



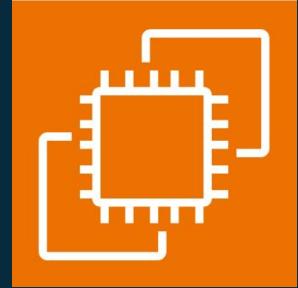
**TECH  
STARTER**

AWS EC2  
Deepdive

# Heutige Inhalte:

- EC2-Instanz Aufbau; Einordnung in “X as a Service”
- Instanz-Typen
- (Security Groups)
- Instanz-Kaufoptionen

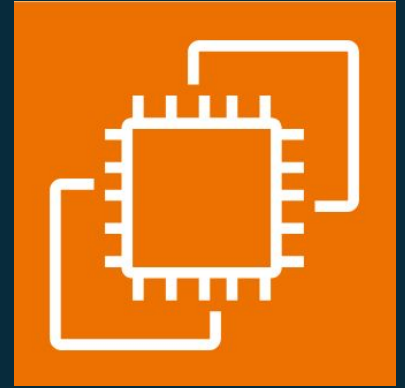
# Amazon EC2



- EC2 = **E**lastic **C**ompute **C**loud
- Für uns: zentralster und wichtigster AWS Service

**Was wisst ihr bereits über EC2?**

**Was macht EC2 so nützlich?**



# X as a Service

Wozu gehört EC2 ?

On-Premise	IaaS (Infrastructure as a Service) (z.B. VM mieten)	PaaS (Platform as a Service) → Ich muss nur Entwickeln	SaaS (Software as a Service) (Email, Netflix, Dropbox,...)
Applikation	Applikation	Applikation	Applikation
Daten	Daten	Daten	Daten
Laufzeitumgebung	Laufzeitumgebung	Laufzeitumgebung	Laufzeitumgebung
Container (z.B. Docker)	Container (z.B. Docker)	Container (z.B. Docker)	Container (z.B. Docker)
Betriebssystem (OS)	Betriebssystem (OS)	Betriebssystem (OS)	Betriebssystem (OS)
Virtualisierung	Virtualisierung	Virtualisierung	Virtualisierung
Server	Server	Server	Server
Infrastruktur (Speicher, Networking)	Infrastruktur (Speicher, Networking)	Infrastruktur (Speicher, Networking)	Infrastruktur (Speicher, Networking)
Einrichtung(Gebäude mit Elektrizität)	Einrichtung(Gebäude mit Elektrizität)	Einrichtung(Gebäude mit Elektrizität)	Einrichtung(Gebäude mit Elektrizität)

# Konfiguration von EC2-Instanzen

- *Amazon Machine Image (AMI)*
  - Also OS und ggf. vorinstallierte und vorkonfigurierte Software
- Instanz-Typ
  - Also RAM-Größe und CPU-Power, sowie Netzwerkbandbreite
- Speicher, in Form von EBS, EFS oder EC2 Instance Store
  - Also Festplatten / Dateispeicher
- Netzwerkeinstellungen
  - Public IP; Security Groups (Firewall)
- User Data / Benutzerdaten
  - Bootstrap Skript, also Skript für initiale Konfiguration (Installationen von Paketen zum Beispiel)



# User Data / Benutzerdaten

- Die Möglichkeit, ein Bootstrap-Skript anzuhängen
- Läuft als root-User

Benutzerdaten - *optional*

Info

Laden Sie eine Datei mit Ihren Benutzerdaten hoch oder geben Sie sie in das Feld ein.

Choose file

```
#!/bin/bash

# Total viel Konfiguration
```

☐ Benutzerdaten wurden bereits base64-kodiert

# Bootstrap

- Eine Reihe von Befehlen (bspw. Bash-Skript)
- Läuft **nur beim ersten Starten** einer Instanz
- Nutzen:
  - Software installieren und konfigurieren
  - Updates laden
  - System weiter konfigurieren (Nutzer / Gruppen, Rechte, ...)
  - **Alles**, was in einem Skript sonst möglich ist



# Aufgabe:

1. Erstelle eine EC2-Instanz mit Standardeinstellungen.  
Schreibe dabei ganz unten (Erweiterte Details > Benutzerdaten)  
ein kurzes Bash-Skript, welches eine Datei “HalloWelt” mit dem Inhalt “Hallo Welt!” im User-Ordner des Standard-Users “ec2-user” erstellt.
2. Verbinde dich mit der Instanz und überprüfe, ob die Datei im richtigen Ordner existiert und “Hallo Welt!” beinhaltet.



# Instanz-Typen

# Aufbau Instanztyp-Bezeichnung

t3.micro  
a1.large

Wir nutzen im  
Kurs nur t2/t3

Legende:

- t: Instanzfamilie (Buchstabenkürzel)
- 3: Generation (Als Zahl)
- micro: "Größe" (Fester Kurzbegriff)



# Instanztypen zuordnen

- Instanzfamilie (manchmal auch “-klasse”)
  - Unterschiedliche Anwendungsfälle
- Instanzgröße
  - Unterschiedliche “Größen” mit unterschiedlich viel Leistung innerhalb der einzelnen Instanzklassen

# Übersicht einiger Instanztypen

Instance-Typ ▾	vCPUs ▾	Architektur ▾	Speicher (GiB) ▾	Speicher (GB) ▾	Speichertyp ▾	Netzwerkleistung
t3.xlarge	4	x86_64	16	-	-	Up to 5 Gigabit
<a href="#">a1.xlarge</a>	<a href="#">4</a>	<a href="#">arm64</a>	<a href="#">8</a>	-	-	<a href="#">Up to 10 Gigabit</a>
<a href="#">a1.4xlarge</a>	<a href="#">16</a>	<a href="#">arm64</a>	<a href="#">32</a>	-	-	<a href="#">Up to 10 Gigabit</a>
c3.xlarge	4	x86_64	7.5	80	ssd	Moderate
c3.4xlarge	16	x86_64	30	320	ssd	High
c3.8xlarge	32	x86_64	60	640	ssd	10 Gigabit
c4.xlarge	4	x86_64	7.5	-	-	High
c4.4xlarge	16	x86_64	30	-	-	High
c4.8xlarge	36	x86_64	60	-	-	10 Gigabit
c5.xlarge	4	x86_64	8	-	-	Up to 10 Gigabit

# Instanztypen - Unterschiede

- *vCPUs*: virtuelle CPU-Kerne, Leistung abhängig von Instanzklasse
- *Architektur*: Kompatibilität mit Prozessor-Architektur XY
- *Speicher (GiB)*: RAM-Größe
- *Netzwerkleistung*: Grobe Angabe der Netzwerkbandbreite

Manche Typen haben einen festen Datenspeicher:

- *Speicher (GB)*: Im Prinzip Festplattengröße
- *Speichertyp*: Festplattentyp (HDD / SSD)

# Unterschiede zwischen den Größen innerhalb einer Instanzfamilie

Instance-Typ ▾	vCPUs ▾	Architektur ▾	Speicher (GiB) ▾	Speicher (GB) ▾	Speichertyp ▾	Netzwerkleistung
m3.medium	1	x86_64	3.75	4	ssd	Moderate
m3.large	2	x86_64	7.5	32	ssd	Moderate
m3.xlarge	4	x86_64	15	80	ssd	High
m3.2xlarge	8	x86_64	30	160	ssd	High

# Wichtige Instanztyp Kategorien



# General Purpose / Allgemeine Zwecke

- Sehr ausgeglichene Instanzen
  - Gleichgewicht zwischen Rechen-, Speicher- und Netzwerkressourcen
- Für moderate CPU-Auslastung mit Leistungsspitzen / Bursts
- Hierunter fallen vor allem die **m**- und **t**-Familien

Anwendungsfälle:

- Testing- / Entwicklungsumgebungen; Web-Server; ...

# Compute Optimized / Für Datenverarbeitung optimiert

- Für sehr intensive, CPU-lastige Workloads
- Hierunter fallen die **c**-Familien

Anwendungsfälle:

- Videocodierung; High-Performance Webserver; Gaming; Machine-Learning; Analytik; ...

# Memory Optimized / Arbeitsspeicheroptimiert

- Für Workloads, die große Datenmengen im Arbeitsspeicher verarbeiten
- Hierunter fallen unter anderem **r**- und **x**-Familien

Anwendungsfälle:

- Open-Source-Datenbanken; Big-Data-Analytik in Echtzeit;  
...

# Storage Optimized / Speicheroptimiert

- Für Workloads, die sehr viele Lese- und Schreibzugriffe auf lokalen Speicher (Festplatten) benötigen
- Hierunter fallen unter anderem die **h**- und **i**-Familien

## Anwendungsfälle:

- Datenbanken (SQL / NoSQL); Big-Data-Analytik; Datenlager; ...

# Wichtige Instanztyp Kategorien

- General Purpose
- Compute Optimized
- Memory Optimized
- Storage Optimized

# Security Groups - Übersicht

- Bilden eine Firewall für unsere Instanzen
- Bestehen aus “Regeln”
- Können ein- und ausgehenden Datenverkehr erlauben
- Eine Regel besteht aus:
  - Protokoll/Port
  - IP(-Range) oder Security-Group-Referenz
- Standard:
  - Ausgehend wird jeder Port an jede IP-Adresse erlaubt
  - Eingehend wird nichts erlaubt (höchstens SSH)

# Security Groups - Zusatzinfos

- Können an mehreren Instanzen gesetzt werden
- Gehören immer zu einer VPC (und somit auch Region)
- Geblockter Datenverkehr wird nicht von der EC2 abgelehnt, sondern erreicht diese gar nicht
  - Also: “time out” vom Server => Security Group einsehen
  - Sonst: “refused” vom Server => Liegt nicht an Security Group



# Aufgabe:

1. Erstelle eine EC2-Instanz mit diesem User-Data-Skript:

```
#!/bin/bash
yum update -y
yum install -y httpd
systemctl start httpd
systemctl enable httpd
echo "<h1>Hello World from $(hostname -f)</h1>" > /var/www/html/index.html
```

Das Skript richtet die Instanz als Webserver ein, sodass über HTTP eine Webseite zu sehen sein müsste.

2. Make it work!



# Instanz-Kaufoptionen

# Einschub: Kosteneinsparung

Generell gilt:

- Flexibilität kostet am meisten
- Commitment auf feste Größen / Zeiträume ist günstiger
- Je länger ein Commitment, desto günstiger
- Je mehr anteilig vorab bezahlt wird, desto günstiger

# Instanz-Kaufoptionen

- On-Demand
- Saving Plans
- Reserved Instances
- Spot Instances
- Dedicated Hosts
- Dedicated Instances
- Kapazitätsreservierung

# On Demand

- Nutzen wir im Kurs fast ausschließlich
  - **Pay-as-you-go**; Bezahlen pro Sekunde pro Instanz
  - Möglichkeit Kosten jederzeit zu stoppen
- 
- Höchster Preis, aber höchste Flexibilität
  - Geeignet für kurze Zeiträume und zum Ausprobieren

# Saving Plans

- Man reserviert eine feste Nutzung, für die man durchgehend bezahlt (10\$/Stunde für ein 1 oder 3 Jahre)
- Wird auf eine Region und Instanzfamilie festgelegt (alles andere flexibel)
- Zeitraum ist 1 oder 3 Jahre
- Nutzung die darüber hinausgeht, wird als On-Demand verrechnet
- Sinnvoll, wenn man einen gewissen Workload durchgehend und langfristig erwarten kann

# Reserved Instances

- Man reserviert eine Instanz für einen festen Zeitraum
  - Zeiträume sind entweder 1 oder 3 Jahre
  - Bezahloptionen zu Beginn: Nichts, Teilweise, oder Alles
  - Erheblich günstiger als On-Demand  
(im Extremfall etwa ein Drittel bis Viertel des Preises)
- 
- Für Langfristigen Nutzen

# Spot Instances

- Nutzt ungenutzte Kapazitäten von AWS Instanzen
- Man wählt einen Instanztypen, einen Preis und hat einen Workload vorbereitet
- Sobald eine Spot-Instanz zu dem gewählten Preis verfügbar ist, wird die Workload verarbeitet
- **Günstigste Variante!** Bis zu einem Zehntel des On-Demand-Preises
- Nur gut, für zeit-unkritische Workloads

# Dedicated Hosts

- Man reserviert einen physischen Server, welchen man beliebig mit EC2-Instanzen füllen kann
- Als **On-Demand** oder Reserviert (**1** oder **3** Jahre) verfügbar
- Generell **TEUER**
- Geeignet für Firmen mit strenger Compliance



# Dedicated Instances

- Wie bei *Dedicated Hosts* werden die Instanzen auf einem einzigen Host bzw. Server erstellt
- Anders als bei *Dedicated Hosts* haben wir nicht unbedingt den ganzen Server reserviert und können die genaue Platzierung der Instanzen nicht nachvollziehen

# Kapazitätsreservierung

- Man reserviert sich eine Kapazität an On-Demand-Instanzen in einer AZ (mit On-Demand-Preis)
  - Jederzeit erstell- und kündbar
  - Keine Discounts!
- 
- Sinnvoll, wenn man geschäftskritische Workloads hat, die jederzeit eine gewisse EC2-Kapazität braucht
  - Man vermeidet hiermit, dass in der AZ eventuell keine Kapazitäten verfügbar sind, wenn man sie braucht

# Instanz-Kaufoptionen

- On-Demand: Pro Sekunde bezahlen, kurze Workloads (teuer!)
- Saving Plans: Auf konstante Nutzung festlegen (1 oder 3 Jahre)
- Reserved Instances: Auf Instanz festlegen (1 oder 3 Jahre)
- Spot Instances: Ungenutzte EC2-Instanzen nutzen (günstigste!)
- Dedicated Hosts: Einen tatsächlichen Server mieten
- Dedicated Instances: Instanz auf reservierter Hardware
- Kapazitätsreservierung: Kapazität in AZ reservieren