Architecting security & governance across your AWS environment, protected by an integrated AWS Identity and Access Management

Marcus Fritsche
AWS Global Solutions Architect
mafritsc@amazon.de
AWS Account

Security, Access & Resource Boundary

API Limits/Throttling

Billing Separation
Account models

One Account

Your Accounts

1,000s of Accounts
Why one account isn’t enough

Many Teams; Different Access

Security / Compliance Controls

Billing

Business Process Isolation (Apps, SaaS)
Goals for a multi-account environment

- Automated
- Scalable
- Self-service
- Guardrails NOT Blockers
- Auditable
- Flexible
Account access & security considerations

Baseline Requirements

**Lock** the AWS Account Credential Management & "Root Account"

**Enable** AWS CloudTrail, AWS Config, Amazon GuardDuty!!

**Federate** and use identity solutions

**Define and map** enterprise roles and permissions

**Establish** InfoSec cross-account roles

**Identify** actions and conditions to enforce governance
What AWS accounts do we need for our secure, compliant multi-account environment?

- Log Archive
- Security
- Shared Services
- Network
- Billing-Admin
- Sandbox
- Dev
- Pre-Prod/QA
- Prod
- Other
AWS Organizations Master

- No connection to Data Center
- Service control policies
- Consolidated billing
- Volume discount
- Minimal resources
- Limited access (e.g. restrict AWS-Orgs role)
Core accounts - OU

AWS Organizations Master

Core Accounts

Foundational Building Blocks
Once per organization
Have their own development life cycle (dev/qa/prod)
Log archive account

AWS Organizations Master

Core Accounts

AWS
Log Archive

Network Path
Optional Network Path

Data Center

Amazon S3 bucket (Versioned, Restricted, MFA delete)

CloudTrail logs

Security logs

Single source of truth

Limited access & alarm on user login
Security account

- **Core Accounts**
  - AWS Security
  - AWS Log Archive

- **Optional data center connectivity**
- **Security tools and audit**
- **GuardDuty Master, FW-Manager**
- **Cross-account read/write**
- **Automated Tooling**
- **Limited access**
Shared services account

Core Accounts
- AWS Security
- AWS Log Archive
- AWS Shared Services

AWS Organizations Master

Connected to DC
DNS
LDAP/Active Directory
Shared Services VPC
Deployment tools
Golden AMIs
Pipeline
Scanning infrastructure
Inactive instances
Improper tags
Snapshot lifecycle
Monitoring
Limited access (IT-Ops)
Networking services

AWS Direct Connect (DX)
AWS DX Gateway
TGW, Shared VPC,
AWS Client VPN

Limited access
Managed by network team
Developer sandbox (OU & SBX-accounts)

Core Accounts
- AWS Security
- AWS Log Archive
- AWS Network
- AWS Shared Services

Developer Accounts
- AWS Developer Sandbox

- No connection to DC
- Innovation space
- Fixed spending limit
- Autonomous Experimentation
Team/group accounts - OU

Based on level of needed isolation

Match your development lifecycle

Think Small

Core Accounts
- AWS Security
- AWS Log Archive
- AWS Network
- AWS Shared Services

Team/Group Accounts

Developer Accounts
- AWS Developer Sandbox

Data Center
- Network Path
- Log Flow
- Optional Network Path
Develop and iterate quickly

Collaboration space

Stage of SDLC
Production applications
Promoted from Pre-Prod
Limited access (RO-only?)
Automated Deployments
Team Shared Services

Grows organically
Shared to the team
Product-specific common services
Data lake
Common tooling
Common services
Multi-account approach

**AWS Organizations Master**

- **Core Accounts**
  - AWS Security
  - AWS Log Archive
  - AWS Network
  - AWS Shared Services

- **Team/Group Accounts**
  - AWS Team Shared Services
  - AWS Prod
  - AWS Pre-Prod

- **Developer Accounts**
  - AWS Developer Sandbox

**Data Center**

- Network Path
- Log Flow
- Optional Network Path

**Orgs**: Account management

**Log Archive**: Security logs

**Security**: Security tools, AWS Config rules

**Shared services**: Directory, limit monitoring

**Network**: Direct Connect

**Dev Sandbox**: Experiments, Learning

**Dev**: Development

**Pre-Prod**: Staging

**Prod**: Production

**Team Shared Service**: Team Services, Data Lake, common Cognito, …
AWS Landing Zone structure - Basic

**AWS Organizations Account**
- Amazon S3 Bucket (manifest file)
- AWS CodePipeline
- AWS Organizations
- AWS SSO

**Shared Services Account**
- Active Directory
- Log Analytics

**Log Archive**
- Security Logs

**Security Account**
- Audit / Break-glass

**Organizations Account**
- Account Provisioning
- Account Access (SSO)
The AWS Landing Zone solution

An easy-to-deploy solution that automates the setup of **new AWS multi-account environments**

- Based on AWS best practices and recommendations
- Initial security and governance controls
- Baseline accounts and account vending machine
- Automated deployment
Account vending machine

Account Vending Machine (via AWS Service Catalog)
- Account creation factory
- User Interface to create new accounts
- Account baseline versioning
- Launch constraints

Creates/updates AWS account
Apply account baseline stack sets
Create network baseline
Apply account security control policy
AWS Control Tower

The easiest way to set up and govern a secure, compliant multi-account AWS environment

An easy-to-deploy solution that automates the setup of new AWS multi-account environments

AWS Control Tower is in preview

The AWS Control Tower preview is currently at capacity.
Access Management
Authorize with IAM-Policies and Secure Control Policies (SCPs)
AWS Organizations (enable all Features Mode)

Developer Accounts
- AWS Security
- AWS Log Archive
- AWS Network
- AWS Shared Services
- AWS Developer Sandbox

Team/Group Accounts
- AWS Security
- AWS Log Archive
- AWS Network
- AWS Shared Services
- AWS Dev
- AWS Pre-Prod

Shared Services
- AWS Team Shared Services
- AWS Prod
- AWS Pre-Prod

Core Accounts
- Single AWS Account

AWS Organizations Master

AWS Single Account

AWS Summit

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.
AWS IAM & AWS Organizations

**AWS IAM**
* IAM Policies
* Manage ARN
* Start from DENIED
* Assigned to Roles & Groups

**Organizations**
* SCPs (Service Control Policies)
* Manage APIs
* Start from ALLOWED
* Assigned to OUs and AWS Accounts
* Not for Root credentials, AWS Support, CloudFront, Alexa, …
AWS IAM & SC Policies

AWS IAM Policy

- Choose a service
- Define actions for the service
- Apply resources for actions
- Specify condition for actions

Organization SCPs

- Effect: Deny or Allow
- Choose a service
- Define actions for the service
- Apply resource = “*”
AWS IAM Policies

- JSON-formatted set of instructions which define permission
- Contain a statement (permissions) that specifies:
  - which **actions** a principal can perform
  - which **resources** can be accessed

```json
{
  "Statement": [{
    "Effect": "effect",
    "Principal": "principal",  who
    "Action": "action",       what
    "Resource": "arn",        where
    "Condition": {
      "condition": {
        "key": "value"  if
      }
    }
  }
}
```

Plenty of AWS managed IAM policies to use or adapt
AWS IAM Policy: Resource & Conditions

• Resources & Services
  Defined uniquely by an Amazon Resource Name (ARN)
  
  ```
  arn:aws:service:region:account:resource...
  ```

• Contain a statement (permissions) that specifies:
  • which **actions** a principal can perform
  • which **resources** can be accessed
IAM-Policies & SCPs

IAM Policies \[\cap\] Organizations SCP = Effective right

- User
- Group
- Role
- OU
- Account
- User
- Service

intersection
SCPs & IAM-Policies to protect Organizations

SCP
- Allow: EC2:*  
- Allow: S3:*  

IAM permissions
- Allow: EC2:*  
- Allow: SQS:*  

Identity-based policy
- Effective permission

Identity-based permission

Identity-based policy

Identity-based permission
Permissions Boundaries for IAM Entities (User or Role)

Set the **maximum permissions** that an identity-based policy can grant to an IAM entity.

The entity can perform only the actions that are **allowed by both** its identity-based policies and its permissions boundaries.
Resource-based Policies

- Effective permission
- Identity-based policy
- Permissions boundary

EC2
Session Policies

Effective permission

Identity-based policy

Permissions boundary

Session policy
AWS Organizations (Cross account access)

- **AWS Organizations Master**
  - Core Accounts
    - Security
    - Log Archive
    - Network
    - Shared Services
  - Team/Group Accounts
    - Team Shared Services
    - Prod
    - Dev
    - Pre-Prod
  - Developer Accounts
AWS Organizations (No Cross account access)

- Log Archive
- Security
- Backups
- PCI
Access Best Practice

• Restrict **Root** and Master Account Access
• Monitor activities as **Root** and in the Org. Master
• Use consolidated User Management / SAML
• Use principal of “Least privilege” (Role-based Access)
• Assign SCPs to OUs and test with dedicated OUs
• Avoid “whitelisting” and “blacklisting”
Next steps – Action required

- Build YOUR AWS account segmentation strategy
- Setup AWS Landing Zone / Control Tower
- Search train your Policy Ninja
- Iterate on SCPs and IAM Policies - automated using scripts!!!
- Use AWS Security Audits & WARs to check and challenge!

?? What did I said, you should not forget?
Next steps – Action required

• Build YOUR AWS account segmentation strategy
• Setup AWS Landing Zone / Control Tower
• Search train your Policy Ninja
• Iterate on SCPs and IAM Policies - automated using scripts!!!
• Use AWS Security Audits & **WARs** to check and challenge!
• **Enable** AWS CloudTrail, AWS Config, Amazon GuardDuty
Please complete the session survey.

... and contact me at mafritsc@amazon.de
Marcus Fritsche
How GuardDuty Works

Threat Detection Types
- Reconnaissance
- Instance Compromise
- Account Compromise

Data Sources
- VPC flow logs
- DNS Logs
- CloudTrail Events

Amazon GuardDuty

Findings
- Threat intel, ML/AI Anomaly Detection
- SIEM And/Or RESPOND

Findings
- HIGH
- MEDIUM
- LOW
AWS Config

- Continuously tracks resource configuration changes
- Evaluates the configuration against policies defined using AWS Config rules
- Alerts you if the configuration is noncompliant with your policies using Amazon SNS and Amazon CloudWatch Events

AWS Config = Continuous Configuration Auditor

Changing resources → AWS Config → AWS Config rules

Normalized

Notifications
API access
History, snapshot
Security and Compliance is a shared responsibility