Modernizing Technology Governance

Module – 2 Workflows

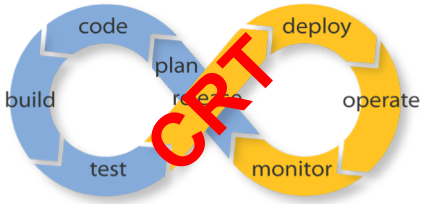
Tradition verses Cloud (Modernized) – Governance

Tradition – Governance

- Information and technology (IT) governance is a subset discipline of corporate governance, focused on information and technology (IT) and its performance and risk management.
- The interest in IT governance is due to the on-going need within organizations to focus value creation efforts on an organization's strategic objectives and to better manage the performance of those responsible for creating this value in the best interest of all stakeholders

Cloud – Governance

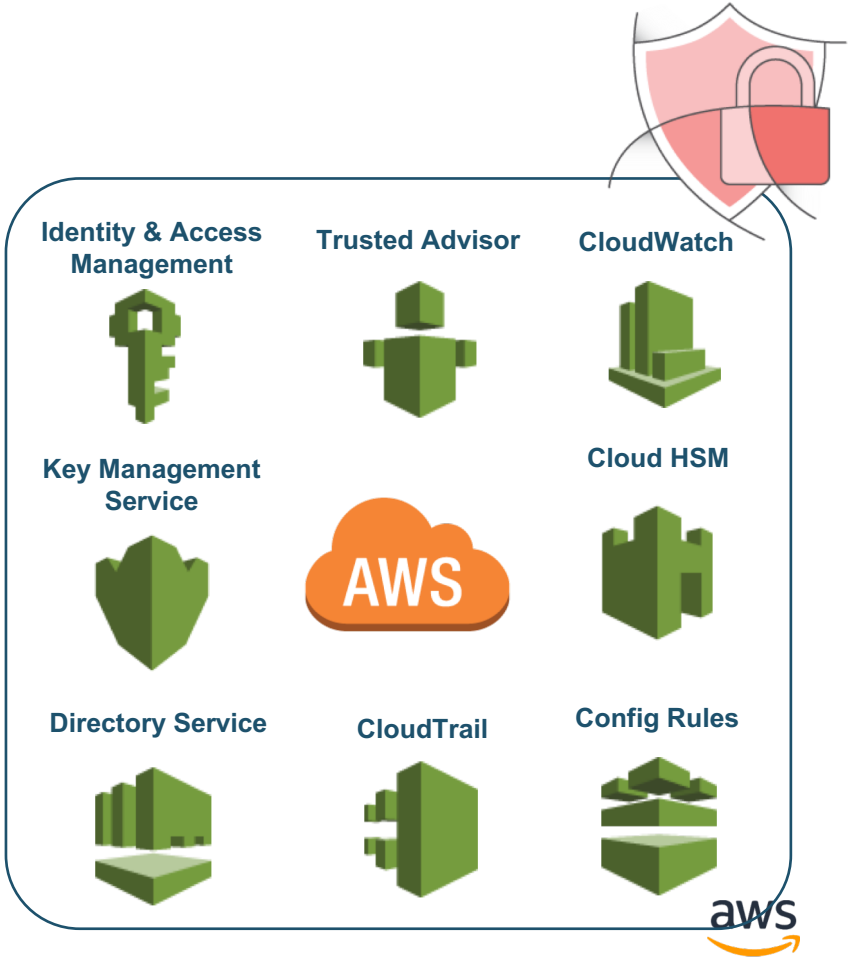
- Technology drives your governance alignment
- Governance is a “Shared Responsibility”
- Automation is the ***Key*** to successful governance
- Pre-Cloud decision making process are paramount (e.g. service selection, policies, frameworks architecture, data protections, etc.).
- Focus is on Continuous Risk Treatments (CRT)



Security by Design

Security by Design (SbD) is a security assurance approach that formalizes AWS account design, automates security controls, and streamlines auditing.

Instead of relying on auditing security retroactively, SbD provides security control built in throughout the AWS IT management process.



Security by Design - *Design Principles*

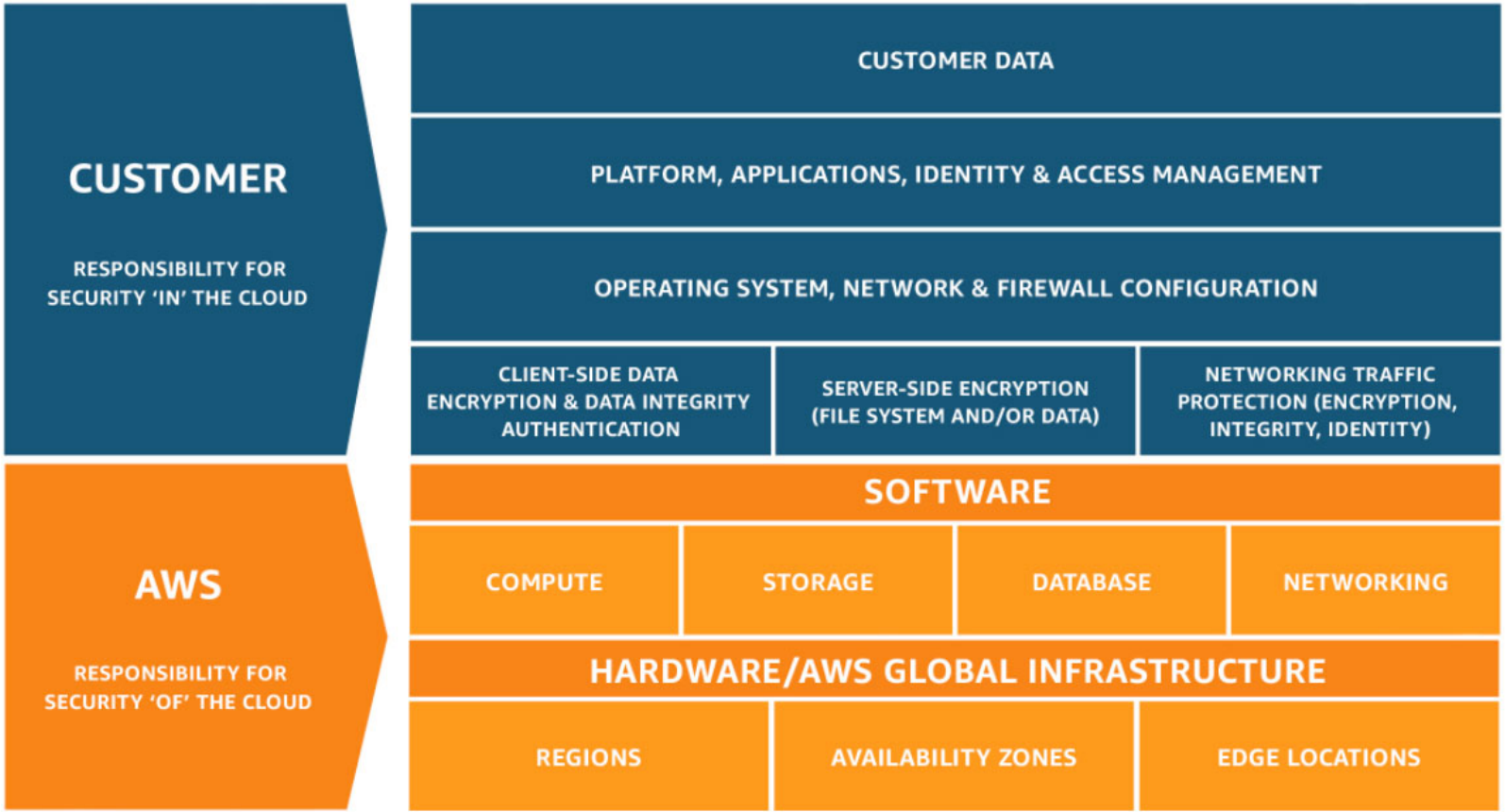
Developing new risk mitigation capabilities, which go beyond global security frameworks, by treating risks, eliminating manual processes, optimizing evidence and audit ratifications processes through rigid automation

- Build security in every layer
- Design for failures
- Implement auto-healing
- Think parallel
- Plan for Breach

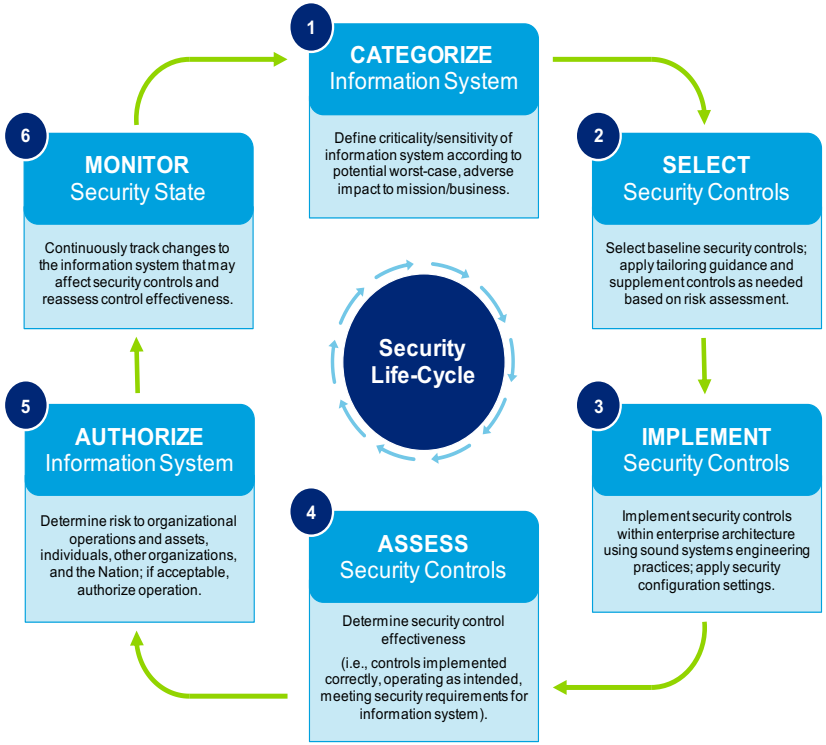
- Don't fear constraints
- Leverage different storage options
- Design for cost
- Treat Infrastructure as Code
    - Modular
    - Versioned
    - Constrained

aws

AWS Shared Responsibility Model



Traditional - Risk Management Framework



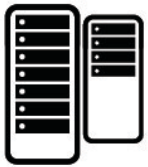
aws

Cloud - Risk Management Framework

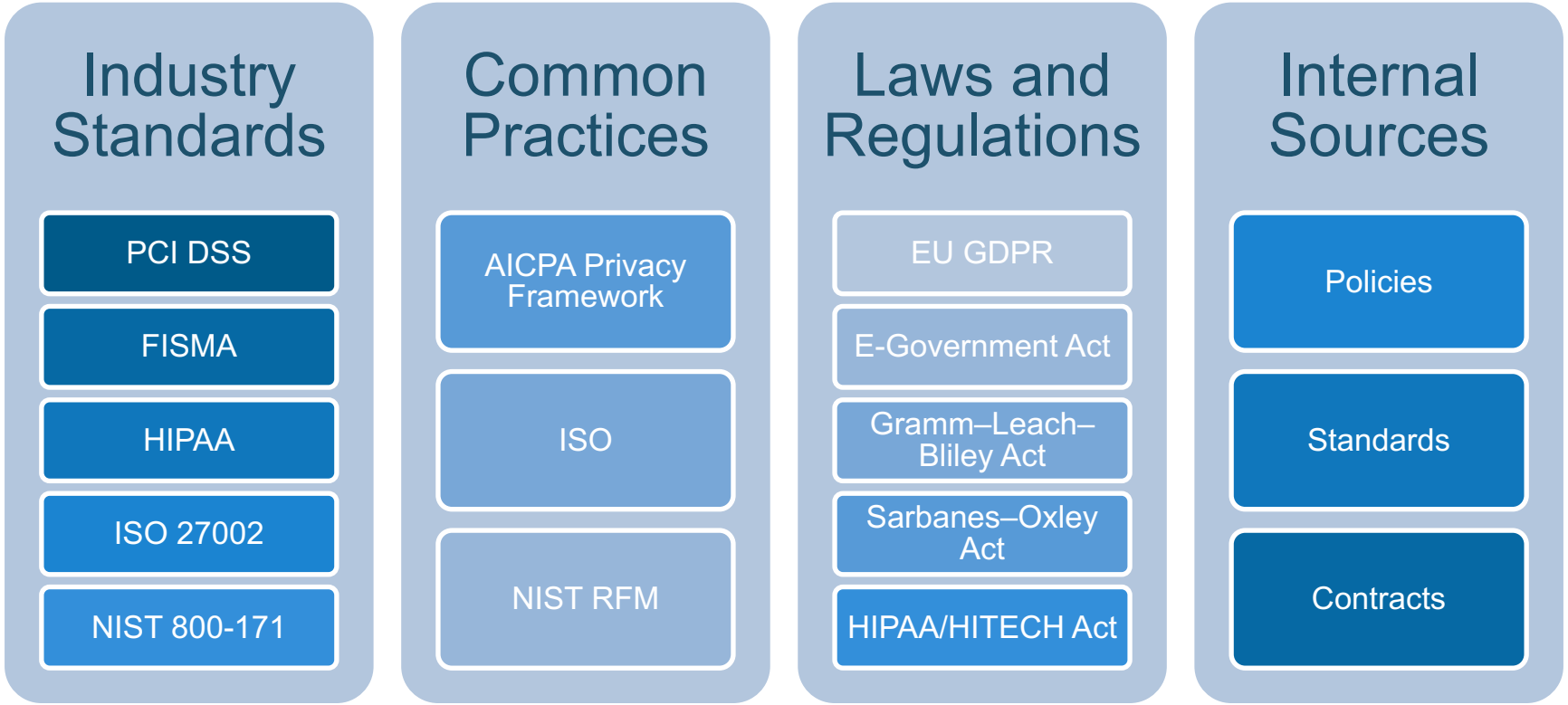
Modernizing Technology Governance (MTG)

1 DECIDE WHAT TO DO (STRATEGY)	 1.1 IDENTIFY STAKEHOLDERS	 1.2 IDENTIFY YOUR WORKLOADS
2 ANALYZE + DOCUMENT	 2.1 RATIONALIZE SECURITY REQUIREMENTS	 2.2 DEFINE DATA PROTECTIONS + CONTROLS	 2.3 DOCUMENT SECURITY ARCHITECTURE
3 AUTOMATE, DEPLOY + MONITOR	 3.1 BUUILD/DEPLOY SECURITY ARCHITECTURE	 3.2 AUTOMATE SECURITY OPERATIONS	 3.3 CONTINUOUS MONITOR	 3.4 TESTING + GAME DAYS
4 CERTIFY	 4.1 AUDIT + CERTIFICATION			

Phase 2 - Analyze, Define and Document

2 ANALYZE + DOCUMENT	 **2.1** RATIONALIZE SECURITY REQUIREMENTS	 **2.2** DEFINE DATA PRODUCTION + CONTROLS	 **2.3** DOCUMENT SECURITY ARCHITECTURE

Rationalizing the Security Requirements



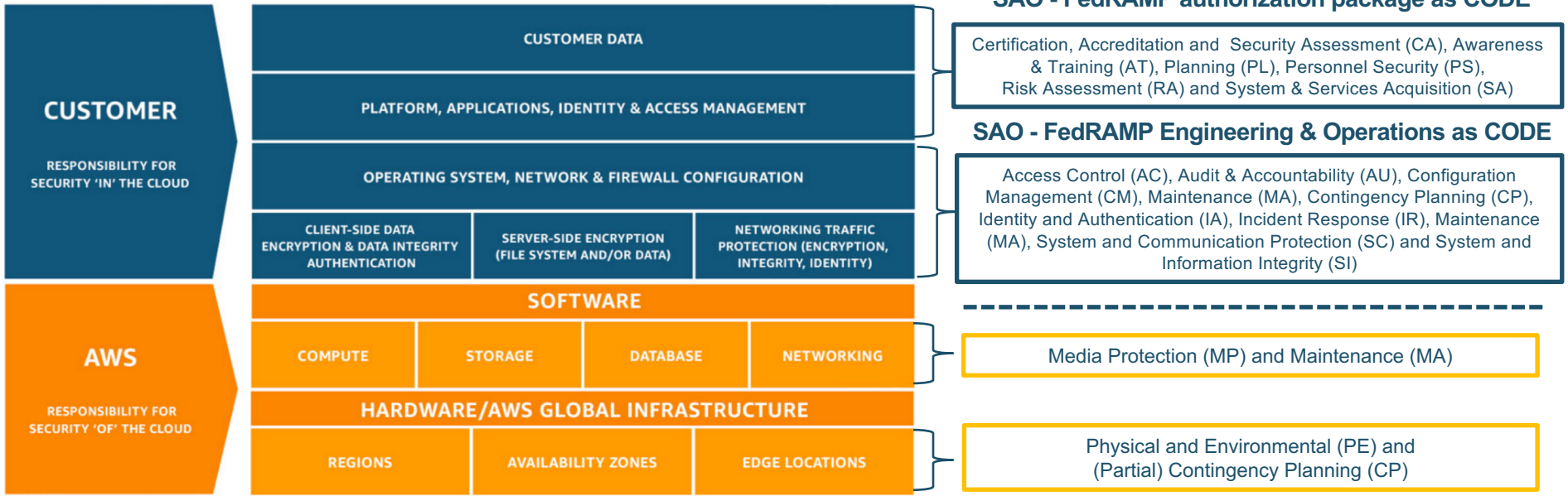
aws

Shared Responsibility Control Types

Control Type	Description
Inherited Controls	Controls which a customer fully inherits from AWS. (e.g. Physical & Environmental)
Shared controls	Controls which apply to both the infrastructure layer and customer layers, but in completely separate contexts or perspectives. In a shared control, AWS provides the requirements for the infrastructure and the customer must provide their own control implementation within their use of AWS services. Examples include: • Patch Management – AWS is responsible for patching and fixing flaws within the infrastructure, but customers are responsible for patching their guest OS and applications. • Configuration Management – AWS maintains the configuration of its infrastructure devices, but a customer is responsible for configuring their own guest operating systems, databases, and applications. • Awareness & Training - AWS trains AWS employees, but a customer must train their own employees.
Customer Specific	Controls which are solely the responsibility of the customer based on the application they are deploying within AWS services. Examples include: • Service and Communications Protection or Zone Security which may require a customer to route or zone data within specific security environments.

https://aws.amazon.com/compliance/shared-responsibility-model/

Shared Responsibility – FISMA, FedRAMP & DoD



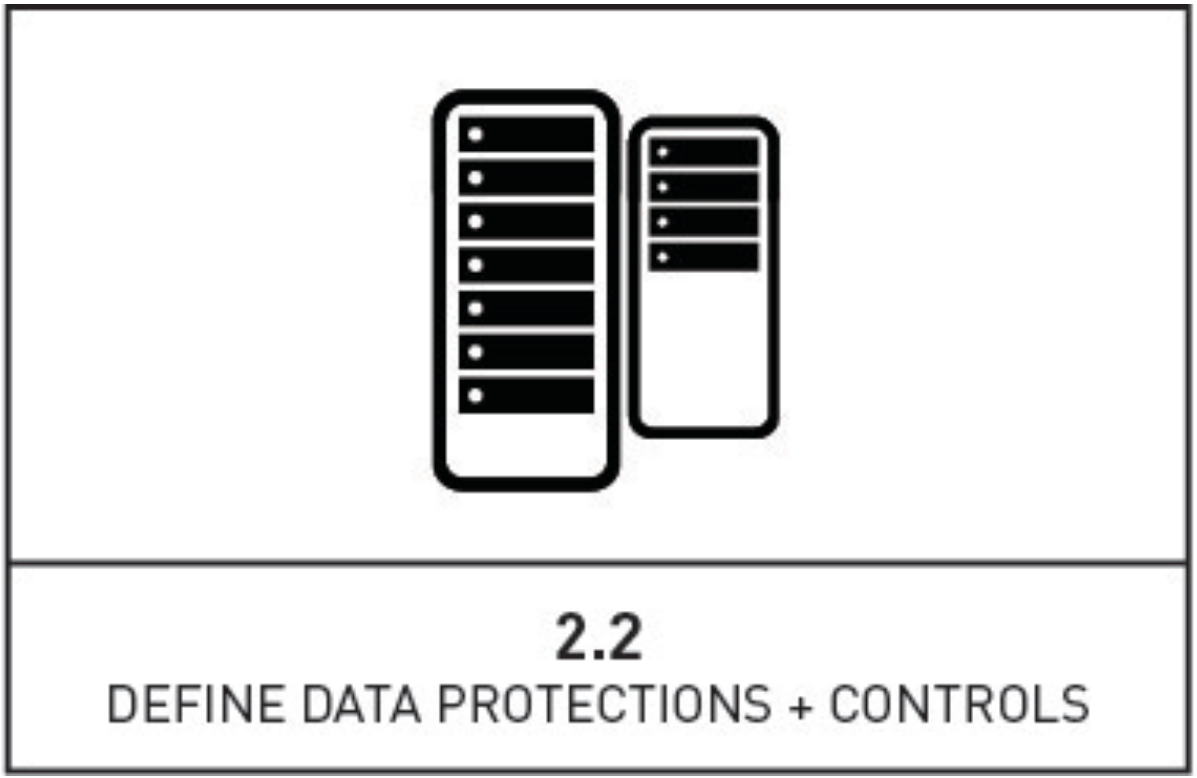
AWS Artifact - FedRAMP Partner Package Customer Responsibility Matrix (CRM)

https://aws.amazon.com/artifact/

AWS – FedRAMP (M & H)
DoD SRG (IL4 & IL5)
DFARS

AWS GovCloud (US)
AWS East/West

Define Data Protections

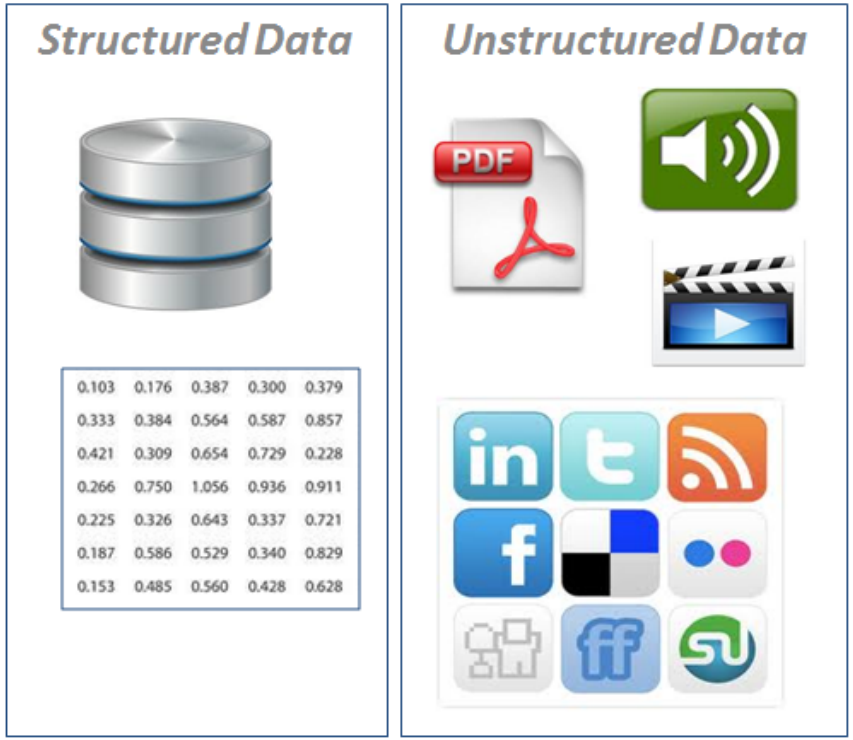


Problem Statement…

Issue #1 – The majority of organization do not have a mature “Data Classification” policy, process or user education schemes for internal use of data.

Issue #2 – Most organizations do not have a clean single source of “Truth” for what is their authoritative source for data. (Structured or Unstructured).

Issue #3 – Most organizations do not have an “Data Lifecycle” policy, procedure and/or operational processes for how data should be derived, protected, used, secured, transferred, achieved and destroyed when no longer relevant.



Data Protection Requirements

There are a number of regulatory, standards and frameworks which can impact data cloud computing.

- US - Health Insurance Portability and Accountability Act (HIPPA)
- US - Health Information Technology for Economic and Clinical Health Act (HITECH Act)
- US - Consumer Data Security and Notification Act (Amendment to Gramm-Leach-Bliley Act)
- EU - Directive 95/46/EC of the European Parliament and of the Council
- EU - Directive 2002/58 on Privacy and Electronic Communications (e.g.-Privacy Directive)
- General Data Protection Regulation (GDPR) (Regulation (EU) 2016/679)
- Canada - The Personal Information Protection and Electronic Documents Act (PIPEDA)
- UK - Data Protection Act 1998 (DPA)
- Australian - The Federal Privacy Act 1988
- Japan - The Act on the Protection of Personal Information (“APPI”)
- Singapore - Personal Data Protection Act 2012
- Philippines - Data Privacy Act of 2012
- South Korea - Personal Information Protection Act (“PIPA”)
- Hong Kong - The Personal Data (Privacy) Ordinance (Cap. 486) (“Ordinance”)

Data Protection Considerations

- **Access Controls**: Ensure organization can manage access to their content, services and resources independently of the cloud service provider.
    - Additionally, ensure cloud provider supports advanced set of access controls (MFA), encryption, and logging features.

- **Contractually**: Ensure the cloud providers does not access or use customer content for any purpose other than as legally required and for maintaining the cloud services.

- **Storage Locations**: Organization should be able to choose the geographic location in which their content will be stored. Cloud Providers should not move or replicate customer content outside of the customer's chosen locations.

- **Security**: Customers should choose how their customer content is secured. Through the use of various encryption of content in transit or at rest.
    - Additionally, organization should ensure they have option to manage their own encryption keys.

(Security Controls + Data Protections)

- Define your architecture capacity in advance
- Align your Test/Dev systems to Production
- Automate your Architecture
- Enable Continuous integration and Deploy
- Create a Tagging Strategy
- Enable a Data Protection architecture
- Test and game days

Defining a Tagging Strategy for Data Protection

Tagging allows organization to assign metadata to their cloud resources in the form of *tags*. Each tag is a simple label consisting of an organizational-defined key and an optional value that can make it easier to manage, search for, and filter resources.

Although there are no inherent types of tags, they enable organizations to categorize resources by purpose, owner, environment, or other criteria. As an example AWS has an outline of commonly used tagging categories and strategies to help AWS customers implement a consistent and effective tagging strategy.

General Best Practices for Tagging

- Always use a standardized, case-sensitive format for tags, and implement it consistently across all resource types.

- Consider tag dimensions that support the ability to manage resource access control, cost tracking, automation, and organization.

- Implement automated tools to help manage resource tags.
- The Resource Groups Tagging API enables programmatic control of tags, making it easier to automatically manage, search, and filter tags and resources. It also simplifies backups of tag data across all supported services with a single API call per AWS Region.

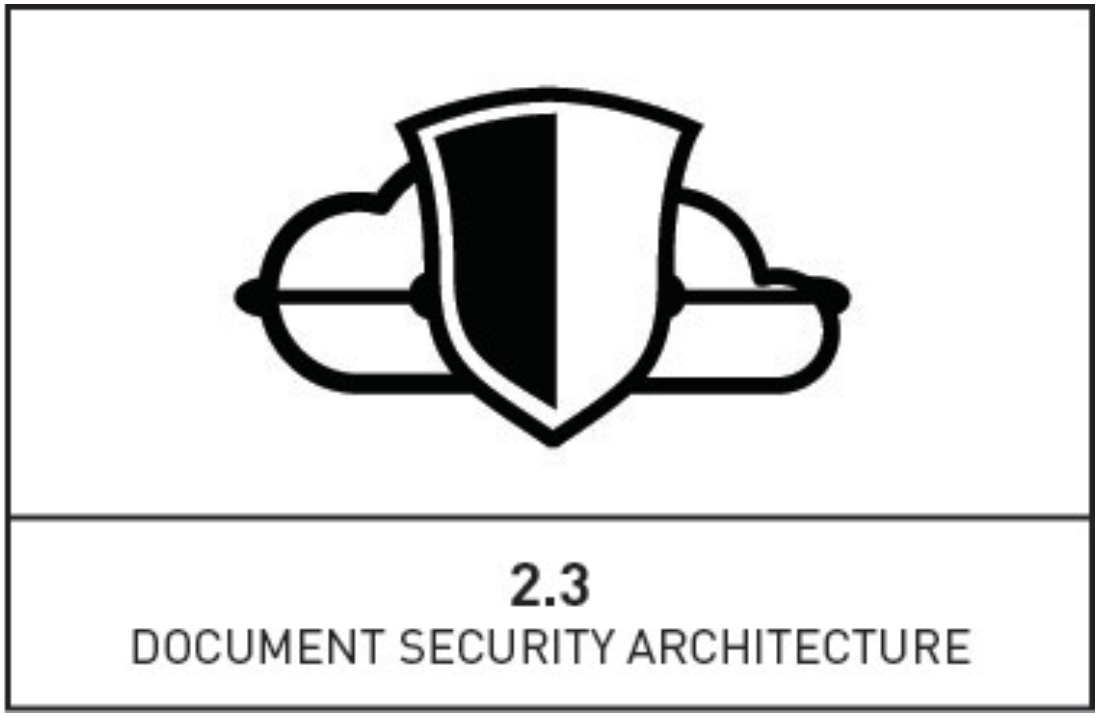
Tagging Categories

Companies that are most effective in their use of tags typically create business-relevant tag groupings to organize their resources along technical, business, and security dimensions.

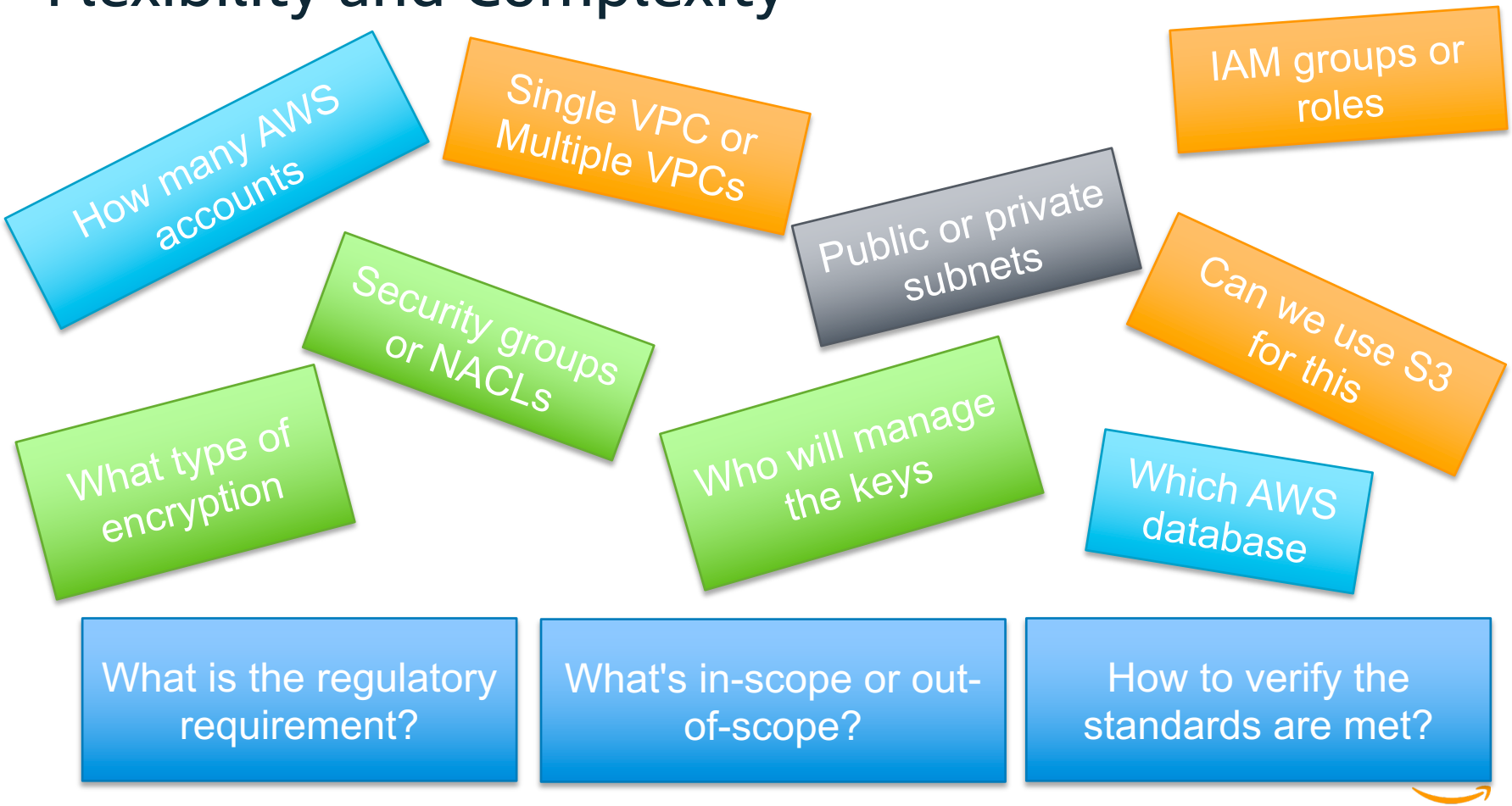
Companies that use automated processes to manage their infrastructure also include additional, automation-specific tags to aid in their automation efforts.

Technical Tags	Tags for Automation	Business Tags	Security Tags
Name – Used to identify individual resources **Application ID** – Used to identify disparate resources that are related to a specific application **Application Role** – Used to describe the function of a particular resource (e.g. web server, message broker, database) **Cluster** – Used to identify resource farms that share a common configuration and perform a specific function for an application **Environment** – Used to distinguish between development, test, and production infrastructure **Version** – Used to help distinguish between different versions of resources or applications	**Date/Time** – Used to identify the date or time a resource should be started, stopped, deleted, or rotated **Opt in/Opt out** – Used to indicate whether a resource should be automatically included in an automated activity such as starting, stopping, or resizing instances **Security** – Used to determine requirements such as encryption or enabling of VPC Flow Logs, and also to identify route tables or security groups that deserve extra scrutiny	**Owner** – Used to identify who is responsible for the resource **Cost Center/Business Unit** – Used to identify the cost center or business unit associated with a resource; typically for cost allocation and tracking **Customer** – Used to identify a specific client that a particular group of resources serves **Project** – Used to identify the project(s) the resource supports	**Confidentiality** – An identifier for the specific data-confidentiality level a resource supports **Compliance** – An identifier for workloads designed to adhere to specific compliance requirements

Security Architecture



Flexibility and Complexity

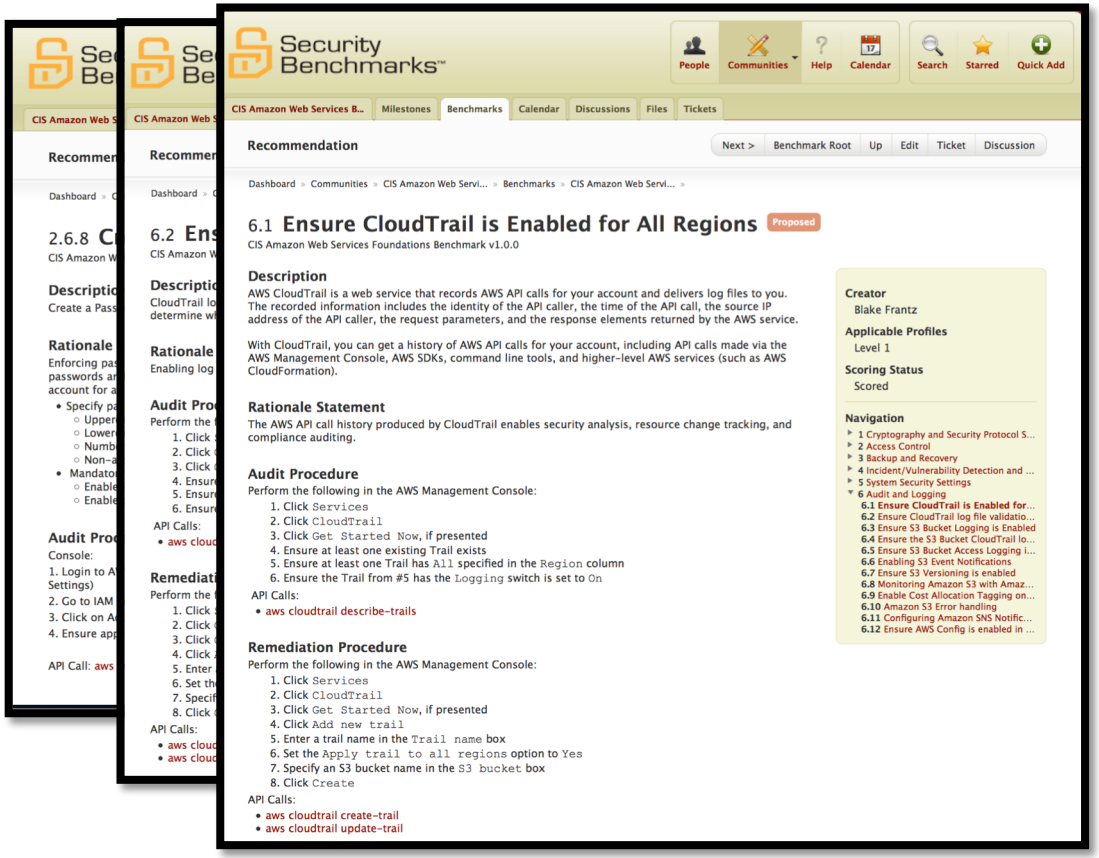


AWS CIS Foundation Benchmark

AWS has partnered with CIS Benchmarks to create consensus-based, best-practice security configuration guides which will align to multiple security frameworks globally.

The Benchmarks are:

- Recommended technical control rules/values for hardening operating systems, middle ware and software applications, and network devices;
- Distributed free of charge by CIS in .PDF format
- Used by thousands of enterprises as the basis for security configuration policies and the de facto standard for IT configuration best practices.



https://www.cisecurity.org/

DevOps and Security Better Together – Architecture

Continuous Integration (CI)
Continuous integration is a software development practice where developers regularly merge their code changes into a central repository, after which automated builds and tests are run. The key goals of continuous integration are to find and address bugs quicker, improve software quality, and reduce the time it takes to validate and release new software updates.

Continuous Delivery (CD)
Continuous delivery is a software development practice where code changes are automatically built, tested, and prepared for a release to production. It expands upon continuous integration by deploying all code changes to a testing environment and/or a production environment after the build stage. When continuous delivery is implemented properly, developers will always have a deployment-ready build artifact that has passed through a standardized test process.

Continuous Risk Treatment (CRT)
Continuous Risk Treatment is a modernized continuous monitoring process and technology approached which is designed to detect, maintain and in selected cases correct security, compliance and threats associated with an organization's solution and service deployment within their operational cloud environment as new needs or cyberthreats emerge.

DevOps – SecOps Cont….

Microservices

The microservices architecture is a design approach to build a single application as a set of small services. Each service runs in its own process and communicates with other services through a well-defined interface using a lightweight mechanism, typically an HTTP-based application programming interface (API). Microservices are built around business capabilities; each service is scoped to a single purpose. You can use different frameworks or programming languages to write microservices and deploy them independently, as a single service, or as a group of services.

Infrastructure as Code

Infrastructure as code is a practice in which infrastructure is provisioned and managed using code and software development techniques, such as version control and continuous integration. The cloud's API-driven model enables developers and system administrators to interact with infrastructure programmatically, and at scale, instead of needing to manually set up and configure resources. Thus, engineers can interface with infrastructure using code-based tools and treat infrastructure in a manner similar to how they treat application code. Because they are defined by code, infrastructure and servers can quickly be deployed using standardized patterns, updated with the latest patches and versions, or duplicated in repeatable ways.

Configuration Management

Developers and system administrators use code to automate operating system and host configuration, operational tasks, and more. The use of code makes configuration changes repeatable and standardized. It frees developers and systems administrators from manually configuring operating systems, system applications, or server software.

Policy as Code

With infrastructure and its configuration codified with the cloud, organizations can monitor and enforce compliance dynamically and at scale. Infrastructure that is described by code can thus be tracked, validated, and reconfigured in an automated way. This makes it easier for organizations to govern changes over resources and ensure that security measures are properly enforced in a distributed manner (e.g. information security or compliance with FedRAMP, PCI-DSS or HIPAA). This allows teams within an organization to move at higher velocity since non-compliant resources can be automatically flagged for further investigation or even automatically brought back into compliance.

aws

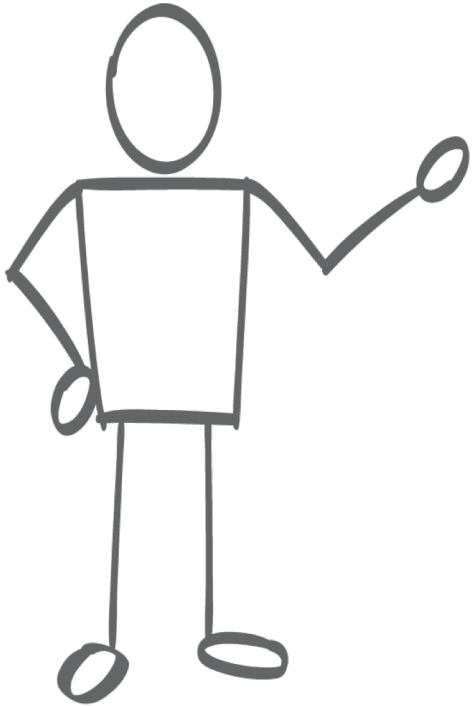
Architecture result…

Compliance as code

AWS and Amazon Partners allows you to codify your compliance with custom rules in AWS Lambda that define your internal best practices and guidelines for resource configurations. Using AWS SAO resources, you can automate assessment of your resource configurations and resource changes to ensure continuous compliance and self-governance across your AWS infrastructure.

Continuous audit and compliance

AWS SAO is designed to help you assess compliance with your internal policies and regulatory standards by providing you visibility into the configuration of your AWS resources, and evaluating resource configuration changes against your desired configurations continuously through the integration AWS services and Amazon Partner Network solutions.



Questions