Platform as Service

Grundidé.

Applikation – Försäljning av iPhones Affärsidé: För detta krävs en PaaS lösning där de verktyg finns tillhanda som löser kommande problem och utmaningar. PaaS är i detta arbete för min del det bästa därför att jag slipper tänka på den grundliga infrastrukturen så som virtuella maskiner och databashantering etc.

Tekniska behov:

- Säker kundinloggning
- Brandväggar och säkert nätverk
- Lagring
- Ekonomiskt perspektiv (indirekt behov) Hantering av data - skalbarhet - optimering – övervakning
- Betallösningar
- Orderhantering Övergripande teknisk lösning:

För att göra på något sätt gå i mål med applikationen krävs lösningar så som:

- Struktur för de tekniska behoven. (ej alla ägg i samma korg)
- Frontend
- Backend (Functions)
- (Skalbarhet – cdn)
- Säkerhet (HTTPS/brandväggar)
- Övervakning av trafik

Microsoft Azure – Platform As A Service

(Efter Azure kommer du kunna läsa om leverantören AWS)

Inledning

Denna rapport handlar om och kommer uppvisa lite kring hur vi designar och sätter upp en molnmiljö i Microsoft Azure. Jag har valt att fokusera på tjänsten Platform as a Service, som är en molnbaserad miljö där utvecklare kan skapa applikationer och driftsätta i molnet. Platform as a Service och tjänsten App Service är ett mycket effektivt verktyg för att snabbt och enkelt köra kod vid skapandet av applikationer. För att göra det kommer denna rapport visa några steg som krävs för att sätta upp en fungerande och säker miljö.

Azure erbjuder ett imponerande utbud av över 200 olika tjänster som sträcker sig från molnlagring och databashantering till avancerade verktyg för maskininlärning och AI. Dessa tjänster utgör grunden för att hantera och utnyttja storskaliga datamängder.

Vad är molntjänster och varför är det så viktigt?

Molntjänster är teknologi som administreras och används i molnet ute på internet för att dela och nå information. Infrastrukturer och datacenter finns runt om i världen och delas in i många olika zoner. I stället för att du som företagare ska sätta upp en egen datahall med servrar, routers, kablar etcetera kan vi alltså använda oss av olika typer av molntjänster och dess resurser från nästan vilken plats som helst i världen så länge du har en internetuppkoppling.

Molntjänster ger utvecklare och användare tillgång till att nå till exempel resurser som applikationer och lagring via det globala internetet. Molntjänster är flexibelt vilket gör att skalbarheten är en nyckelfaktor här. Om ditt företag behöver mer lagringsutrymme eller andra typer av resurser går detta administrera via skalbarhet. Med skalbarhet kan du alltså gå från 200 till 400 gigabyte lagringsutrymme för att inte gå miste om någon viktiga data eller öka CPU resurser för att inte låta din applikation bli överbelastad. Det gäller dock att din miljö är konfigurerad på rätt sätt.

Det kan vara kravbilden och budget som styr hur din miljö är konfigurerad men molntjänsterna i dag har faktureringsmodeller som ”Pay-as-you-go”, vilket betyder att du betalar bara för de resurserna du använder vilket gör ”cloudcomputing” i många lägen kostnadseffektivt. Mer om detta i avsnittet kostnader.

Med den snabba digitaliseringen och utveckling som skett är efterfrågan på molntjänster betydligt större idag än för många år sedan. Fler och fler går över till molnet vilket det finns många argument till att göra så som att företagen själva slipper tillhandahålla mark, lokal, personal, hård och mjukvara och som tidigare nämnts att den skalbarhet, flexibilitet och kostnadseffektiviteten det kan innebära.

I dagens samhälle är det nästan ett ”krav” med snabb leverans och utveckling av applikationer vilket gör det möjligt med tjänster som Platform As A Service då vi slipper underhålla en hel datahall med allt vad detta innebär.

Metod:

Först ska vi se över kostnader och de olika faktureringsmodeller som vi behöver ta hänsyn till när vi skapar vår Azure-miljö. Detta är helt och hållet företags kravbild och behov som styr detta.

Kostnader och Faktureringsmodeller:

Valet av faktureringsmodell beror på organisationens storlek, budget och vad syftet är med tjänsterna som krävs. Företaget bör även ta hänsyn till användningsmönster och behov av flexibilitet och kostnadsförutsägbarhet.

Microsoft Azure erbjuder flera faktureringsmodeller för att passa olika behov och krav. Två av dom är till exempel:

Pay-As-You-Go: Detta är den mest flexibla modellen där du betalar för de resurser du använder, vanligtvis per timme eller per minut. Det finns ingen långsiktig förpliktelse, och du kan skala upp eller ned efter behov. Om företaget har tider på året där det krävs mycket resurser för stunden är denna modell bra. Många företag säljer till exempel biljetter till olika event eller låt oss ta exempel ”black Friday”. Här krävs mer resurser i form av CPU vilket då företaget bara betalar för just denna period.

Enterprise Agreement: Detta är ett avtalsbaserat program för stora organisationer som ger rabatter baserat på förbetald användning och en förpliktigad användningsvolym under en viss period. Inkluderar vanligtvis kund och teknisk support samt möjligheten att konsolidera fakturering över flera avdelningar eller geografiska områden.

När vi har budgetar att förhålla oss till kan vi inte att välja och vraka resurser och sätta upp en miljö som vi vil, detta kommer att resultera i en extremt dyr kostnad. Vi behöver se över krav och behov när vi skapar vår Azure-miljö

Det är viktigt att när vi sätter upp resurser över till exempel olika avdelningar eller geografiska områden att vi använder av det som kallas för ”TAGS”. Här kan vi lägga in vilka resurser och se vilka avdelningar som använder vad. Genom de olika tags kan vi alltså övervaka kostnader med mera.

Med hänsyn till kravbilden ser vi om det finns några problem eller bekymmer med de olika faktureringsmodellerna. Att förlita sig på tredjeparts kan i vissa fall skapa vissa bekymmer eftersom vi PaaS tjänst inte tillhandahåller infrastruktur och datahantering, detta kan innebära risk. Ska vi betala för något som möjligtvis skapar problem eftersom vi själva inte kan styra över infrastrukturen.

Värdera och analysera kostnader och risker vid övervägandet och användning av Platform as a service och säkerställ att den verkligen passar kravbildens behov och budget.

Det finns alltid för och nackdelar med det mesta. När vi pratar om automatiseringen kan detta bidra med att sänka vissa kostnader genom att automatisera resurser vid behov och att vi dels slipper mycket av det manuella arbetet som i sig också är en kostnad. Det vi bör tänka på är att en rätt konfigurerad miljö är oftast bättre än en mänsklig då vi minimerar risken för felkonfiguration.

Med hjälp av verktyg och tjänster som till exempel virtuella maskiner och containers kan vi skapa säkra och moderna molninfrastrukturer när vi skapar vår miljö, vi kommer dock tillbaka till de kostnader det medför och hur kravbilden och behov ser ut. Det finns många aspekter att ta hänsyn till här vilket gör det extremt viktigt att en tydlig och analyserad kravbild om vad som ska utföras och hur budget ser ut.

Att sätta upp en Azure-miljö:

För detaljerad konfiguration: Se bifogade JSON filer.

I mitt försök att sätta upp och konfigurera min miljö har jag genom en strukturerad plan och tankesättet ”från grunden och utåt” jobbat mig igenom de olika moment för att nå ett slutresultat. Det har funnits begränsningar och problem på vägen som tas upp till viss del i denna rapport.

Network security group: Det allra första jag har gjort är att sätta upp en ”Network Security Group”. Denna resurs är en central del i den säkerhet vi vill bygga som möjliggör en strukturerad och kontrollerad miljö. Med en NSG ges ökad säkerhet och en större kontroll på den nätverkstrafik som flödas inom nätverket. Det vi måste tänka på är att regler och standarder ska följas vilket gör NSG till en utmärkt resurs för detta ändamål. Hur denna konfigureras är helt beroende på den kravbild som företaget ställer.

Virtuellt nätverk: Efter min NSG har ett virtuellt nätverk skapats. I detta virtuella nätverk ska det i senare skede sättas upp subnät som har konfigurerats för att isolera resurser för att skapa en säker miljö.

Subnät: För att på något sätt skapa en säker och isolerad miljö till de olika resurserna har några subnät skapats. Detta för att öka säkerheten och att rätt trafik transporteras i nätverket.

Det som är bra med virtuella nätverk är att det är flexibelt vilket gör att vi snabbt kan skapa och konfigurera och förändra inställningar för enheter som ska tillåtas anslutning.

Brandvägg: En brandvägg har inte satts upp då kostnaden vart för dyr. Dock är det viktigt att nämna att detta oftast är ett krav som möjliggör att vi kan styra vilken trafik som är tillåten och vad för trafik som ska blockeras.

Webbapp Fire Wall: När en app service skapas har jag valt att ge extra skydd för mitt nätverk genom en WAF. Från det publika internetet vill jag ha så liten tillgång som möjligt till min app servic och detta görs med att sätta upp en så kallad WAF. Detta är inget som finns i min miljö men den är lika viktig som en vanlig brandvägg ut mot det publika.

Lagringskonto: Möjliggörandet för att hantera lagring av data har jag valt att ha med lagringskonto och databaser. Lagringskonto och databaser med tillhörande databasserver har jag separerat i olika subnät för att isolera dessa resurser från varandra. Detta dels för säkerhet, dels för prestanda och skapa möjligheter till skalbarhet.

App Service: För att möjliggöra drift av applikationer har resursen web application service med tillhörande ”App Service plan” satts upp. Eftersom vi vill skapa en säker miljö och minska risken för intrång har en Web Application Firewall satts upp i samband med detta och det är till största del för att skydda applikationen för attacker från internet.

Nat Gateway: Genom en nätverks-gateway som är kopplad mellan min Webb applikation och ett specifikt subnät kan trafiken styras inom det virtuella nätverket som i

sin tur ger åtkomst till andra resurser i andra subnät om det skulle krävas. Att dela upp och isolera är en säkerhetsaspekt som vi inte får lägga åt sidan.

Med implementeringarna som gjort för en bra och säker miljö är dessa helt nödvändiga för att köra och hantera webapplikationer på ett säkert och effektivt sätt.

Övervakning och loggar: Det som vi nu har gått igenom är hur vi sätter upp en säker och robust miljö. Det krävs också en noggrann övervakning och loggning för att på något sätt hantera och optimera den miljö som vi har. Det finns tre delar som är vikta att ta hänsyn till. Loggar, övervakning och skalning. Dessa tre sätter vi i någon form av kategori då det är loggning och övervakning som ger oss information på hur skalningen ska gå till och vad det är som faktiskt krävs, dock återigen tillbaka till kravbilden och den budget som finns.

Med loggning kan vi logga olika aktiviteter och analysera vad som händer över tid, denna data kan innehålla flera olika av händelser. Övervakningen övervakar prestandan i systemet i realtid vilket möjliggör att vi kan följa användningen av CPU-kraft, minne och nätverkstrafik. Detta kan vi följa i en resurs i Azure som heter ”Metrics” som går att ställa in på hur ofta övervakningen ska ske.

Via loggning och övervakning skapas möjligheten att förstå hur och vad vi ska skala och vilka resurser som behöver skalas upp eller ut. Behövs fler databaser kan vi skala upp eller behöver mer CPU-kraft kan vi skala ut. Vilket som är bäst för situationen är det svårt att svara på här och nu utan det är beslut som kan tas via estimeringar och med hänsyn till krav och budget.

Tjänsten ”Alert” är ett viktigt verktyg för att säkerställa att inte kostnader sticker i väg. Om din miljö inte är tillräckligt bra och konfigurerad på rätt sätt finns det risk att kostnader kan dra iväg utöver budget. Därför kan vi använda oss av alerts och jag har en alert som jag satte upp på100$.

Resultat

Innan vi går vidare och ser över miljön och vad den innehåller ska vi se över på lite resurser och teknologier som jag inte kunnat implementera men som är fullt möjligt.

När webbapplikationen har implementerats har tankarna om ”App Service Enviroment” kommit på tankarna och hur detta skulle kunnat ha implementerats. Kunskaperna om hur detta skulle sättas upp i en säker nätverkstopologi har förmodligen inte funnits och därför har jag lagt detta åt sidan men hade gärna velat veta hur detta fungerar.

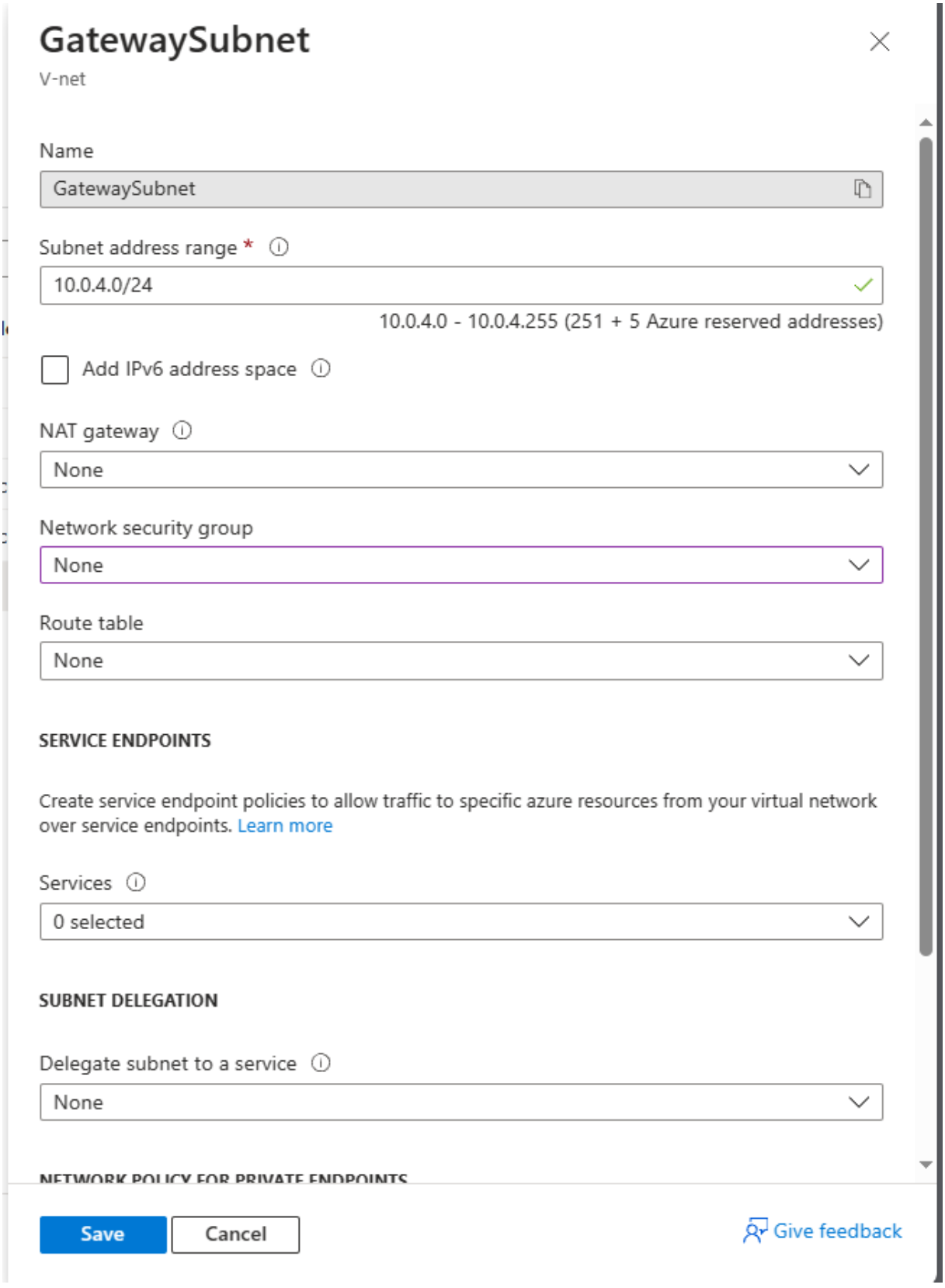
Eftersom vi i en Paas miljö inte använder oss av Virtuella maskiner i normala fall har inte detta implementerats i min Azure miljö. Men teknologier som Virtuella maskiner och containers är till stor hjälp i vissa fall.

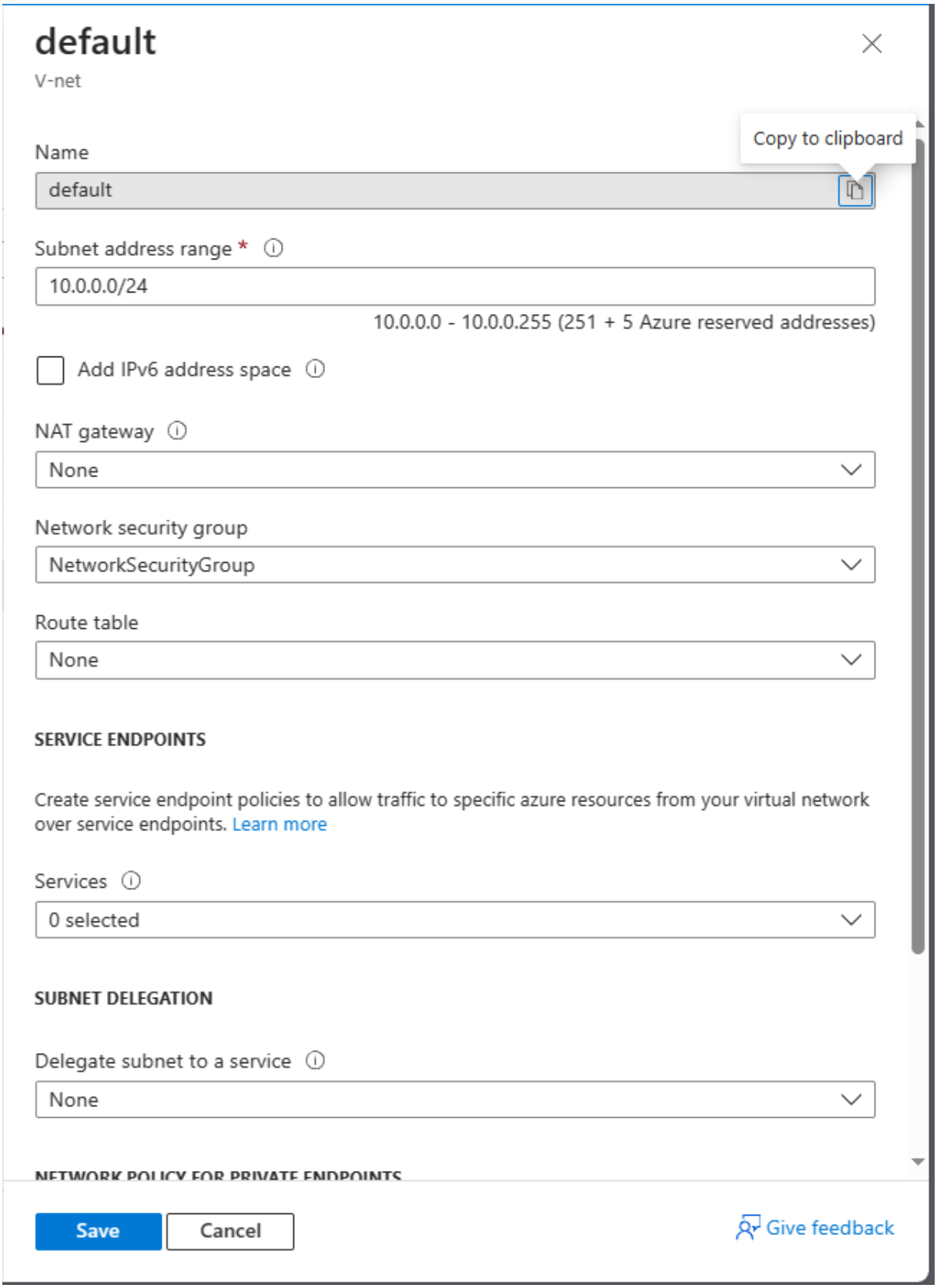
Virtuella maskiner används när vi ska simulera fysiska datorer. En VM sätts upp med eget operativsystem och applikationer som körs i en isolerad miljö. Det är ett kraftfullt paket som tar mycket minne och lagringsutrymme vilket gör att det tar lång tid att starta och stoppa dessa tjänster. Containers däremot kräver inte samma mängd resurs och kraft som en Virtuell maskin vilket gör att det går mycket snabbare att starta en container. Containers går att likställa med en liten låda som innehåller det som krävs för att köra en applikation. Det går alltså fylla en container med innehåll som du kan flytta över till dator eller virtuella maskiner som enkelt går att köra. En container är självständig och är lätta att distribuera och köra i vilken dator som helst.

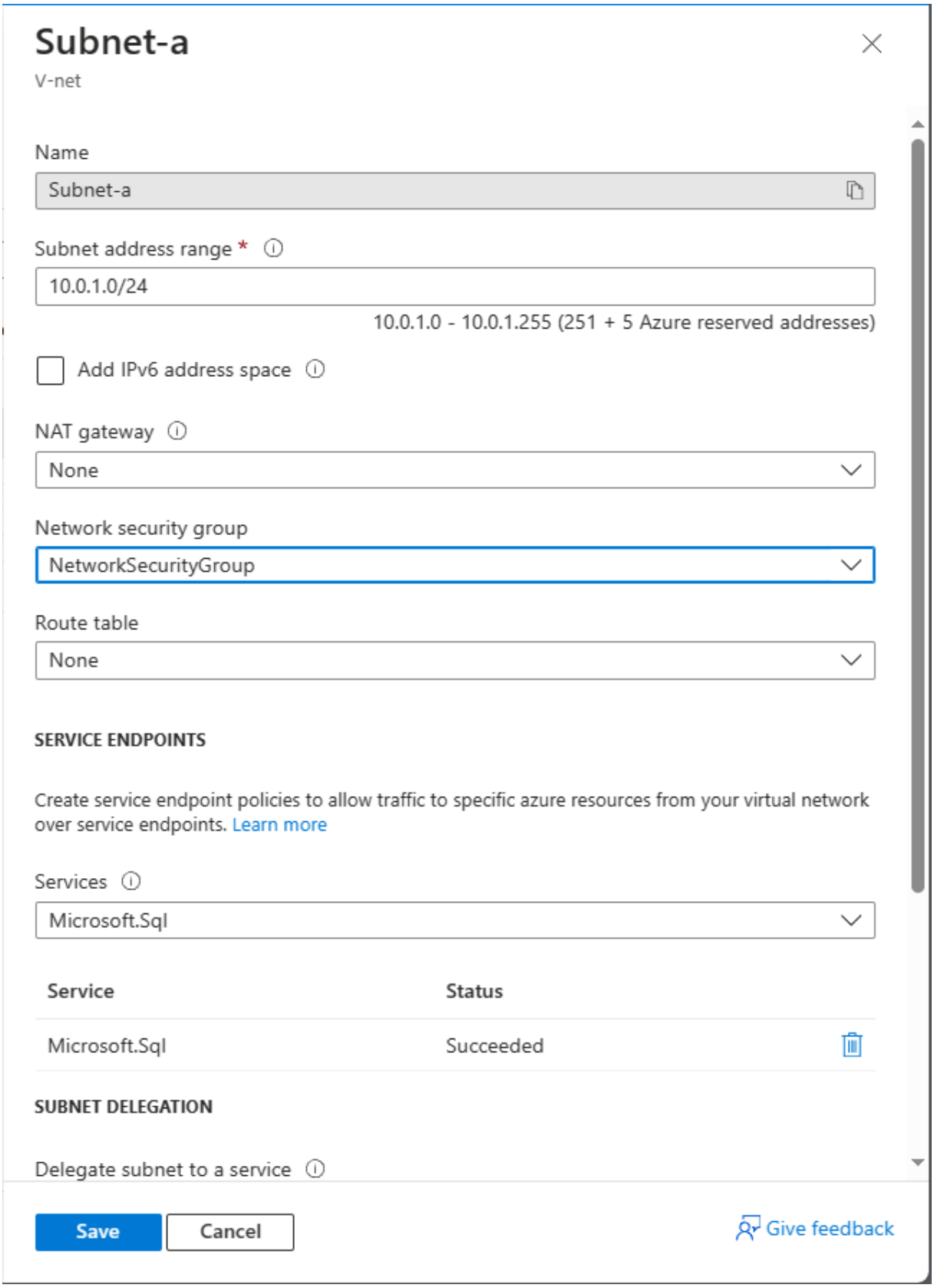
Jag har även försökt lägga till ett till virtuellt nätverk för att skapa någon slags integration mellan två virtuella nätverk där subnätet för applikationen skulle legat i det. Dock har jag haft strul med detta.

Här kan vi se hur mitt virtuella nätverk med tilldelade subnäten ser ut. Som bilden uppvisar ligger alltså inte min webbapplikation i NSG som tillhör de tilldelade subnäten. Detta är för att webbapplikationen är en global tjänst som ligger utanför vilket gör att jag måste ha Gateway till subnätet för just den specifika resursen.

Virtuellt nätverk med de underliggande subnäten och anslutningar.




default
V-net
Copy to clipboard
Name
default
Subnet address range *
10.0.0.0/24
10.0.0.0 - 10.0.0.255 (251 + 5 Azure reserved addresses)
Add IPv6 address space
NAT gateway
None
Network security group
NetworkSecurityGroup
Route table
None
SERVICE ENDPOINTS
Create service endpoint policies to allow traffic to specific azure resources from your virtual network over service endpoints. Learn more
Services
0 selected
SUBNET DELEGATION
Delegate subnet to a service
None
Save
Cancel
Give feedback


Subnet-a
V-net
Name
Subnet-a
Subnet address range *
10.0.1.0/24
10.0.1.0 - 10.0.1.255 (251 + 5 Azure reserved addresses)
Add IPv6 address space
NAT gateway
None
Network security group
NetworkSecurityGroup
Route table
None
SERVICE ENDPOINTS
Create service endpoint policies to allow traffic to specific azure resources from your virtual network over service endpoints. Learn more
Services
Microsoft.Sql
Service
Status
Microsoft.Sql
Succeeded
SUBNET DELEGATION
Delegate subnet to a service
Save
Cancel
Give feedback

Subnet-B

V-net

Name

Subnet-B

Subnet address range *

10.0.2.0/24

10.0.2.0 - 10.0.2.255 (251 + 5 Azure reserved addresses)

Add IPv6 address space

NAT gateway

appservice-gateway

Network security group

NetworkSecurityGroup

Route table

None

SERVICE ENDPOINTS

Create service endpoint policies to allow traffic to specific azure resources from your virtual network over service endpoints. Learn more

Services

0 selected

SUBNET DELEGATION

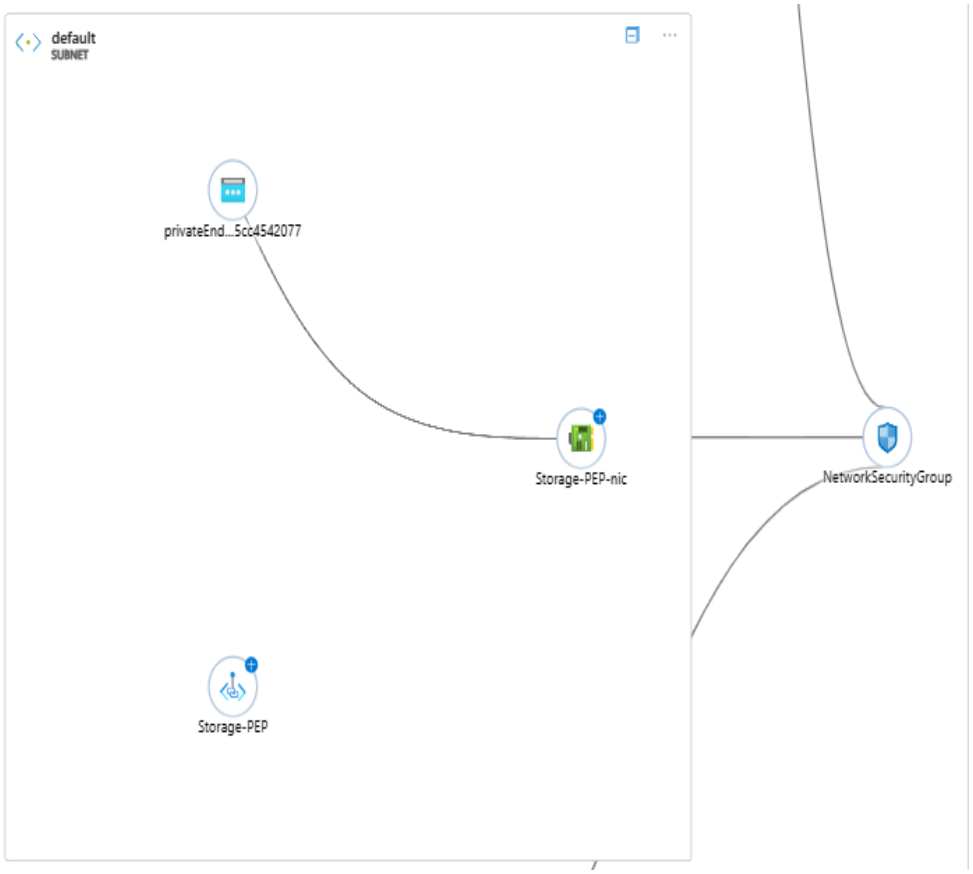
Delegate subnet to a service

Microsoft.Web/hostingEnvironments

NETWORK POLICY FOR PRIVATE ENDPOINTS

Save Cancel Give feedback


V-net
VIRTUAL NETWORK
Subnet-a
SUBNET
privateEn...983676232
database-pep-nic
database-pep


default
SUBNET
privateEnd...5cc4542077
Storage-PEP-nic
NetworkSecurityGroup
Storage-PEP

VNet Integration

webappproject

Disconnect Refresh Troubleshoot

VNet Configuration

Securely access resources available in or through your Azure VNet. Learn more

Route All

Disabled

When Route All is enabled, all outbound traffic from the app will be affected

VNet Details

VNet NAME	V-net
LOCATION	North Europe

VNet Address Space

Start Address	End Address
10.0.0.0	10.0.255.255

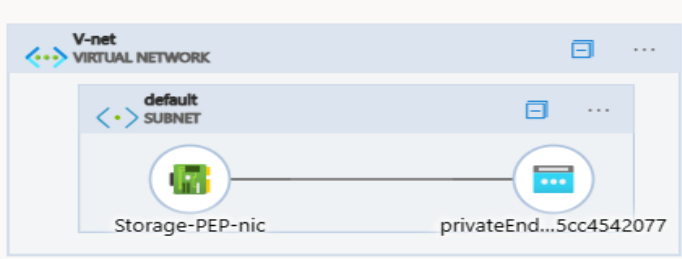
Subnet Details

Subnet NAME	VNet-Integration

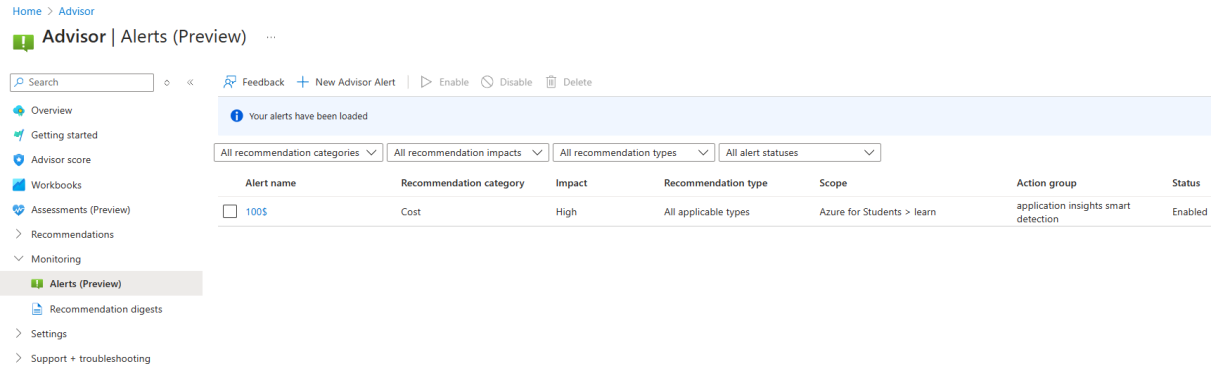
Subnet Address Space

Start Address	End Address
10.0.3.0	10.0.3.255

Bilderna här visar subnätet där mitt lagringskonto är kopplat till som styrs via en service link och en network interface för ökad säkerhet.

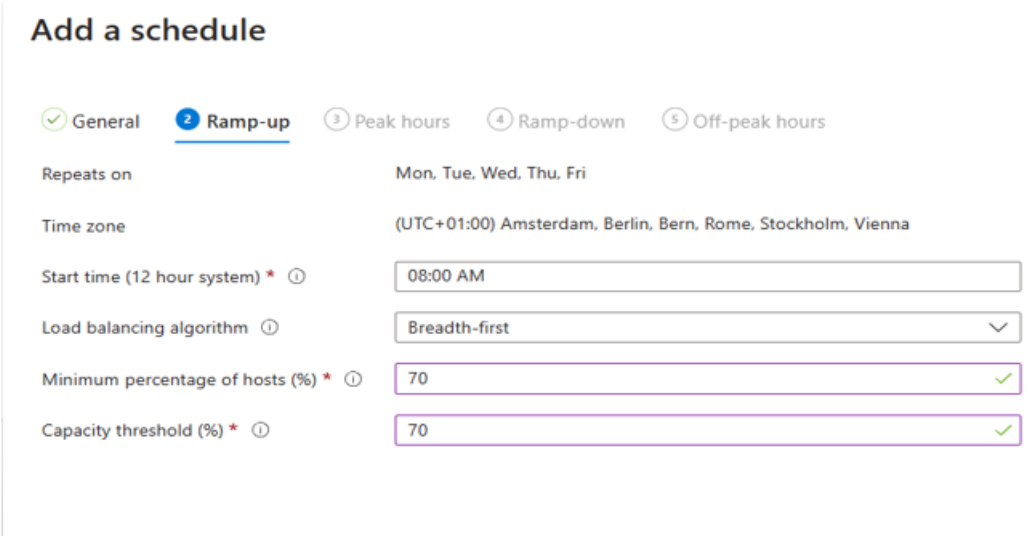


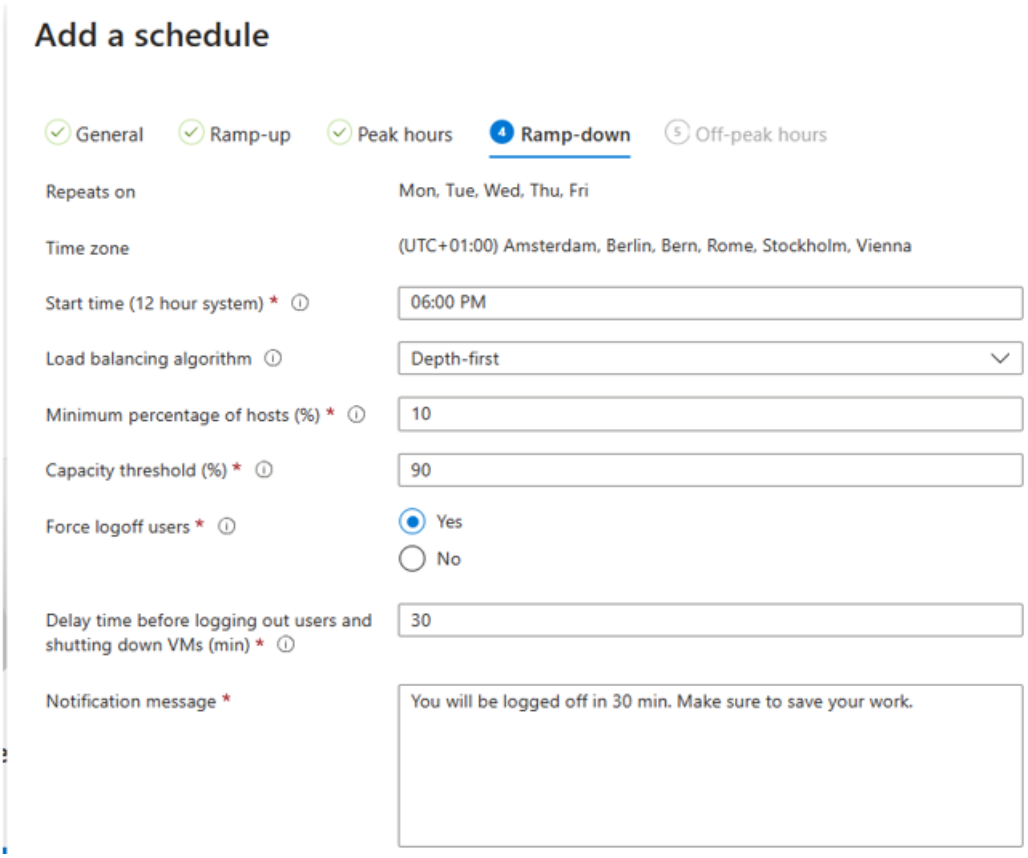
Här ser vi ett exempel på hur det går att sätta upp Alerts. Om kostanden uppgår till 100$ vill jag ha en varning till adresserad email.



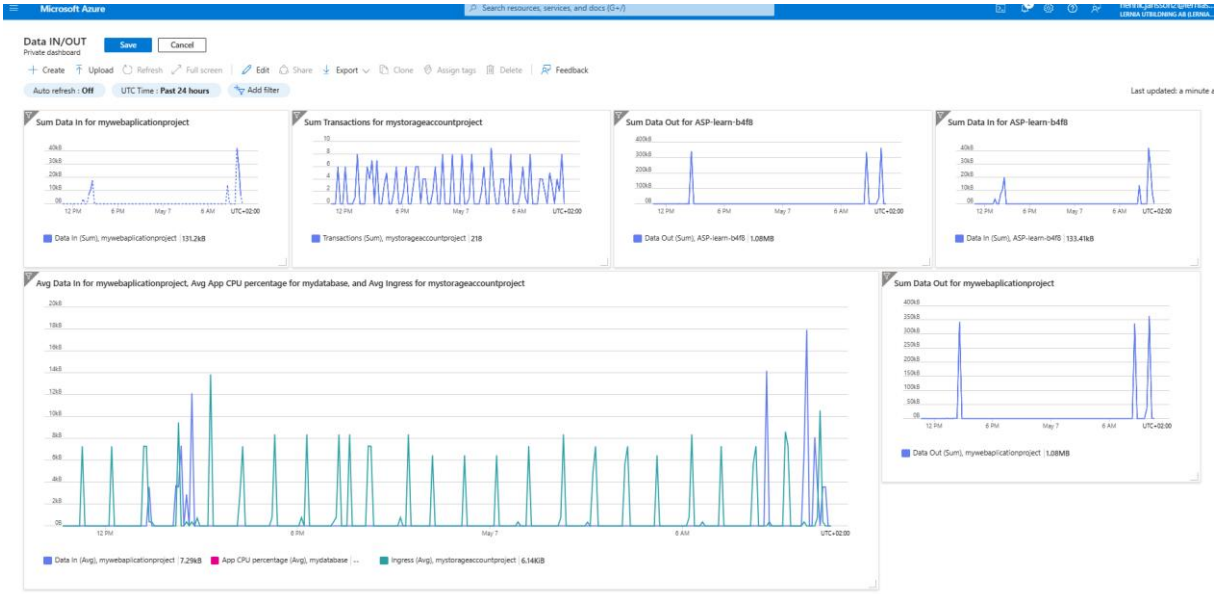
Exempel på hur en lastbalanserare och skalning kan sättas upp.
Detta är inget som min miljö innehåller då jag haft svårt och inte har skapat mig kunskaperna för detta om hur detta ska sättas upp.

Ramp-up & Ramp-Down





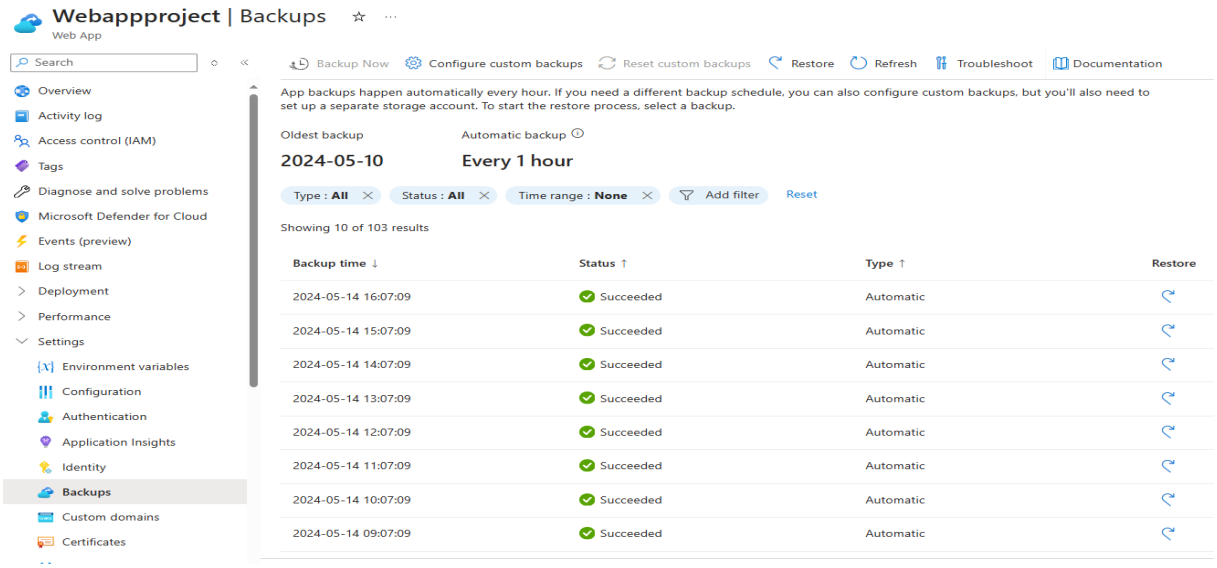
Azure har över 200 resurser och möjligheterna att få data och analysera detta är väldigt stor. Som vi kan se i detta exempel har jag handplockat några av de resurser jag har.

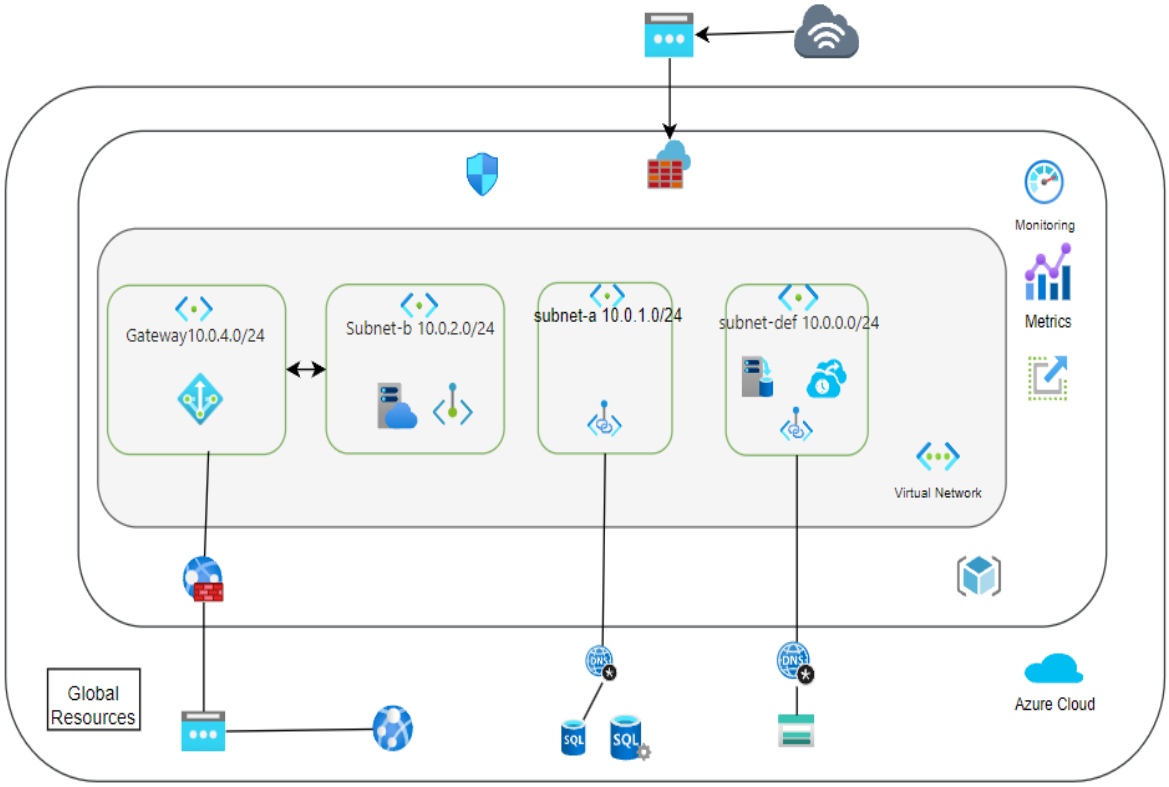


Här kan vi se hur en ”Metric” kan se ut efter vi gjort övervakning och logging. Detta sätts upp för specifika resurser eller i grupp där vi kan se data in/out , CPU användande och mycket mer. En bra verktyg som kan ge oss en bild över hur vår miljö mår. Det vi vill undvika till varje pris är långsamma och hög belastade miljöer som blir tröga eller stängs ner.

Backups för App Service

Jag har satt min backup i detta fall varje timma. Detta beror helt och hållet på hur kravbilden ser ut och hur detta ska administreras.





Se fullständig konfiguration i bifogade JSON filer.

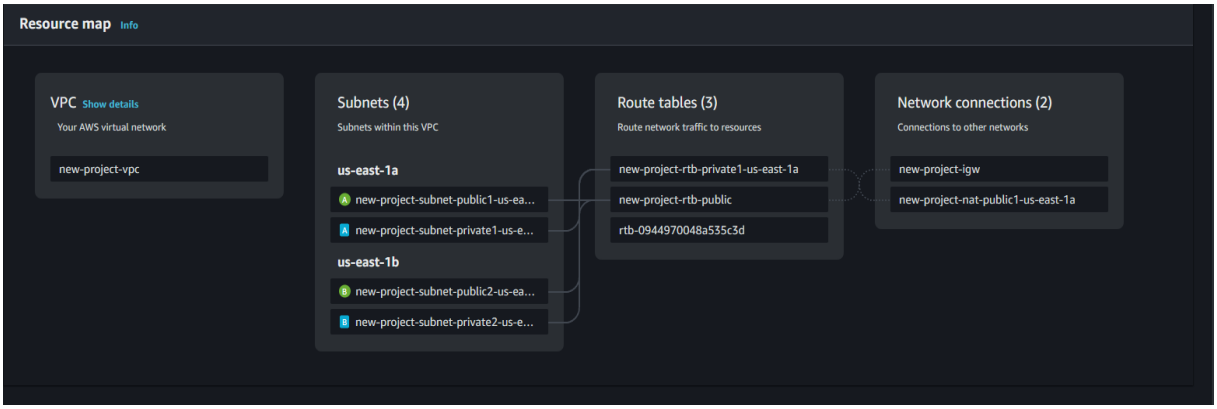
Amazon Web Service (AWS)

Här kommer jag uppvisa hur jag har använt AWS som molntjänstleverantör och hur en miljö kan sättas upp.

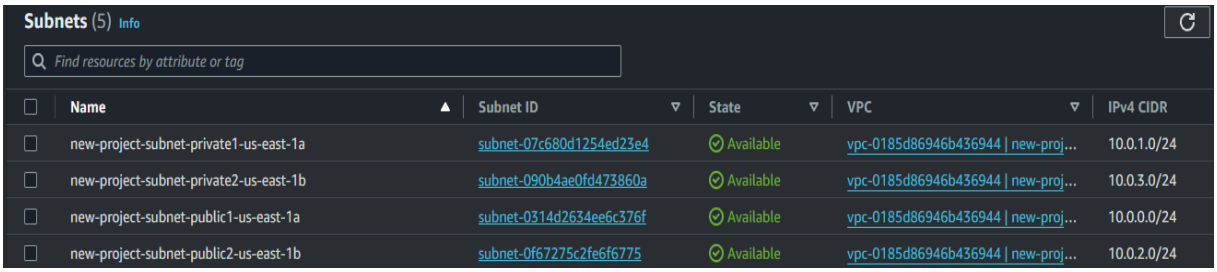
Metod:

VPC: När vi skapar vårt VPC kan vi välja vilken zon vi vill ha vårt nätverk i. I detta projekt är vi väldigt begränsade och vald zon är North.Virginia. När vi skapar ett VPC med tillhörande subnät så skapas automatiskt nätverksanslutningar. Som vi kan se på bilden visar det våra subnät och ett route table och anslutningar. Dessa två anslutningar möjliggör anslutning från det publika internetet.

När vi skapar vårt VPC sätts en ACL (Access Control List), en lista där vi kan kontrollera och styra åtkomsten till olika resurser som finns inom vårt VPC. En viktig säkerhetsaspekt att ta hänsyn till. Det som är bra med denna tjänsten är att vi kan spåra och logga trafik inom miljön.

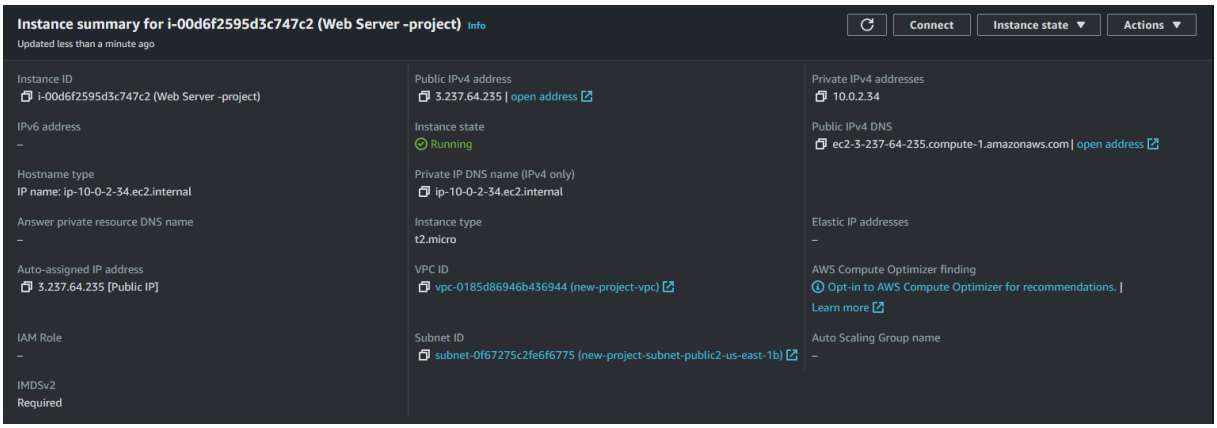


SUBNÄT:



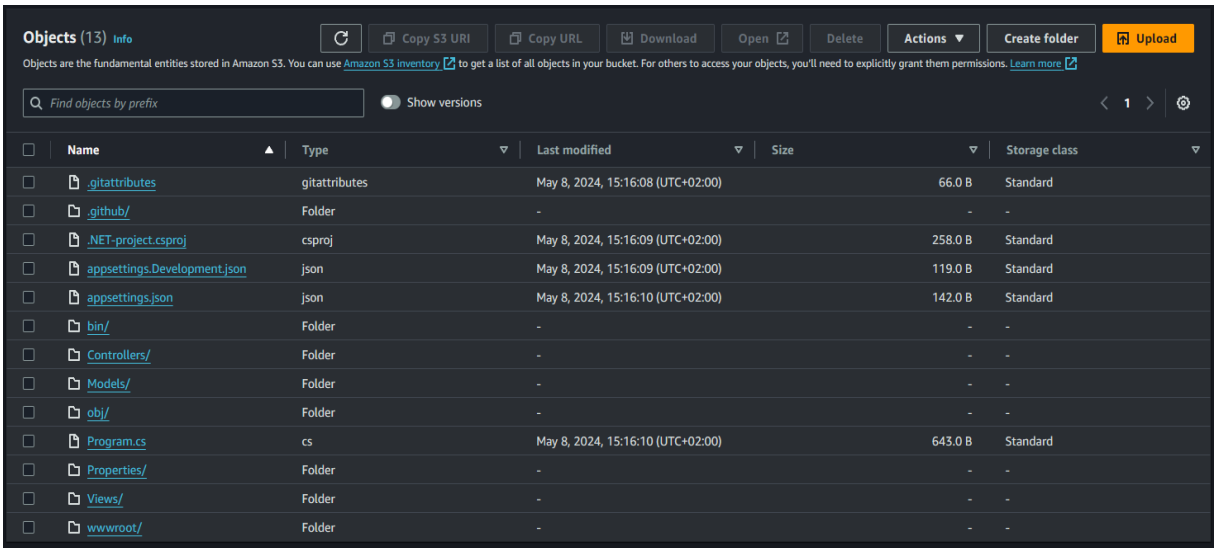
EC2 INSTANCE: Är en virtuell server som är en helt central del i AWS miljön. EC2 tillhandhåller operativsystem, lagringsalternativ och några andra alternativ beroende på vilka krav som ställs. EC2 är flexibel när det kommer till skalbarhet, säkerhet och integration med andra tjänster inom AWS. Som tidigare nämnt i Azure i de faktureringsmodellerna som finns är EC2 en Pay-As-You-Go. Du betalar för de instanserna som skapas och skalas upp/ned.

EC2 instansen hjälper företag att bygga och köra applikationer i nätet.

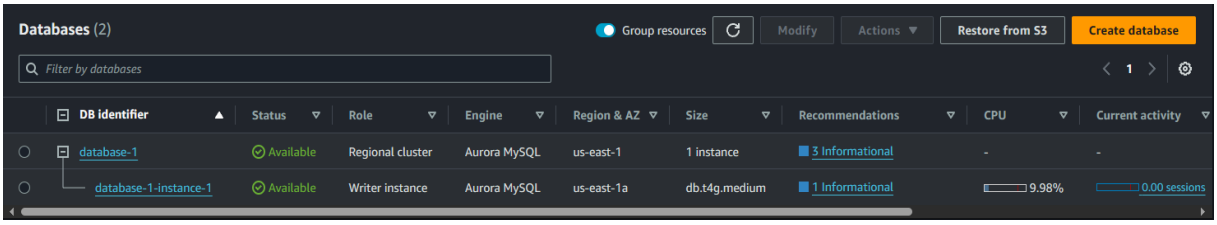


S3 BUCKET: Detta är grundläggande för just AWS. För att på något sätt få tillgång till lagring krävs en S3 bucket där filer, bilder, videos och databaser etc sparas. Du kan ha flera S3 och varje S3 får då en specifik adress som du kan hämta dina data från.

Det som är bra med S3 bucket är att teknologin är flexibel vad det gäller skalbarhet, säkerhet, redundans och mångisidighet. S3 bucket kan användas till webbar, backups och arkivering etc.



Cluster och Databas instans: När en datasbas skapas så skapas automatiskt ett cluster för din miljö. Detta cluster får regional som default och därefter kan vi skapa instanser som kan ligga i olika AZ. Det är här samlingen sker av data på ett strukturerat sät. Här lagrar och hämtar vi information. Databasen är en viktig och avgörande roll i dagens digitala värld. Databasen som skapats i denna miljö kan vi se ligger i Availibility Zon 1a. Om vi väljer att skapa fler instanser kan vi alltså isolera dessa i olika AZ och skapa lägre latens men även bättre säkerhet.



Network Security Group: En anslutning mellan RDS och EC2 har satts upp och en säkerhetsgrupp har skapats för att öka säkerheten. Säkerhetsgrupperna som sätts upp på detta sätt är en viktig del och en praxis att följa då vi minskar risken för oönskad trafik. När vi skapar säkerhetsgrupper på detta vis skapas flexibilitet och det hjälper oss även att hålla en strukturerad och säker miljö då vi kan anpassa åtkomsten.

Connection summary Info

You are setting up a connection between RDS database awseb-e-5vydd35bpe-stack-awsebrdsdatabase-65iuuirvukex and EC2 instance i-0064bc703fac66fb4.

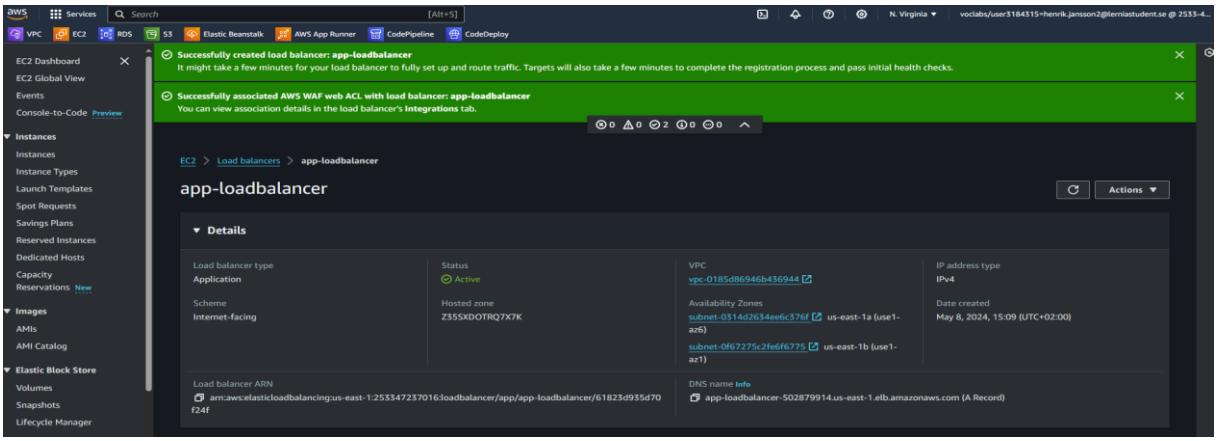
To set up a connection between the database and the EC2 instance, VPC security group *rds-ec2-1* is added to the database, and VPC security group *ec2-rds-1* is added to the EC2 instance.

Bold indicates an addition being made to set up a connection.

Changes to RDS database: awseb-e-5vydd35bpe-stack-awsebrdsdatabase-65iuuirvukex

Attribute	Current value	New value
Security group	rds-awseb-e-5vydd35bpe-stack-awsebrdsdbsecuritygroup-15516m8mrv6ok-vcog	rds-awseb-e-5vydd35bpe-stack-awsebrdsdbsecuritygroup-15516m8mrv6ok-vcog, **rds-ec2-1**

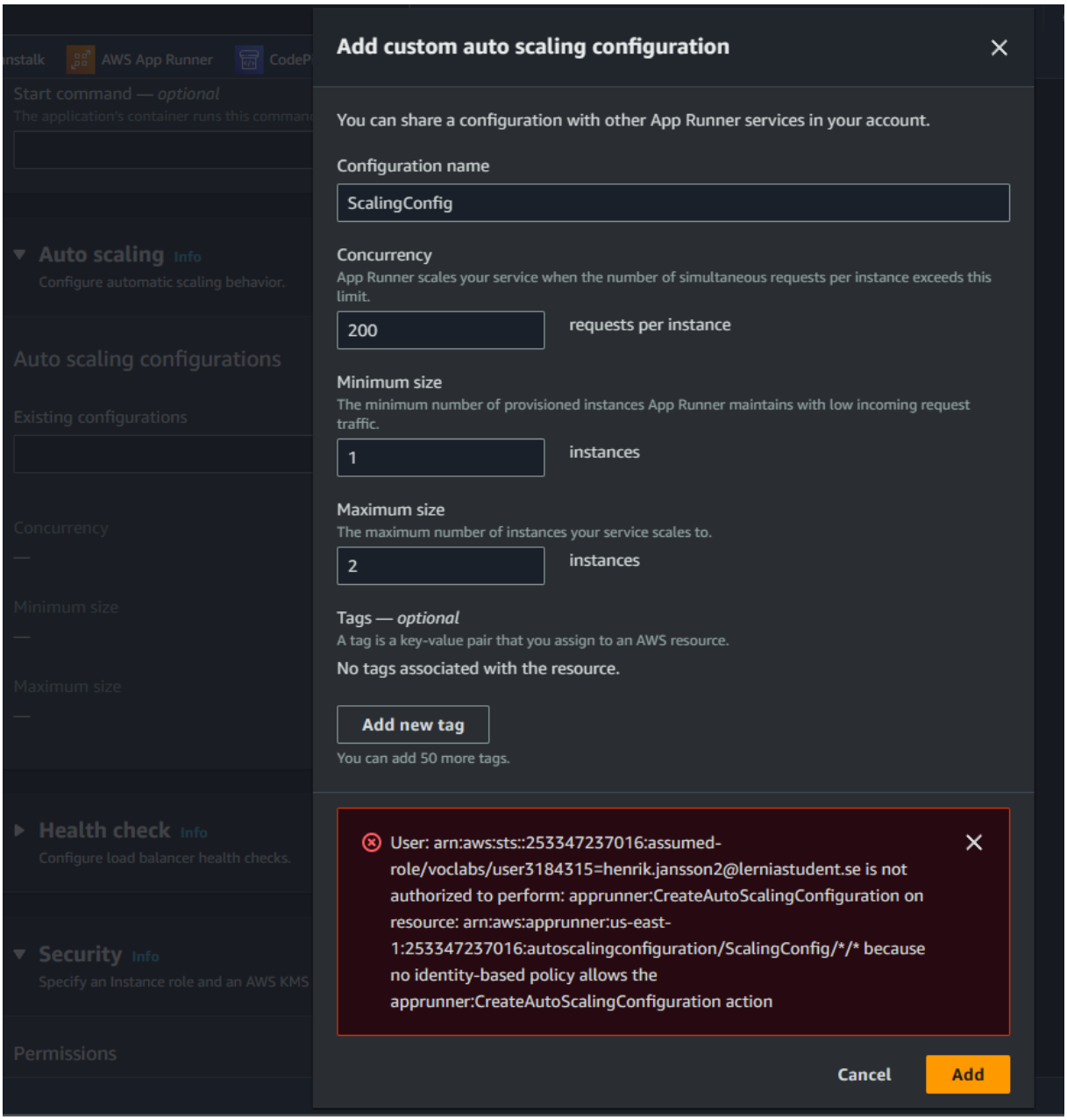
För att inte överbelasta nätverket kan vi använda oss av något som heter lastbalanserare. Det finns några typer av lastbalanserare som Applikations balanserare och nätverksbalanserare. Lastbalanserarna kan sköta delar som skalning, flexibilitet och säkerhet i nätverket.



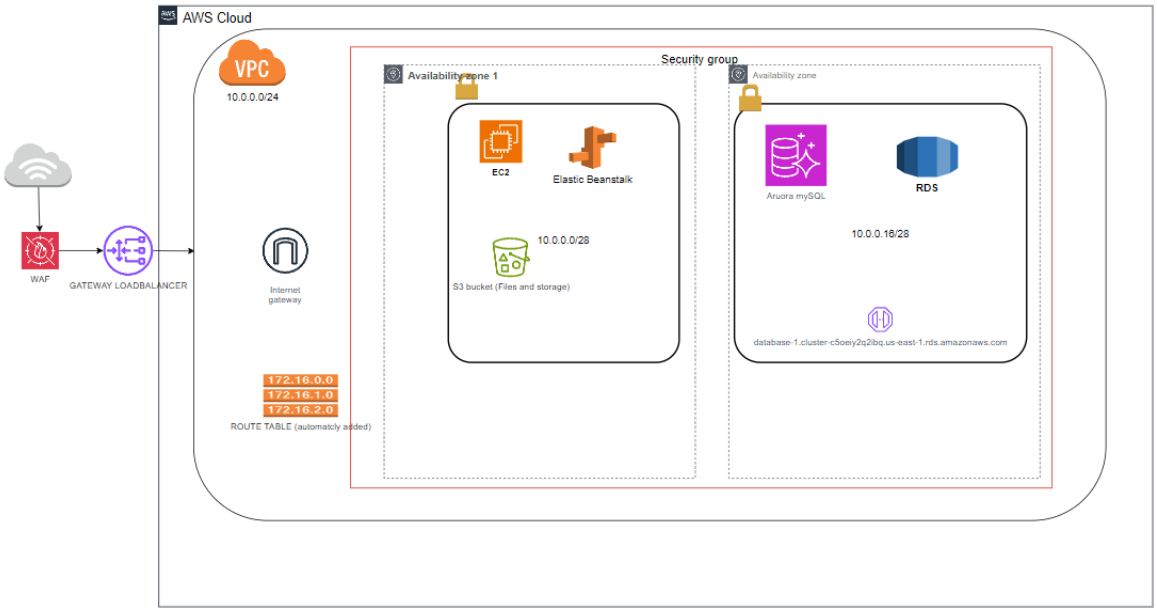
Under projektets gång har jag försökt implementera CI/CD funktioner få min AWS att köra repositories från GitHub. Med detta har otroliga problem upptäckts med alla tjänster. Några av de tjänsterna där försök att implementera är AppRunner, Lambda, ECR, ECS. Då jag inte haft kunskapen om alla dessa teknologier och resurser har detta fått läggas åt sidan.

Rättigheterna och behörigheterna har inte varit fullständiga vilket har påverkat mina resultat. Om fullständiga eller rätt behörigheter för mig så hade miljön fungerat på ett helt annat sätt, dock har kunskapen som jag sa tidigare inte varit 100% heller.

AutoScaling: Denna teknik används som det låter, att automatiskt skala upp/ner eller in/out beroende på vad för resurser vi använder. I en automatisk skalning kan vi hantera saker som EC2 instanser, databaser men även optimering av kostnader.


Add custom auto scaling configuration
You can share a configuration with other App Runner services in your account.
Configuration name
ScalingConfig
Concurrency
App Runner scales your service when the number of simultaneous requests per instance exceeds this limit.
200
requests per instance
Minimum size
The minimum number of provisioned instances App Runner maintains with low incoming request traffic.
1
instances
Maximum size
The maximum number of instances your service scales to.
2
instances
Tags — optional
A tag is a key-value pair that you assign to an AWS resource.
No tags associated with the resource.
Add new tag
You can add 50 more tags.
User: arn:aws:sts::253347237016:assumed-role/voclabs/user3184315=henrik.jansson2@lerniastudent.se is not authorized to perform: apprunner:CreateAutoScalingConfiguration on resource: arn:aws:apprunner:us-east-1:253347237016:autoscalingconfiguration/ScalingConfig/*/* because no identity-based policy allows the apprunner:CreateAutoScalingConfiguration action
Cancel
Add
AWS App Runner
Start command — optional
Auto scaling Info
Configure automatic scaling behavior.
Auto scaling configurations
Existing configurations
Concurrency
Minimum size
Maximum size
Health check Info
Configure load balancer health checks.
Security Info
Specify an Instance role and an AWS KMS
Permissions

AWS Cloud



Diskussion

Att arbeta med kända molnteknologier som Microsoft Azure och AWS är väldigt krävande där du behöver ha goda kunskaper och vara insatt i ämnet. Med de tillgångar vi haft under tiden då vi har försökt sätta upp miljöer har jag märkt att det är väldigt noga att studera behovet och vad som krävs inom företaget och hur vi ska använda tjänsterna hos de olika leverantörerna. Både är unika på sitt sätt då jag vill påstå att Microsoft har ett mer användarvänligt gränssnitt än vad AWS. Båda leverantörerna levererar i slutändan samma produkter men på lite olika sätt där jag föredrar att arbeta mer med Azure än AWS. Beroende på den tiden som lagts ner är detta endast ur min synvinkel, jag är övertygad om jag någon gång i framtiden jobbar med AWS kommer jag förmodligen ändra min åsikt om vilken molntjänstleverantör som är bäst för ändamålet.

Kom ihåg att hålla dokumentationen uppdaterad när Azure-miljön utvecklas, och involvera relevanta intressenter i att granska och validera dokumentationen regelbundet. Detta är något som jag har varit dålig på och som jag tror hjälper utvecklare och alla de som arbetar i dessa miljöer borde ta stor hänsyn till. Nu i efterhand förstår jag vikten om att ha allt dokumenterat för att själv göra det enkelt både för mig och andra att följa flödet.

Slutsats

Enligt mig själv får jag nog säga att jag inte har gjort tillräckligt. Det finns absolut mer att göra i den miljö som har skapas för att jag hade känt mig helt nöjd. Eftersom det har varit enormt mycket information på kort tid så känner jag att jag inte har lärt mig och förstått det som har presenterats i kursmaterialet under tidens gång. Jag vill dock poängtera att jag har lärt mig och fått mycket kunskap om hur molnet fungerar mot vad jag visste innan.

En sak jag definitivt kommer ha i åtanke när jag hör ordet moln är att verkligen vara noggrann och tänka efter en gång extra om jag skulle fördjupa mig mer inom detta område. Att ha tungan rätt i mun och verkligen följa de steg som krävs för en säker och robust infrastruktur är något som jag har en otrolig mycket mer förståelse nu än vad jag tidigare haft.