# Workload Identity Use Cases

***IETF 118***

Justin Richer
Bespoke Engineering

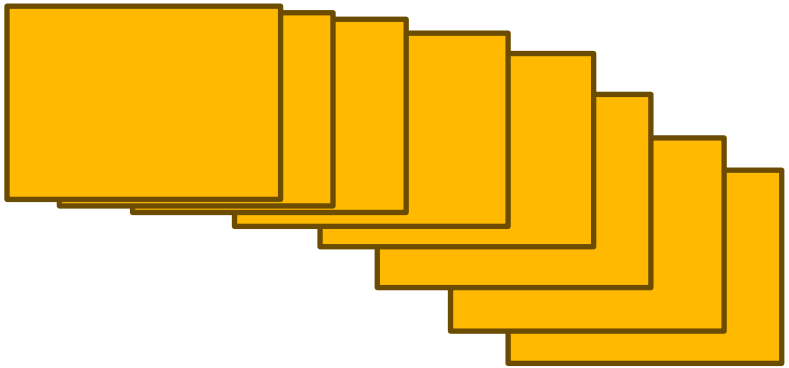
Pieter Kasselman
Workload Identity Enthusiast
Microsoft

# What is a workload?

# What is a workload?

"a single piece of software, deployed with a particular configuration for a single purpose"

# What is a workload?

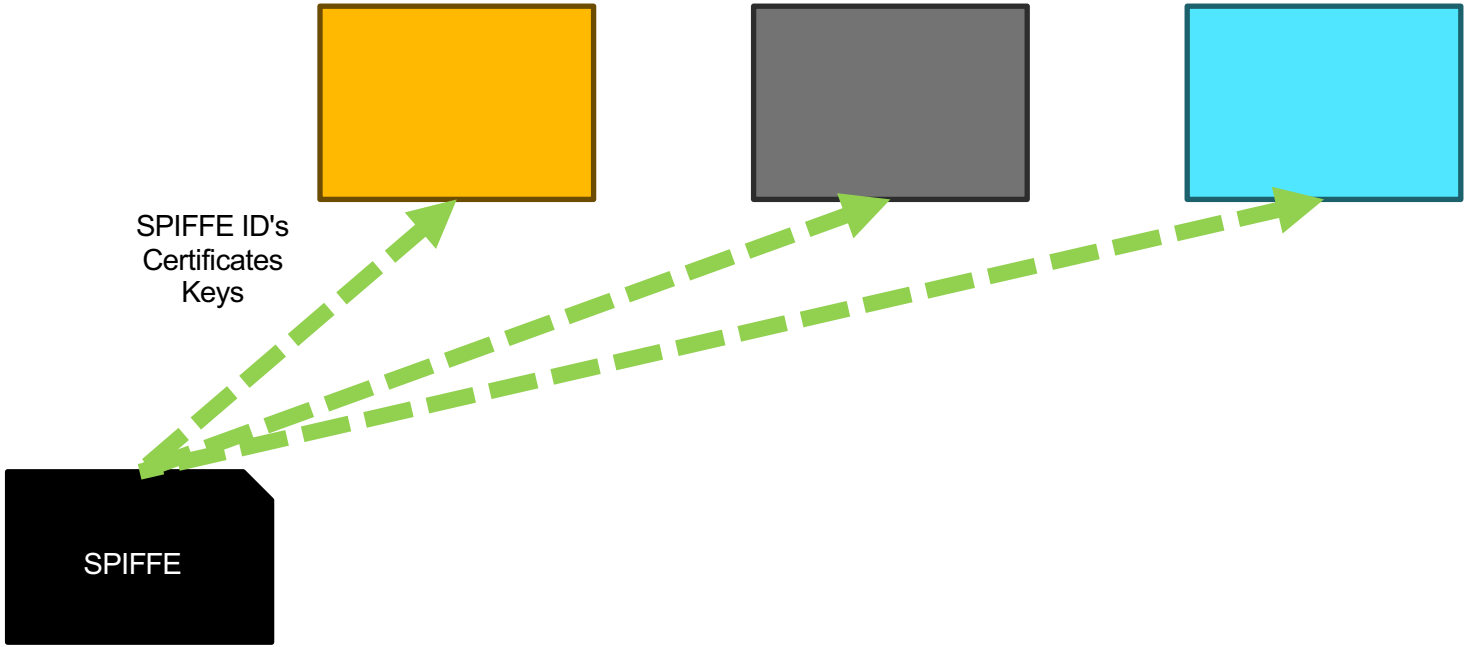


"it may comprise multiple running instances of software, all of which perform the same task"

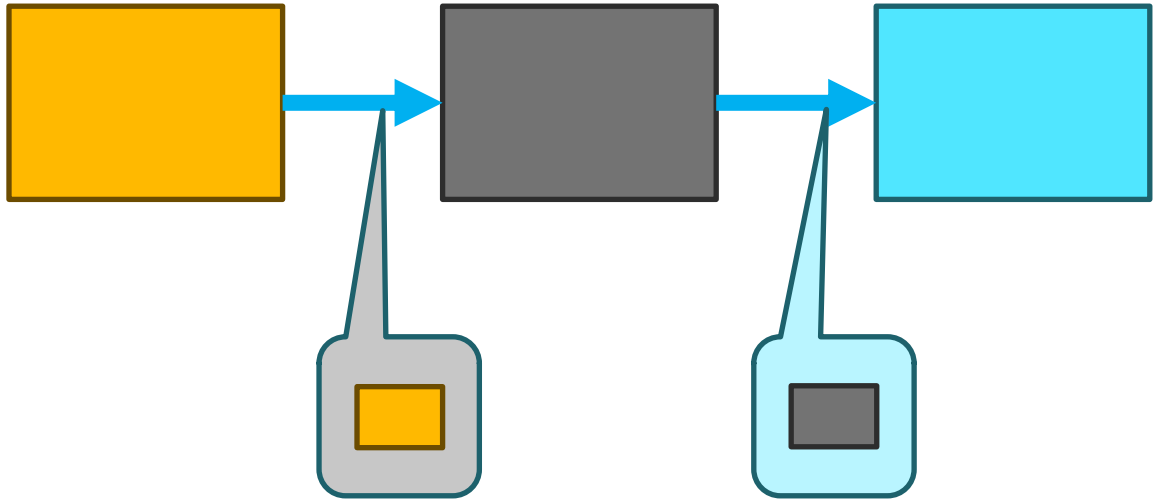
# So what's identity got to do with this?

# Let's say we have three cloud services

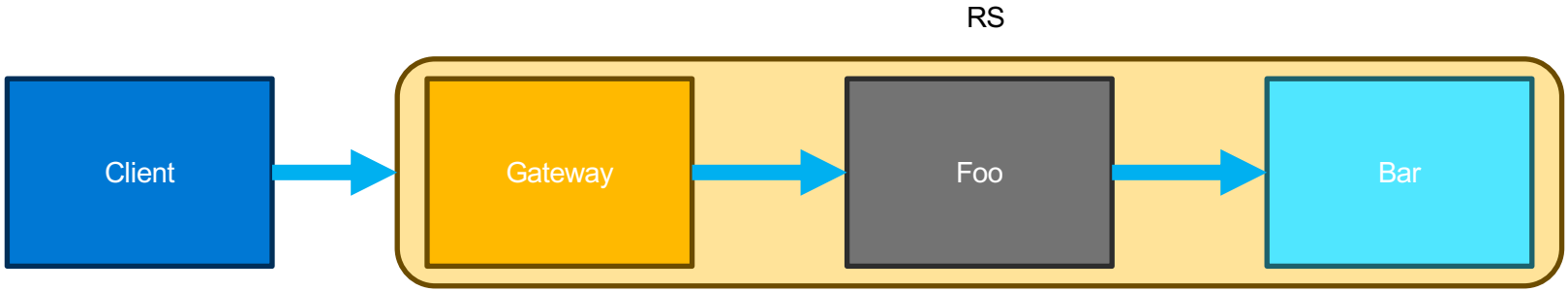
# Technology like SPIFFE can let us identify parts of the workload



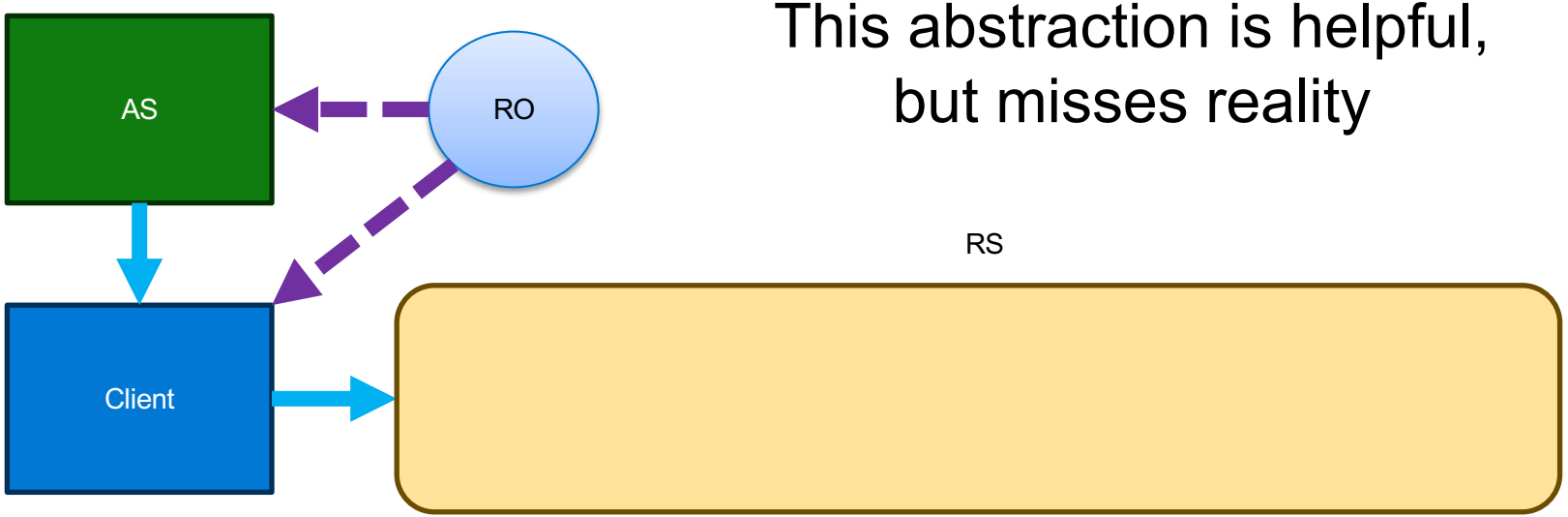
Now let's have the services talk to each other:
Each service can know its caller.



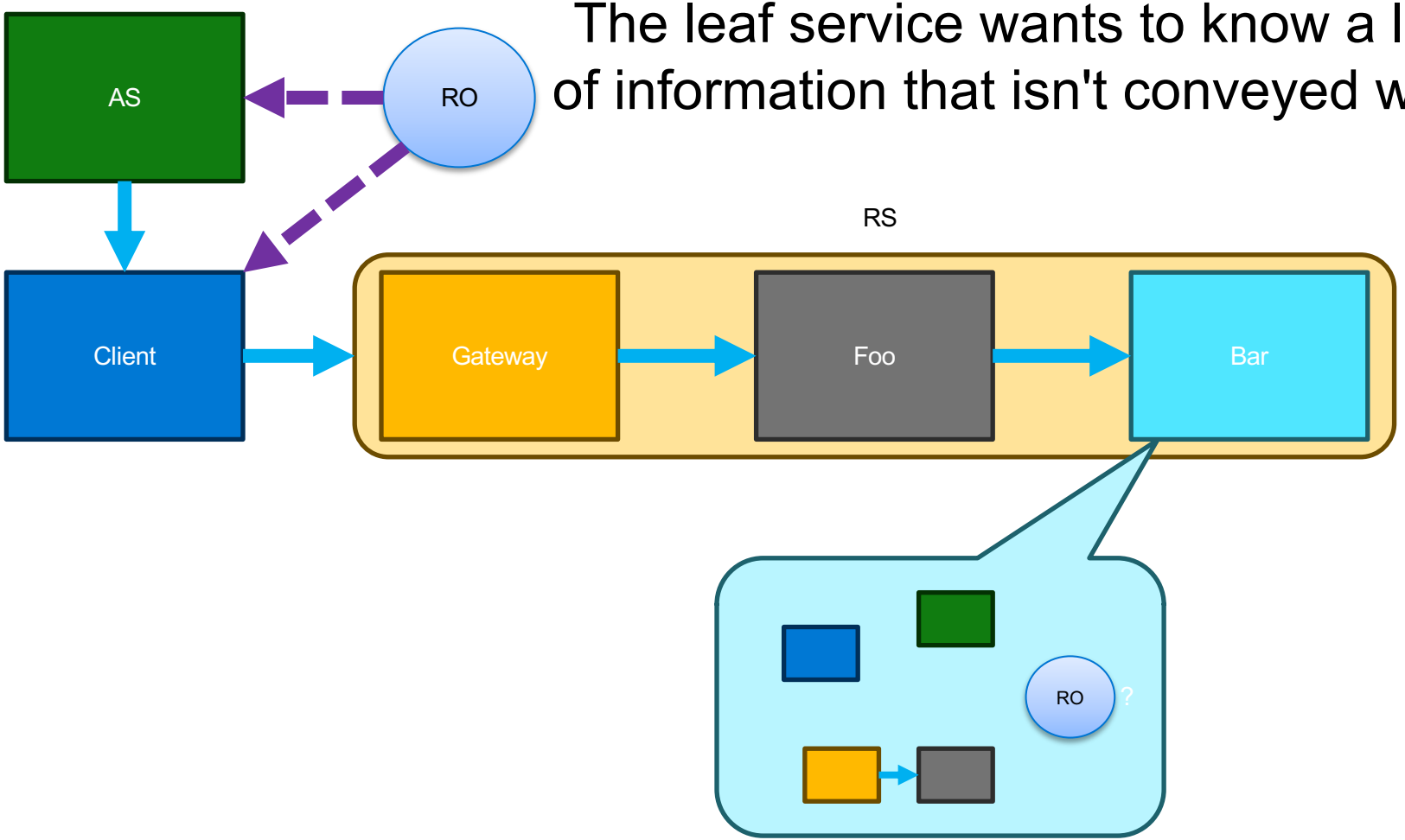
# In OAuth and similar protocols we abstract the workload



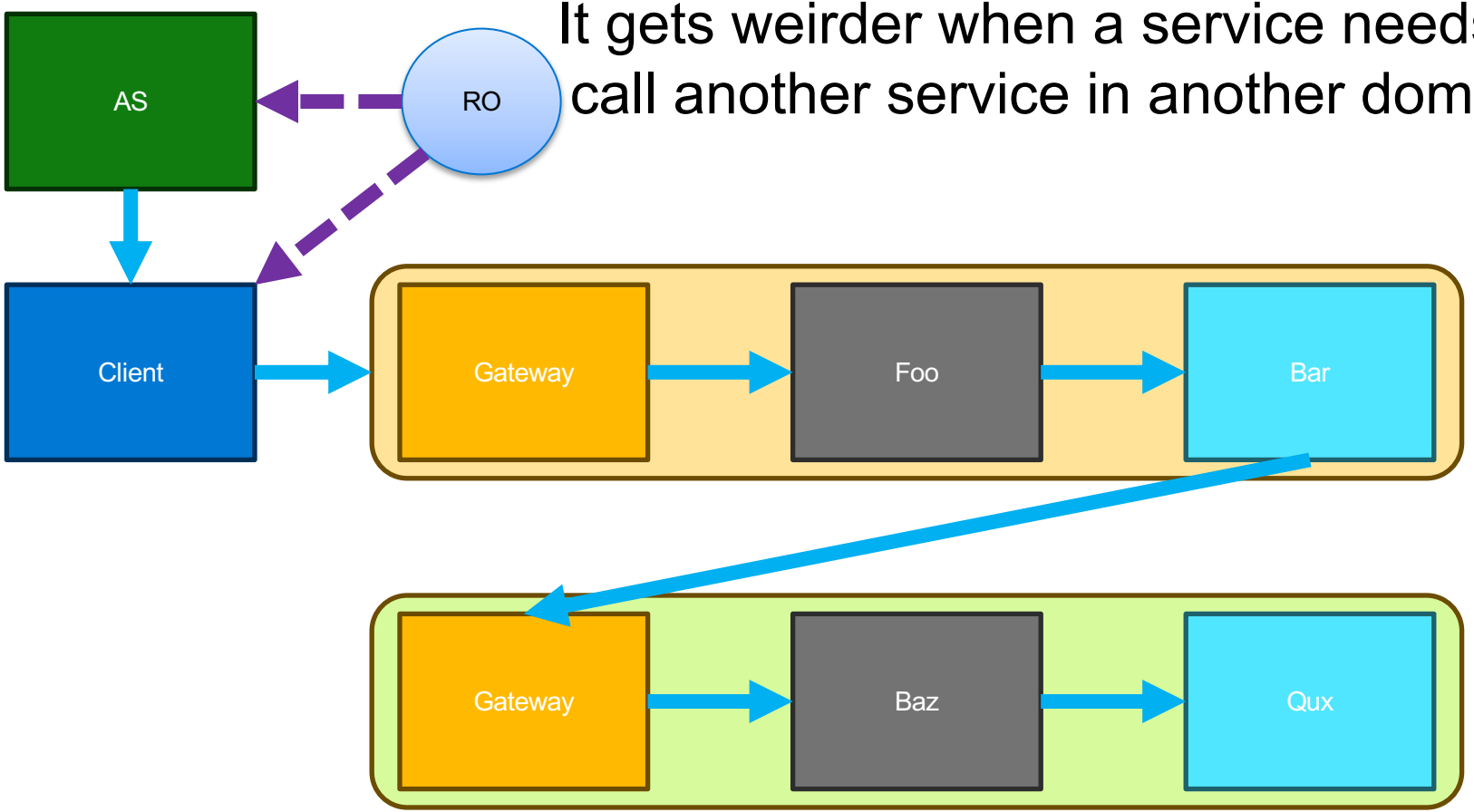
# This abstraction is helpful, but misses reality



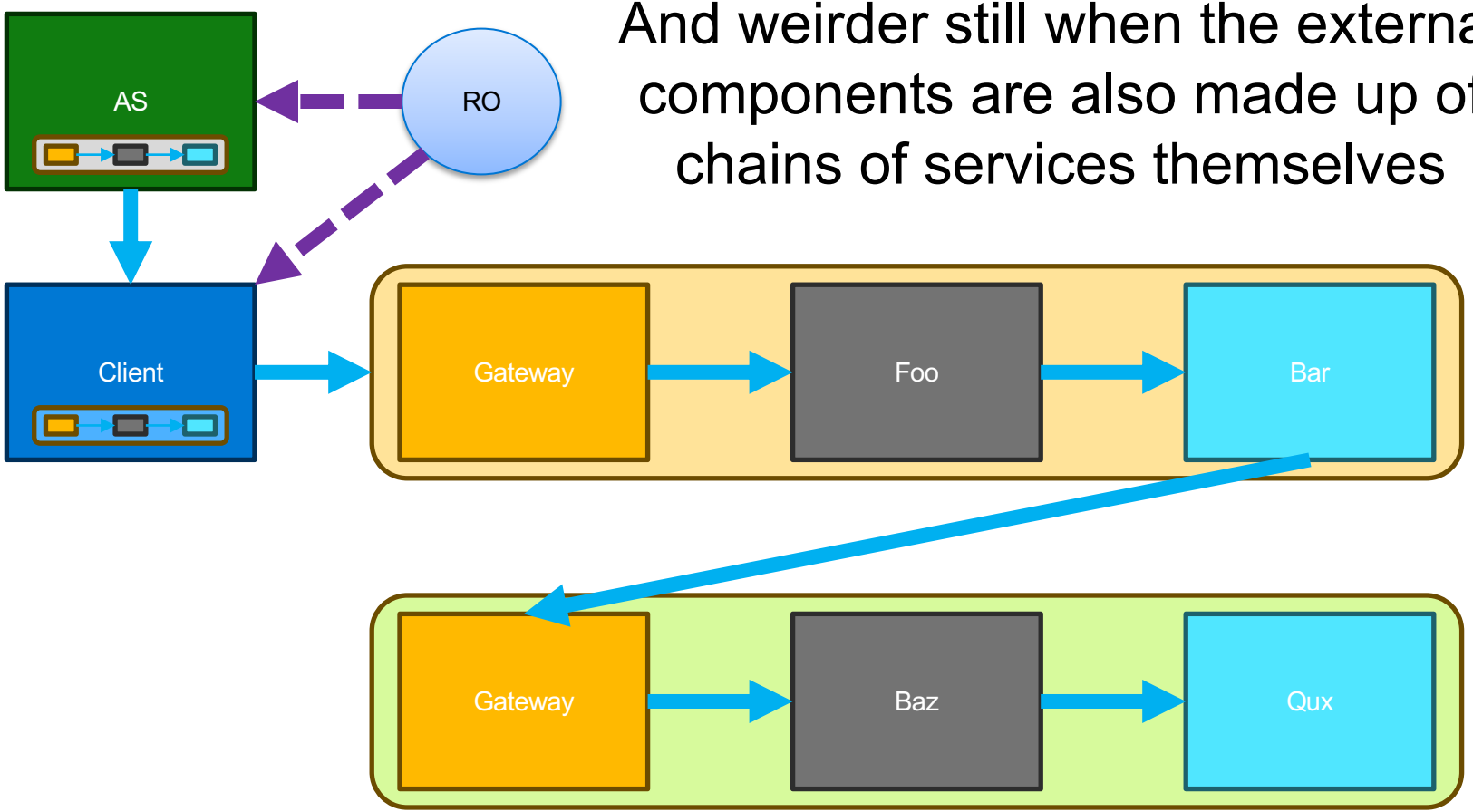
# The leaf service wants to know a lot of information that isn't conveyed well



It gets weirder when a service needs to call another service in another domain



# And weirder still when the external components are also made up of chains of services themselves



# WIMSE Use Cases

- Constrained Credential Security
- Cross-workload Access
- Chain of Custody for Request
- Local Authentication and Authorization Decisions
- Audit Logs
- Consistent Entity Identification
- Authorization
- General requirements

https://datatracker.ietf.org/doc/draft-gilman-wimse-use-cases/

# Constrained Credential Security

- Prevent token replay
- If a workload credential is compromised, I can’t re-use it.
- Workload authentication using asymmetric credentials

# Cross-workload Access

- No OAuth client registration (10k+ workloads)
- Access workloads from different service providers
- Access workloads in different cloud providers (multicloud)

# Chain of Custody for Requests

- Need to know if the data owner authorized access
- Authorize on-behalf-of logged in user
- Authorize on-behalf-of user as scheduled job
- Authorize if specific service was called (e.g. fraud)
- Authorize if request entered through a specific point of entry

# Local Authentication and Authorization

- Low latency (<<10ms)
- Resilient to network disruptions
- Operate in disconnected state (retrieve/establish trust in keys)
- Authorize if specific service was called (e.g. fraud)
- Account onboarding is not time sensitive, but runtime use is.

# Audit Logs

- Remediate a specific workload without impacting others.
- Audit trail for account\identity onboarding
- Reconcile logs when a disconnected entity is re-connected.

# Consistent Entity Identification

- Each network entity needs to be identified unique.
- Discover metadata of another trust domain

# Authorization

- Need to know origin and integrity of calling software.
- Authenticate based on workload ID documents
- Carry rights/policies/privileges to a disconnected entity
- Delegate permissions from caller to service
- Append only record (not mandatory to add)

# General Requirements

- Observability should be a requirement.
- Accountability to the overarching policy and framework
- Effect changes in the system based on signals from the application plane.
- Definition of information encapsulated in the document