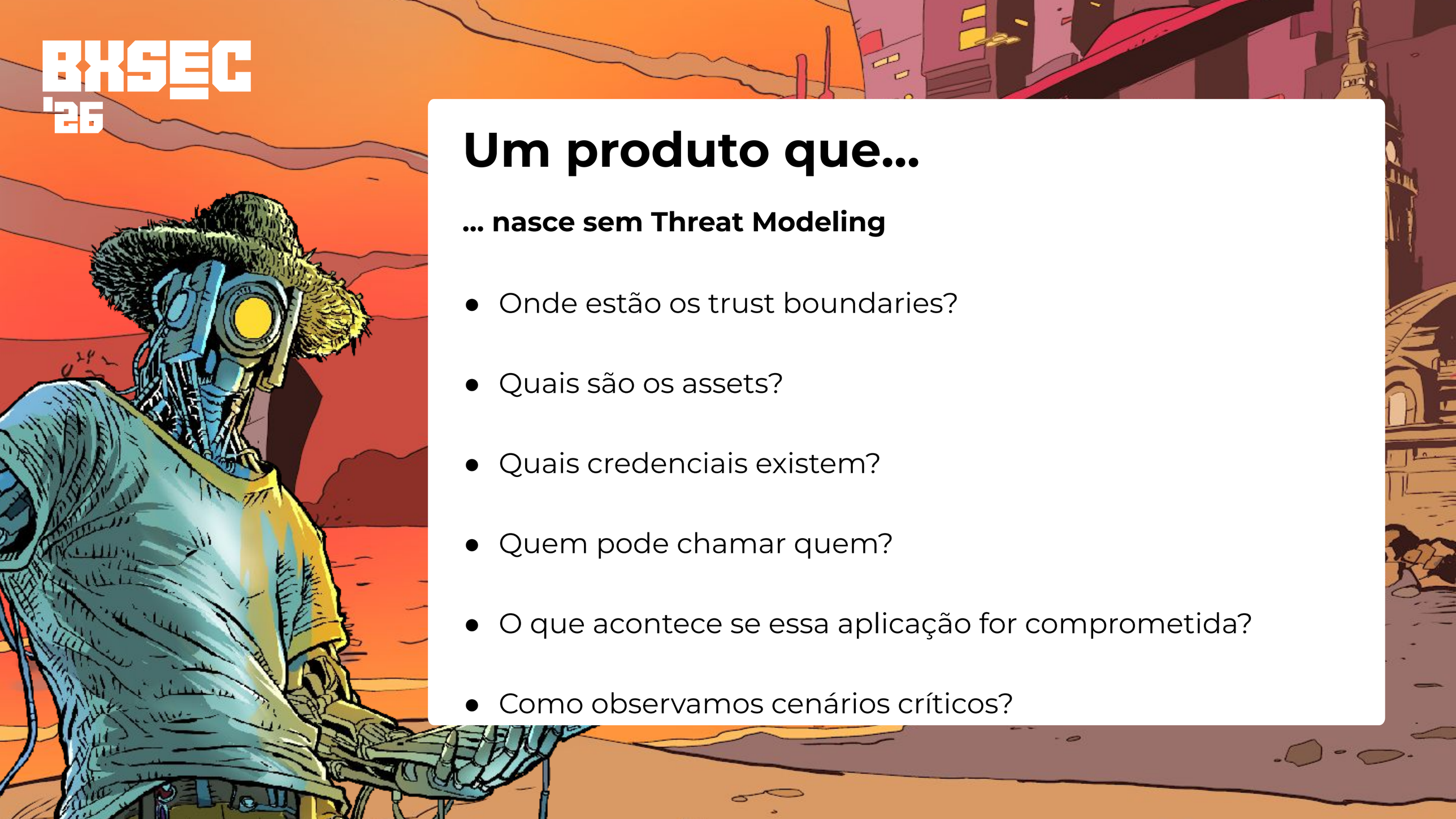
- Estou na rede local, estou seguro
- Estou na VPN, estou seguro
- “Essas informações não servem para nada, é uma POC apenas!”
- *Ninguém no mundo sabe que essa aplicação roda “aqui na minha máquina”**

mas acessa um bucket corporativo com um acesso mágico

Um produto que...

... possui “Governança desgovernada”

- A corporação não sabe que esse produto existe
- Aplicação sem guardrails mínimos de segurança
- Alto potencial destrutivo (Blast Radius)
- Lida incorretamente com Credenciais de Cloud Providers



Um produto que...

... nasce sem Threat Modeling

- Onde estão os trust boundaries?
- Quais são os assets?
- Quais credenciais existem?
- Quem pode chamar quem?
- O que acontece se essa aplicação for comprometida?
- Como observamos cenários críticos?

Um produto que...

... possui uma estratégia de entrega ruim

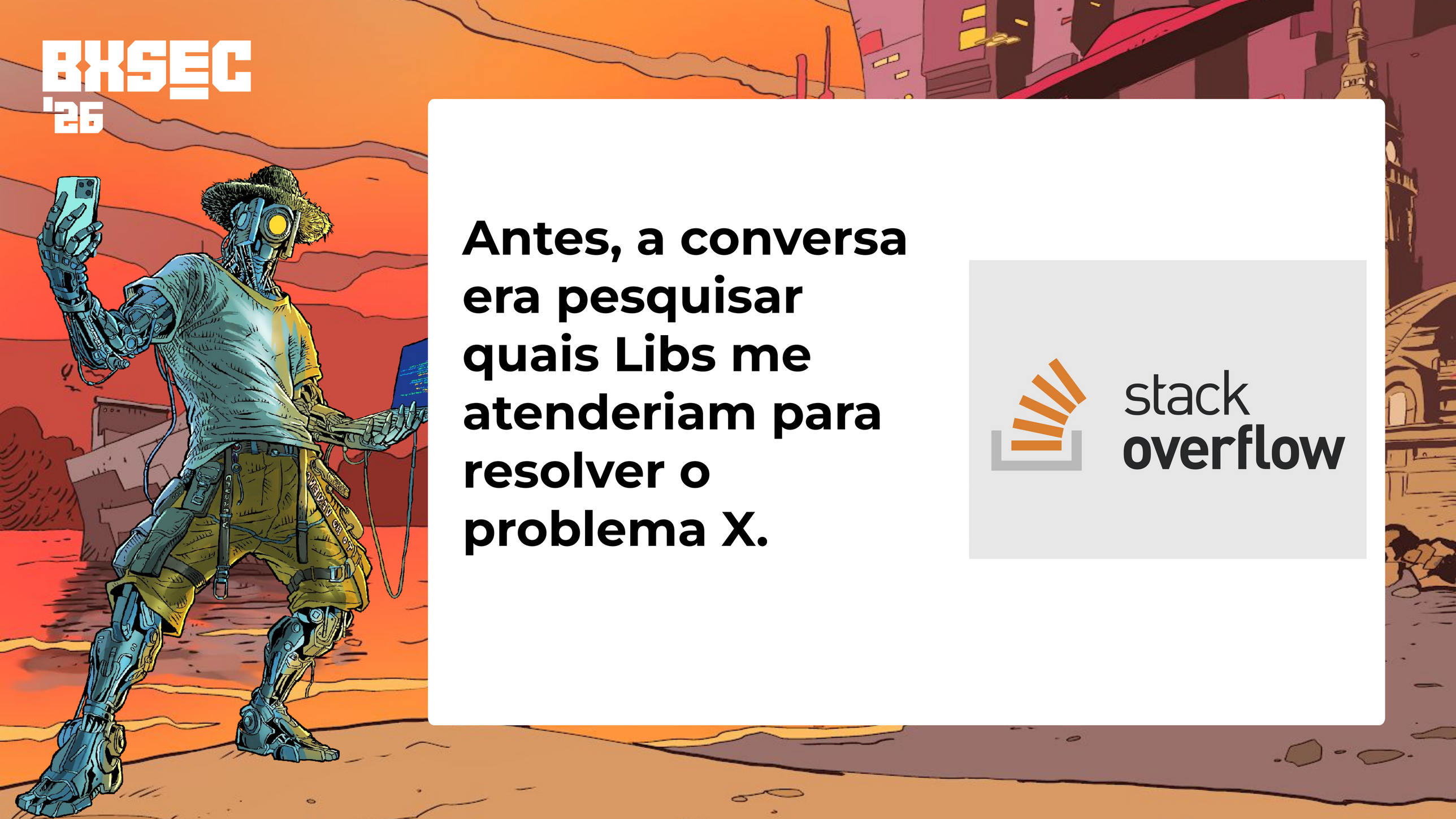
Lembra dos cavalos? Antes a gente já criava software baseado em relações de confiança.

```
$ npm init

$ pip install -r requirements.txt

$ docker-compose up -d
```

BXSEC '26



Antes, a conversa era pesquisar quais Libs me atenderiam para resolver o problema X.

stack overflow

BXSEC '26

“IA, crie uma API (no framework x) com JWT”

Hoje a IA pode criar uma aplicação com estratégia de deploy completa.

Mas foram feitas as perguntas corretas?

“IA, crie uma API (no framework x) com JWT”

Hoje a IA pode criar uma aplicação com estratégia de deploy completa.

Mas foram feitas as perguntas corretas?

Quantas decisões implícitas de segurança foram tomadas?

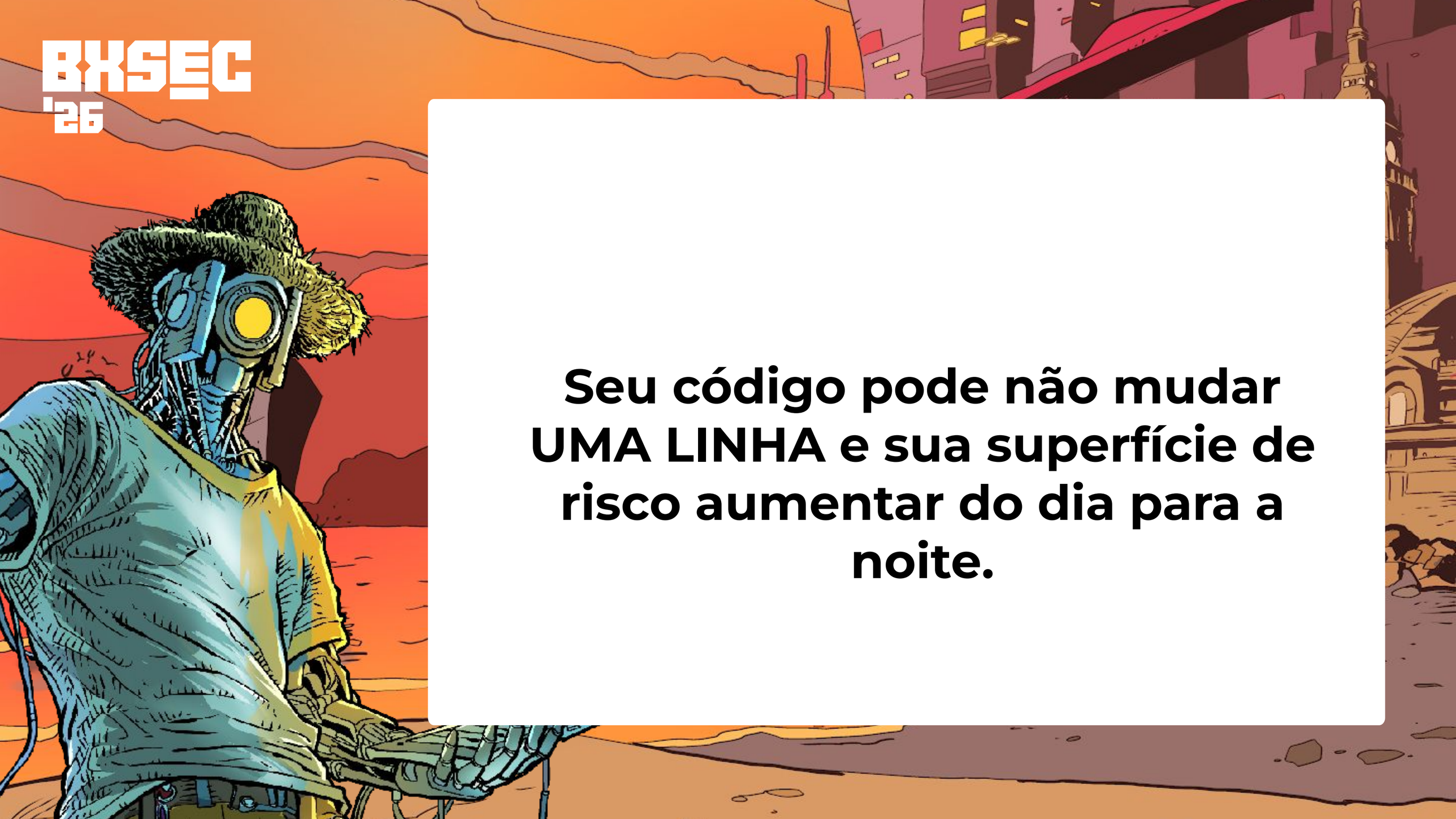
“IA, crie uma API (no framework x) com JWT”

Hoje a IA pode criar uma aplicação com estratégia de deploy completa.

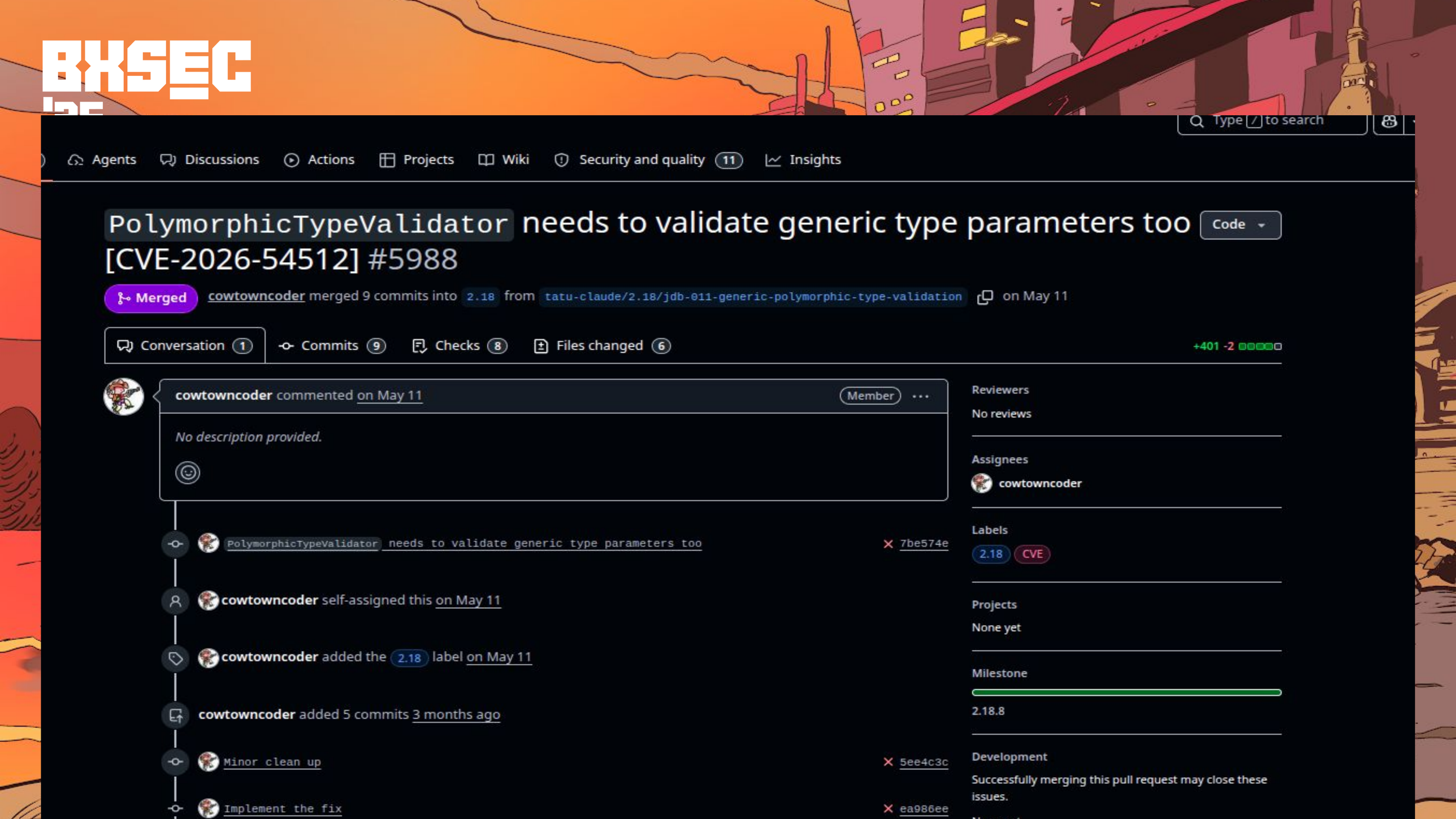
Mas foram feitas as perguntas corretas?

Quantas decisões implícitas de segurança foram tomadas?

Qual foi meu apetite em aumentar ou reduzir a fricção para adicionar componentes a cadeia?



Seu código pode não mudar UMA LINHA e sua superfície de risco aumentar do dia para a noite.



Agents Discussions Actions Projects Wiki Security and quality 11 Insights

`PolymorphicTypeValidator` needs to validate generic type parameters too [CVE-2026-54512] #5988

Code

Merged cowtowncoder merged 9 commits into `2.18` from `tatu-claude/2.18/jdb-011-generic-polymorphic-type-validation` on May 11

Conversation 1 Commits 9 Checks 8 Files changed 6 +401 -2

cowtowncoder commented on May 11 Member

No description provided.

`PolymorphicTypeValidator` needs to validate generic type parameters too 7be574e

cowtowncoder self-assigned this on May 11

cowtowncoder added the 2.18 label on May 11

cowtowncoder added 5 commits 3 months ago

Minor clean up 5ee4c3c

Implement the fix ea986ee

Reviewers
No reviews

Assignees
cowtowncoder

Labels
2.18 CVE

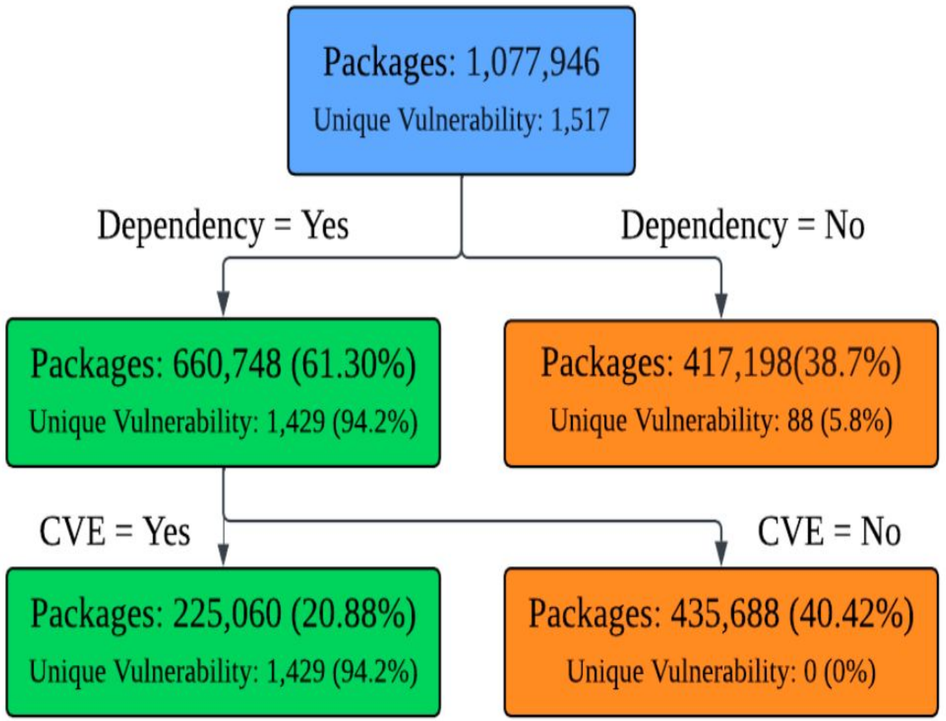
Projects
None yet

Milestone
2.18.8

Development
Successfully merging this pull request may close these issues.

oto dado interessante:

https://arxiv.org/abs/2604.17668



Original Sin of npm: A Study on Vulnerability Propagation in JavaScript Dependency Networks

Michael Robinson
Data61, CSIRO
Australia
michael.robinson@data61.csiro.au

Sajal Halder
Data61, CSIRO & RMIT University
Australia
sajal.halder@rmit.edu.au

Muhammad Ejaz Ahmed
Data61, CSIRO
Australia
ejaz.ahmed@data61.csiro.au

Muhammad Ikram
Macquarie University
Australia
muhammad.ikram@mq.edu.au

Seyit Camtepe
Data61, CSIRO
Australia
seyit.camtepe@data61.csiro.au

Hyoungshick Kim
Sungkyunkwan University
South Korea
hyoung@skku.edu

Abstract

In software development, the widespread adoption of open-source software has led to a proliferation of reusable modules and package libraries such as the Node Package Manager (npm). While this approach can significantly reduce development time, it also introduces the risk of vulnerability inheritance where a dependent *parent package* inherits vulnerabilities from a referenced *dependency package*. Understanding vulnerability propagation is essential for assessing how vulnerabilities spread across components of a software package. This supports more accurate impact analysis and enhances threat detection and mitigation. There is a need for frameworks that capture propagation across multiple layers and dynamic environments, along with comprehensive simulation tools that reflect real-world software systems. In this paper, we investigate how a small number of vulnerable JavaScript packages contribute to the creation of a disproportionately large number of vulnerable packages. This paper presents insights from 1,515 reported vulnerabilities gathered from a custom-built vulnerability database containing 1,077,946 JavaScript packages sourced from 'npm-follower' and their associated dependency networks. Dependency networks were constructed using the deps.dev API, with vulnerabilities identified by parsing package names and version numbers through the Google Open Source Vulnerability API.

Our findings reveal that 61.30% (660,748) of packages are reliant on one or more dependency packages, and 21.60% (232,836) of total packages have at least one known vulnerability throughout their dependency networks–of which most (42%) are of High severity. Moreover, we found that the number of vulnerabilities and affected packages on npm since 2015 has increased linearly–with R^2 values of 0.97 and 0.96, respectively. We also found that it takes, on average, approximately 4 years and 11 months to fix a vulnerable package from when the first vulnerable version is published on npm – al-

25% of vulnerability cases and the top-23 most frequent accounting for 50%. Based on these findings, we propose recommendations for developers and package managers to mitigate the threat and occurrence of vulnerabilities within the npm dependency network and the broader software repository community.

CCS Concepts

• **Security and privacy → Software and application security**.

Keywords

npm, JavaScript, software dependencies, vulnerability propagation

ACM Reference Format:
Michael Robinson, Sajal Halder, Muhammad Ejaz Ahmed, Muhammad Ikram, Seyit Camtepe, and Hyoungshick Kim. 2026. Original Sin of npm: A Study on Vulnerability Propagation in JavaScript Dependency Networks. In *ACM Asia Conference on Computer and Communications Security (ASIA CCS '26), June 1–5, 2026, Bangalore, India.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3779208.3785388

1 Introduction

Software repositories serve as essential components for managing both private and public software projects. Private repositories are accessible and modifiable only by authorized users, typically owned by individuals or companies. Public repositories, on the other hand, are open to all users. Within publicly accessible packages is open-source software (OSS), characterized by decentralized development that allows numerous developers to contribute to and modify the software. This collaborative approach offers significant benefits, such as accelerated development times and reduced costs for firms that would otherwise need to develop and maintain software in-house [16]. Accordingly, open-source development has

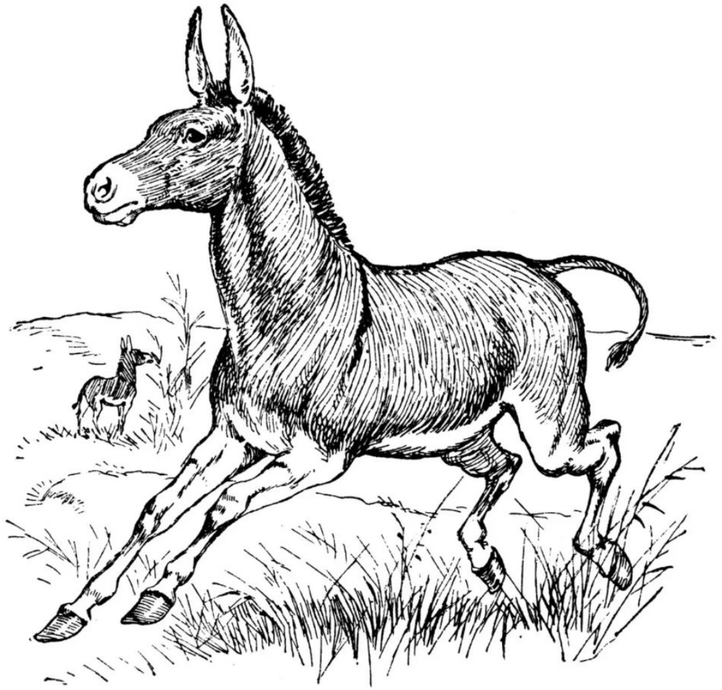
Toda dependência é também uma relação de confiança.

(Supply Chain lição 1)

BXSEC '26

PROMPT ENGINEERING NÃO É SOFTWARE ENGINEERING.

Living the extraortinary in the AI era

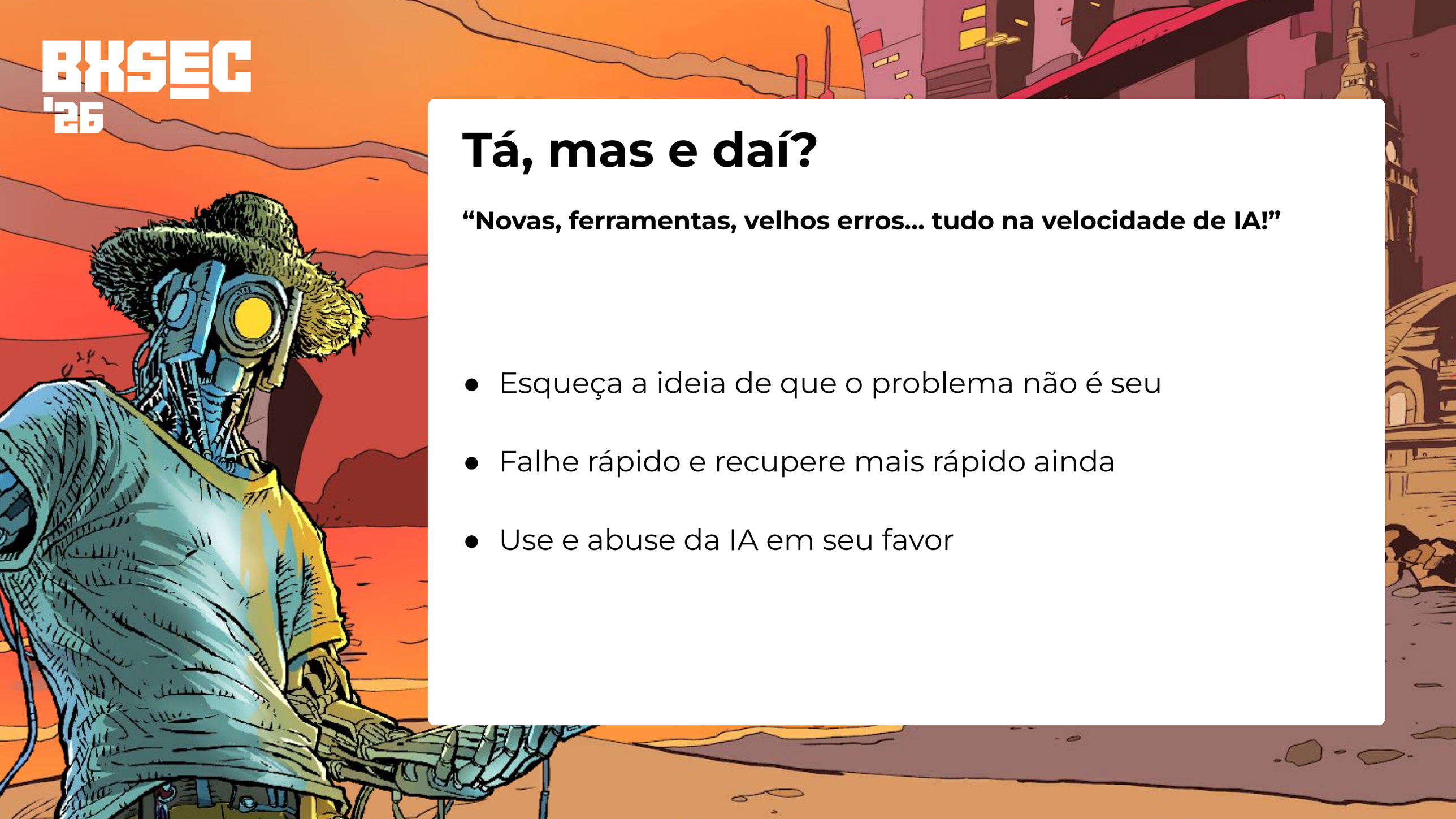


Coding with no fear of being hacked

because Shift Left and Threat Modeling are hype

Idéia 3:

A IA criou, mas quanto maior a velocidade de desenvolvimento, maior precisa ser a maturidade de engenharia.



Tá, mas e daí?

“Novas, ferramentas, velhos erros... tudo na velocidade de IA!”

- Esqueça a ideia de que o problema não é seu
- Falhe rápido e recupere mais rápido ainda
- Use e abuse da IA em seu favor

Tá, mas e daí?

“Novas, ferramentas, velhos erros... tudo na velocidade de IA!”

- Botão de Pânico
  - O Centro de Operação de Segurança sabe dessas toggles?
  - Ele consegue atuar na sua aplicação de forma proativa?

- Cybersecurity não é mais dever do DevSecOps ou do SOC, é de todas as pessoas que fazem parte do ciclo de entrega de software

Esse já é o grande diferencial mercadológico

BXSEC '26

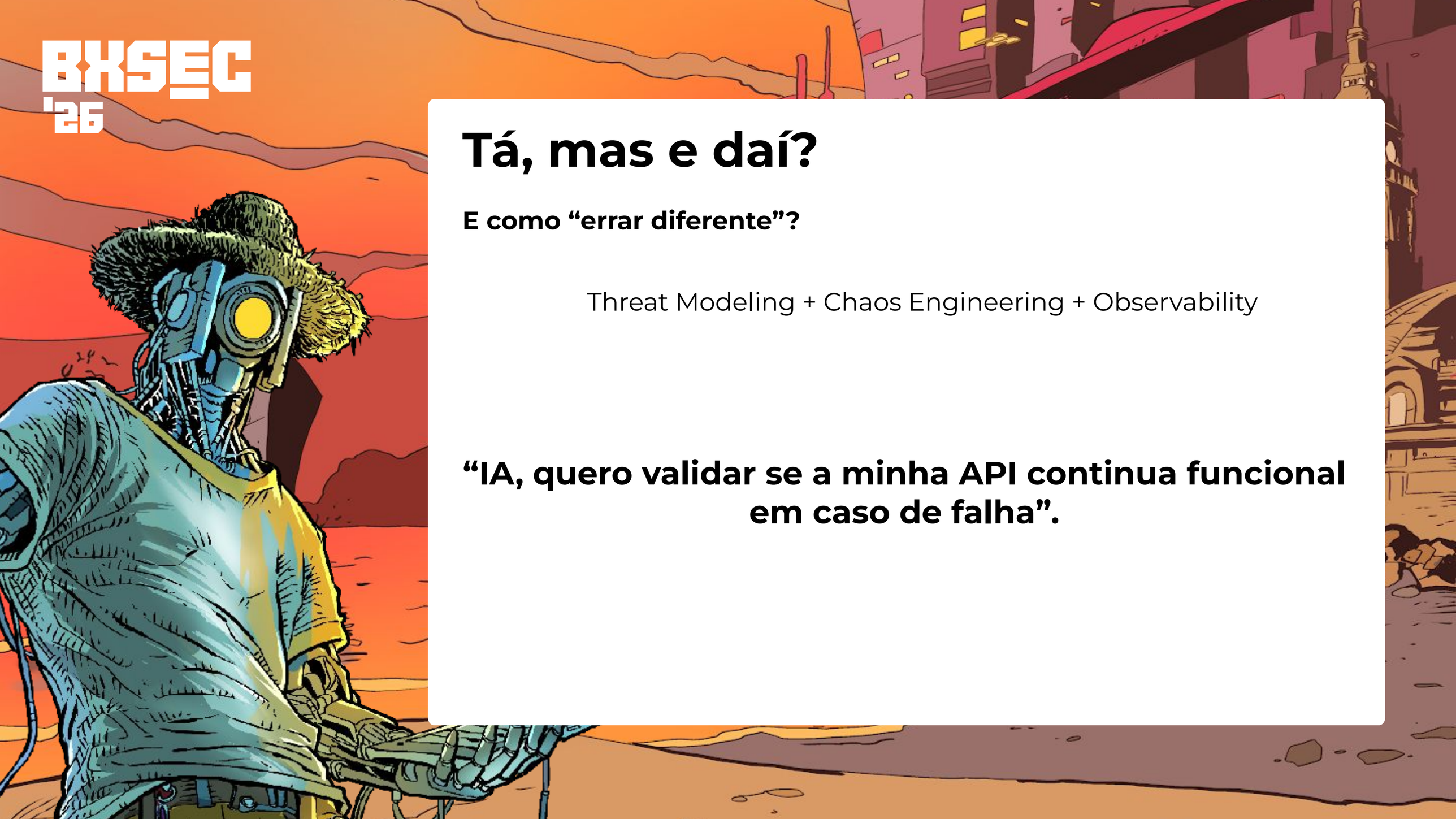
Tá, mas e daí?

“Novas, ferramentas, velhos erros... tudo na velocidade de IA!”

- Incidentes de segurança não ocorrem por ataques sofisticadissimos, mas por sistemas projetados de forma insegura

- Apesar da IA, nada mudou muito... a não ser:
  - A velocidade do ataque
  - A nossa velocidade em projetar sistemas de forma insegura

- A IA tornou código mais fácil de produzir. Não tornou boas decisões de engenharia mais fáceis de produzir.

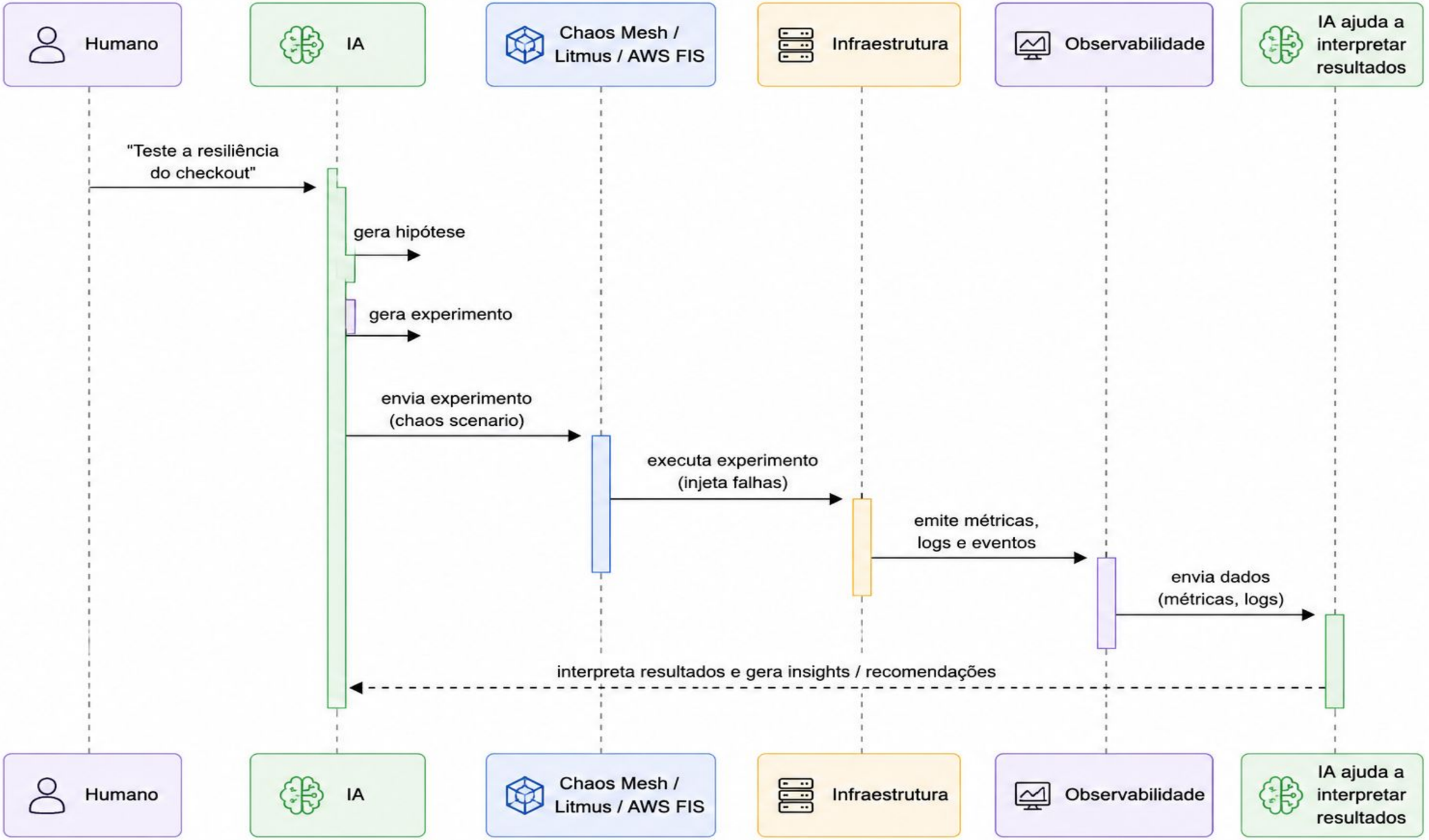


Tá, mas e daí?

E como “errar diferente”?

Threat Modeling + Chaos Engineering + Observability

“IA, quero validar se a minha API continua funcional em caso de falha”.


Humano
IA
Chaos Mesh / Litmus / AWS FIS
Infraestrutura
Observabilidade
IA ajuda a interpretar resultados
"Teste a resiliência do checkout"
gera hipótese
gera experimento
envia experimento (chaos scenario)
executa experimento (injeta falhas)
emite métricas, logs e eventos
envia dados (métricas, logs)
interpreta resultados e gera insights / recomendações
Humano
IA
Chaos Mesh / Litmus / AWS FIS
Infraestrutura
Observabilidade
IA ajuda a interpretar resultados

BXSEC '26

Como “errar diferente”?

Threat Modeling

O que pode dar errado?

(STRIDE, DREAD, PASTA ou tudo junto)

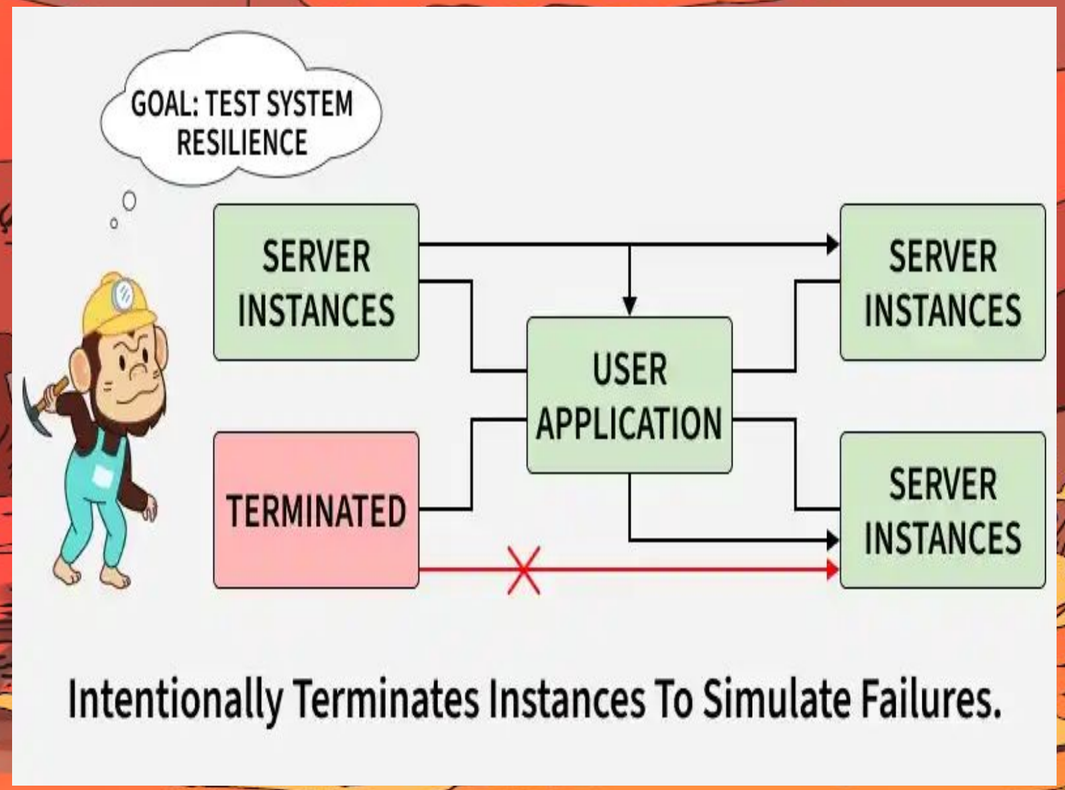
BXSEC '26

Como “errar diferente”?

Threat Modeling

Qual o risco para o negócio, qual o impacto e onde devemos gastar $$$?

** CISO sensato(a)*



Como “errar diferente”?

Chaos Engineering

Vamos “causar ordenadamente” para descobrir que estamos certos?

Como “errar diferente”?

Observabilidade

O que realmente aconteceu?

BXSEC '26

Como “errar diferente”?

Guardrail humano precisa existir!

Ela propõe onde você pode quebrar, mas não decide o apetite a risco!

BXSEC '26

Soberania Cognitiva

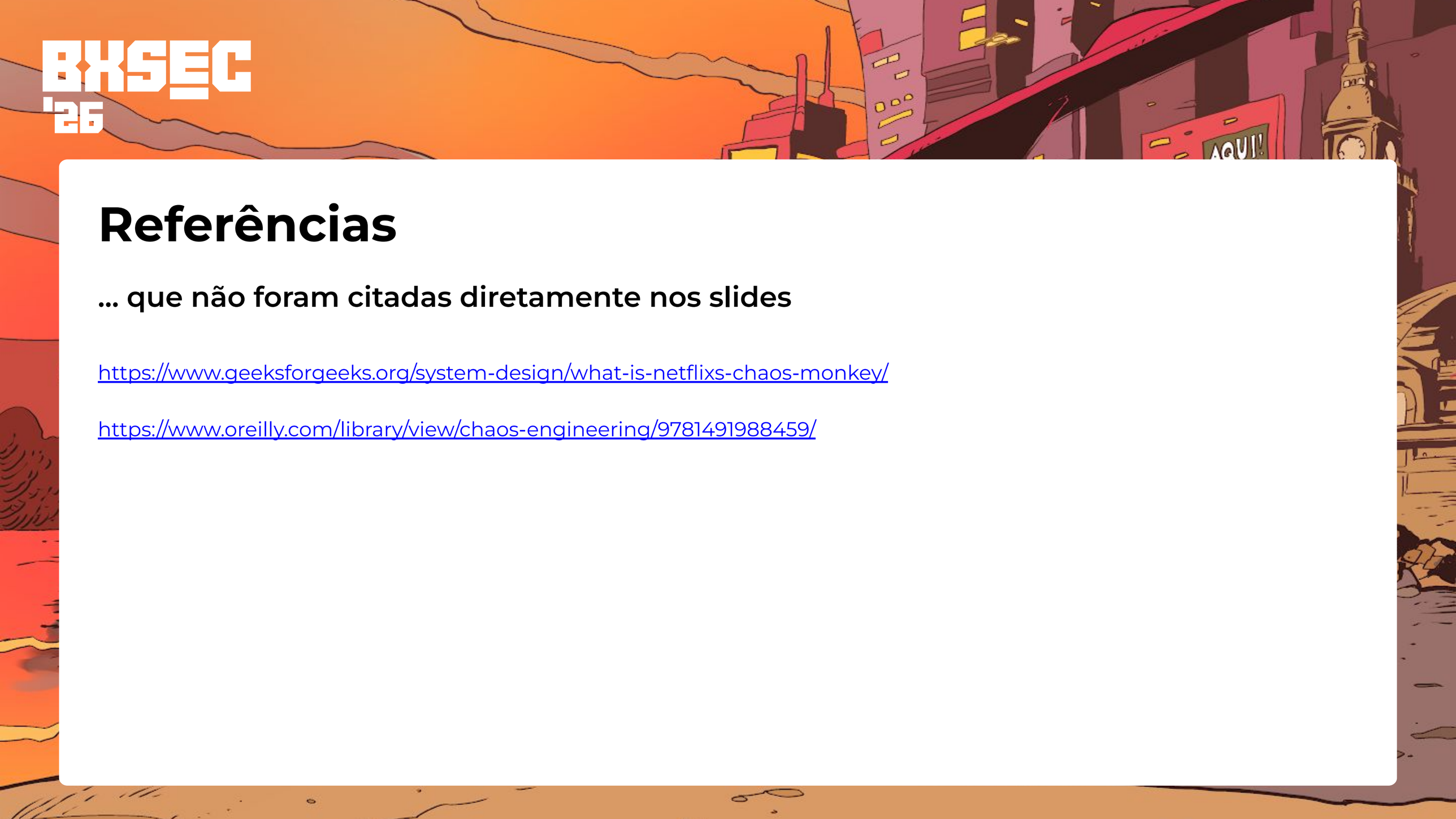
https://soberaniacognitiva.com.br/

"A soberania cognitiva representa um risco civilizatório, pois o que está em jogo é o próprio 'sapiens' do Homo sapiens: nossa capacidade de pensar por conta própria, sentir com profundidade e imaginar coletivamente o futuro."

Vanessa Mathias, da White Rabbit

BXSEC '26

Obrigado!



Referências

... que não foram citadas diretamente nos slides

https://www.geeksforgeeks.org/system-design/what-is-netflixs-chaos-monkey/

https://www.oreilly.com/library/view/chaos-engineering/9781491988459/