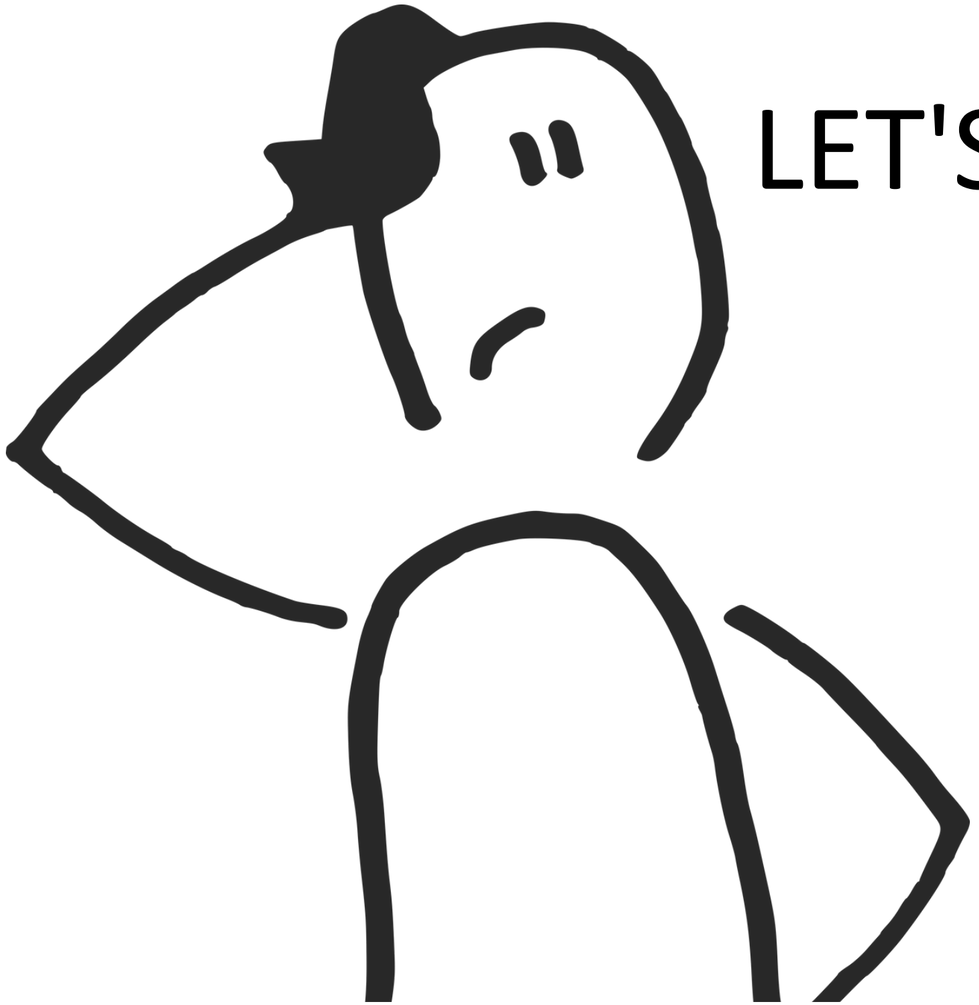
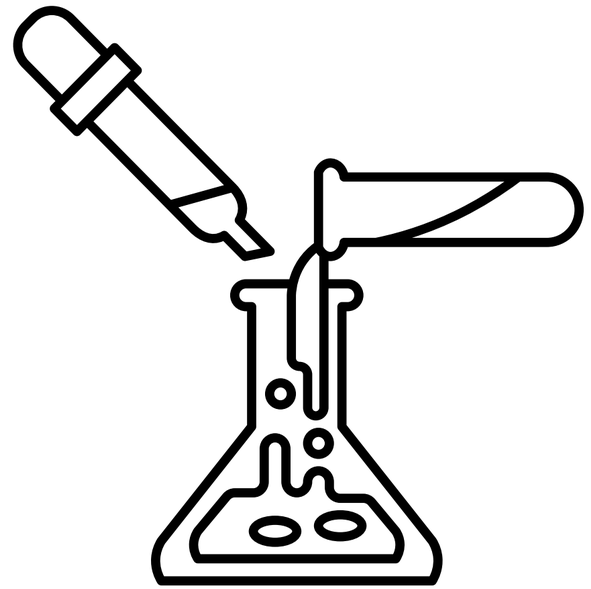
DESIGNING RESILIENT MICROSERVICE ARCHITECTURES WITH ECS AND SERVICE CONNECT

Avinash Dalvi

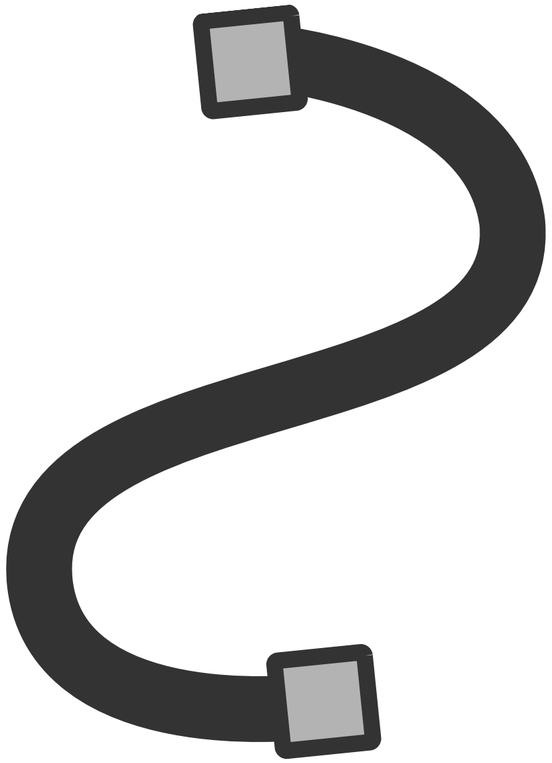
Head of technology, Nushift Technologies



LET'S START WITH A THOUGHT EXPERIMENT...

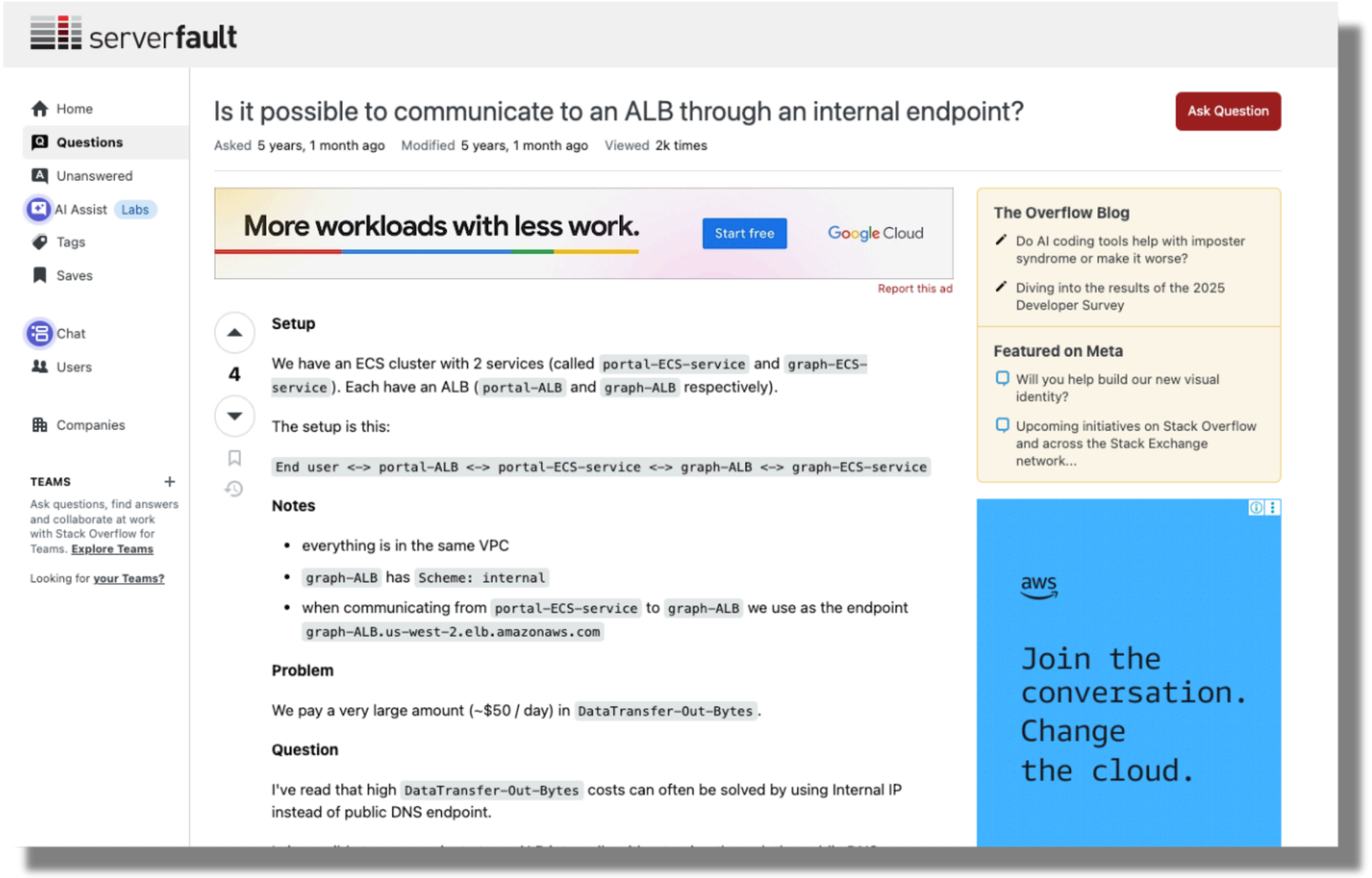


How do you wire multiple services?

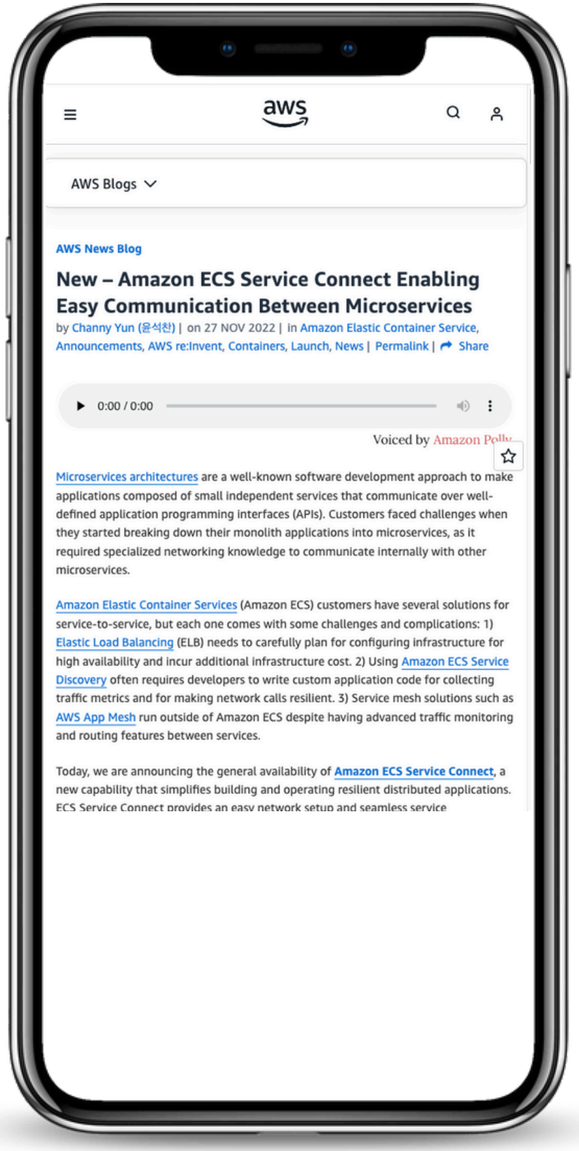


- Spin up an internal load balancer?
- Do you use an environment variable with the service URL?
- Manually manage hostnames in a config file?

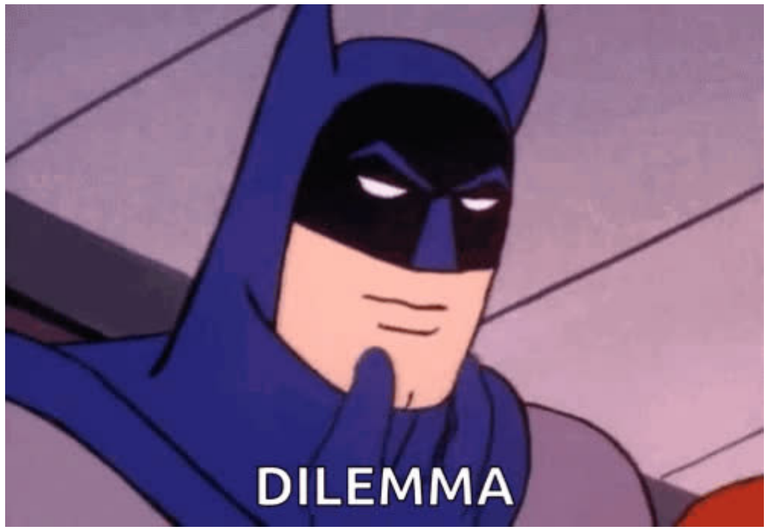
but here's the twist


serverfault
Home
Questions
Unanswered
AI Assist
Labs
Tags
Saves
Chat
Users
Companies
TEAMS
Ask questions, find answers and collaborate at work with Stack Overflow for Teams. Explore Teams
Looking for your Teams?
Is it possible to communicate to an ALB through an internal endpoint?
Ask Question
Asked 5 years, 1 month ago Modified 5 years, 1 month ago Viewed 2k times
More workloads with less work.
Start free
Google Cloud
Report this ad
4
Setup
We have an ECS cluster with 2 services (called portal-ECS-service and graph-ECS-service). Each have an ALB (portal-ALB and graph-ALB respectively).
The setup is this:
End user <-> portal-ALB <-> portal-ECS-service <-> graph-ALB <-> graph-ECS-service
Notes
everything is in the same VPC
graph-ALB has Scheme: internal
when communicating from portal-ECS-service to graph-ALB we use as the endpoint graph-ALB.us-west-2.elb.amazonaws.com
Problem
We pay a very large amount (~$50 / day) in DataTransfer-Out-Bytes.
Question
I've read that high DataTransfer-Out-Bytes costs can often be solved by using Internal IP instead of public DNS endpoint.
The Overflow Blog
Do AI coding tools help with imposter syndrome or make it worse?
Diving into the results of the 2025 Developer Survey
Featured on Meta
Will you help build our new visual identity?
Upcoming initiatives on Stack Overflow and across the Stack Exchange network...
aws
Join the conversation. Change the cloud.

Then in 2022, AWS launched something called Service Connect — a native, elegant way to handle service-to-service communication in ECS



The Classic Dilemma



- Environment variables (works until it doesn't)
- Public ALB routing (feels wrong, costs money)
- Internal ALBs ($$ adds up fast)
- Custom service discovery (complexity nightmare)

Service A → ALB → Target Group → Service B

The problems with this approach:

$20/month per ALB × number of internal services

Extra network hop for every internal call

Managing target groups, health checks, and routing rules

Single point of failure for service communication

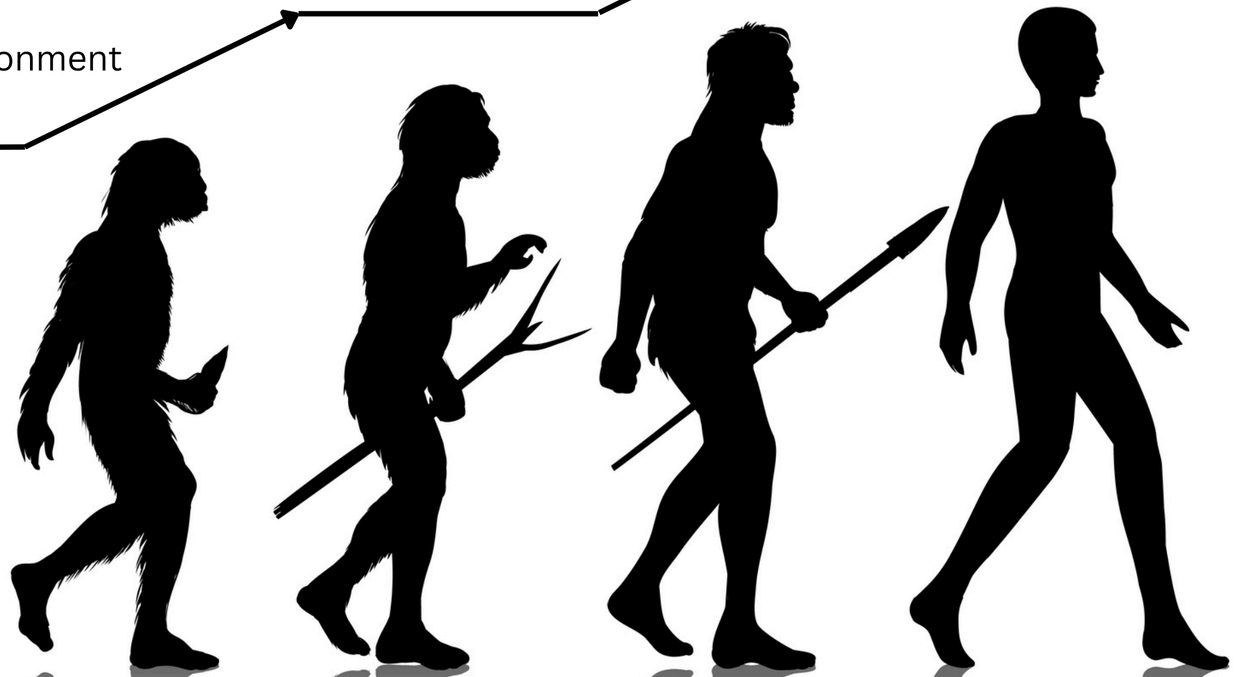
The Evolution of My Thinking

2019: "I'll just use environment variables for now"

2020: "Maybe I need a service discovery solution"

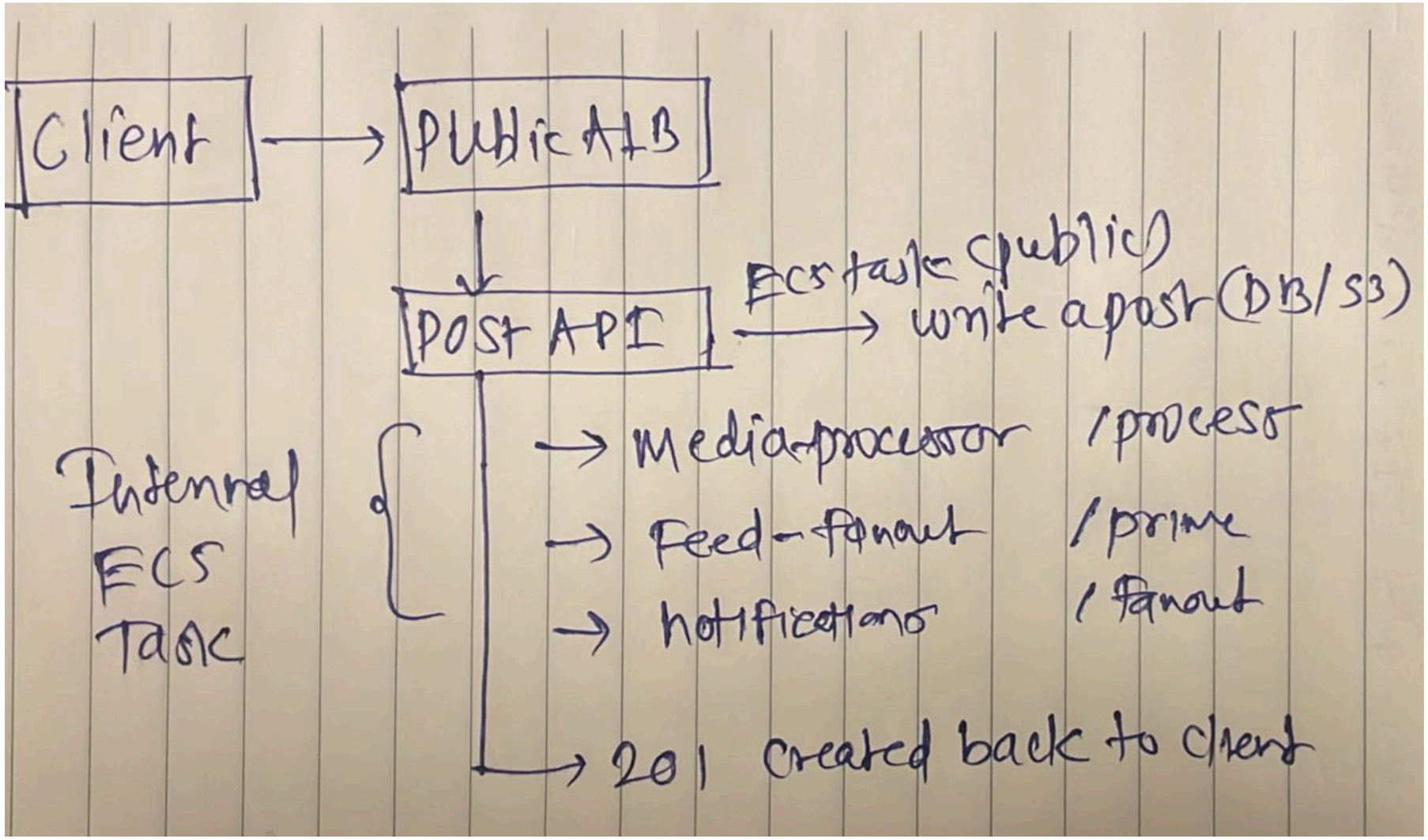
2021: "What about AWS Service Discovery?"

2022: "Wait... what's this Service Connect thing?"



How its really works ?

The "Finally!" Moment



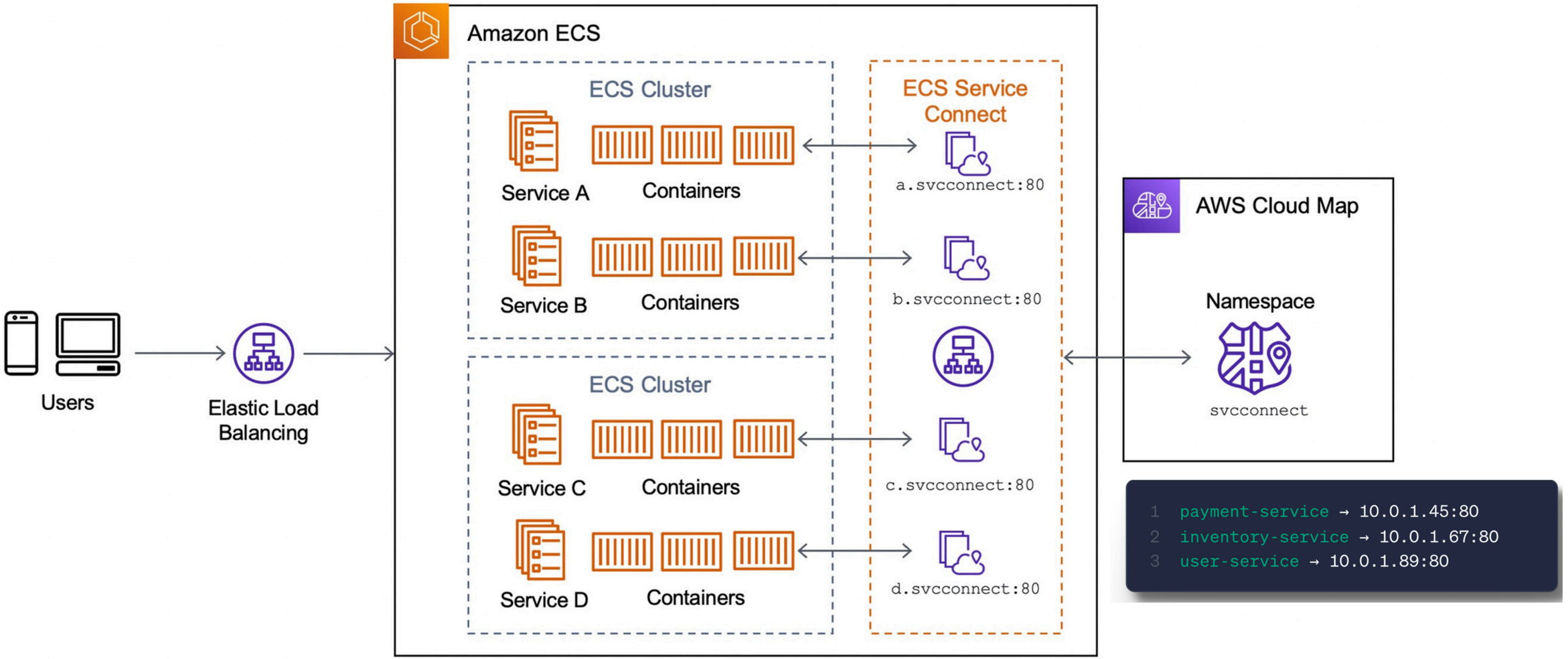
Before

```
PAYMENT_URL = os.environ.get('PAYMENT_SERVICE_URL', 'http://localhost:3001')
response = requests.get(f"{PAYMENT_URL}/api/process-payment")
```

After

```
# No environment variables needed!
response = requests.get("http://payment-service/api/process-payment")
```

How It Actually Works




```
1  payment-service → 10.0.1.45:80
2  inventory-service → 10.0.1.67:80
3  user-service → 10.0.1.89:80
```

What it takes to set this up

- Define a Cloud Map namespace (e.g. app.local)
- Add ServiceConnectConfiguration to your ECS service
- Register each service with a portName
- Set one as client-only or server (like Redis)

When NOT to use Service Connect

External Service Communication

- Service Connect is for internal ECS-to-ECS communication only
- Still need ALBs/NLBs for external traffic

Cross-Region Communication

- Service Connect works within a single region
- Use API Gateway or direct ALB calls for cross-region

When NOT to Use Service Connect

Legacy Service Integration

- Services not running on ECS can't participate
- Consider hybrid approaches during migration

Complex Routing Requirements

- Advanced traffic splitting (beyond simple load balancing)
- Complex authentication/authorization rules

The Migration Strategy

Internal services → Service Connect

Keep ALBs for external traffic

Gradually migrate edge services

Start with Non-Critical Services

- Begin with internal admin tools
- Graduate to business-critical services
- Monitor extensively during transition

> *Don't bet the farm on your first attempt*

Health Check Configuration is Critical

```
yaml# Too aggressive - causes unnecessary failovers
healthCheckGracePeriodSeconds: 10

# Too lenient - slow failure detection
healthCheckGracePeriodSeconds: 300

# Just right - balanced approach
healthCheckGracePeriodSeconds: 60
```

Naming Conventions Matter

```
# Bad - conflicts with AWS services
discoveryName: "sqs-processor"

# Good - clear and unique
discoveryName: "order-processor-service"
```

Monitor the Envoy Proxy

- CPU and memory usage of the Service Connect agent
- Connection pool statistics
- Retry and circuit breaker metrics

Who's actually using this?

Open Healthcare Network (OHC)

Service Connect simplified the configuration and management, reducing the architecture's complexity. Regarding pricing, Service Connect is cost-effective as it uses AWS Cloud Map, which is free for ECS Service Connect. Removed ALB for Redis, cut cost, simplified internal connectivity

Why This Matters – Final Takeaways

- Stop duct-taping service communication - You don’t need to wire ECS services with ALBs, env vars, and fragile DNS hacks anymore.
- Service Connect is built-in, secure, and cost-effective
- Native service discovery, health-aware routing, IAM control — no extra infrastructure.
- Start small, scale clean - Use Service Connect for internal traffic first. Migrate gradually. The DX and ops benefits stack up fast.

Should You Try Service Connect?

- Do your ECS services need to talk to each other internally?
- Are you spinning up internal ALBs just for internal APIs?
- Are you manually updating service URLs during deploys?
- Do you want built-in health-aware service discovery?

= Try ECS Service Connect

References

- https://docs.ohc.network/blog/aws-service-connect
- https://medium.com/@joudwawad/aws-ecs-deep-dive-c8f773af0bf6
- https://fivexl.io/blog/ecs-service-connect-encryption
- https://aws.amazon.com/blogs/aws/new-amazon-ecs-service-connect-enabling-easy-communication-between-microservices/


THANK YOU!
Connect with me to learn more!
AWS
FOR PRODUCT BUILDERS