

Cloud Essentials 101

NA Partner Channels – IBM Cloud
Technical Evangelism Team



© IBM 2018

Welcome everyone to the Cloud Essentials overview session!

Agenda

- 1 • Objectives and Introductions
- 2 • Cloud Overview
- 3 • Cloud Architectures
- 4 • Microservices
- 5 • Serverless
- 6 • IBM Cloud

© IBM 2018

Today's agenda:

We will review the objectives for the session, then we will take a really brief look at what is cloud, and the basics of cloud architecture. From there we will make a dive into Microservices and serverless architectures, before concluding with a look at the IBM Cloud specifically and then we will go through our lab.

Course Objectives

- Upon completion of this presentation, you should be able to:
 - Describe basic Cloud information:
 - Types of Clouds
 - Types of Cloud Service Models
 - Cloud Architecture basics
 - Describe Microservices and microservice and serverless architectures
 - Understand IBM Cloud specific terminology
- Upon completion of the labs, you will have:
 - Created an IBM Cloud Account
 - Navigated the UI and Provisioned Basic IBM Cloud Services
 - Connected some APIs on the cloud platform

© IBM 2018

Just a quick review of our objectives for the day:

When we finish, you should be able to describe the basics of cloud architectures and types of cloud platforms

You'll be able to describe and understand the uses of and functions of a microservice, and serverless architectures

When we get through with the lab we will have Created a free IBM Cloud account, navigated and used the UI to deploy services, and built some basic API connections using the cloud platform

About Us



IBM Technical Evangelism Team

 [@IBMWolfPack](#)

 [team-wolfpack](#)

© IBM 2018

The IBM WolfPack is the technical innovation team dedicated to making it easy for you to find the right platform, tools, and education so you can change the world.

What is the Cloud?

Nobody understands the cloud!



WHAT IF I TOLD YOU

**THE CLOUD IS JUST SOMEONE
ELSE'S COMPUTER?**

© IBM 2018

The structure of cloud itself isn't earth shattering. It is computers running in someone else's data center so that you do not have to manage them. In addition there has been a major growth in services that cloud platforms provide that address specific needs that are complex or cumbersome to "hand-code" from scratch.

Cloud Models



PUBLIC

Public and open-by-design

Fast and on Demand

Easily configure, deploy and scale applications, services, and infrastructure

Shared environments with community-driven features, and shared platforms and infrastructure



HYBRID

Multi-cloud model options

A Mix of Many Options

Instant scalability of public cloud with support for critical enterprise integration

Features and flexibility of the public cloud with local control and management

Cloud to cloud and multi-cloud platform for resiliency and access to unique services



PRIVATE

Management and deployment options

Dedicated Flexibility

On Prem: Some features of public cloud, running on prem

Hosted: Dedicated "slices" of a public cloud running in a provider's environment

Cloud Pizza Architecture Models

Homemade

Type of Pizza
Dining Table
Soda
Electric/Gas
Oven
Gas/Electricity
Pizza Dough
Tomato Sauce
Toppings
Cheese

Take and Bake

Type of Pizza
Dining Table
Soda
Electric
Oven
Gas/Electricity
Pizza Dough
Tomato Sauce
Toppings
Cheese

Delivery

Type of Pizza
Dining Table
Soda
Electric
Oven
Gas/Electricity
Pizza Dough
Tomato Sauce
Toppings
Cheese

Dine-In

Type of Pizza
Dining Table
Soda
Electric
Oven
Gas/Electricity
Pizza Dough
Tomato Sauce
Toppings
Cheese

© IBM 2018

So what does this look like in practice?

<click> let's take an example of Pizza – when you want pizza and you are making it at home, you decide on the type of pizza, and then you need to provide all the things to make it and serve it. Home Made pizza is good, and it is exactly what you want, but it takes a long time to make.

<click> so let's get take and bake! I need to know my pizza type still, and I need to serve and cook it, but someone else makes it. That's easier, but I still have to wait for it to come out of the oven and if my oven is broken... no pizza.

<click> so let's do delivery! I pick the type of pizza, and it shows up ready to eat. I just need to set the table and maybe have some drinks. Faster still, but man I want it RIGHT NOW.

<click> dine in it is! Everything given to me, I just choose the type of pizza and eat. Fast, simple, and exactly what I need! <click to switch slide>

Cloud Architecture Models

On-Premises

Your Settings
Applications
Data
Runtime
Middleware
O/S
Networking
Virtualization
Storage
Servers

Infrastructure as a Service

Your Settings
Applications
Data
Runtime
Middleware
O/S
Networking
Virtualization
Storage
Servers

Platform as a Service

Your Settings
Applications
Data
Runtime
Middleware
O/S
Networking
Virtualization
Storage
Servers

Software as a Service

Your Settings
Applications
Data
Runtime
Middleware
O/S
Networking
Virtualization
Storage
Servers

© IBM 2018

That Pizza Model is exactly what the evolution of cloud platforms has been like:

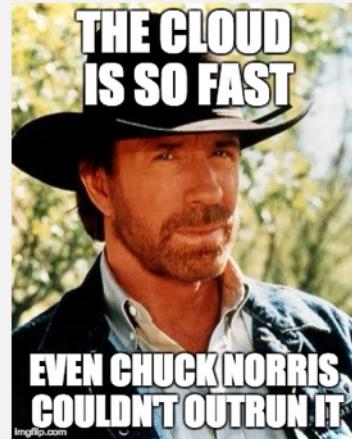
On prem = bake at home

IaaS is Take and Bake

PaaS is the delivery model, and lastly Software as a Service is full dine in. I always have to take care of what I want on my pizza (configurations), but someone else does the rest.

Top 10 Reasons to move to Cloud

- 1. Flexibility
- 2. Disaster Recovery
- 3. Automatic Software and Security updates
- 4. Capital Expenditure Free
- 5. Increased collaboration
- 6. Work from anywhere
- 7. Document Control
- 8. Security
- 9. Competitiveness
- 10. Environmentally friendly

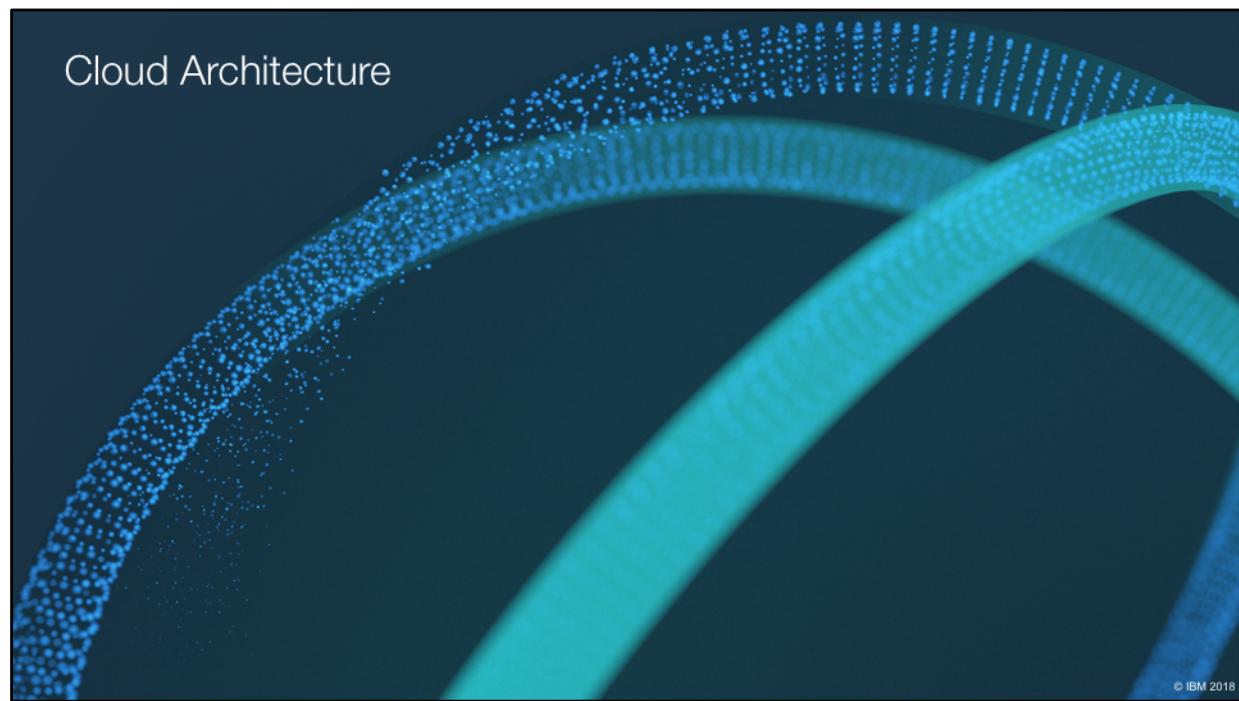


© IBM 2018

<Click for list to appear>

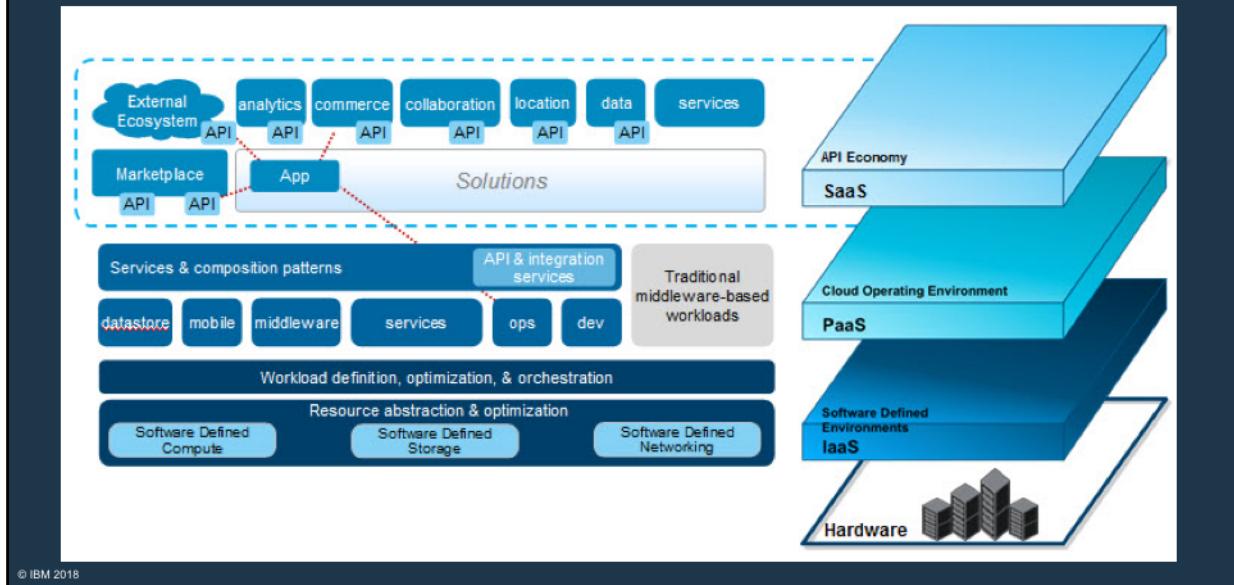
Key points – flexible, faster, Reduced capex

<click> for memes to reinforce those 3 key points



So let's look at the cloud architecture and see what makes up a cloud

The Cloud “Layer Cake” Architecture



This is a cloud. Any cloud. IBM, MS, AWS, they all look like this under the hood in some form.

Hardware at the bottom, IaaS, PaaS, and SaaS on top in order to make up the various service layers.

Increasingly people are moving to the top level and leaving the rest to the provider. That is the rise of the API economy and microservice architecture we have heard about. There are lots of huge advantages to moving to cloud, but in order to take advantage of them you need to have this architecture in mind.

<click to change slide>

The #1 Cloud Mistake

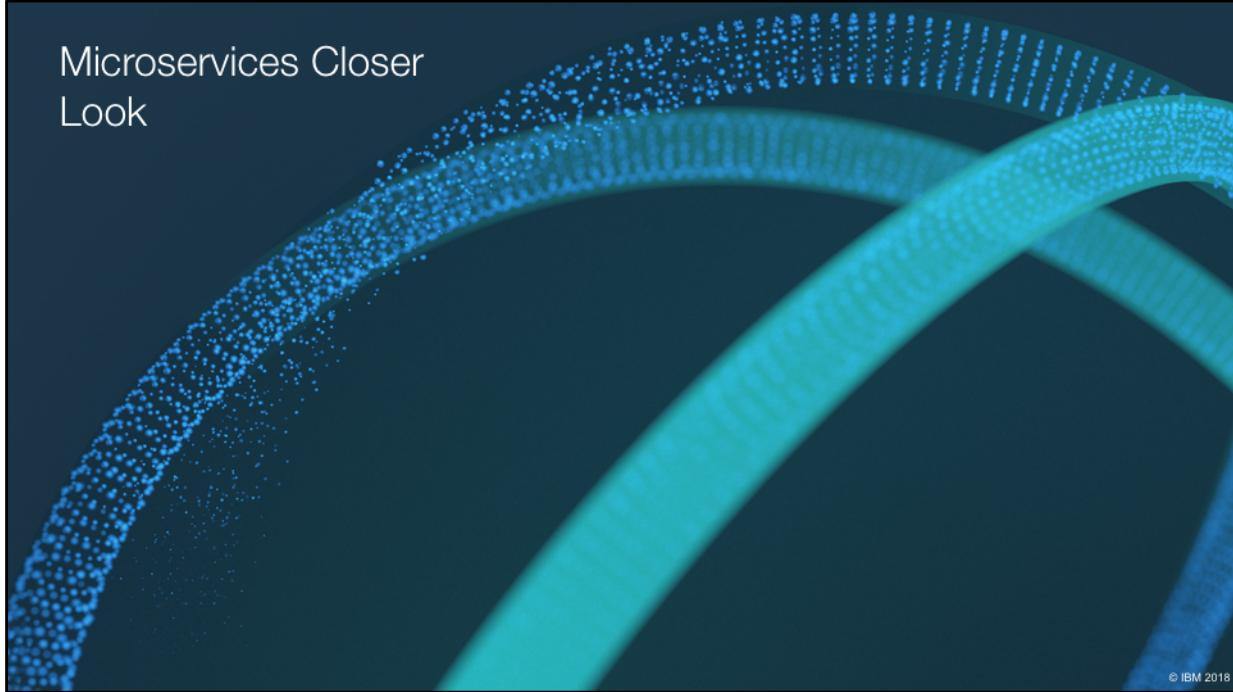


Cloud offers a lot of benefit, but only when you really embrace the full “move to cloud”

This takes a lot of time and unfortunately a lot of organizations just get here (POINT to “Lift and Shift”). There is good opportunity to move higher up the chain, but that’s not always easy and a lot of people bog down at just getting things replatformed and never even get to a recode.

<Click> the mistake is just dropping things in cloud “as is”. Why? Because putting lipstick on a pig? <click> yup – still a pig...

Microservices Closer Look

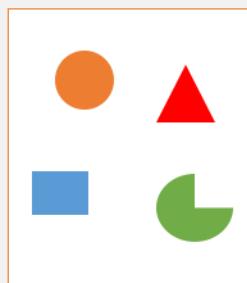


Let's dig into the idea of replatforming with microservices

Cloud Requires Rethinking Architectures

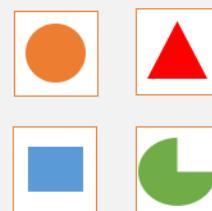
- Continuous Delivery and a Cloud First approach demands the use of Microservices

Monolith



All functionality in a single process

Microservices



Functionality broken down into separate services

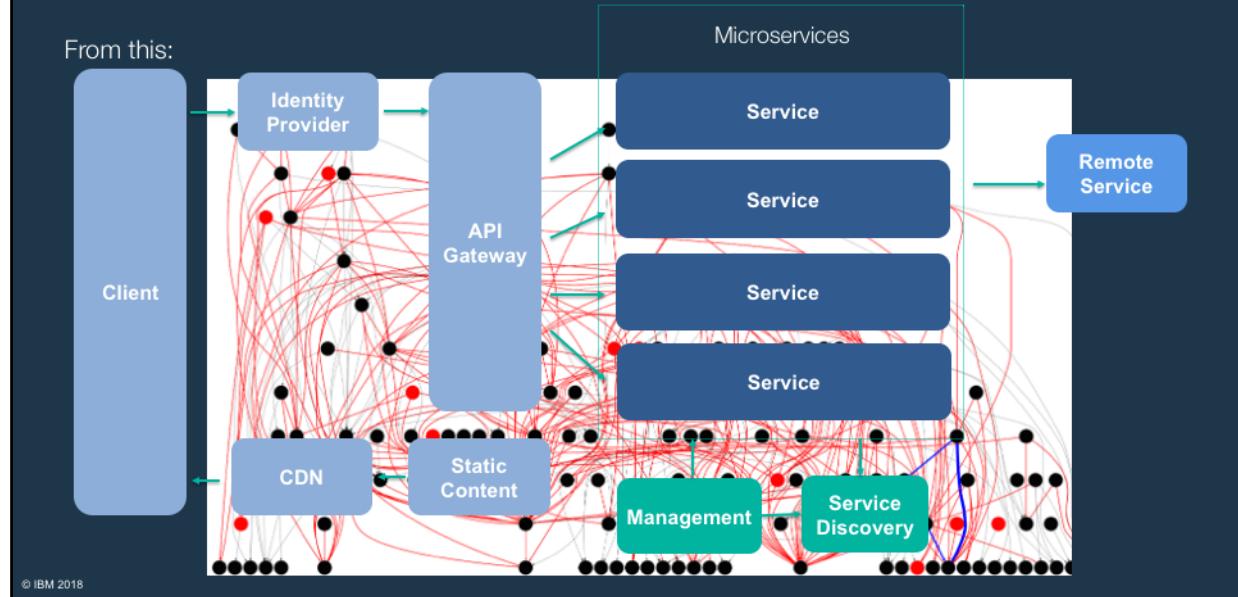
© IBM 2018

You have got to get the new way of thinking in your head

<click> Monoliths served us well for decades. They were the backbone of Web 1.0 and the genesis of everything we have today, but they were also VERY fragile and hard to manage. Everything was in one place and in a clustering of functionality in a single monolithic code base.

<click> Microservices take those functions and break them into their smallest functionality and use those as discrete services to make something operate. <Click to advance>

Microservices Architectures



Microservices allow you to go from This

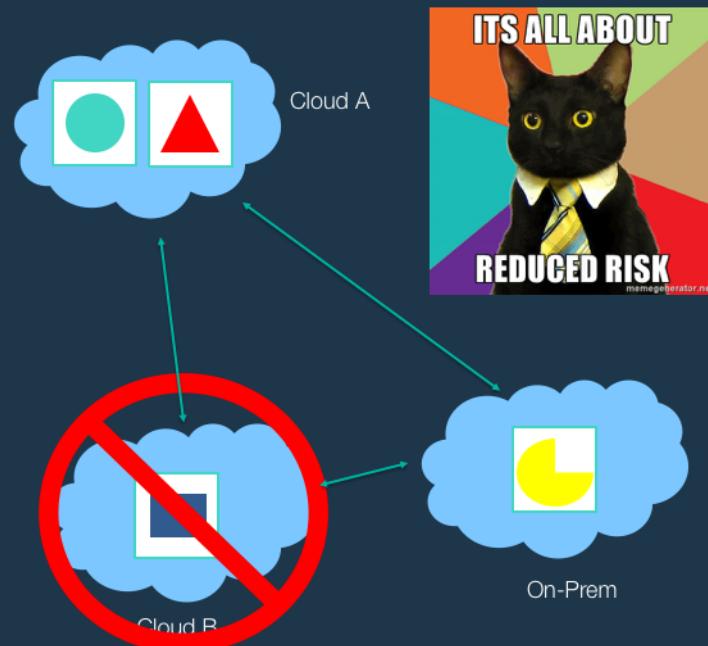
<click> to this. APIs working together with smaller functionality, and simpler to manage application parts, versus a giant single stack of code.

Considerations

Functionality that is:

- Broken down into separate services
- Managed by smaller teams
- Has Physical separation
- Has separate service providers

Requires consistent development and testing process, and architecture awareness to maintain integrity and service levels



Microservices make a lot of things easier. <click>

They divide things up and make them simpler and easier to manage. They provide slimmer code bases to maintain,

<click>can be distributed for load and scale,

<click>and generally make troubleshooting easier by making it easier to identify troublemakers.

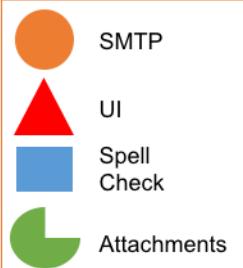
<click> but they require a consistent and repeated process in order to minimize defects – thus the rise of DevOps and things like Continuous integration and code pipelining.

<click> which are all about reducing the risk of failure

Microservice Example

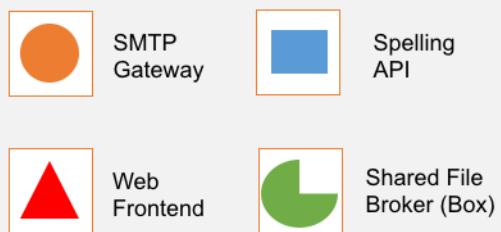
MICROSERVICES TAKE THE SMALLEST LEVEL OF FUNCTIONALITY AND DEPLOY IT AS A SEPARATE SERVICE

Monolithic Application



All functionality in a single process

Microservice Application



Functionality broken down into separate services

Let's say I wanted to create an email application <click> here's a monolith version of it:

The basic parts I need to send and receive mail

<click> now let's build it out of microservices

Same parts, smaller footprints, less failure/impact if a single service is broken <click to advance slide to next>



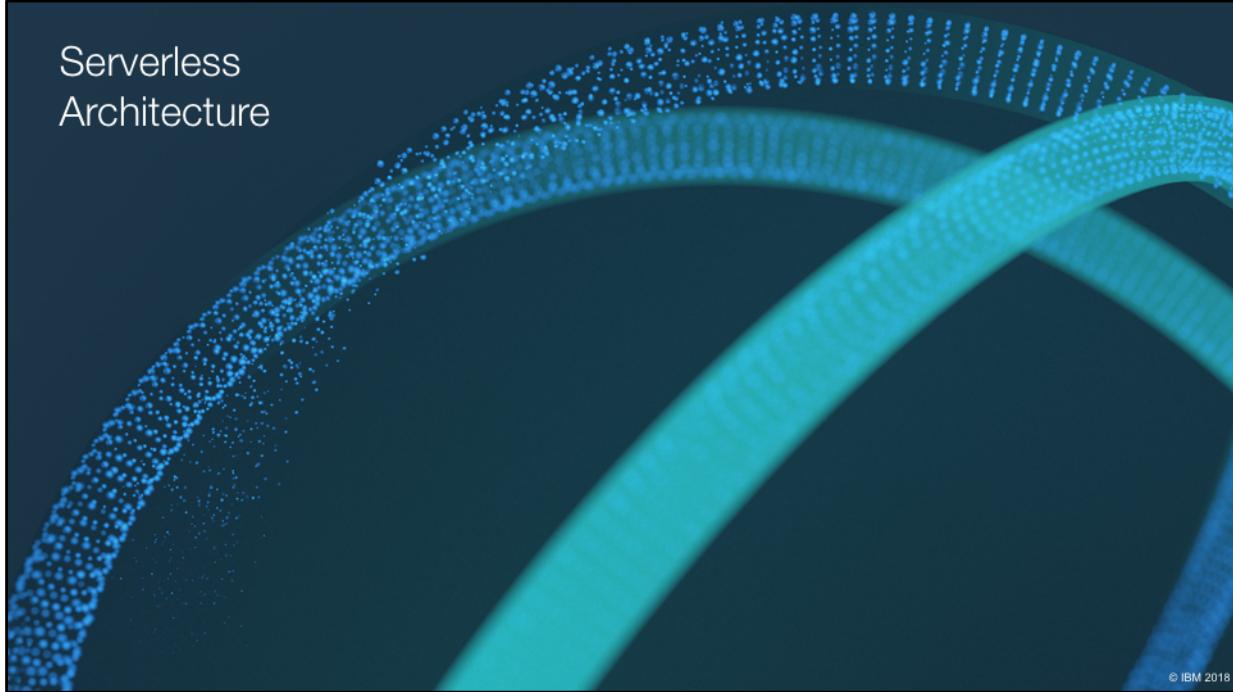
Now let's expand our minds. We are going to do a little exercise based on Netflix, one of the true pioneers of microservices.

(go to whiteboard and do the Netflix exercise - start with distro centers and physical DVDs and upgrade when done, come back and <click> to show the “death star” map of Netflix Microservices)

Sure there are a LOT of them, but each service may only be a few lines of API calls and code. Easily maintained by a single person. By spreading the load we reduce the burden on any one resource, human and code alike.

PAUSE FOR ?'s HERE

Serverless Architecture



Now let's look at Serverless!

Wait.... Serverless?

NOT REALLY

- Serverless abstracts the final layer of management into it's smallest footprint
 - There are still servers, you just don't manage them
 - You actually don't access them at all (and CAN'T)

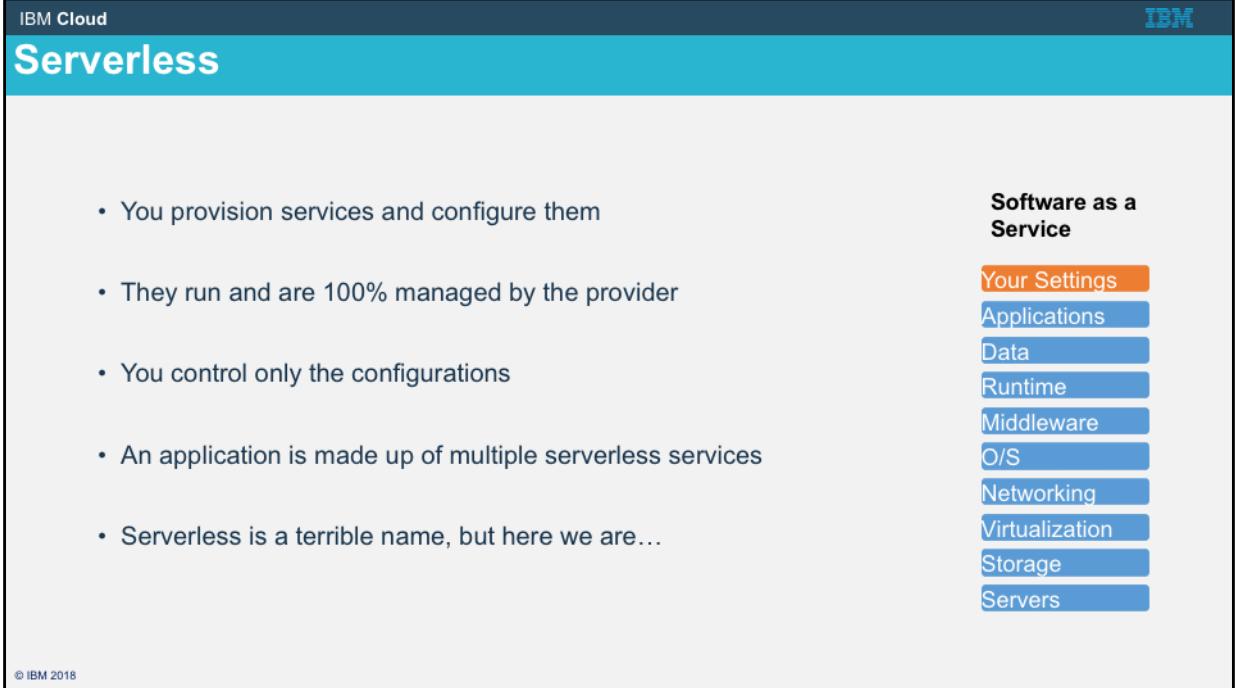


© IBM 2018

Um.... LOL WUT?

<CLICK>Serverless is not *really* serverless.

<CLICK>It is the final abstraction of services away from management and the last slimming of the footprint of management and configuration into the smallest footprint possible – the servers are there, you just don't (and in fact CAN't) see them at all.



The screenshot shows a web interface for IBM Cloud Serverless. At the top left is the "IBM Cloud" logo, and at the top right is the "IBM" logo. Below the header, the word "Serverless" is prominently displayed in a large, bold, white font on a teal background bar. The main content area contains a bulleted list of characteristics of serverless computing:

- You provision services and configure them
- They run and are 100% managed by the provider
- You control only the configurations
- An application is made up of multiple serverless services
- Serverless is a terrible name, but here we are...

To the right of the list is a vertical sidebar titled "Software as a Service" in bold. The sidebar items are arranged in a descending staircase pattern of colors: orange, blue, and teal. The items listed are:

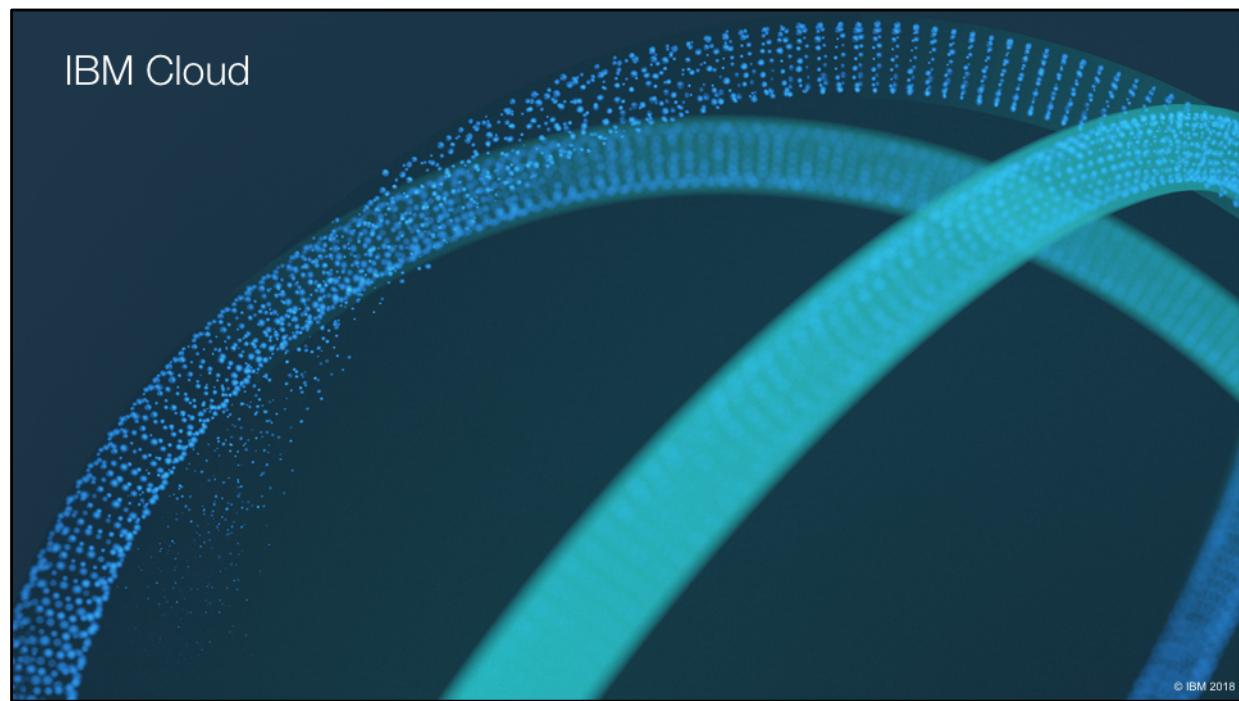
- Your Settings
- Applications
- Data
- Runtime
- Middleware
- O/S
- Networking
- Virtualization
- Storage
- Servers

At the bottom left of the main content area, there is a small copyright notice: "© IBM 2018".

The last note on serverless <CLICK>

You provision and configure them, they are managed 100% by the provider, there are multiple serverless services in an application, and it is a dumb name...

?s again



So let's talk about IBM's cloud in specific and get ready for the lab exercises!

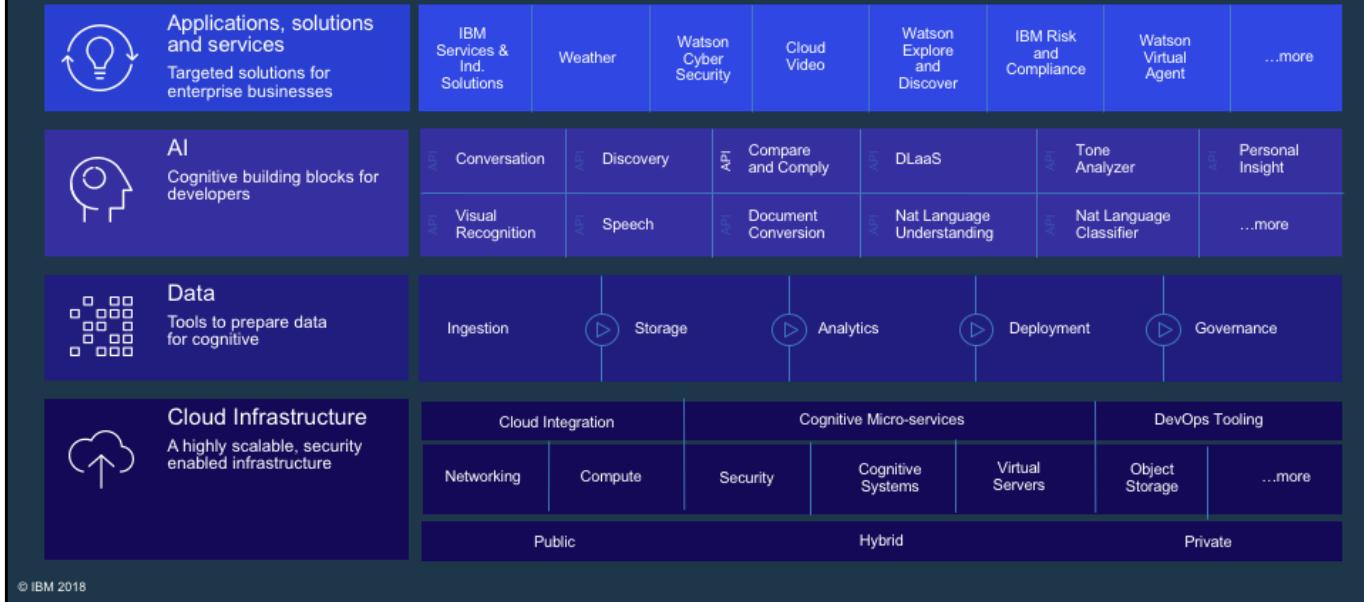
IBM Cloud Orientation

- Formerly Known as Bluemix
- Comprehensive catalog with Infrastructure and Platform Services, IBM developed, and Third Party Solutions

The screenshot shows the IBM Cloud Catalog interface. On the left, there's a sidebar with 'All Categories' and a search bar. The main area is titled 'Infrastructure' and contains sections for 'Compute', 'Storage', and 'Network'. Under Compute, there are 'Bare Metal Server' and 'Virtual Server'. Under Storage, there are 'Block Storage', 'File Storage', and 'Object Storage'. Under Network, there are no visible options.

IBM's cloud is a collection of infrastructure and services – our unique offering is, of course, Watson and the related API stack, but we also feature a number of 3rd party and IBM developed solutions, such as IoT, Blockchain, and Industry specific offerings.

IBM Cloud architecture – platform and services for business



© IBM 2018

Talking Points:

Our architecture is designed as an integrated whole across 4 dimensions.

- Our Cognitive **applications** are trained with more data and expertise than any competitor.
- Our sophisticated **AI** services are foundational to our architecture, and we allow for customized models. Our competitors only offer low-level AI without industry-specificity.
- Client **data** will never be used to train another client's AI. Our architecture allows for data use that is both open and with regulatory governance.
- Our **cloud** has been purpose-built to allow for accelerated AI training.

IBM Cloud

IBM Cloud Account

- **Organization:** groups all spaces together. Organization has administrative task.
- **Space:** contains application and services.
- **User account:** consists of username and password; username are usually email.
- **Quotas:** are size constraints for organization and spaces. used to control cost.

IBM Cloud Account

Organization

Space

Space

User 1

User 2

Organization

Space

Space

Space

User 1

Organization

Space

User 2

© IBM 2018

Cloud accounts are made up of multiple parts:

<CLICK> Accounts have organizations and spaces which are used to separate various functionalities and services into manageable groups – either logical, or simply hierarchical

<CLICK> User accounts can have permissions to multiple organizations

<CLICK> Quotas are for ensuring resource availability and usage restrictions and can be applied to a single space or a whole organization as needed

Let's Do Some Hands-On!

- Deploy some services
- Build some connections
- Write some code!

