7 Maßnahmen zur Kostenreduzierung in AWS

April 2020

Inhaltsverzeichnis

1. Einleitung....3
2. Konsolidierte Faktorierung....4
3. UngenutzteRessourcen....5
4. Rightsizing....6
5. Autoscaling....6
6. Reservierungen und Spot Instanzen....7
7. Lizenzkosten....9
8. Trusted Advisor & Cost Explorer....11
9. Fazit....11
10. Literaturverzeichnis....13

1. Einleitung

Amazon Web Services bietet viele Vorteile gegenüber Rechenzentren im Eigenbetrieb. Einer dieser Vorteile ist die nutzungsbasierte Abrechnung (Pay-per-Use). Verschiedene Abrechnungsmodelle erhöhen jedoch die Komplexität dieses eigentlich relativ simplen Modells. Spätestens beim Betrieb vieler produktiver Anwendungen sollte die Infrastruktur hinsichtlich möglicher Kosteneinsparungspotenziale überprüft werden. Damit dies geschehen kann, müssen zunächst die Berechnungsfaktoren betrachtet werden.

Grundsätzlich werden bei den AWS Diensten die CPU- und Speicher Nutzung sowie der ausgehende Netzwerkverkehr berechnet. Dies geschieht je nach Service in leicht abgewandelter Form [1]. Managed Services, wie zum Beispiel der vollverwaltete serverlose Containerdienst Fargate, verursachen bei reiner Betrachtung der Infrastruktur mehr Kosten, als vergleichbare EC2-Instanzen, ersparen dem Kunden jedoch die Konfigurations und Verwaltungsaufwände.

Kostenoptimierungen können grundsätzlich durch die Justierung dieser Stellhebel sowie durch Skaleneffekte, Eliminierung ungenutzter Ressourcen und einiger anderer Möglichkeiten realisiert werden.

Im Folgenden werden sieben kurzfristig realisierbare Einsparungspotenziale aufgezeigt.

2. Konsolidierte Faktorierung

Die Verwendung von mehr als nur einem AWS Account kann aus verschiedenen Blickwinkeln sinnvoll sein:

Ein gesonderter Benutzeraccount und eine Trennung nach Umgebungen kann die Sicherheit erhöhen, da ein kompromittierter Account, in welchem die Test Umgebung betrieben wird, nicht gleichzeitig zu einer kompromittierten Produktionsumgebung führt. Auch ist die Rechtevergabe transparenter, da Entwickler ggf. gar nicht auf den Produktionsaccount zugreifen dürfen und sich nur im Entwicklungs- und Testaccount aufhalten sollen. Sicherlich ist dies auch über eine ausgeklügelte Rechtevergabe im Identity- and Access Management (IAM) Service realisierbar, kann aber bei einer kleinen Unachtsamkeit zu einer falschen Rechtevergabe führen.

Eine Trennung auf Accountebene kann auch für die Abbildung der Organisationsstruktur dienlich sein. Somit können Anwendungen der Buchhaltung in einem anderen Account betrieben werden als die Anwendungen der IT-Abteilung.

Der Service AWS Organizations ermöglicht es bei einem Multi-Account Betrieb Kostenvorteile zu erzielen. Durch die sogenannte konsolidierte Faktorierung (engl. Consolidated Billing) werden die Kosten aller verknüpften Accounts über einen zentralen Zahlungsaccount und eine Rechnung abgerechnet. Dies führt dazu, dass auch die genutzten Leistungen addiert und Skaleneffekte erzielt werden.

Angenommen es bestehen drei Accounts: Entwicklung, Test und Produktion. Wenn jeder dieser Accounts 40 TB S3 Speicher verwendet, entstehen Kosten in Höhe von $1003,52 pro Account pro Monat, was einer Gesamtsumme von $3010,56 entspricht. Durch die konsolidierte Fakturierung werden statt 3 x 40TB die konsolidierten 120TB berechnet. Dies führt zu reduzierten Gesamtkosten von $2938,88, also einer monatlichen Einsparung von $71,68. Die Einsparung ist in diesem Beispiel darauf zurückzuführen, dass AWS ab einer verwendeten S3 Speichermenge von 100TB einen Nachlass von 4,5% auf alle folgenden GB einräumt [2].

Dies ist nicht nur bei S3 der Fall, sondern gilt auch für die meisten anderen Services und unter anderem auch für Kosten, die durch ausgehenden Netzwerkverkehr entstehen.

Auch wenn dies im Vergleich zu den Gesamtkosten keine große Summe ist, so kann dennoch ohne viel Aufwand und ohne technische Architekturänderungen eine sofortige Kostenreduktion erzielt werden.

3. UngenutzteRessourcen

Eine einfache und naheliegende Methode, um Kosten einzusparen ist das Entfernen ungenutzter Ressourcen. Während Compute Ressourcen, wie EC2 Instanzen oder Lambda Funktionen keine Kosten verursachen, wenn diese gestoppt sind, gilt dies nicht für alle Ressourcen.

Verwendeter Speicherplatz in Form von EBS Volumes oder S3 Objekten verursacht Kosten solange der Speicherplatz belegt ist, unabhängig davon, ob auf die Objekte zugegriffen wird. Es kann sich also durchaus lohnen einen Blick auf die genutzten Ressourcen zu werfen und ungenutzte Objekte auszusortieren.

Ein häufiger Kostenverursacher sind EBS und RDS Snapshots. Diese werden üblicherweise automatisiert als Backup erstellt, können jedoch auch manuell gesichert werden. Für manuelle Snapshots gibt es keine automatischen Löschautomatismen, außer wenn diese eigenständig implementiert wurden. Dies führt dazu, dass sich diese schnell zu einem relevanten Posten auf der Rechnung entwickeln können. Es empfiehlt sich, regelmäßig die Snapshots auf Relevanz zu prüfen und veraltete Snapshots zu löschen.

Es kann keine generelle Empfehlung für das Alter eines Snapshots ausgesprochen werden: Während einige Snapshots auch noch nach einigen Monaten verwendet werden können, sind andere schon nach wenigen Tagen so veraltet, dass eine Wiederherstellung nicht sinnvoll ist.

Des Weiteren sollte regelmäßig über nicht angebundene EBS Volumes mit dem Status available geschaut werden. Dieser Status ist ein Indikator dafür, dass das Volume nicht verwendet und somit nicht mehr benötigt wird.

Obwohl S3 Speicherplatz vergleichsweise günstig ist, sollte auch hier sparsam mit den verwendeten Ressourcen umgegangen werden und der belegte Speicher auf den Prüfstand gestellt werden.

All diese Maßnahmen können im Rahmen eines Clean-Up Days einmal im Monat oder im Quartal organisiert werden, sodass sich erst keine großen Mengen an ungenutzten Ressourcen ansammeln. Im Idealfall werden Automatismen z. B. In Form von S3 Lifecycle Rules oder AWS Backup genutzt, welche den Account sauber halten.

Grundsätzlich ist es empfehlenswert alle Ressourcen mit Tags zu versehen, sodass diese eindeutig einem Projekt zugeordnet werden können. Wenn ein Projekt endet oder eine Anwendung abgelöst wird, kann anhand der Tags eine umfangreiche und vollständige Löschung aller zugehöriger Ressourcen durchgeführt werden.

4. Rightsizing

Amazon Web Services bietet die Möglichkeit, innerhalb kürzester Zeit die Ressourcen auf die benötigten Anforderungen zu skalieren. Hierdurch müssen keine Reserven für Auslastungsspitzen vorgehalten werden, was einen enormen Vorteil zu einem klassischen Rechenzentrum bietet.

Gerade in Lift & Shift Projekten kommt es häufig vor, dass zu viele Serverkapazitäten bereitgestellt werden, da die Bezugsgröße für die Bemessung der Instanzen die zuvor genutzten Server sind.

Dies ist ein valides und zunächst auch korrektes Vorgehen, wenn entweder die Server schon korrekt bemessen waren oder entsprechende Entscheidungsgrundlagen in Form von Auslastungsmetriken fehlen. Mithilfe von CloudWatch kann die reelle Auslastung der Instanzen analysiert und entsprechende Anpassungen vorgenommen werden.

Ziel sollte es sein, die Instanzen so zu bemessen, dass die regelmäßige durchschnittliche Auslastung bedient werden kann und bei Auslastungsspitzen eine horizontale Skalierung[1] durchgeführt werden kann. Durch die hierbei eingesparten Überkapazitäten können die Kosten signifikant reduziert werden.

5. Autoscaling

Bei der Nutzung von AWS Diensten sollte das Überwachen und Nachjustieren zu einer regelmäßigen Aufgabe werden, um Kostenvorteile zu erzielen. Während dies zu Beginn noch eine manuelle Tätigkeit sein kann, bietet AWS alle benötigten Werkzeuge, um diesen Prozess zu automatisieren.

Mit Autoscaling kann auf CloudWatch Metriken reagiert werden, sodass automatisiert neue Instanzen bereitgestellt werden, sobald eine Metrik einen Grenzwert über- oder unterschreitet. Hierbei ist zu berücksichtigen, dass die

1 Horizontale Skalierung ist das Bereitstellen neuer Instanzen und die **Lastverteilung** auf mehrere Maschinen. Vertikale Skalierung ist das Erhöhen von Ressourcen bzw. der Instanzklasse.

Bereitstellung von neuen Instanzen einige Zeit in Anspruch nehmen kann. CloudWatch muss die Grenzwertüberschreitung zunächst feststellen, was abhängig von der konfigurierten Granularität des Monitorings abhängt. Außerdem muss die Überschreitung über einen definierbaren Zeitraum erfolgen. Sobald CloudWatch dies feststellt, benötigt Autoscaling eine gewisse Zeit, um die Instanz zu provisionieren. Dies variiert, je nach Umfang der Userdata und des verwendeten Amazon Machine Images (AMI).

Sobald mit einem sehr hohen Anstieg der Nachfrage innerhalb kürzester Zeit zu rechnen ist, sollte die Erkennungszeit reduziert werden in dem kurze Prüfzyklen, geringe Grenzwerte und eine geringe Anzahl an aufeinanderfolgenden Grenzüberschreitungen konfiguriert werden. Des Weiteren sollte das verwendete AMI alle benötigten Abhängigkeiten beinhalten, um die Bereitstellungszeit durch möglichst wenige Initialisierungsschritte zu reduzieren.

Somit hilft Autoscaling bei der Reduzierung manueller Aufwände und Kosten.

6. Reservierungen und Spot Instanzen

Eine bei AWS gestartete EC2 Instanz wird standardmäßig nach dem „on-Demand“ Abrechnungsmodell berechnet. Das bedeutet, dass sie nach der Verwendung gestoppt werden kann und dadurch nur der genutzte Zeitraum abgerechnet wird. Neben on-Demand Instanzen gibt es noch reservierte Instanzen (RI) und Spot Instanzen.

Beim RI-Abrechnungsmodell, wird eine Instanz für einen Zeitraum von einem oder drei Jahren reserviert, wobei ein Jahr mit einem Preisnachlass von 40% und drei Jahre mit einem Nachlass von 60% honoriert werden. Die hohen Rabatte lassen sich vor allem über einen hohen Reservierungszeitraum und Vorauszahlungen realisieren.

In dem gewählten Zeitraum kann der Instanztyp nicht verändert werden. Es können andere Instanztypen verwendet werden, diese werden dann nach dem on-Demand Modell abgerechnet. Außerdem gibt es eine konvertierbare Option, bei der es möglich ist den Instanztypen zu ändern. Hierbei wird jedoch eine geringere Einsparung erreicht [1].

Reservierte Instanzen eignen sich vor allem bei konstantem Bedarf, bei dem sich die Anforderungen an eine Instanz innerhalb des gewählten Zeitraums nicht ändern. Außerdem ist der Einsatz von Ris bei standardisierten Umgebungen sinnvoll, bei denen wenig unterschiedliche Instanzklassen

verwendet werden. Das Modell eignet sich besonders gut, um in Multi-Account Umgebungen Kosten einzusparen, da eine reservierte Instanz bei der Gesamtrechnung berücksichtigt wird.

Genau wie reservierte Instanzen, die eine langfristige Bindung belohnen, wird eine spontane Übernahme von überschüssigen Kapazitäten durch die Verwendung von Spot Instanzen honoriert.

Spot Instanzen werden über einen Marktplatz bei AWS versteigert. Der Kunde gibt sein Höchstgebot und die gewünschten Instanztypen an und erhält eine Instanz sobald der Marktpreis unter dem Höchstgebot ist. Steigt der Marktpreis wieder über das Höchstangebot, wird die Instanz innerhalb einer Zeitspanne von zwei Minuten entzogen und terminiert. Dem Kunden wird immer nur der Marktpreis in Rechnung gestellt, nicht das Höchstangebot.

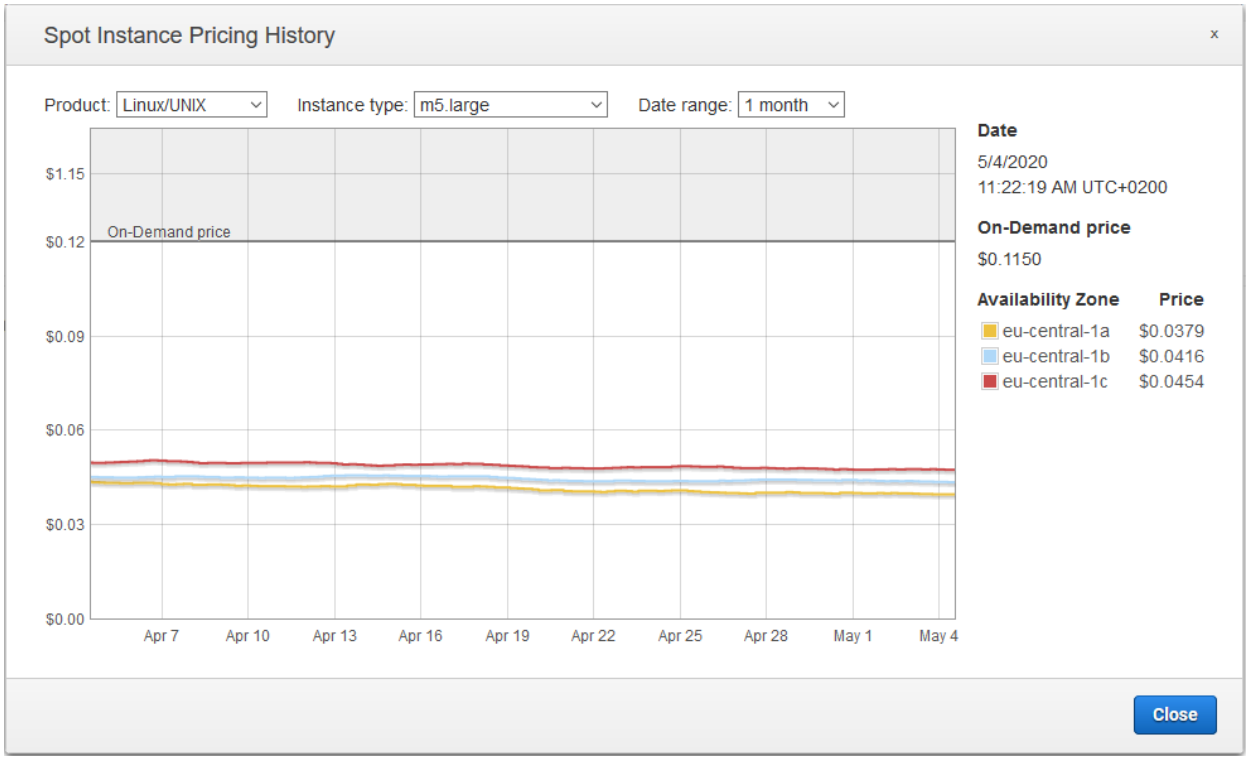

Abbildung 1: Entwicklung des Spot Marktes im April/Mai 2020

Durch Spot Instanzen kann eine Einsparung von bis zu 90% erreicht werden. Aus diesem Grund lohnt es sich die bestehende Architektur zu hinterfragen und Möglichkeiten zu finden, um Spot Instanzen einzusetzen. Abbildung 1 zeigt einen Ausschnitt der Preisentwicklung im April 2020. Hier wird ein Preisnachlass von bis zu 68% im Vergleich zum on-Demand Preis realisiert.

Typische und schnell umzusetzende Anwendungsfälle für Spot Instanzen sind

- Worker Instanzen, die eine Queue abarbeiten,
- Batch Jobs,
- zusätzliche kurzfristige Kapazitäten bei Auslastungsspitzen.

Darüber hinaus lassen sich aber auch normale Anwendungen auf Spot Instanzen betreiben. Eine einfache Möglichkeit, um erste Erfahrungen mit Spot zu sammeln ist die Verwendung als Instanz in einem Docker Cluster (zum Beispiel im Zusammenspiel mit ECS oder EKS).

Auch wenn die Instanz jederzeit in einem Zeitraum von zwei Minuten entzogen werden kann, zeigt Abbildung 1, dass die Instanz einen ganzen Monat ohne Unterbrechungen betrieben werden konnte, wenn das Höchstangebot auf den on-Demand Preis gesetzt wurde.

7. Lizenzkosten

Vor allem nach der Migration von Altsystemen, sollte eine Anpassung der verwendeten Drittsysteme auf Open Source Alternativen in Betracht gezogen werden. Während es unwahrscheinlich ist, dass EC2 Windows Systeme auf Linux umgestellt werden können, ist es häufig möglich lizenzpflichtige Linux Distributionen durch lizenzfreie Varianten zu ersetzen.

Zum Vergleich der Lizenzkosten wird im Folgenden der m5.large (2 vCPUs, 8 GB RAM) Instanztyp in der Region Frankfurt (eu-central-1) genutzt. Tabelle 1 zeigt die Kosten der verschiedenen Betriebssysteme. Diese Kosten sind zum Stand April 2020 aufgeführt.

Windows	Red Hat Enterprise Linux (RHEL)	SuSE Linux Enterprise (SLES)	Ubuntu / Amazon Linux
$0,207	$0,175	$0,215	$0,115

Tabelle 1: Kosten einer EC2 Instanz (m5.large; eu-central-1) [3]

Tabelle 2 zeigt die Kostenentwicklung bei einem Wechsel des Betriebssystems.

Bei nur temporär laufenden EC2 Instanzen, wie es bei Batch Jobs der Fall sein kann, ist eine Umstellung auf Ubuntu oder Amazon Linux sogar mit einer Veränderung des Abrechnungszeitraumes verbunden: Während

andere Systeme pro Minute abgerechnet werden, folgen Amazon Linux und Ubuntu einem Abrechnungstakt pro Sekunde.

Von / Nach	SLES	Windows	RHEL	Ubuntu / Amazon Linux
SLES	0%	-4%	-19%	-53%
Windows	+4%	0%	- 15%	-44%
RHEL	+23%	+18%	0%	-34%
Ubuntu	+87%	+80%	+52%	0%

*Tabelle 2: **Kosteneinsparungsmatrix** EC2 Betriebssysteme*

Auch bei RDS Datenbanken können Lizenzkosten durch einen Wechsel auf eine Open Source Datenbank eingespart werden. Tabelle 3 zeigt die Kosten einer r5.large RDS Instanz im Multi AZ Modus in der Region Frankfurt. Die darauf folgende Tabelle 4 vergleicht diese Kosten miteinander und zeigt die Einsparpotentiale (negative Werte) auf. Ein Wechsel von Oracle auf PostgreSQL führt demnach zu einer **Kostenersparnis von 49%**.

Oracle SE 2	PostgreSQL	MySQL/MariaDB
$1,188	$0,61	$0,58

Tabelle 3: Kosten einer RDS Instanz (r5.large; eu-central-1) [4]–[7]

Von / Nach	Oracle SE2	PostgreSQL	MySQL / MariaDB
Oracle SE2	0%	-49%	-51%
PostgreSQL	+95%	0%	-5%
MySQL	+105%	+5%	0%

Tabelle 4: Kosteneinsparungsmatrix RDS

Ist ein Datenbankwechsel auch nach ausführlichen Tests nicht möglich, kann bei Oracle Datenbanken die Anzahl der CPU Kerne reduziert werden, um somit auf anderem Wege Lizenzkosten einsparen zu können [8].

Auch wenn ein Wechsel nicht ohne Weiteres kurzfristig durchführbar ist, lohnt es sich die Zeit für entsprechende Kompatibilitätstests zu investieren und einen Betriebssystems oder Datenbank Wechsel vorzunehmen.

8. Trusted Advisor & Cost Explorer

Der Service Trusted Advisor hilft den Kunden mit Business oder Enterprise Support Plan bei der Einsparung von Kosten. Streng genommen ist der Trusted Advisor Service selbst keine Kosteneinsparungsmaßnahme, jedoch unterstützt dieser ungemein bei der Erreichung einer geringeren Rechung.

Dieser Service gibt Hinweise auf ungenutzte Loadbalancer, EC2 Instanzen mit überschüssigen Kapazitäten, nicht genutzten Elastic IP Adressen und vieles mehr.

Dieses Werkzeug gibt nützliche Hinweise, um überschüssige Ressourcen zu identifizieren, um diese anschließend dem Bedarf anzupassen.

Für Kunden ohne Business und Enterprise Support bietet der AWS Cost Management Service eine Möglichkeit, um Instanzen zu identifizieren, bei denen eine Kosteneinsparung durch reservierte Instanzen möglich ist. Zusätzlich wird eine geschätzte monatliche Einsparung angezeigt.

9. Fazit

Amazon Web Services bietet einige Möglichkeiten, um in kurzer Zeit die monatlichen Kosten zu reduzieren. Neben klassischen Kostenreduktions-strategien, wie beispielsweise der Reduzierung von überschüssigen Ressourcen, gibt es auch administrative Mechanismen, wie die konsolidierte Faktorierung zur Nutzung von Rabatten.

Des Weiteren lassen sich durch strategische Entscheidungen langfristig die Kosten deutlich reduzieren, indem auf die Nutzung von Open Source Produkten gesetzt wird. Außerdem kann durch Flexibilität (Spot-Instanzen) bzw. langfristiger Bindung (reservierte Instanzen) eine beachtliche Kosteneinsparung realisiert werden.

Durch die Vielfältigkeit der Amazon Cloud ist es möglich verschiedene Wege zu beschreiten, die zum gleichen Ziel führen. Die Wege unterscheiden sich im Wesentlichen durch die entstehenden Kosten. Um den richtigen Kosten-Nutzen-Effekt zu erzielen, ist es wichtig, dass die einzelnen Kostenfaktoren der Amazon Services tiefgehend verstanden werden.

Durch meine tägliche Praxis und meine Erfahrung kenne ich die verschiedenen Stellhebel, die zur Optimierung der Kosten betätigt werden können. In meinen Projekten konnte ich durch konsequente Anwendung

von Kosteneinsparungsmaßnahmen die AWS-Rechnung meiner Kunden deutlich verschlanken und reduzieren.

Falls auch Sie sehr hohe monatliche Kosten durch die Nutzung von AWS verbuchen müssen, kontaktieren Sie mich gerne für einen unverbindlichen Termin.

Übrigens: Häufig führen Kostenreduktionen der Gesamtrechnung auch zu geringeren Supportkosten, wodurch noch mehr eingespart werden kann.

10.Literaturverzeichnis

[1] Amazon Web Services, „How AWS Pricing Works“, Amazon Web Services, 2018. [Online]. Verfügbar unter: https://d1.awsstatic.com/whitepapers/aws_pricing_overview.pdf.
[2] Amazon Web Services, „Amazon S3 Simple Storage Service Pricing“. https://aws.amazon.com/de/s3/pricing/.
[3] Amazon Web Services, „Amazon EC2 Pricing“, *EC2 Instance Pricing*, Apr. 30, 2020. https://aws.amazon.com/ec2/pricing/on-demand/?nc1=h_ls.
[4] Amazon Web Services, „Amazon RDS for Oracle Pricing“, Apr. 30, 2020. https://aws.amazon.com/rds/oracle/pricing/.
[5] Amazon Web Services, „Amazon RDS for PostgreSQL Pricing“, Apr. 30, 2020. https://aws.amazon.com/rds/postgresql/pricing/.
[6] Amazon Web Services, „Amazon RDS for MySQL Pricing“, Apr. 30, 2020. https://aws.amazon.com/rds/mysql/pricing/.
[7] Amazon Web Services, „Amazon RDS for MariaDB Pricing“, Apr. 30, 2020. https://aws.amazon.com/rds/mariadb/pricing/.
[8] Amazon Web Services, „Introducing Optimize CPUs for Amazon RDS for Oracle“, Apr. 30, 2020. https://aws.amazon.com/about-aws/whats-new/2018/06/introducing-optimize-cpus-for-amazon-rds-for-oracle/.