We're Not Building a Piano: Full-Stack Infrastructure Understanding – Part 2

In our previous post (Design Patterns for Resilient Application Infrastructure – Part 1) we explored the basics of N+1 concept for node loss in a cluster and discussed what our series is going to cover. Let's start by mapping out the full-stack view as the IT architect will need to.

An IT architect needs to understand the physical layer (servers, storage, network), the data layers (relational databases, NoSQL databases), the application layers (application logic, code logic and code deployment), and the access layers (front-end load balancing and caching)

There are many stacks we have been exposed to over our IT careers. Today's "stack" is one that spans physical, virtual, cloud, and application infrastructure. We will expand this model as we go in the series, but let's start with this as our initial set of building blocks to work from:

- Access
- Application
- Data

Inside each block will be many services that support the needs at that logical layer.

Access Layer

The access layer is often known as the presentation layer. This layer includes our many facets of network access to reach our application front-end. The access layer may include just raw Layer-3 networking directly to a HTTP server, or it may also use proxies, distributed load balancers, firewalls bridging a DMZ and more. Understanding the needs of your application presentation will drive a lot of the underlying architectural decisions for the access layer. Some of your designs may also leverage existing technologies in place.

Application Layer

The access layer draws content from the actual application itself. Is your application a single node which hosts all code in a monolithic instances virtual machine? It may be a variety of services that are distributed for resiliency and to decouple from each other for better portability. How are your applications stored, deployed and updated? These are considerations for the application layer.

Data Layer

Applications are typically backed by data repositories. It could be just for read access or to interactively create/update/manage data in the back-end. Data may be distributed close to the application itself, and may be in a variety of forms. Relational databases, flat files, key-value stores, and many more options. Data can also be aggregated by other API-enabled services which makes understanding data and application architectures challenging.

Understanding Requirements for Application Resiliency

Put aside cost and complexity for a moment, and think about raw requirements. If you need resilient application infrastructure, you need to think about each of the three layers this is our next challenge

as we explore some options for distributed architectures to protect and deliver resiliency using some more direct product exploration and the more physical layer.

Using the CLP (Conceptual, Logical, Physical) layers for our applications means we can properly assess and architect each of the layers in the IT stack to meet the application resiliency requirements.

NEXT POST: Understanding the Access Layer