# Purpose

Abstract

Strategy-level document specifying CUSTOMER’s approach to deploying applications on Amazon Web Services infrastructure. Documents CUSTOMER’s current approach to architecture, provisioning, and operations on AWS; and also provides the next steps to increase cloud adoption maturity.

Customer Sample– Cloud Playbook

Delivered by:

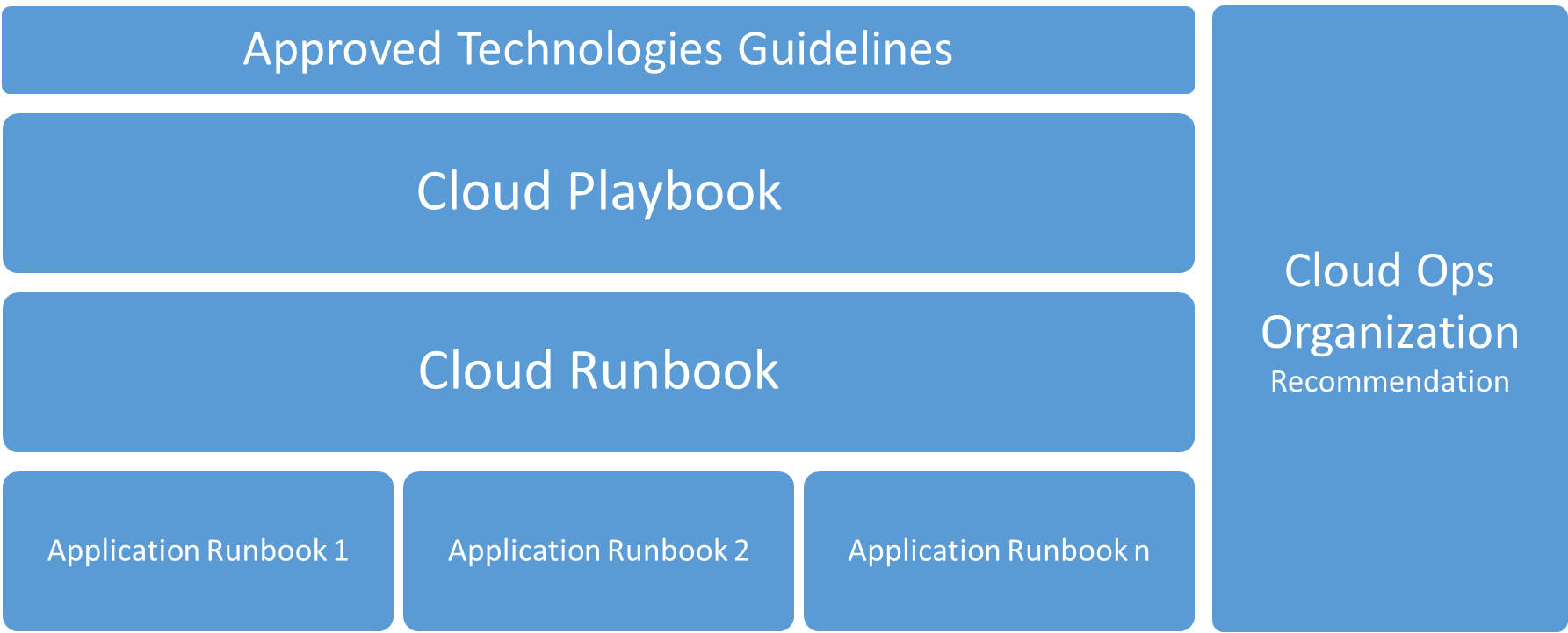
Professional Services

The purpose of this document, referred to herein as “Cloud Playbook” is to be a strategy-level document specifying CUSTOMER’s approach to deploying applications on Amazon Web Services infrastructure. The Cloud Playbook documents CUSTOMER’s current approach to architecture, provisioning, and operations on AWS; and also provides the next steps to increase overall cloud adoption maturity through iterations and incremental improvement.

The intent is for this to be a “living” document that is referred to often and continuously updated as the organization’s experience with AWS increases and the CUSTOMER business functions that surround it mature.

# Related Documents

The Cloud Playbook is part of a series of interrelated documents that address the topics of cloud adoption from technology selections, strategy formation, and cloud execution, all the way through application-specific Runbook details. The level of detail included in these documents is illustrated below, starting with the highest level strategy information at top, pushing down through the most specific and detailed at the bottom.



* Approved Technologies Guidelines represent a framework to guide the organization in selection of technologies to insure an efficient use of resources, improve time to market and the ability to maintain service level operations.
* Cloud Playbook is Strategy-level document specifying CUSTOMER’s approach to deploying applications on Amazon Web Services infrastructure. Documents CUSTOMER’s current approach to architecture, provisioning, and operations on AWS; and also provides the next steps to increase cloud adoption maturity.
* Cloud Runbook is an execution level document that provides the technical detail of how to operate AWS infrastructure.
* Application Runbook(s) provide a view into application specific exceptions from the standard cloud guidelines as outlined in the Cloud Runbook.
* Cloud Ops Organization outlines the resources, skills and teams that are involved in driving the cloud adoption (COE) as well as those who manage the day to day provisioning, usage, monitoring, and cost of cloud resources (Cloud Engineering).

# Operations Overview

As organizations adopt cloud services, the operations organization must transform in order to fully realize the benefits of cloud technologies. Operationalizing your cloud service offering includes adapting the services offered, the manner in which they are offered as well as who and how they are offered. This document is to be a living document that outlines the IT service management features like proactive monitoring, patch, continuity, and financial management, and processes like change and incident management. There are options identified that can be implemented dependent upon the maturity of your organization.

For each category there is a section which identifies the initial strategy that Customer has selected as well as an identified growth opportunity. The Cloud Runbook will expand upon the option Customer has selected and provide prescriptive steps to implement the desired approach.

The specific categories are as follows:

|  |  |
| --- | --- |
| Operations Category | Sub-categories |
| Platform Operations | Tagging  Application Service Level Classification  Application Recovery Classification  Patch Management  Platform Logging |
| Provisioning & Service Catalog | Infrastructure Provisioning  Service Catalog |
| Availability Management | Backup & Recovery (DR)  High Availability |
| Application Lifecycle Management | Application Architecture Patterns  Security Group Patterns  Platform Monitoring  License Management  Network Architecture Patterns  Load Balancers  Firewall Management  Application Logging  Resource Cost Optimization  Configuration Management |
| Financial Management | Chargeback Process  Billing Method  Budgeting and Forecasting  Reporting |

To be optimal, service management and operational functions must be proactive and supported by automation, as opposed to reactive and supported by manual human intervention. This applies both to deploying resources and automating responses to potential issues by designing for failure. Repetitive manual tasks should be reduced through the use of automation to allow operations teams to focus on value-add work.

Operational areas that are important to address early in the cloud journey include Financial Management, Platform Operations, Monitoring & Incident Resolution, Provisioning & Service Catalog, Availability Management and Application Lifecycle Management. Similar to the iterative approach of agile development, there will be an iterative approach to cloud operations.

# Platform Operations

|  |  |  |
| --- | --- | --- |
| Activity | Traditional Approach | Cloud Approach |
| Tagging | Tagging of all resources across an enterprise is not a common feature found in most organizations today. Most organizations utilize a CMDB to track the assets according to cost center or project codes, entering the relevant attribute information in the CMBD manually as the equipment is purchased and then again when it’s provisioned. Usually updates to the CMDB are manual and lag the actual change, especially in scenarios where CMDB’s are used to track the inter-dependencies of applications. | Tags are the building blocks of cloud resource reporting, but they also are functional and are used in everything from auto-scaling to license management. Due to the ephemeral nature of resources in the cloud, it is not necessary to log cloud resources in the same static pattern as physical resources. Instead, a tagging strategy is employed to log resources for tracking, dependency, and chargeback purposes. Additionally, the attributes of the cloud resources being utilized can be readily identified using AWS Config, alleviating the need for manual entry and low-value data entry time.  Key Element to a Cloud Governance Framework   * Cost Transparency * Automation Enablement * Policy Enforcement * Application Correlation * Object Organization   Tags are used by many groups including; Finance, Operations, IRM/Security, Application Owners, and others. Document the Key Fields identified for each report:   * Field Values, Length, Formats * Logical Association of the fields * Typical fields to look for:   + Line Of Business   + Cost Center   + Version   + Owner   + Compliance Domain   + Name   + Environment   + Application   + Tier   Please see the AWS whitepaper which explores various tagging strategy and best practices: <https://d0.awsstatic.com/aws-answers/AWS_Tagging_Strategies.pdf> |
| Application Service Level Classification (SLO/SLA) | Most organizations will define several application service level classifications to communicate the desired uptime of an application (SLO/SLA) of an application. In traditional approaches the SLO/SLA is measured in minutes/days and achieving these metrics is an expensive proposition. | Most customers when moving to the cloud will utilize their existing classification process, with the addition of specifying their classifications with AWS resource tags. In cloud approaches service levels are generally identified in terms of minutes/hours and can be cost effectively architected to achieve these levels.  Patterns:   * Utilize existing classification process without any changes * Ensure AWS services being utilized adhere to your service level classifications * Allows for future improvements with service uptime to grow with evolving business needs |
| Application Recovery Classification (RTO/RPO) | Most organizations will define several application recovery classifications to communicate the desired level of acceptance regarding data loss in the event of a disaster for an application. These are calculated using Recovery Point Objective (RPO) and Recovery Time Objective (RTO). | Most customers when moving to the cloud will utilize their existing classification process, with the addition of identifying their classifications with AWS resource tags.  Patterns:   * Utilize existing classification process without any changes * Ensure AWS services being utilized adhere to your recovery level classifications * Allows for future improvements with recovery times to grow with evolving business needs |
| Patch Management | Most organizations have a standard monthly patch management cadence to maintain or update operating system and off the shelf applications such as database and web hosting software. Critical patches often follow an exception process based on potential business impact.  Typically patching is performed across the enterprise using tools such as Chef, Puppet, and Ansible, Salt, private repos, WSUS, and Microsoft SCCM. | Some organizations take advantage of their existing investment in patch management tools and processes. Often when multiple tools and differing processes are used to address patch management, organizations coalesce around a single enterprise configuration management solution. Below are some of the common options used to address patch management:  Options:   * Some organizations when moving to the cloud will initially utilize their existing patch management process assuming their tool is cloud-aware and/or has the capability to perform the needed functions on the cloud * When using Auto Scaling with EC2, patching can be simplified by updating a single custom Amazon Machine Images (AMI) and re-launching. * As you adopt services such as Amazon RDS and Amazon ElasticBeanstalk you can take advantage of managed platform updates which handle the patch management for these services for you. * As you adopt fully managed services such as Amazon DynamoDB, Amazon SQS, and AWS Lambda, patch management is handled entirely for you. * Many customers today utilize a number of patch management tools and have consolidated around a single solution such as AWS OpsWorks or Chef. |
| Platform Logging | Most organizations do not have a consistent approach or tooling when it pertains to platform logging. However, a pattern of aggregating logs to provide visibility into the enterprise view with various third party tools (ie. Splunk) has begun to emerge. | Organizations building or migrating applications in the cloud have an opportunity to design centralized platform logging as part of the initial architecture.  This brings new insight into the health of a given environment and reduces incident resolution times and drive improvements to the solution.  Below are the options available to support a platform logging solution in the cloud.  Options:   * Amazon CloudWatch is a logging and monitoring service. CloudWatch tracks metrics, monitors log files, sets alarms and reacts to changes in your AWS resources. Amazon CloudWatch is used to gain system-wide visibility into resource utilization, application performance, and operational health * Amazon CloudWatch logs can be consumed by your existing log aggregate tool. * AWS Marketplace and the open source community offer a variety of solutions for centralized logging and analytics |
| Configuration Management | Configuration Management Databases (CMDB) was originally designed as a database to store records of configuration items (CI) throughout their lifecycle as well as their relationships with other configuration items. The CMDB has been traditionally populated with data from static physical environments with long lifecycles. This traditional configuration data is not going to be appropriate for optimized operations in a cloud environment due to the ephemeral nature of cloud services like IP addresses, instance names, and storage volumes. The industry is changing to adapt to capturing not just configuration and relationship information, but also performance information on infrastructure, applications, and services.  Configuration Management Databases will need to be evaluated and structured or modified to reflect data that is really needed to run operations in the cloud. With the dynamic nature of the cloud and capabilities such as auto-scaling and build-test-destroy, Configuration Items may be provisioned and decommissioned within hours. Organizations need to determine what the critical data elements are for each component, application or service; how often they want to update the CMDB; and how to utilize that data. Cloud providers offer multiple ways of configuration management – integration between the provider and client tool, federation of tools, or use of the provider tools. An appropriate tagging strategy and consistent and accurate tagging implementation are critical to identifying and tracking cloud resources.  A CMDB for cloud computing will be even more valuable when micro-services and container architectures are deployed in complex, dynamic, ever-changing cloud configurations. Documenting the relationships among CI’s is paramount when performing any type of compliance assessment, threat impact assessment, or troubleshooting efforts.  In addition, a new type of CI will be seen as more automation is created. Infrastructure as code provides the foundation for IT infrastructure deployment automation and should be captured in industry-standard notation and stored in configuration management tools, just like code. From there the organization can move into continuous integration / continuous deployment (CI/CD) to combine infrastructure and software deployment automation. As code templates are developed and used in the service catalog and to deploy full application stacks, those templates will need to be put through processes such as change control when modified. | Most organizations embrace some of the changes involved in their configuration management process due to the elastic nature of using cloud resources. Review the options below to determine the level of configuration management integration with cloud resources. See Tagging section of this document for more information.  Options:   * Enable AWS Config service to provide dynamic configuration change history and integrate the information into your existing CMDB. This could be a simple pointer to AWS Config   + Ensures the CMBD is always up to date through an automated process.   + Support new capabilities such as resources that automatically are provisioned and de-provisioned with a short life cycle such as Amazon EC2 Auto Scaling. * As your organizational adoption matures, it is possible to build API based integration between AWS and your existing CMDB. A second layer of deeper information that may not be supported or valuable to log in the master CMDB system regarding the cloud infrastructure state, changes, API calls, etc. Instead, this information can be logged in a CMDB-like application hosted by AWS (e.g. AWS Config). Access to this information can be hot linked to the master CMDB and quickly viewed at any time. * Alternatively, it is possible to use AWS Lambda scripts that auto-update their CMDB when resources are deployed or terminated   + If a resource isn't properly tagged, a script can shut it down |

|  |  |
| --- | --- |
| Initial AWS Approach for CUSTOMER (Refer to Runbook for execution details) | |
| Tagging | * CUSTOMER will follow the AWS recommended approach to use tags to identify application service level, cost center, environment type, cost center, organization and more. Specific details of the tagging strategy are described in the Cloud Runbook. * An emphasis on enforcing tagging at launch time to ensure all costs are accounted for and allocated appropriately will be accomplished through required CloudFormation variables and the internal code review.   Growth Path:   * In addition to enforcing tagging at launch, using AWS Lambda (serverless computing) scripts will be used to search and report on resources that are not tagged. |
| Application Service Level | * CUSTOMER will use the existing workflow and process for identifying the Service Class of an application and will also continue to use the Application Recovery Classification using Recovery Point Objective (RPO) and Recovery Time Objective (RTO) as a measure. The currently established RTO/RPO’s are detailed below. * Once the application is identified into a Service Class, that classification will determine the architecture template (see: Application Architecture Patterns section of this document) that best fits the application requirements. * Service Level is directly related to RTO/RPO below. |
| Application Recovery Classification (RTO/RPO) | Current CUSTOMER Recovery Tier Standards:   * Recovery Tier 1 (Mission)   + Time Objective: 24 hours   + Point Objective: ~4 hours * Recovery Tier 2 (Mission)   + Time Objective: Up to 48 hours   + Point Objective: 24 hours * Recovery Tier 3 (Business)   + Time Objective: Up to 72 hours   + Point Objective: 24 hours * Recovery Tier 4 (Business)   + Time Objective: up to 168 hours   + Point Objective: 24 hours * Recovery Tier 5 (Operational)   + Time Objective: 169+ hours   + Point Objective: 24 hours   Growth Path:   * Current RTO/RPO to be evaluated in the future against the cost/benefit of increased availability strategies using AWS. It’s understood that better RTO/RPO’s are attainable a lesser cost on AWS than with traditional architectures with physical data center limitations. Given this, CUSTOMER will take their current Service Class metrics under review in the near future to evaluate whether their RTO/RPO’s are valid given the new capabilities on AWS. |
| Patch Management | * CUSTOMER will explore the use of fully managed (e.g. RDS, DynamoDB, Elastic Beanstalk, etc.) and/or serverless (Lambda) cloud services for every project in an effort to reduce the future need for patching servers on every application. * When patching is required, CUSTOMER delineate between applications that are treated like “pets” vs. “cattle” and will use at least three slightly different approaches for managing and deploying patches to AWS EC2 Compute instances. These approaches will all use scripting and should be repeatable across many instances at one time. These approaches are detailed in the Cloud Runbook.   Growth Path:   * Continued maturity down the path of using more cloud managed service that do not require patching; and when that is not an option, continued maturity in creating application architectures/choosing software vendors that provide the ability to be cloud aware. |
| Platform Logging | * CUSTOMER will use CloudWatch logs with CloudTrail and VPC Flow Logs to review network, security, and system related health and load. Platform Logging will leverage the tagging strategy to report and subsequently notify, regarding resources outside the normal operating ranges.   Growth Path:   * These logs will be sent to the existing CUSTOMER Splunk implementation for further analysis, storage, and or action. |
| Configuration Management | * CUSTOMER will use AWS Config to view real-time AWS resource inventory, configuration history, and enable configuration change notifications. * AWS infrastructure will initially not be stored in the CUSTOMER enterprise CMDB due to the rate of change and the ephemeral nature of the AWS resources; including: instance types, ID’s, IP’s, storage volumes, test/dev environments, etc.   Growth Path:   * This real-time view will enable security and governance of the platform once these individuals are trained and/or are able to consume the outputs of AWS Config into their current systems. An accompanying service, AWS Config Rules will enable CUSTOMER to continuously monitor configuration changes to AWS resources and provide a new dashboard to track compliance status, so an administrator can quickly determine when and how a resource went out of compliance. * CUSTOMER may also explore the case and options for writing Change and Configuration Management details back to their enterprise CMDB |