

<DevSum>

Building Practical Zero Trust APIs with .NET9 and Azure

Roy Cornelissen

active
SOLUTION
Cornerstone

Agria
Djurförsäkring

VONAGE
Part of Ericsson

**RaySearch
Laboratories**



Duende.

bulbul

SOFTRONIC

Polisen

Photo by [the blowup](#) on [Unsplash](#)

Hi, I'm Roy

Cloud Solution Architect at Xebia

roy.cornelissen@xebia.com

[@roycornelissen](https://twitter.com/roycornelissen)

[linkedin.com/in/roycornelissennl](https://www.linkedin.com/in/roycornelissennl)

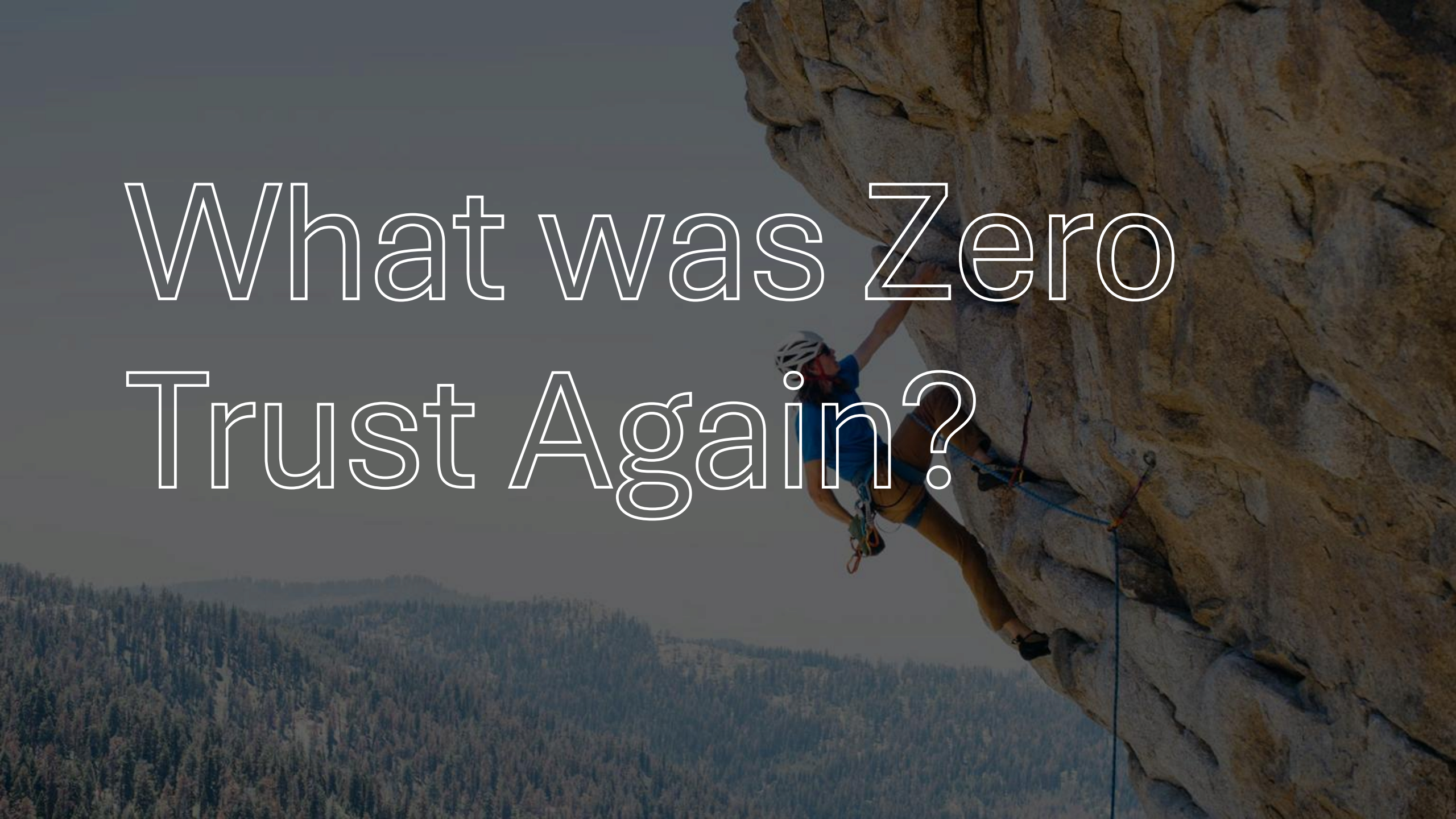
The background is a 10x10 grid of numbers from 1 to 100, arranged in rows of 10. Overlaid on this grid are numerous small, cylindrical wooden chips, each with a number from 1 to 100 printed on its top surface. The chips are scattered across the grid, with some numbers appearing multiple times. The entire image has a dark, semi-transparent overlay.

14,985,121,175

The background is a close-up, slightly blurred image of a Go board. It shows a grid of intersections with many Go stones placed on them. The stones are light-colored with red circular markings containing black numbers. The numbers visible include 16, 18, 25, 26, 35, 45, 75, 76, 86, and 98. The stones are arranged in a way that suggests a game in progress or a specific board position.

**Have I Been
Pwned**

What was Zero
Trust Again?



The concept

Zero Trust is a **security concept** based on the principle of "**never trust, always verify.**" It assumes that no component - whether hardware or software - should be implicitly trusted, regardless of whether it is inside or outside the network perimeter.

Every access request must be continuously authenticated, authorized, and validated to ensure security.



Stephen Marsh

April 1994



**Zero Trust, and the
fact that there is
always a Handsome
Prince...**





John Kindervag - Forrester Research

2010



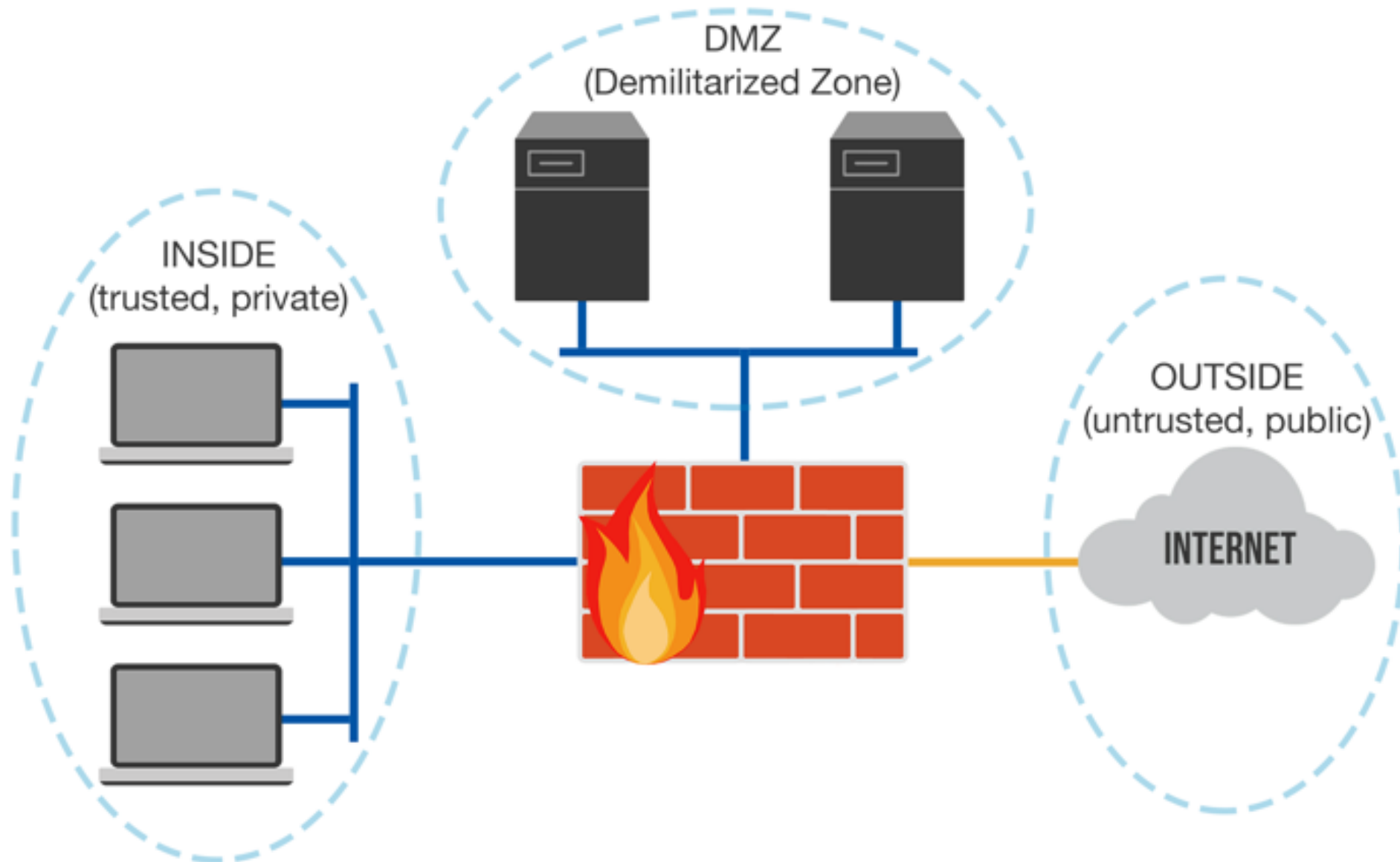
Zero Trust Architecture - NIST SP 800-207

2018

<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207.pdf>

Perimeterless Security

A stylized globe with a network overlay. The globe is rendered in a golden-brown, textured style, showing the continents of Africa and Asia. It is surrounded by a complex network of thin, glowing lines and dots, suggesting a global network or data flow. The background is a dark, deep blue with a subtle pattern of small, light-colored dots, resembling a starry sky or a digital space.





Zero Trust



Explicitly Verify

Zero Trust teaches us to never trust, and always verify



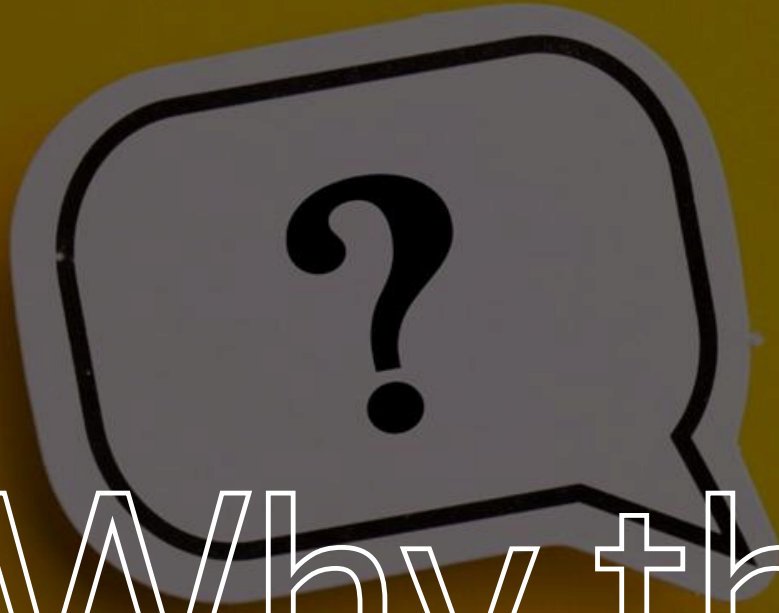
Limit User Access

Zero Trust uses the principle of least privilege access and limits users



Assume Breach

Zero Trust always assumes breach and verifies each request



Why this talk?



A man with curly hair and glasses, wearing a plaid shirt and tie, sits at a desk in a cluttered office. Behind him is a large server rack filled with equipment. The office is filled with various items like a desk lamp, a calendar, and a 'NO!' sign. The text 'IT Ops only goes so far' is overlaid in a large, white, outlined font.

IT Ops only goes so far

And there is more to secure development
than “use Authn and Authz”



Practical Use Case

Mini Bank: Payment handling... overly simplified

Few notes from our CTO

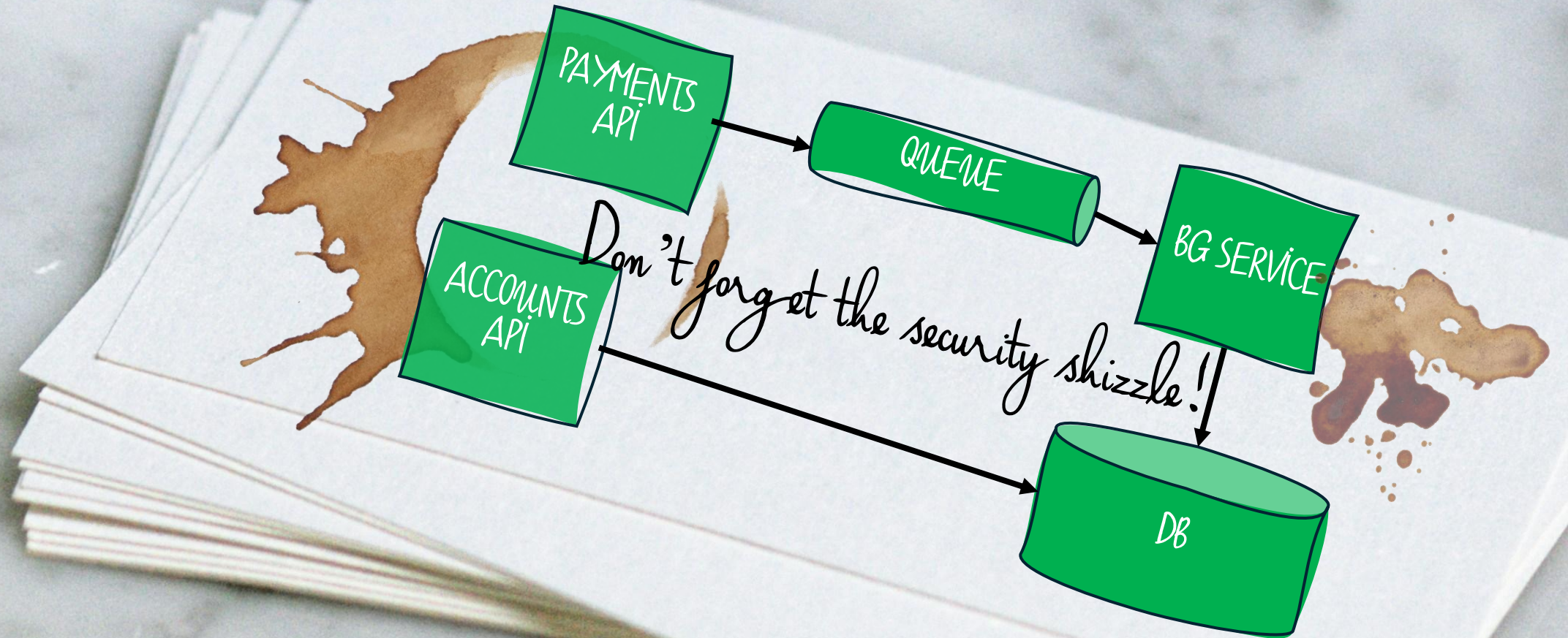
“I’m gonna need you to build this”

- Payments & Accounts API should be available
- Use .NET 9 with C# and host it in containers
- Azure Subscription is setup by Platform team
- There’s an agreed drawing somewhere on a paper napkin

Oh, and
make it
secure

That’d be
great,
mmmokay?

Agreed Drawing





Let's look at the
code for a second



Now let's make it
secure

Never Trust, Always Verify

never trust, always verify

Identity: hook up to Corporate OAuth

Data Classification: Ensure both input and output validation

Anomalies: Set bounds to our input data

Anomalies: Signal strange behavior



Use least privilege access limit impact and secure data

Just enough access: Only HTTP requests to the endpoints

Just enough access: Restrict the access to the queue and database

Just enough access: Just read specific keys from Vault

Just enough access: Avoid excessive requests

Out of band: Minimize your hosting surface

Assume breach prevent 'lateral movement'

Segmenting access: Isolate service in its own segment

Encrypt: Ensure TLS connections everywhere

Avoid: Unwanted outbound access

Avoid: Access to the host

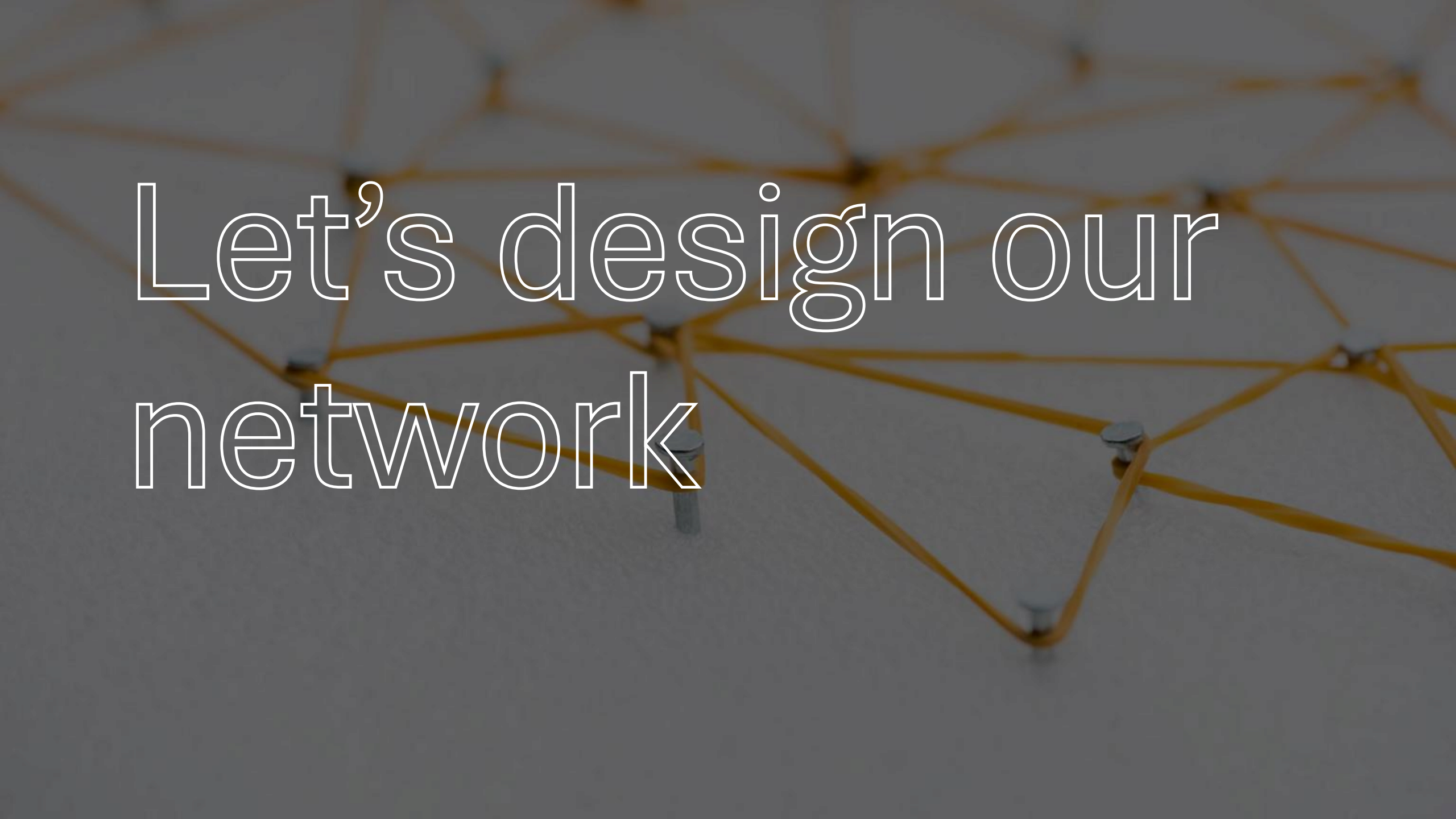
Use Analytics to spot anomalies: Setup threat detection



How big is your container?

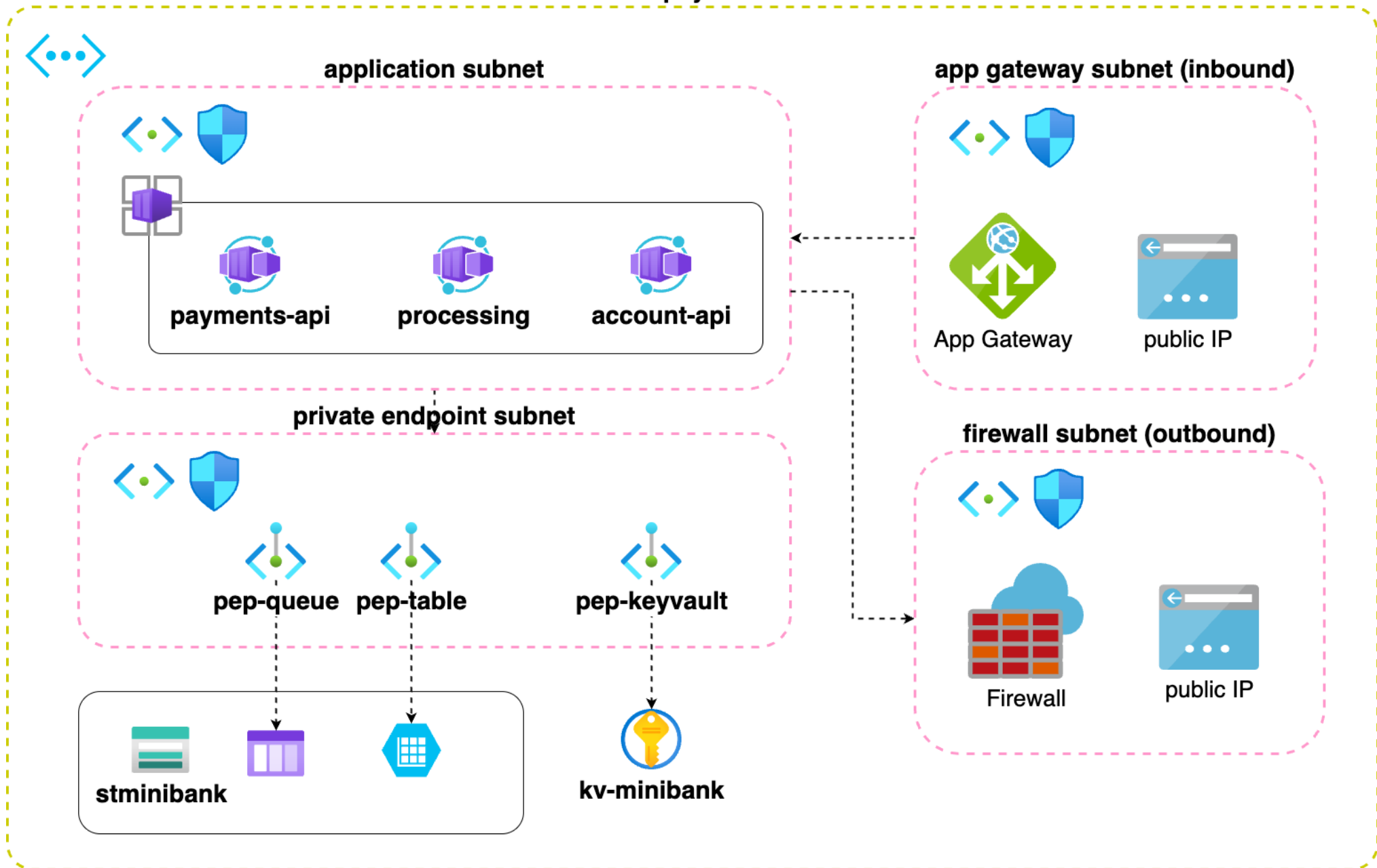
SBOOM

 CycloneDX

A network diagram consisting of several grey circular nodes connected by yellow lines. The nodes are arranged in a non-uniform pattern, with some having multiple connections. The background is a light grey color.

Let's design our
network

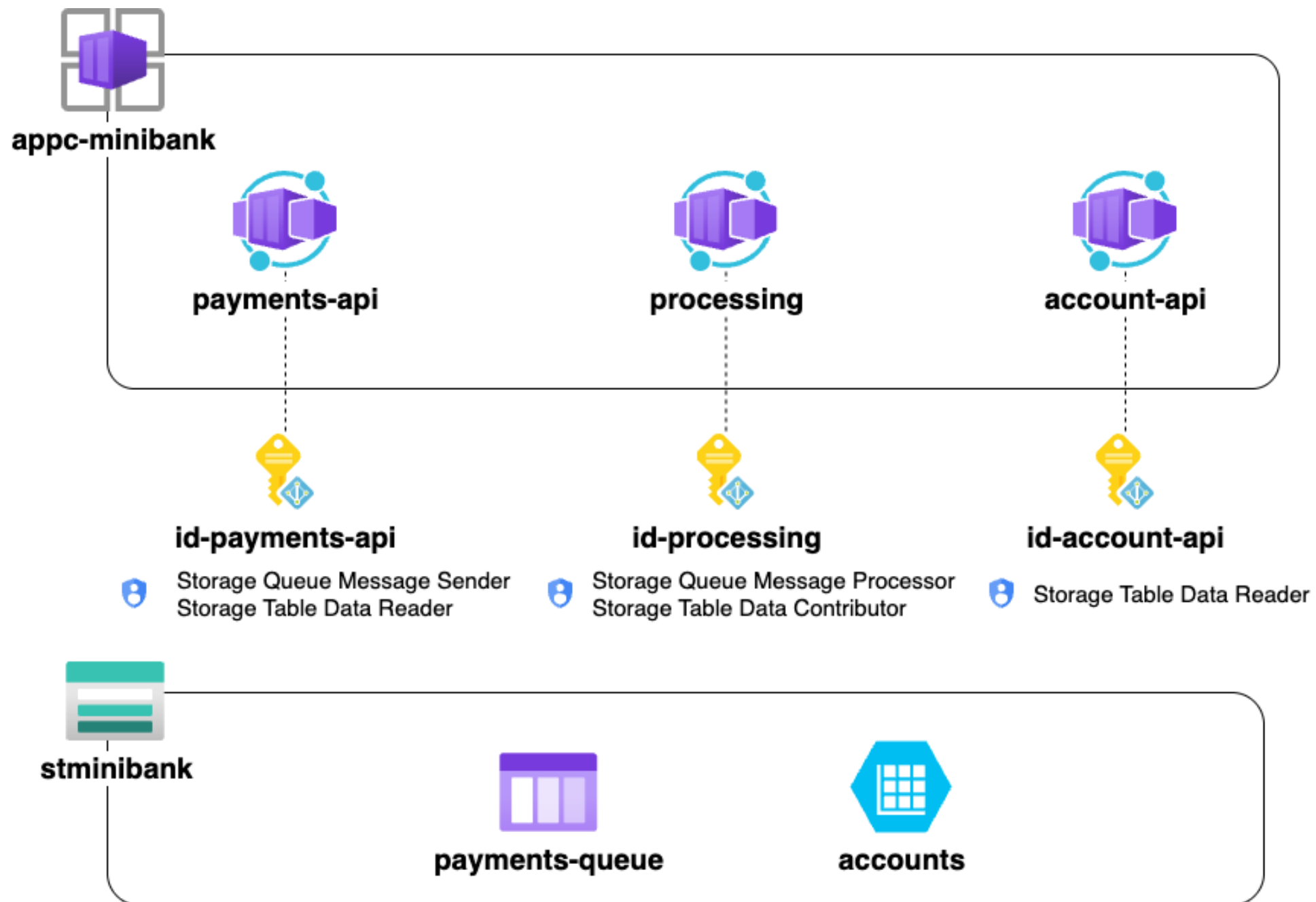
minibank payments vnet





IT Ops approves







How does that
look in the portal?



See it in action

Wrap Up: Zero Trust as a Developer

- **Verify explicitly:** always authenticate and authorize based on all available data points, including user identity, location, device health, service or workload, data classification, and anomalies. Verify both input and output.
- **Use least-privilege access:** limit user access with just-in-time and just-enough-access, risk-based adaptive policies, and data protection to help secure data and improve productivity.
- **Assume breach:** minimize attack surface in containers, verify end-to-end encryption and use analytics to gain visibility, detect threats, and improve defences.

A man with a beard and a light blue shirt, looking surprised with his hands outstretched. The image is overlaid with a semi-transparent dark red background.

Are we done?

There is always more to do

Thank you!

Roy Cornelissen

roy.cornelissen@xebia.com

[@roycornelissen](#)

[linkedin.com/in/roycornelissennl](https://www.linkedin.com/in/roycornelissennl)

Demo: github.com/roycornelissen/ztdemo

Photo by [Morvanic Lee](#) on [Unsplash](#)