CLOUD & INFRASTRUCTURE PLAYBOOK · PART 3

Architecture Neighborhoods: Risk-Based Security Zoning That Survives a Migration

Not every app is a good neighbor. Zone every environment into neighborhoods by risk, security, and criticality, and place each app where it belongs — so a migration keeps its security posture.

THE ARCHITECTURE NEIGHBORHOOD FRAMEWORK — A PRACTITIONER'S MODEL

- **High exposure / low trust — internet-facing, DMZ, partner**
- **Regulated data — PCI, PII, confidential**
- **Mission-critical — crown jewels, fraud & risk, control plane**
- **Internal / general — business apps, shared services**
- **Non-prod — dev / test**

Tom Ward
Enterprise Architect · Cloud & Infrastructure

linkedin.com/in/tomward
July 2026

Drop a few hundred applications into a data center — or an availability zone, or a cloud region — and the real question isn't "which rack?" It's "which neighborhood?" You don't seat an internet-facing, low-trust application next to your crown-jewel data, any more than a city zones a chemical plant next to a school. The reason is the same: **shared blast radius.** Applications in a neighborhood share a security boundary, a set of controls, and a failure domain — so who their neighbors are is a risk decision, not a matter of convenience.

> **WHAT A NEIGHBORHOOD IS**
>
> **A neighborhood is a zone within an environment — not a destination. Each data center, availability zone, colo, or cloud region is carved into a dozen or more of them, and every application is placed in the one that matches its profile. The profile — not the platform, and not the team that owns the app — decides where it lives.**

Five characteristics define a neighborhood. Most applications are pulled toward a zone by the strongest one or two.

1. **Risk & exposure.** Internet-facing or internal-only? An application reachable from the public internet lives in a different trust world than one that isn't — and the two should never share a boundary.
2. **Security & trust.** A neighborhood is a set of controls, not just a location: segmentation, inspection, identity, least privilege. The zone enforces them so the app doesn't have to reinvent them.
3. **Data privacy & classification.** Cardholder data (PCI), customer PII, confidential, public. Regulated data pulls an application into a controlled neighborhood regardless of anything else about it.
4. **Business criticality.** Crown jewels or dev/test? Criticality sets how much isolation the neighborhood needs and how small its blast radius has to be.
5. **Regulatory & residency.** What the law requires — data residency, sovereignty, audit boundaries. Often this closes the question before the other four are even weighed.

The Neighborhoods, by Band

A dozen or more neighborhoods usually cluster into five risk bands. The exact list varies by enterprise; the bands rarely do.

High exposure / low trust

OUTERMOST

Internet-facing & DMZ · public web and API · partner / B2B gateways. Reachable from outside, so tightly inspected and segmented — and kept far from anything sensitive.

Regulated data

COMPLIANCE-BOUND

Cardholder data (PCI) · customer PII · sensitive analytics. The controls are set by an obligation, not a preference; the neighborhood exists to make the obligation provable.

Mission-critical

CROWN JEWELS

Core banking & systems of record · fraud and risk engines · the management/control plane. Smallest blast radius, strongest isolation — a compromise here is an existential event.

Internal / general

THE BULK

Internal business applications · shared services. The largest population by count, and the easiest to place — but still zoned, so it can't become a soft path to the bands above.

Non-production

ISOLATED

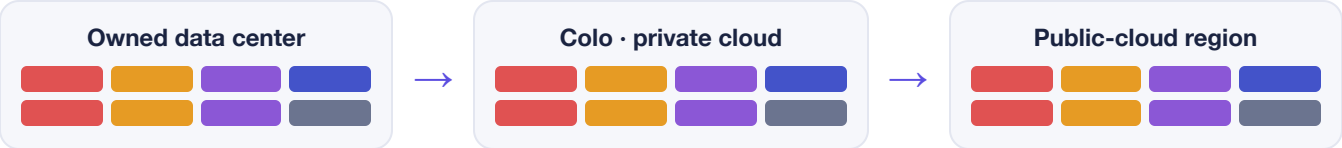
Dev / test / staging. Separated hard from production, because non-prod is where controls are loosest and lateral movement is easiest.

The principle underneath: shared blast radius

Every application in a neighborhood shares one security boundary and one failure domain. That is the whole point — and the whole risk. **Who an app's neighbors are is a security decision, not a placement of convenience.** Get it wrong and one weak tenant becomes the way into a strong one.

The Same Map in Every Environment

Here is the part that makes zoning more than good hygiene — and the part most migrations miss. The neighborhoods don't change when the estate does. The same map applies whether an application lives in an owned data center, a colocation private cloud, or a public-cloud region.



So a migration isn't a re-architecture of your security posture — it's **re-homing each application into the equivalent neighborhood** in the new environment. In public cloud the neighborhood is expressed as landing zones, subscriptions, VPCs, and policy; in a data center as VLANs, security zones, and segmentation. The *expression* changes; the map doesn't. Get the neighborhoods right once, and the risk and security posture survives the move.

Two layers — don't confuse them

Environment is the destination; neighborhood is the zone within it. "Do we move this to colo or to Azure?" is the environment decision. "Which zone does it land in once there?" is the neighborhood decision. You need both — and getting the second one wrong is how a clean, on-time, on-budget migration quietly weakens a bank's security posture, with no one noticing until it's tested.

The Zoning Test

Ten checks before you zone an environment — or migrate one. Score one point per honest "yes." Below seven, you're placing applications by convenience, and your blast radius is larger than you think.

- ☐ Every application has a **named neighborhood** — not just a rack, a subnet, or a resource group.
- ☐ Neighborhoods are defined by **profile** — risk, security, data class, criticality, regulatory — not by team or convenience.
- ☐ **Internet-facing, low-trust workloads never share a neighborhood** with crown-jewel data.
- ☐ Each neighborhood is a real boundary — **shared segmentation, controls, and failure domain** — not just a label.
- ☐ Regulated data (PCI, PII) sits in a neighborhood whose controls **actually meet the obligation**, and can prove it.
- ☐ The **blast radius of each neighborhood is known and bounded** — one compromise doesn't cross the boundary.
- ☐ The **same neighborhood map exists in every environment** you run — owned DC, colo, and cloud.
- ☐ A migration plan **re-homes each app to its equivalent neighborhood**, not "wherever it lands."
- ☐ Your most sensitive workload has a **defined equivalent neighborhood in the target environment** — if not, that gap is your migration risk.
- ☐ Zoning standards run through **architecture governance with conformance**, so neighborhoods don't erode over time.

Going Deeper: Frameworks & Companions

The disciplines this model builds on, and a companion note. All links verified July 2026.

NIST SP 800-207 — Zero Trust Architecture

The reference for trust zones, segmentation, and least privilege — the security spine of a neighborhood.

csrc.nist.gov/pubs/sp/800/207/final

CISA — Zero Trust Maturity Model

Segmentation as a maturity pillar — a practical yardstick for how well your neighborhoods actually contain risk.

cisa.gov/zero-trust-maturity-model

Microsoft — Azure Landing Zones

How a neighborhood is expressed in public cloud: subscriptions, management groups, and policy guardrails.

learn.microsoft.com/azure/cloud-adoption-framework/ready/landing-zone

AWS — Security Reference Architecture

The AWS counterpart: accounts, organizational units, and guardrails as segmentation — the same zoning, expressed in AWS.

docs.aws.amazon.com/prescriptive-guidance/latest/security-reference-architecture

The TOGAF® Standard

Carrying zoning standards through architecture governance, with conformance, so neighborhoods hold over time.

opengroup.org/togaf

Companion — You Don't Choose Five Nines

Once an app is zoned, its reliability tier picks the deployment archetype. Zoning and posture, in sequence.

archreviews.com/cloud-playbook/availability-ladder

About the Author

Tom Ward is an Enterprise Architect with 20+ years across Fortune 500 financial services, retail, and government — including Wells Fargo and Bank of America — specializing in EA governance, reference architectures, data-center and hybrid-cloud placement, security zoning, and legacy modernization, with recent hands-on GenAI and mobile delivery work.

More field notes: archreviews.com/field-notes · Connect: linkedin.com/in/tomward