Revolutionizing a bank

Introducing service mesh and a secure container platform in a brownfield organization

Robin van Zijll & Janna Brummel (@jannabrummel)
ING is a global financial organization, active in 41 countries. This talk is about the retail bank of NL with...
- 9 million debit cards
- 8 million retail customers
- 7 million ATM transactions/month
Mobile banking is used by 4.5 million customers. Together, they log in 6 million times a day (100+ TPS).
Working for a bank impacts our engineering

- ING aims to be a tech company with a banking license and engineering culture.
- Our technological landscape is diverse.
- IT risk and security are of extreme importance for both regulators and customers.
- Technological diversity and emphasis on IT risk have encouraged build-your-own solutions and a fear of public hosting
  ....until now
Questions we will answer in this talk

**WHY**
- develop a secure container platform?
- use public cloud, containers and a service mesh?

**WHAT**
- are our guiding principles in development?
- technologies are used in our platform?

**HOW**
- do we take care of IT Risk and Security?
- do we continue our work?
What problems do we aim to solve?

The future of banks is not as we know them today.

To make and keep our customers happy we need to be real-time, available and deliver great functionality fast.

Right now, we waste time not coding for customers. We lose time on proving we are in control with documents and maintaining full stacks.
To solve our problems, we started work on a container platform based on 3 enablers:

**Public Cloud** helps us to speed up development of our platform and to use resources efficiently.

**Containers** to facilitate platform independence, operational simplicity and to increase speed.

**Service Mesh** for ease of change, consistency across the fleet, improved observability and security and connectivity/network features.
Why did we choose the **public cloud** as part of container platform to solve our problems?

It’s **cost effective**: less infrastructure development and maintenance and we’d pay for actual usage, not for data centers.

It’s more **scalable**: easy to get more resources and return them.

It’s **highly available**: public clouds offer availability of at least 99.9%.

We can reuse **best practices**: proven technologies from industry are likely to be delivered as part of public cloud services. This saves us time and makes it easier to learn from others.
Why did we choose **containers** as part of container platform to solve our problems?

**Platform independence**: containers can run on any OS or infrastructure

**Less full-stack** engineering knowledge of engineers required, no maintaining of VMs

**More operational simplicity**: moving from pets to cattle

**Increased security, speed and scalability**: containers take less time to (re)start or die, easy to duplicate and are designed to be immutable
Why did we choose **service mesh** as part of container platform to solve our problems?

**Ease of change** without redeployment resulting in consistency across the fleet (in our case: 300 squads)

**Centralized control** combined with Site Reliability Engineering expertise helps us to improve observability across our application landscape

We can introduce and expand **traffic and network features**

**Engineers do not need to worry** about security on infrastructure level and will spend less time on IT Risk
Only when combining service mesh with public cloud, we will accelerate engineering productivity.
There are four steps that we need in our journey

1. CONTAINERIZE APPLICATION
2. PIPELINE FOR CONTAINER
3. RUN CONTAINER ON PLATFORM
4. DEFINE TEAM POLICIES
This work is spread over 4 teams: we are one of those

1. CONTAINERIZE APPLICATION
2. PIPELINE FOR CONTAINER
3. RUN CONTAINER ON PLATFORM
4. DEFINE TEAM POLICIES

5 ENGINEERS
1 PRODUCT OWNER
1 CHAPTER LEAD
We use these principles in our way of working

- We work with industry standards
- We work with open source products and practices
- We automate toil wherever and whenever we can
<table>
<thead>
<tr>
<th>OBSERVABILITY</th>
<th>Prometheus</th>
<th>Grafana</th>
<th>elastic stack</th>
<th>Grafana Loki</th>
<th>kiali</th>
<th>RIGOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPLICATION DEFINITION &amp; DEVELOPMENT</td>
<td>HELM</td>
<td>Azure Pipelines</td>
<td>docker</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ORCHESTRATION &amp; MANAGEMENT</td>
<td>Azure Kubernetes Service (AKS)</td>
<td>Istio</td>
<td>envoy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROVISIONING</td>
<td>Terraform</td>
<td>Azure Registry</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INFRASTRUCTURE</td>
<td>Microsoft Azure</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
With our platform, we can introduce 3 new benefits for teams to be more in control of IT Risk

**Advantage 1: Chaos Engineering**
- Availability will increase by observing systems during controlled experiments
- Observability tools and chaos engineering functionalities out of the box

**Advantage 2: Zero trust**
- By enabling microsegmentation, vulnerability will be limited to the segment
- Service mesh will restrict traffic to designated paths and all traffic is monitored

**Advantage 3: Everything pipeline**
- No access to running workloads will reduce risk and mitigating risk
- We enforce the use of pipeline for any action being done on any service
<table>
<thead>
<tr>
<th>Category</th>
<th>Tools/Logos</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OBSERVABILITY</strong></td>
<td>Prometheus, Grafana, elastic stack, Grafana loki, kiali, RIGOR</td>
</tr>
<tr>
<td><strong>APPLICATION DEFINITION &amp; DEVELOPMENT</strong></td>
<td>Helm, Azure Pipelines, docker</td>
</tr>
<tr>
<td><strong>ORCHESTRATION &amp; MANAGEMENT</strong></td>
<td>Azure Kubernetes Service (AKS), Istio, envoy, Project Calico</td>
</tr>
<tr>
<td><strong>PROVISIONING</strong></td>
<td>Terraform, Azure Registry, Twistlock</td>
</tr>
<tr>
<td><strong>INFRASTRUCTURE</strong></td>
<td>Microsoft Azure, ING</td>
</tr>
</tbody>
</table>
EVERYTHING PIPELINE

OBSERVABILITY
- Prometheus
- Grafana
- elastic stack
- Grafana loki

APPLICATION DEFINITION & DEVELOPMENT
- HELM
- Azure Pipelines
- docker

ORCHESTRATION & MANAGEMENT
- Azure Kubernetes Service (AKS)
- Istio
- envoy
- Project Calico

PROVISIONING
- Terraform
- Azure Registry
- Twistlock

INFRASTRUCTURE
- Microsoft Azure

ING
We foresee a hybrid environment combining local infra and public cloud

<table>
<thead>
<tr>
<th>Category</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>APIs, front ends and services developed in house</td>
<td>45%</td>
</tr>
<tr>
<td>COTS, DBs</td>
<td>35%</td>
</tr>
<tr>
<td>Mainframe or strategic apps we want to keep in the bank</td>
<td>20%</td>
</tr>
</tbody>
</table>

First target group for our platform

Second target group of our platform

we expect this border to shift in future

Likely to stay on local ING infra
What’s next?
Questions?

All icons used are from flaticon.com

Please rate our session 😊