
AWS
re:Invent
DECEMBER 1 – 5, 2025 | LAS VEGAS, NV

MAM301

Transforming monoliths with cell-based architecture

Adrian Begg

Senior Migration & Modernization Solutions Architect

EMEA Migration & Modernization, AWS

Inho Kang

Senior Migration & Modernization Solutions Architect

APJ Migration & Modernization

Agenda

What is cell-based architecture ?

What problems do cell-based architectures aim to solve ?

What are key design challenges and patterns for cell-based architecture ?

What tools or resources can help ?


“Everything fails, all the time.”
Werner Vogels
CTO, Amazon.com
aws

Limiting the blast radius of failure

"Blast radius is defined as the **maximum impact** that might be sustained in the event of a system failure, or faulty change."

Cell-based Architecture

Cell-based architecture aims to **limit** the blast radius by partitioning the system into isolated, self-contained units to prevent issues cascading to others

How do we protect against failure ?

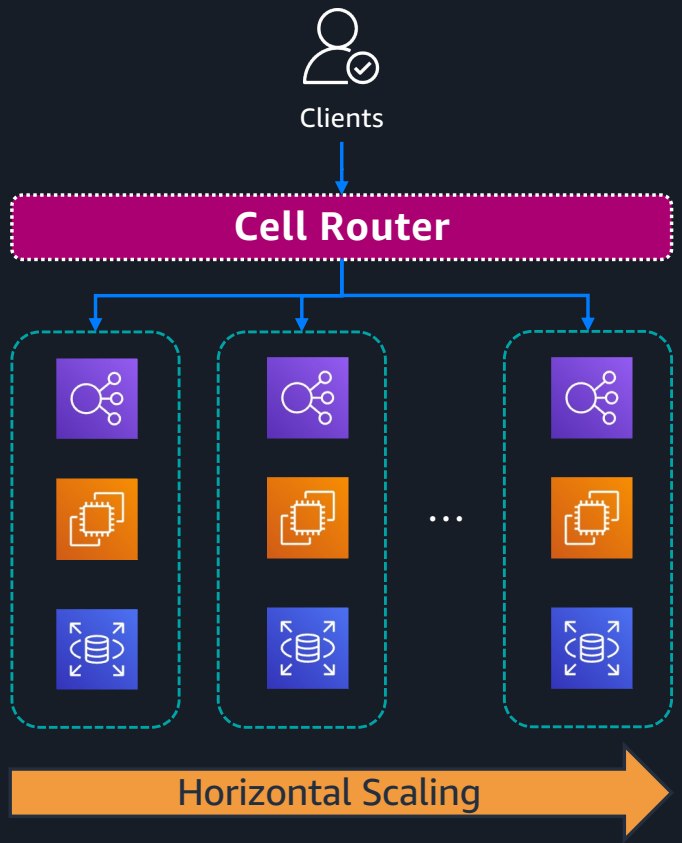
Failure Mode	Means of protection
Rack Down	Autoscaling Groups
AZ Failure	Multi-AZ Deployments
Regional Failure	Region Failover
Bad Deployment	Canary Deployments
Human Failure	Processes, Runbooks
Poison Pill	Extensive testing

Focus for cell-based arch.

Typicall only for Compute

Can’t be fully mitigated.

Cell-based Architecture Overview



- Isolated cells contain full application stack with partitioned data
- Failures contained within individual cells (limited blast radius)
- Linear horizontal scaling by adding new cells

No architecture decision you take comes without trade-offs.

Your job as software architect is to identify the least painful option on the table.

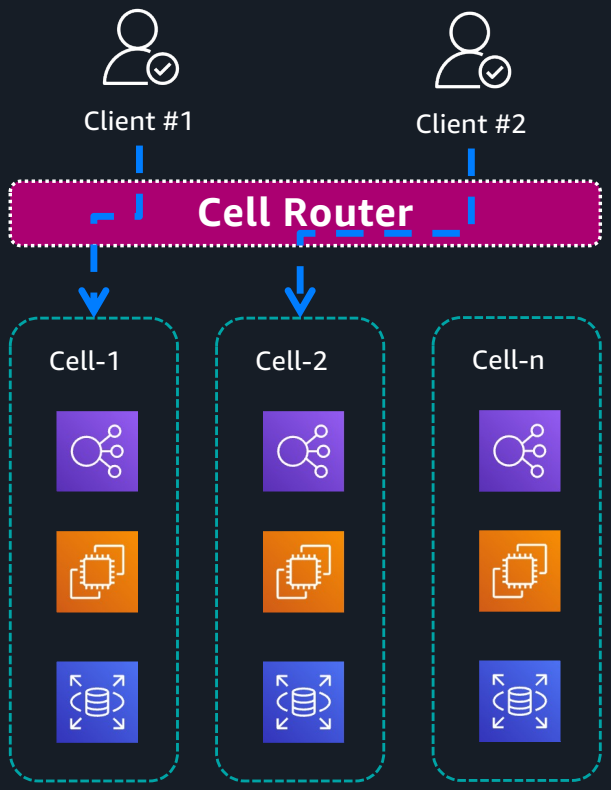
Key Design Decisions

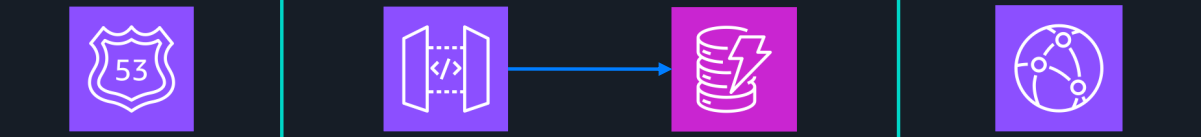
KEY DESIGN DECISIONS

Key questions to answer

- What problem are you trying to solve ?
- How will you implement the cell router ?
- Cell count and sizing ?
- What is your customer migration mechanism ?
- How will you operate and monitor ?

Cell Routing and Sharding



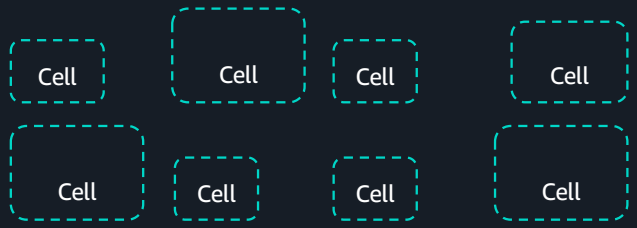


- Cell Router is discovery and routing mechanism for clients
- Single point of failure : design for high availability
- Keep thin, simple, and highly responsive
- Shard key granularity is critical

Cell Size

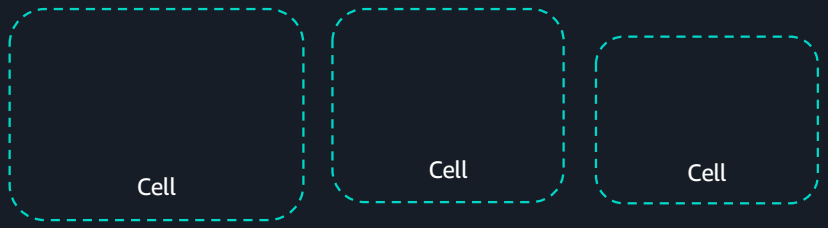
Smaller cells

- Lower blast radius
- Easier to test and deploy
- Complex to operate at scale



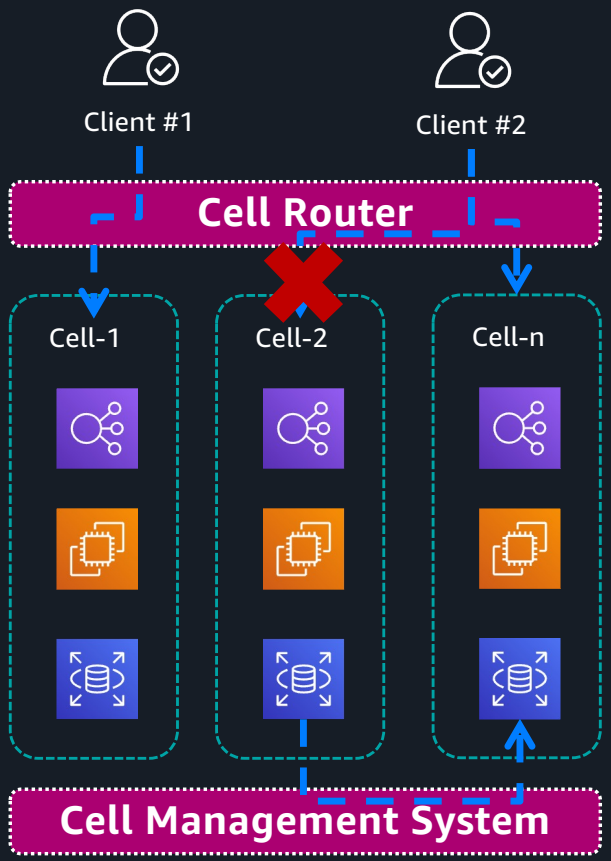
Larger cells

- Better cost efficiency
- Accommodates large customers
- Fewer splits/less operations



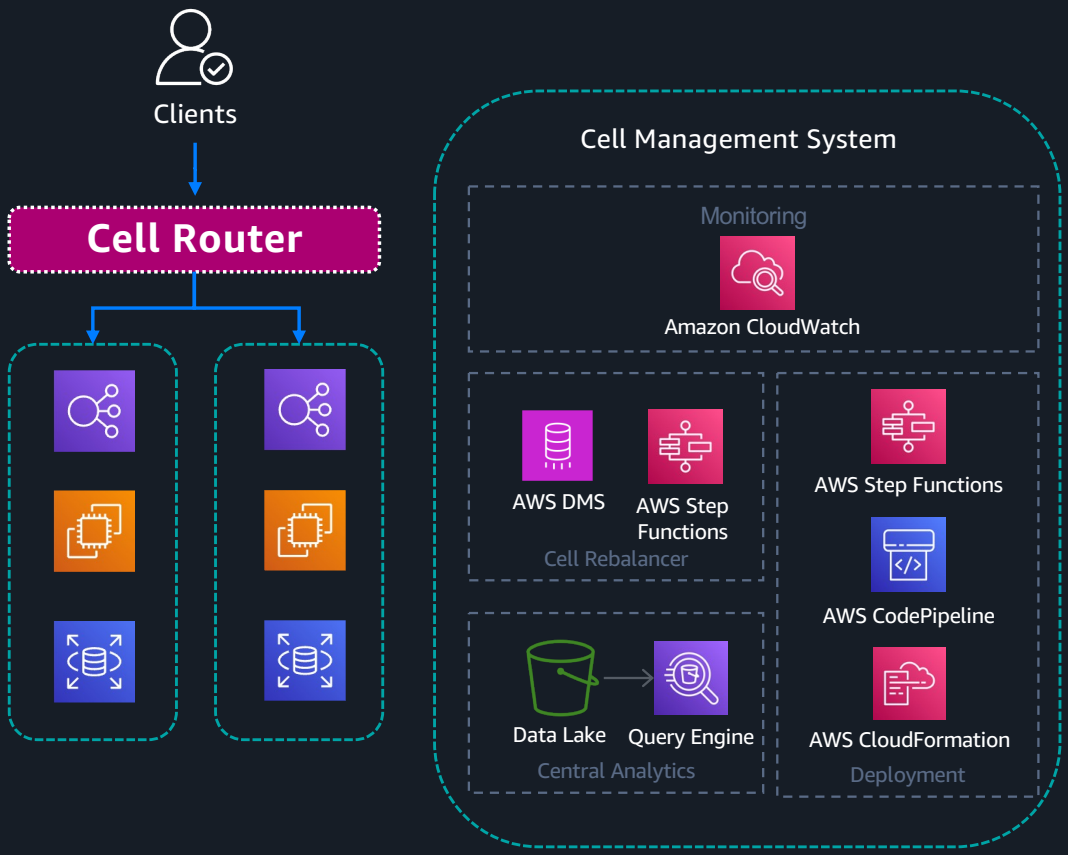
Define a max size; actual cell utilization fluctuates constantly

Migration and Heat Management



- Cells should be balanced - unbalanced cells negate blast radius benefits
- Migration should be automated, transparent, and routinely tested
- Use canary accounts to continuously validate migration process
- Consider migration complexity early: duration, downtime, customer impact

Observability and Operations



- Per-cell and aggregated observability required for health monitoring
- More components = increased operational complexity; automate heavily
- Cell management system is critical infrastructure, not an afterthought
- Design for isolation while enabling cross-cell management and networking

Key Takeaways

Key Takeaways

1. Keep routing layer simple and highly available - it's a single point of failure
2. Maintain balanced cells - imbalance eliminates blast radius benefits
3. Design migration early; automate and prove with continuous canary testing
4. Invest in robust cell management, observability, and automation
5. Consider cell sizing tradeoffs based on your workload characteristics
6. Weigh benefits (resilience, scale) against increased complexity and cost

Dive deeper

Dive Deeper in Cell-Based Architecture

AWS Prescriptive Guidance

Cloud design patterns, architectures, and implementations

https://go.aws/42TVycw

AWS Solutions Library

Guidance for Cell-Based Architecture on AWS including code sample

https://go.aws/43AtGKD

Speaker Contact & Presentation Resources

https://bit.ly/4qwe9FN

AWS Workshop

Building resilient and scalable SaaS applications with a cell-based architecture

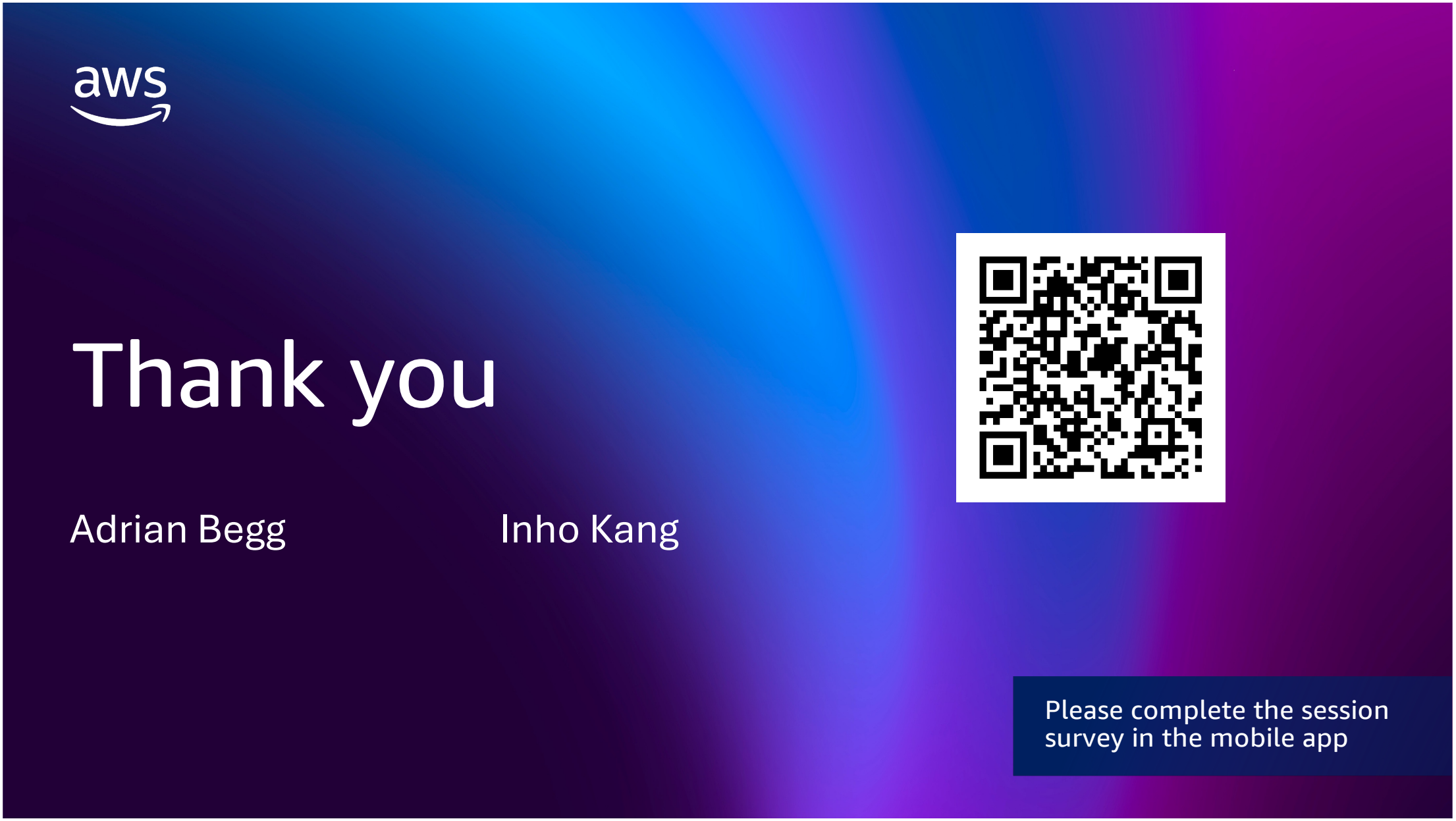
https://catalog.workshops.aws/cells-for-saas

Millions of Tiny Databases

Amazon Science describing Physalia, used in Amazon Elastic Block Service

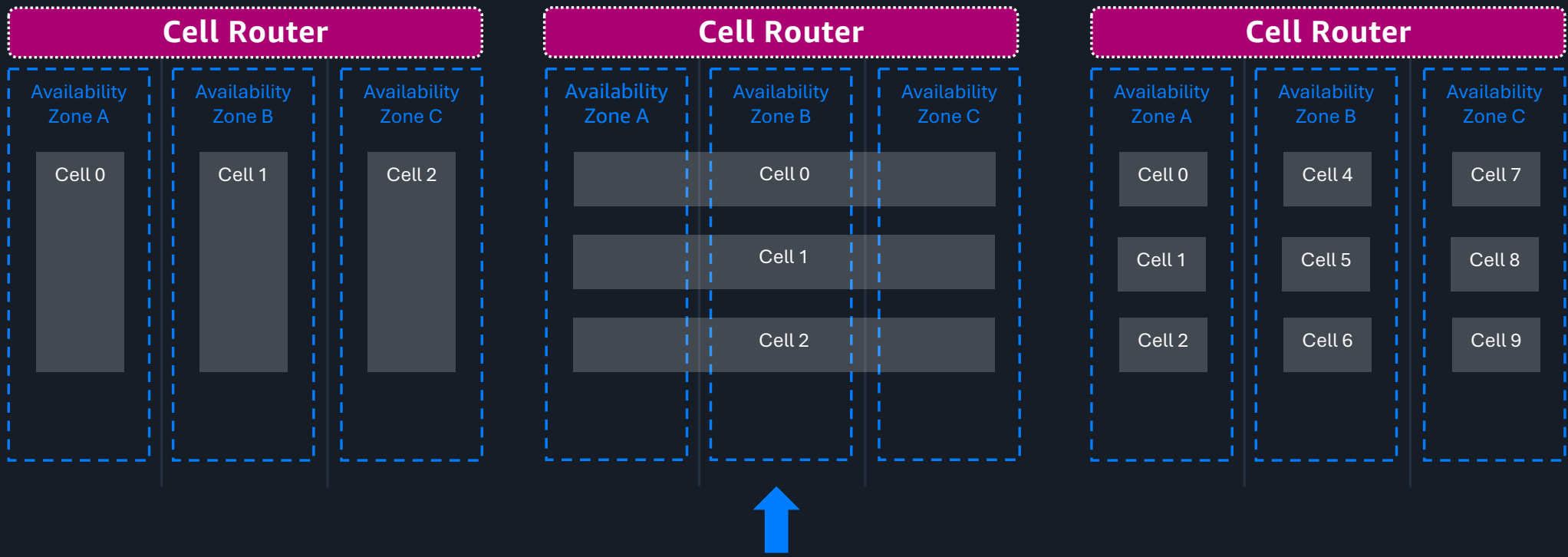
https://bit.ly/4qnAK7d

aws


aws
Thank you
Adrian Begg
Inho Kang
Please complete the session survey in the mobile app

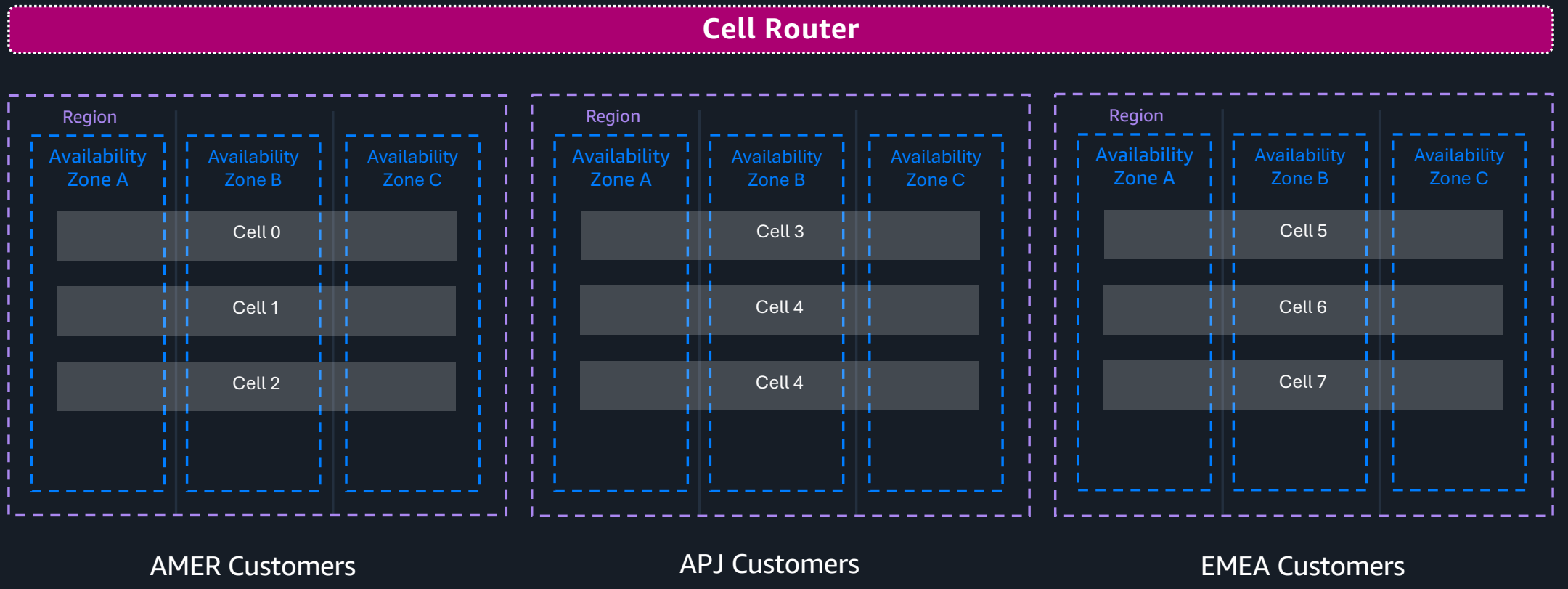
Appendix

Building Cells within an AWS Region

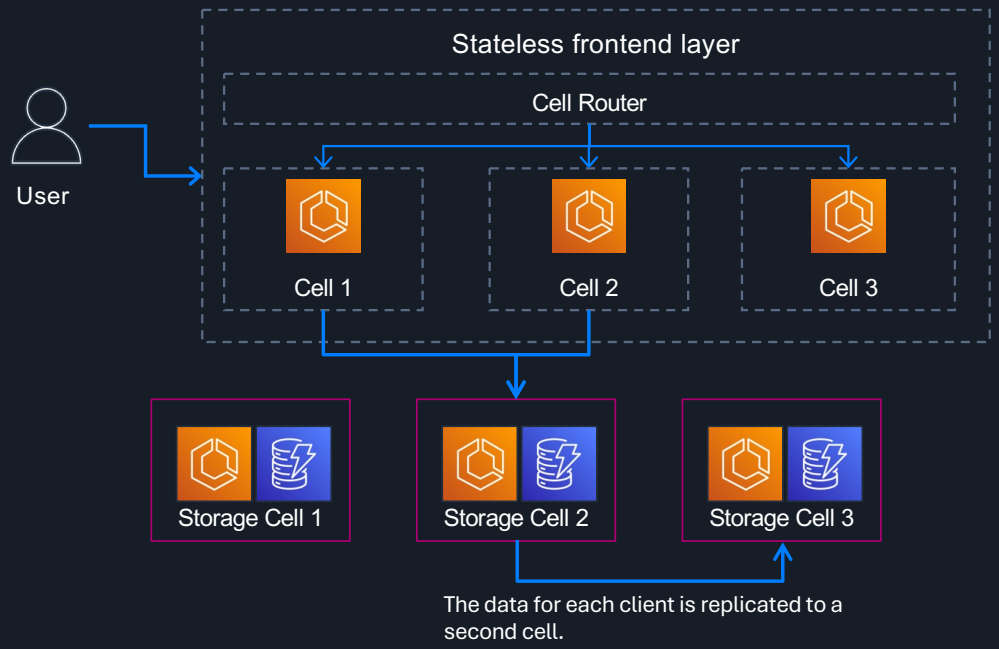


Availability Zones provide redundancy, cells provide compartmentalization, design for the combination

Building regional cells on AWS

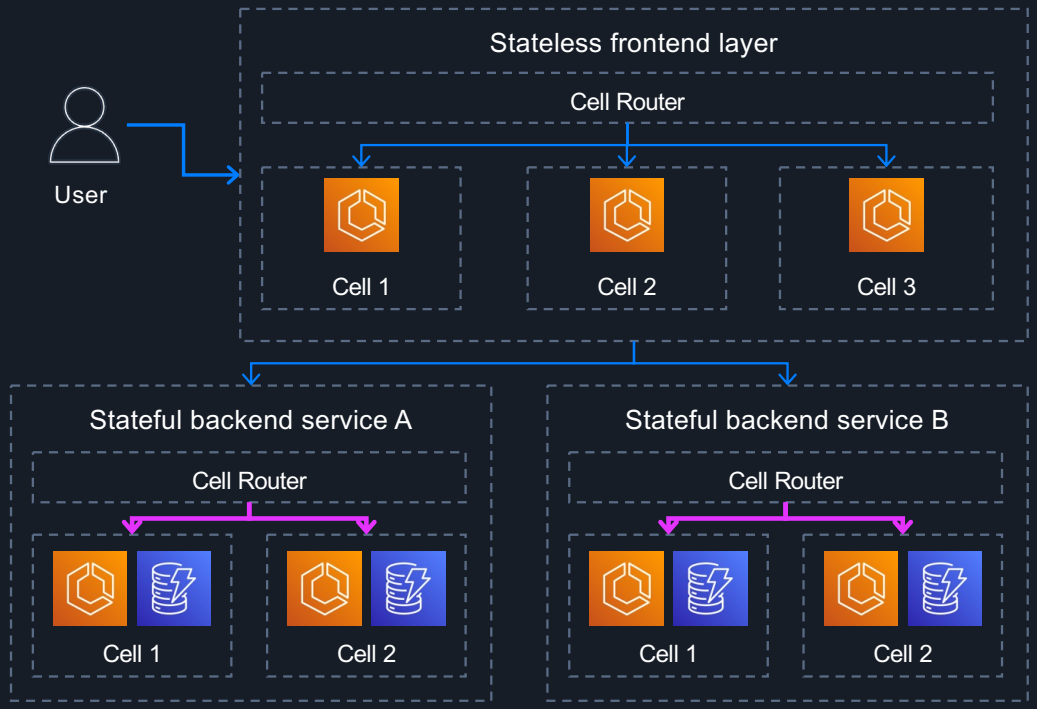


Complex designs : Decoupling storage as cells



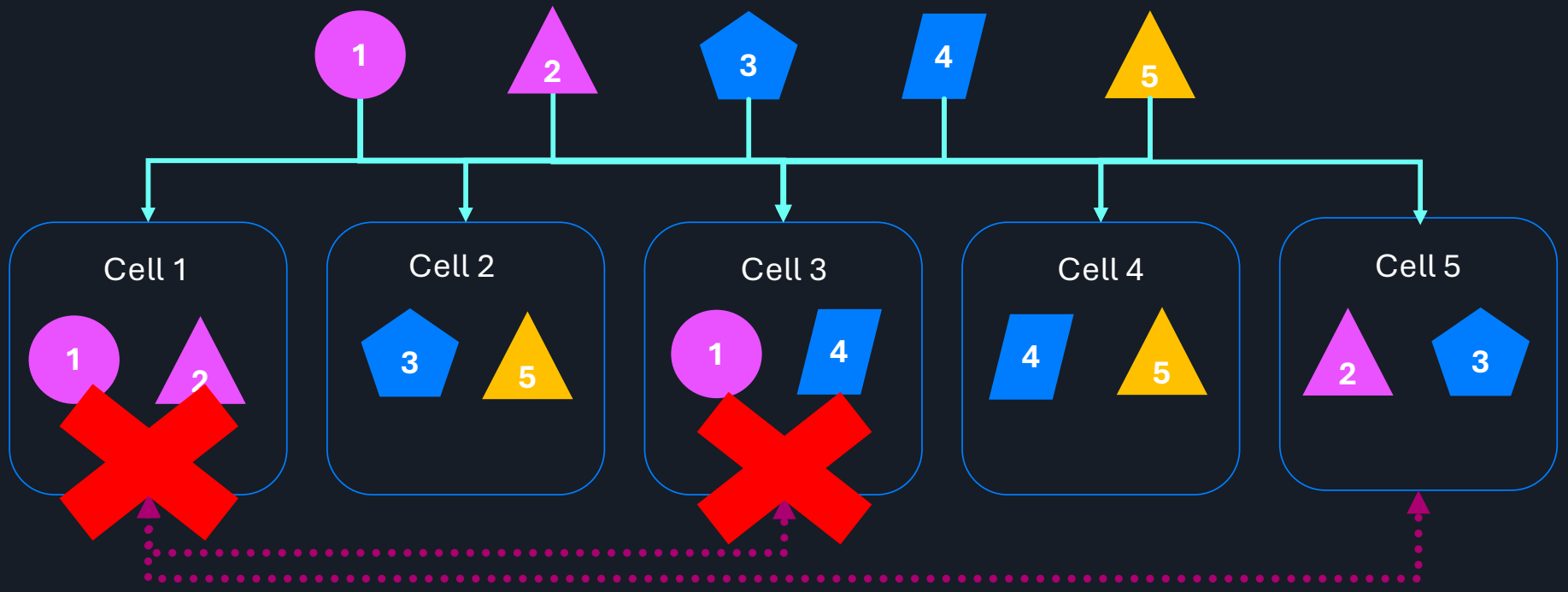
- More complex design, stateless frontend allows advanced routing
- Scale storage layer independently
- Makes front-end more disposable, migration easier

Complex designs: Scaling out cell-based architecture to more complex systems



- More complex cell-based systems can look like microservices.
- We can copy a lot of patterns from that world.
- Complexity will increase with more layers.

Shuffle Sharding



Customers = 5, Shards = 5, Overlap =1, Blast radius = 20%

https://go.aws/4895y3n