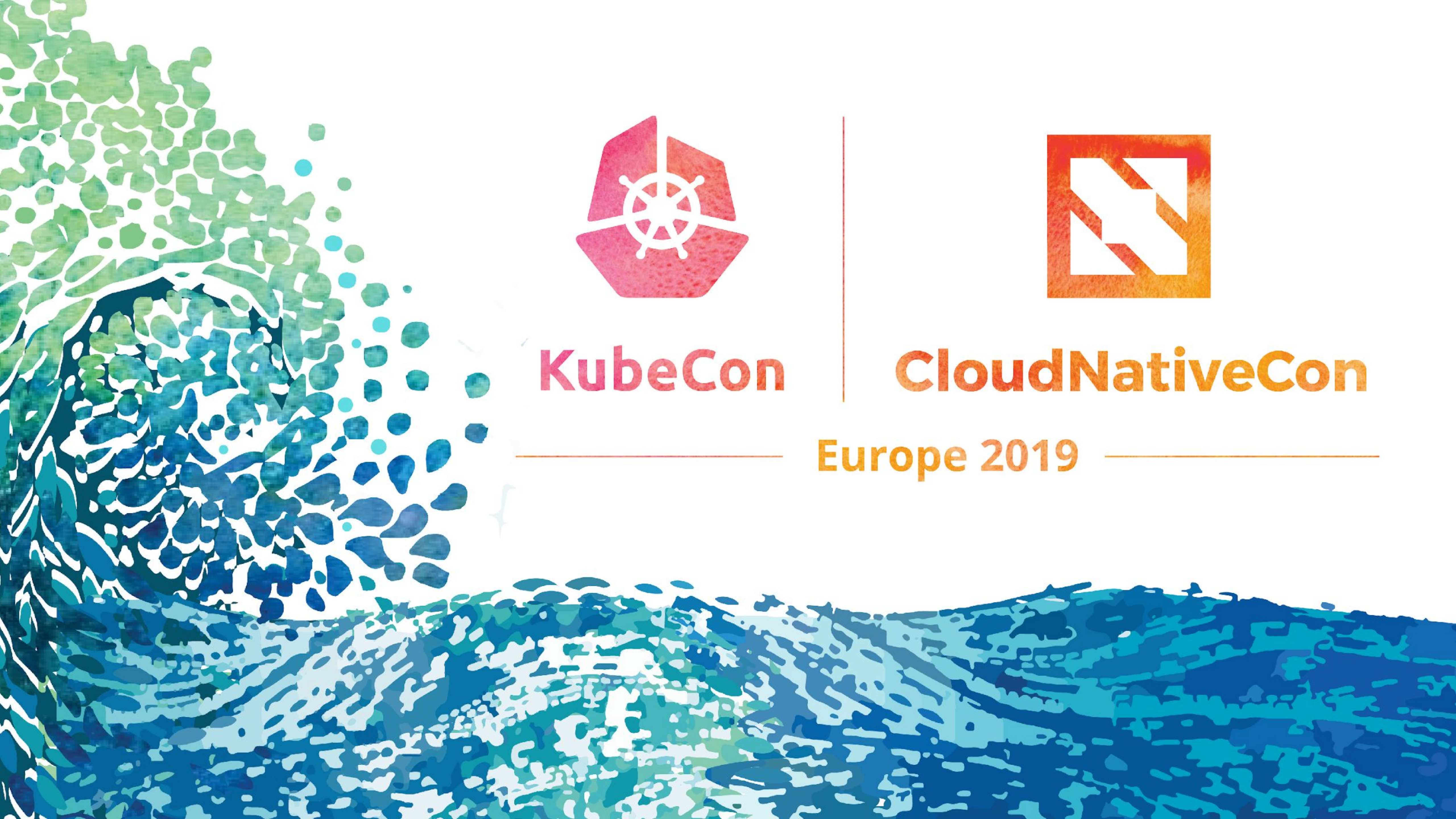


KubeCon | CloudNativeCon

Europe 2019

KubeCon | CloudNativeCon

Europe 2019

Caller ID in Kubernetes

Mike Danese

Intros

Mike Danese

- Software Engineer at Google working on GKE Security
- Chair and Tech Lead of Kubernetes SIG-Auth
- Tech Lead on GKE Identity
- Seattleite

Roadmap of this talk

1. State a (big) problem
2. Discuss the relevance of “authentication”
3. Explore the features of Kubernetes that assist in building authentication solutions
4. Explore Istio as an authentication solution that builds on Kubernetes
5. Identify the gaps

Objective

Provide value to customers!

The problem

No Bugs
+ No User Data
+ No Bad Actors
= No Problem!

The elephant

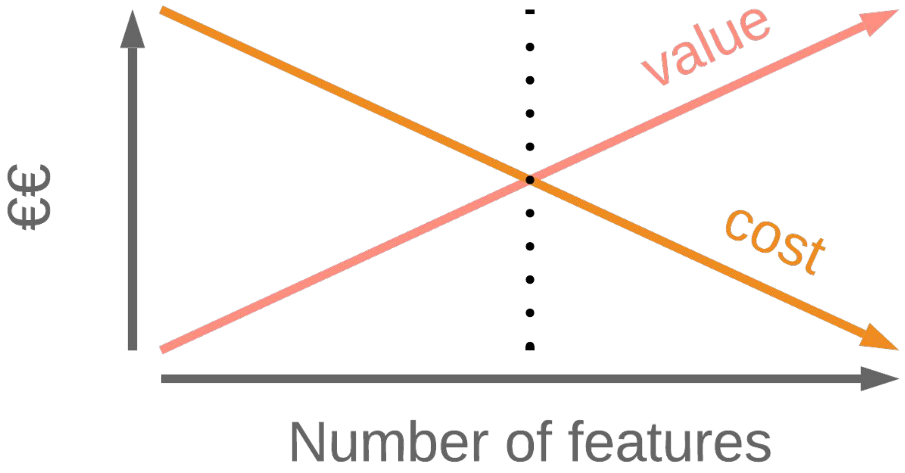
User Data

The problem

Reality Check

- Features make room for exploitable bugs
- Many valuable features require handling sensitive data

Exploit Economics



The problem

Reality Check

- Social Engineering
- Exploitable Bugs
- Supply Chain Compromises
- Insider Risk
- Physical Risk

The problem

Consequence of Failure

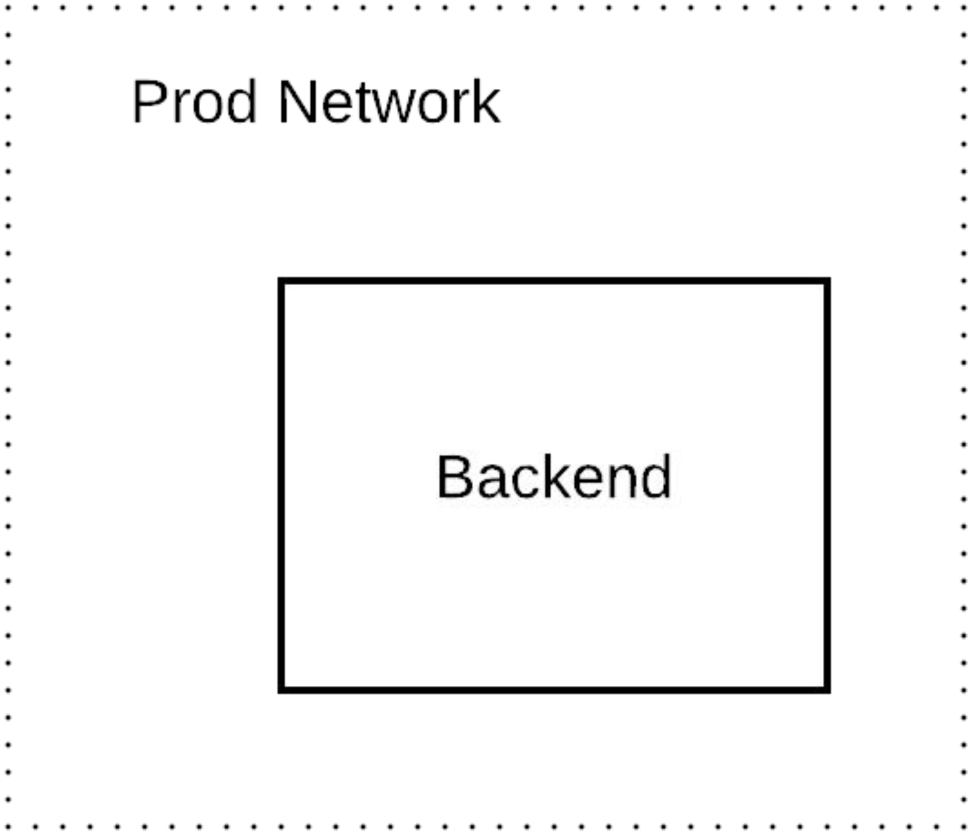
- Reputational harm
- Financial damage
- Legal liability

Failure hurts your ability to create value for customers.

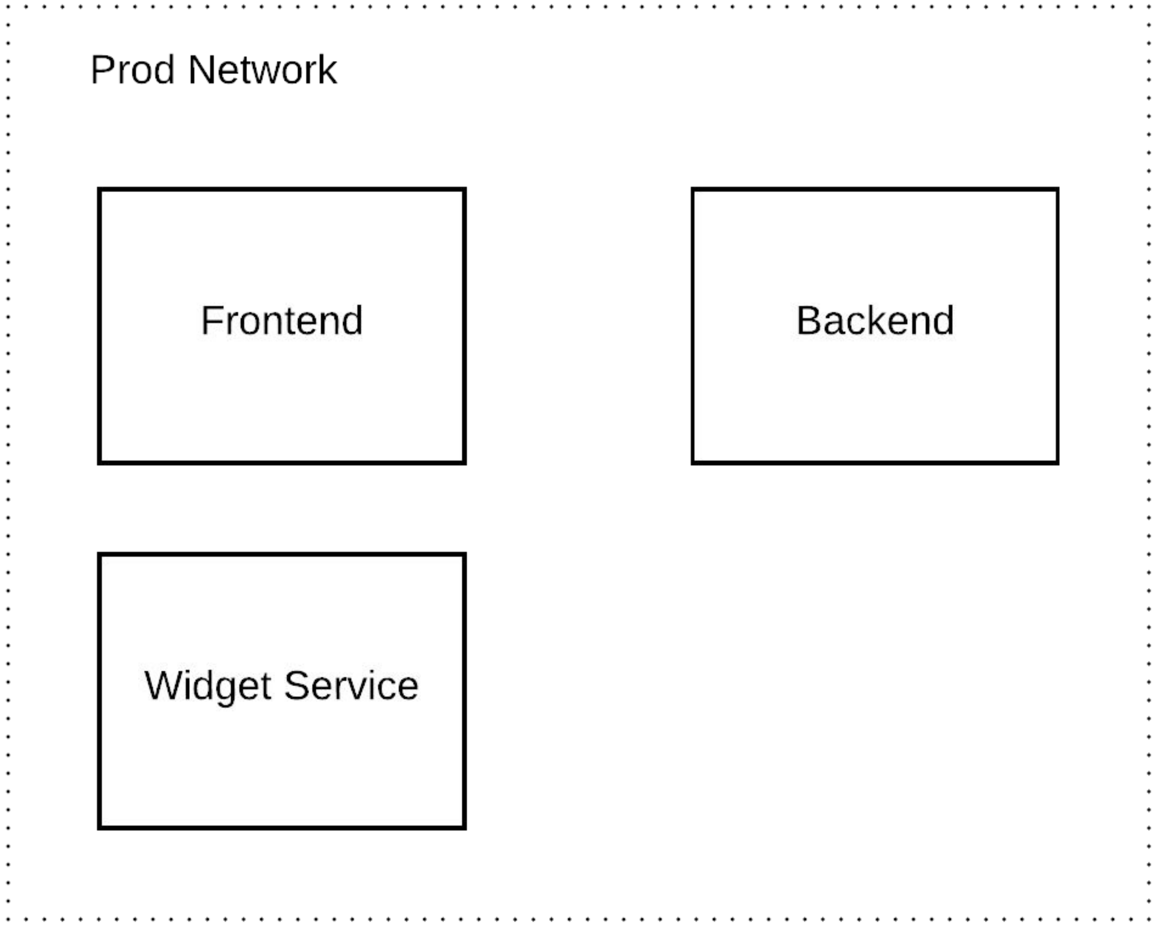
The problem

How do we create an environment that maintains a sufficiently high level of assurance on *user data*?

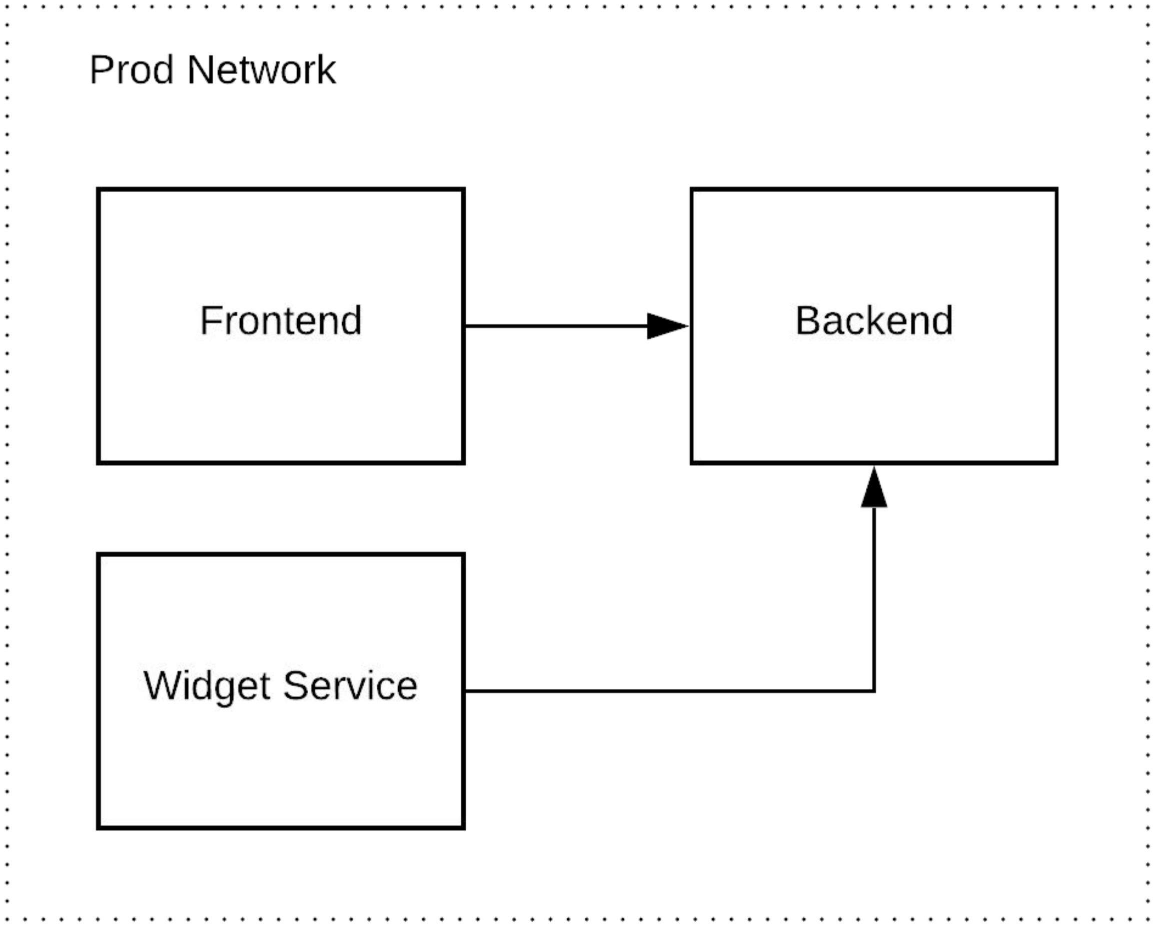
The game of telephone



The game of telephone



The game of telephone



Feature Creep

Grant access to data

To service Frontend

Kubernetes Service Accounts

Native Service Identity

- All pods run as a service account
- Defined access control

Kubernetes Service Accounts

Service Account Tokens

- Automatic distribution/rotation rooted in Kubelet trust
- Support for attenuation
  - Fast expiration
  - Audience binding

Service Account Tokens

Major Downside

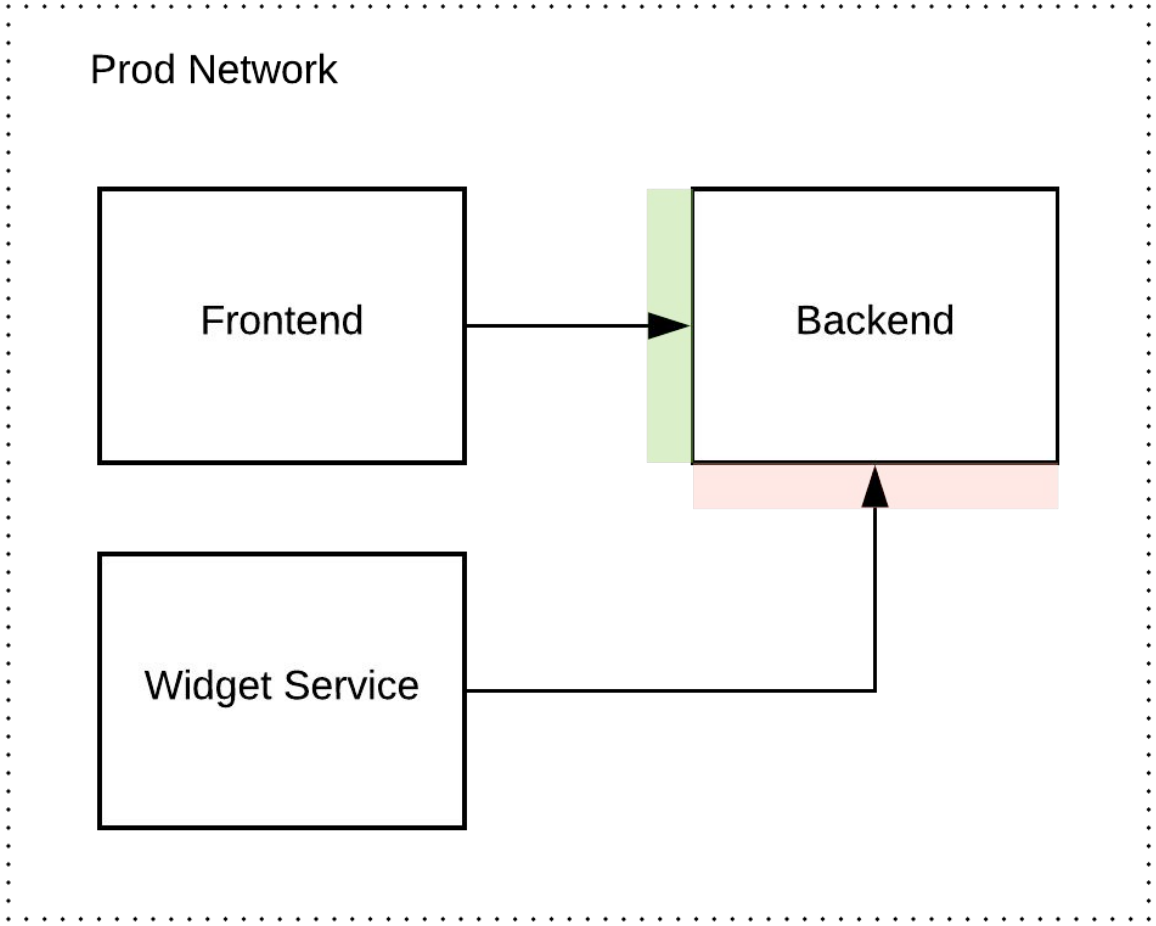
- Replayable
- Don’t solve server authentication

Certificates

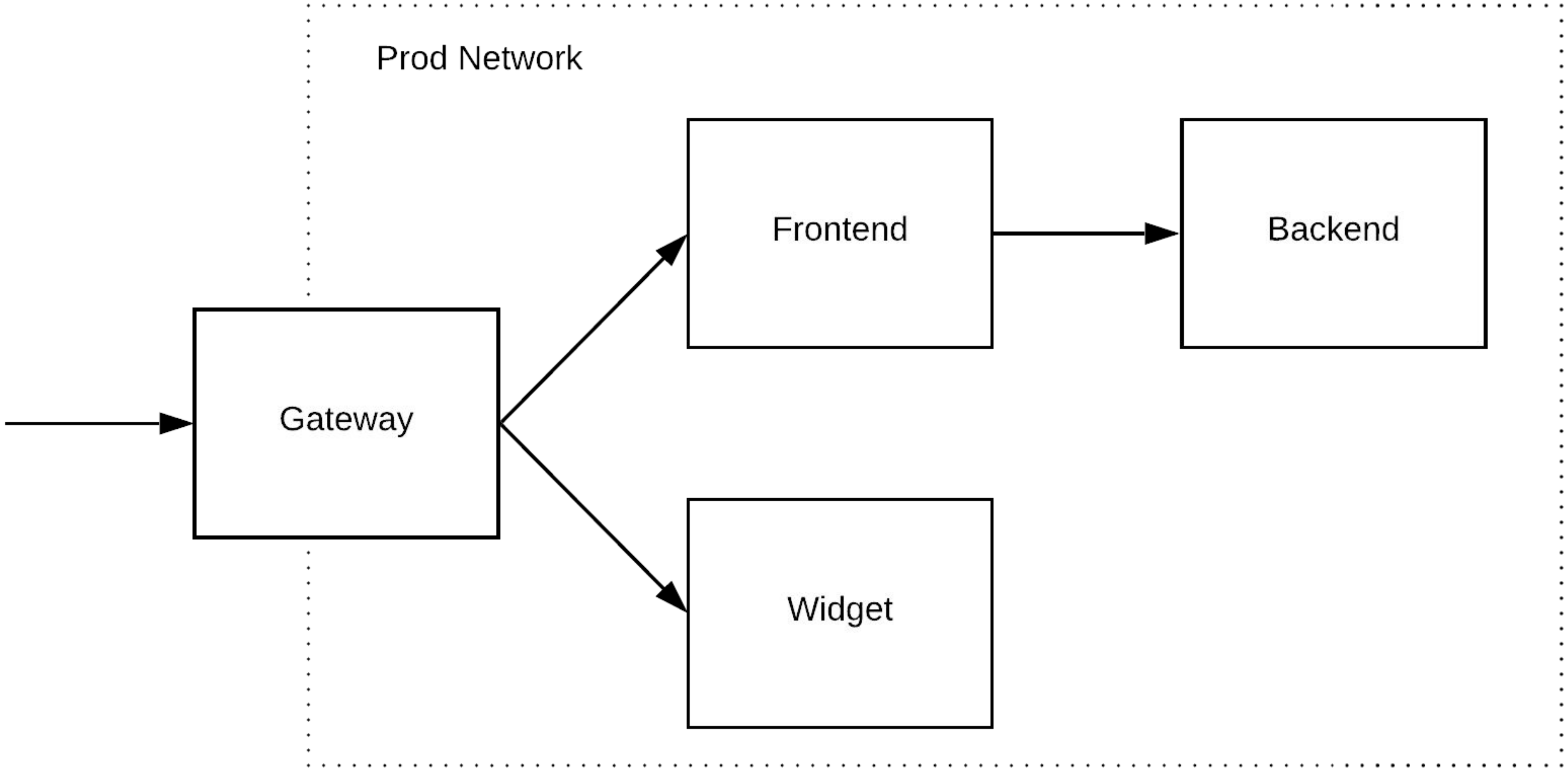
Mutual TLS

- Provides server authentication
- Channel bound
- Kubernetes Certificates API is flexible but requires some integration
- Istio does all the heavy lifting for you

The game of telephone



The game of telephone

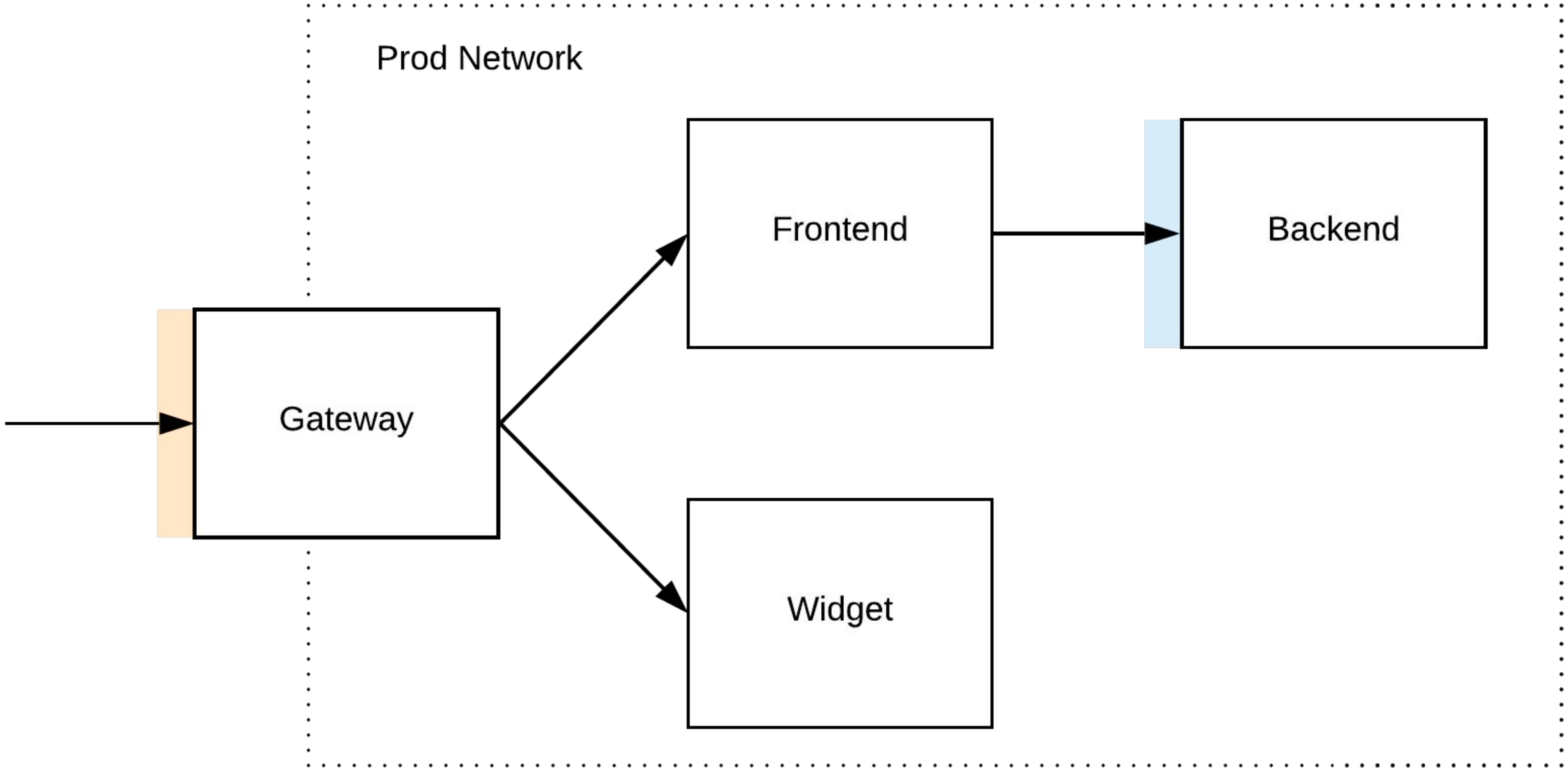


Feature Creep

Grant access to user data

With proof of user interaction

The game of telephone

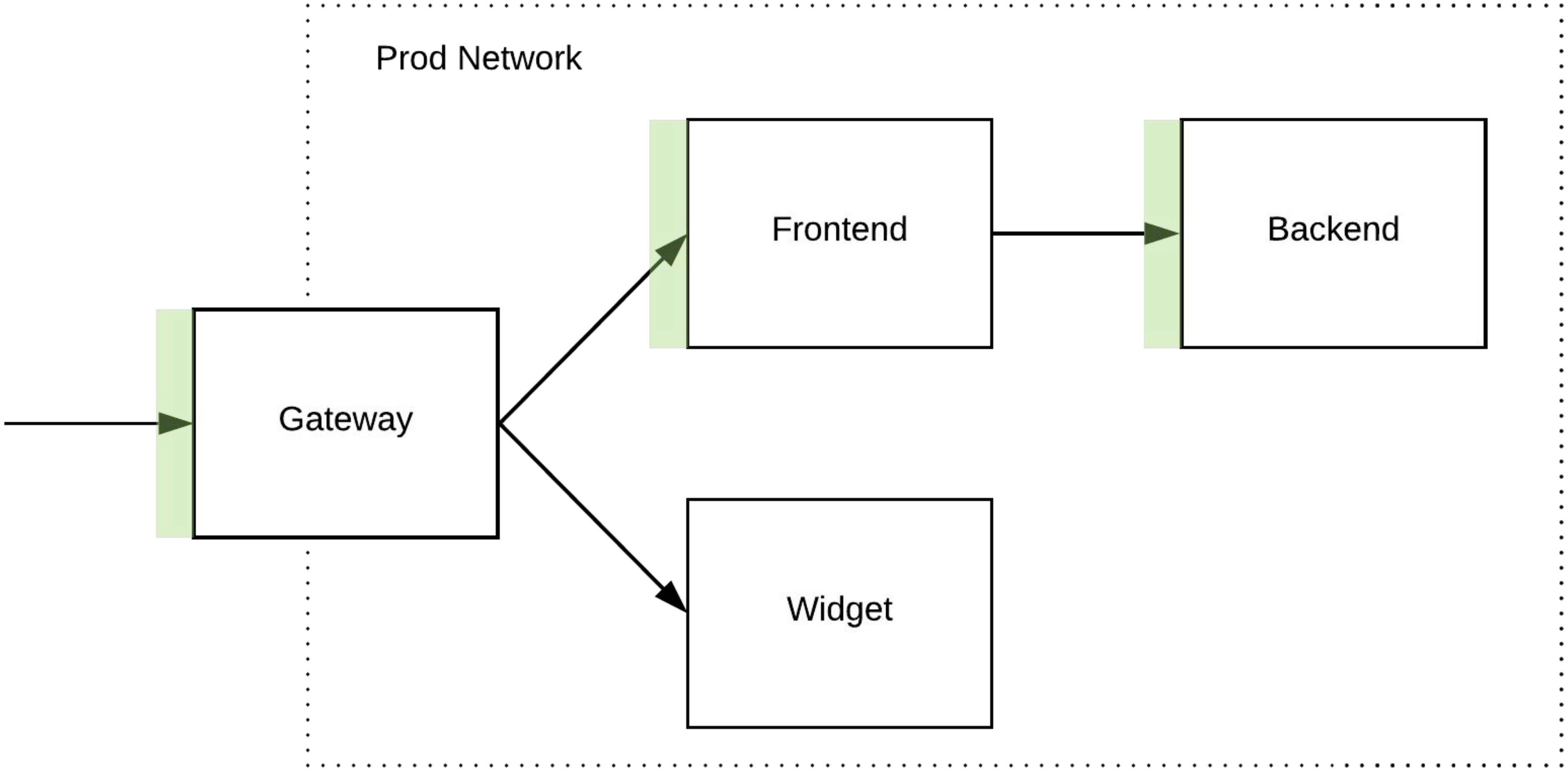


Istio RCToken

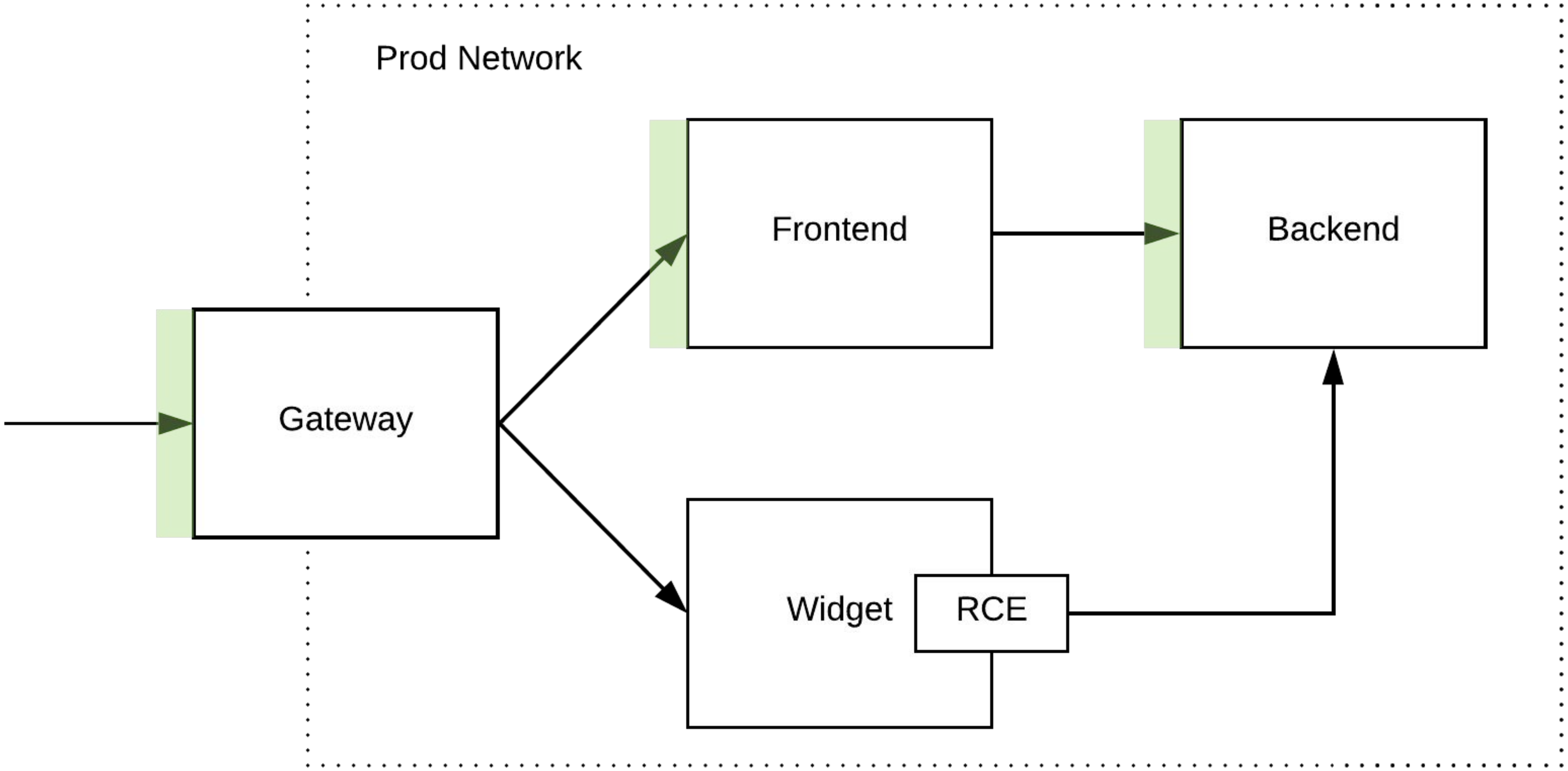
Captures context on ingress

- Supports
  - identity attributes like end user
  - general attributes like source IP
- Asserts attributes in a package that can be validated anywhere in the mesh

The game of telephone



The game of telephone

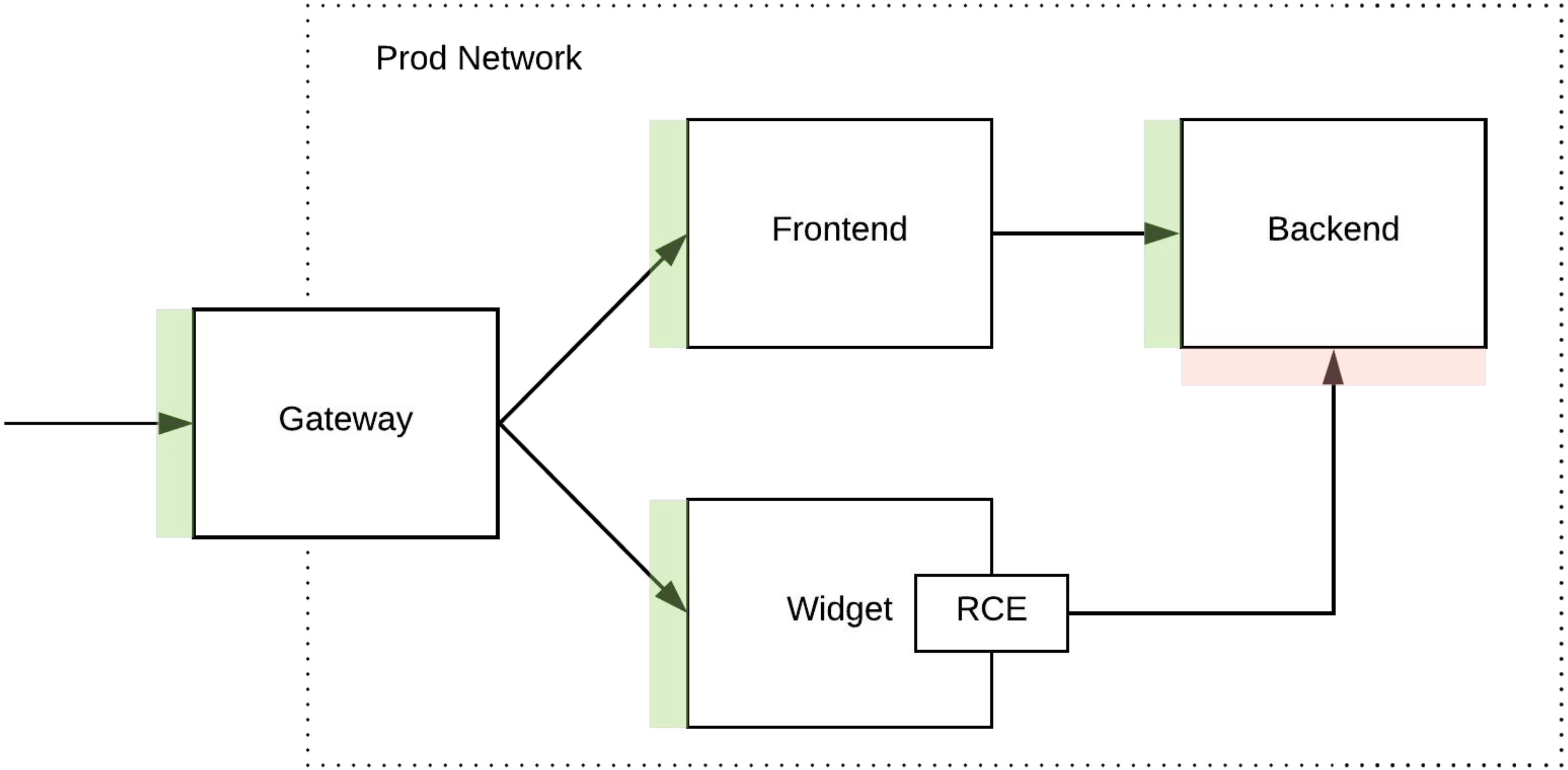


Authentication Feature Creep

Grant access to user data

To service Frontend with proof of user interaction

The game of telephone



Feature Creep

Grant access to User A’s data

- To Service B
  - If User A recently interacted with Service B
  - If request originated in my prod VPC
  - If service B was verifiably built by my CI system

Feature Creep

Grant access to user A's data

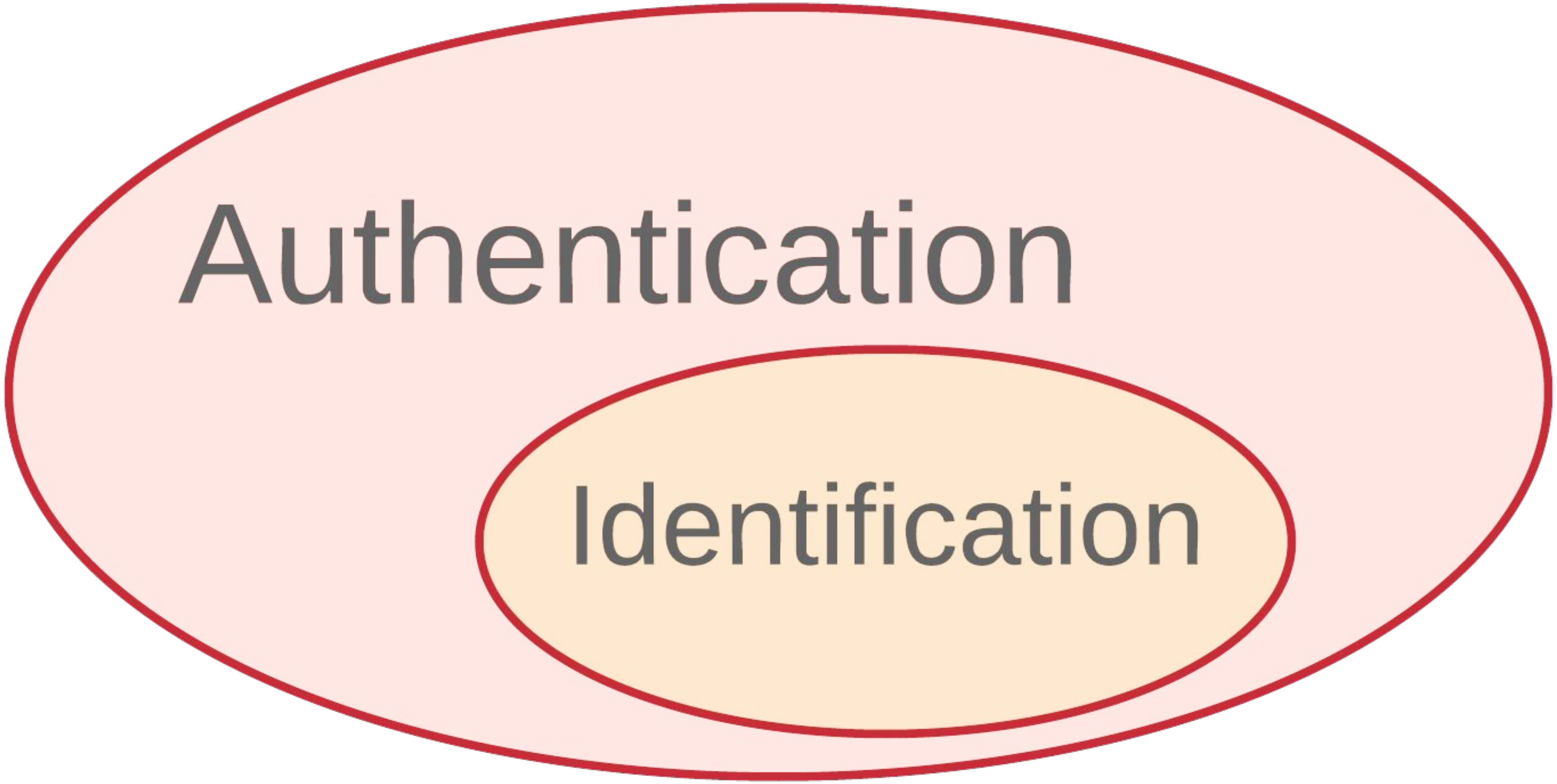
- To Employee C
  - With associated justification
    - e.g. support ticket, bug ID, page ID
  - If request originated on company issued device
  - Between the hours of 5-9PM M-F

What is authentication?

A principled approach:

- Verify authenticity of interesting attributes of a context with some degree of certainty

Identification ⊂ Authentication



Identification ⊂ Authentication

Not all attributes can be directly inferred from where they need to be consumed everywhere.

Strong identity and trust gets useful attributes to where they need to be consumed

Back to the problem

How do we create an environment that maintains a sufficiently high level of assurance on *user data*?

Is authentication the answer?

Bad news:

No, not even close. The complexity of the problem requires:

- A holistic approach
- Sustained diligence

And nothing is perfect.

Is authentication the answer?

Good news:

However, it is foundational in a holistic approach. It enables:

- Granular authorization
- Complete audit history

Closing thoughts

What makes for a good solution?

- Easy to adopt
- Hard to use incorrectly
- Generally useful, built on open standards
- Easy to evolve and extend (in and out of core)

Closing thoughts

What belongs in Kubernetes?

- Extension points that allow experimentation in systems built on Kubernetes.
- Improvements that harden core infrastructure (but move cautiously)

Closing thoughts

Shout out!

- SPIFFE and SPIRE
- SIG Auth