

well-architected-framework.md

# AWS Well-Architected Framework

## Inhalte

1. Was ist das AWS Well-Architected Framework?
2. Die Funktionen des AWS Well-Architected Framework
3. Die 5 Säulen des AWS Well-Architected Framework
  - i. Betriebliche Spitzenleistung (Operational Excellence)
  - ii. Sicherheit (Security)
  - iii. Zuverlässigkeit (Reliability)
  - iv. Leistungseffizienz (Performance Efficiency)
  - v. Kostenoptimierung (Cost Optimization)

## Was ist das AWS Well-Architected Framework?

- Das AWS Well-Architected Framework ist ein Ansatz, um die Architektur von Anwendungen auf AWS zu verbessern.
- Hilft dir dabei zu entscheiden ob deine Architektur auf AWS optimiert ist.
- Erhöhen des Verständnisses für die Architektur und die Kosten (Best practices).
- Thematisieren **grundlegende Bereiche** die oft vernachlässigt werden.
- **Bewerten** von Architekturen anhand konsistenter Prinzipien
- **Kein AWS Service** sondern ein Ansatz um die Architektur von Anwendungen auf AWS zu verbessern.

## Die Funktionen des AWS Well-Architected Framework

### Bietet nicht

- Informationen zur implementierung
- Architektonische Muster

## Bietet

- Fragen, bei denen es darum geht, architektonische Entscheidungen zu treffen
- Dienstleistungen und Lösungen, die für jede Frage relevant sind
- Verweise auf relevante Ressourcen

## Die 5 Säulen des AWS Well-Architected Framework

---

1. **Betriebliche Spitzenleistung** (Operational Excellence)
  - Geschäftswert generieren
2. **Sicherheit** (Security)
  - Schutz und Überwachung von Systemen
3. **Zuverlässigkeit** (Reliability)
  - Wiederherstellung nach Fehlern und Minimierung von Unterbrechungen
4. **Leistungseffizienz** (Performance Efficiency)
  - Sparsamer Einsatz von Ressourcen
5. **Kostenoptimierung** (Cost Optimization)
  - Vermeidung unnötiger Ausgaben

## 1. Betriebliche Spitzenleistung (Operational Excellence)

---

- Generierung eines positiven Geschäftswertes
- Zur kontinuierlichen Verbesserung der unterstützten Prozesse und Verfahren

### Hauptthemen

- Änderungen verwalten
- Auf Ereignisse reagieren
- Auf Änderungen reagieren

### Best practices

- Definiere Infrastruktur als Code (IaC)
- Mache kleine, häufige und umkehrbare Änderungen
- Konstante Verbesserung der Prozesse
- Gehe von Fehlern aus
- Lerne von Operation-Failures

### Beispiel

- Gesmate Infrastruktur wird über AWS Konsole (UI) erstellt
- Probleme:
  - Keine Versionskontrolle
  - Keine automatisierte Skalierung
  - Keine automatisierte Wiederherstellung
  - Keine automatisierte Überwachung
  - Keine automatisierte Änderungsverwaltung
- Lösung:
  - Infrastruktur als Code (IaC)
  - CloudFormation, AWS CDK, Terraform

## 2. Sicherheit (Security)

---

- Überwachen und schützen von Systemen
  - Informationen
  - Systeme
  - Komponenten
- Bereitstellung von Unternehmenswert durch
  - Risikobewertung
  - Minderungsstrategien

### Hauptthemen

- Authentifizierung und Autorisierung
- Tracing (Nachverfolgung)
- Schutz vor Systemen und Diensten
- Schutz der Vertraulichkeit, Integrität und Verfügbarkeit von Daten

### Schlüsselbereiche der Cloud Sicherheit

- Identity and Access Management (IAM)
- Erkennungskontrollen
- Infrastrukturschutz
- Datenschutz
- Vorfallreaktion

### Best practices

- Implementiere eine starke IAM (Identity and Access Management) Strategie

- Aktivierung der Rückverfolgbarkeit
- Implementierung des Prinzips der geringsten Rechte ("least privilege")
- Konzentration auf die Sicherung der Systeme und Dienste
- Automatisierung

## Beispiel

- Gesamtes Team nutzt den selben AWS Admin User
- Probleme:
  - Keine Überwachung von Nutzeraktivitäten
  - Keine Kontrolle über die Zugriffe
  - Keine Kontrolle über die Berechtigungen
  - Keine Kontrolle über die Daten
- Lösung:
  - Verschiedene AWS User pro Teammitglied
  - IAM User - Authentication
  - AWS IAM Policies -> Authorization
  - AWS CloudTrail -> Tracking von Nutzeraktivitäten

## 3. Zuverlässigkeit (Reliability)

---

- Wiederherstellung nach Infrastrukturfehlern oder Service-Unterbrechungen
- Dynamisch Datenverarbeitungsressourcen bereitstellen, um die Nachfrage zu decken (Auto Scaling)
- Abschwächung von Unterbrechungen wie:
  - Fehlkonfiguration
  - Vorübergehende Netzwerkprobleme

## Best practices

- Wiederherstellungsverfahren testen
- Automatische Wiederherstellung
- Horizontale Skalierung
- Kapazitäten genau definieren
- Verwalten von Änderungen in der Automatisierung

## Beispiel

- Datenbank Backup wird manuell durchgeführt

- Probleme:
  - Keine automatische Wiederherstellung
  - Letztes Backup kann lange zurückliegen -> Hoher Datenverlust bei Ausfällen -> Recovery Point Objective (RPO)
- Lösung:
  - Automatisiertes Backup
  - AWS RDS Multi AZ -> High Availability
  - AWS Backup -> Automatisierte Wiederherstellung

## 4. Leistungseffizienz (Performance Efficiency)

---

- Rechenressourcen effizient nutzen -> Systemanforderungen erfüllen
- Diese Effizienz beizubehalten, wenn sich die Nachfrage ändert und sich die Technologien entwickeln (Auto Scaling)

### Fähigkeiten

- Anpassbare Lösungen wählen
- Überprüfen für kontinuierliche Innovation
- Überwachen von AWS-Services
- Beachten von Kompromissen

### Best practices

- Demokratisieren fortschrittlicher Technologien (Managed Services)
- Globale Reichweite in Minutenschnelle erzielen
- Verwenden einer *serverlosen Architektur* (FaaS)
- Häufige Experimente durchführen
- Mechanische Sympathie verwenden (Best tool for the job)

### Beispiel

- Statische Website wird mit einem EC2 Instanz betrieben
- Probleme:
  - Keine automatische Skalierung
  - Keine automatische Wiederherstellung
  - Eine EC2 Instance = Single Point of Failure -> nicht hochverfügbar (High Availability)
  - Nicht effizient, da EC2 Instanz immer läuft, auch wenn keine Anfragen kommen
  - Nicht effizient, da für static content keine EC2 Instanz benötigt wird

- Lösung:
  - AWS S3 -> Static Content Hosting
  - AWS CloudFront -> Content Delivery Network (CDN)
  - AWS Route 53 -> DNS
  - **Globale Infrastruktur in Minuten**

## 5. Kostenoptimierung (Cost Optimization)

---

- Vermeidung unnötiger Ausgaben
- Kostenbewusstsein
- Vermeidung von suboptimaler Ressourcen

### Fähigkeiten

- Kostengünstige Ressourcen verwenden
- Angebot der Nachfrage anpassen
- Steigerung des Kostenbewusstseins
- Vorgänge im Laufe der Zeit optimieren

### Best practices

- Einführung eines Verbrauchmodells
- Messung der Gesamteffizienz
- Verringerung der Ausgaben für den Rechenzentrumsbetrieb
- Ausgaben analysieren und zuordnen
- Verwenden von Managed Services

### Beispiel

- Kurz laufende, nicht vorhersehbare Jobs, die jederzeit abgebrochen werden können, werden mit On Demand EC2 Instanzen ausgeführt
- Probleme:
  - Nicht kosteneffizient, da EC2 Instanzen nur kurz laufen
  - On Demand EC2 Instanzen sind teurer als Spot Instanzen (bis zu 70%)
- Lösung:
  - Spot Instanzen verwenden, um Kosten zu sparen **falls spontane Ausfälle kein Problem darstellen**

# Zusammenfassung

---

- Das Well-Architected Framework bietet einen konsistenten Ansatz zur Bewertung von Cloud-Architekturen und Anleitungen zur Implementierung von Designs.
  - Im Well-Architected Framework werden grundlegende Fragen formuliert, die es Ihnen ermöglichen zu verstehen, ob eine bestimmte Architektur gut mit den bewährten Methoden für die Cloud übereinstimmt.
  - Das Well-Architected Framework basiert auf fünf Säulen:
    - Operational Excellence
    - Sicherheit
    - Zuverlässigkeit
    - Leistungseffizienz
    - Kostenoptimierung.
  - Jede Säule beinhaltet eine Reihe von Designprinzipien und bewährten Methoden.
-