Azure cloud Architectuur voor Data Science

Binnen PLUGIN en AI Ondersteund Coderen (AIOC) maken we gebruik van een federatieve infrastructuur wat betekent dat de data binnen de muren van het ziekenhuis blijft en wij de modellen en applicaties uitvoeren bij het ziekenhuis zelf. Om dit te kunnen bewerkstelligen gebruiken we twee Azure services:

- **Azure Container Registry** Voor cloud opslag van de DHD docker images met onze code die vanuit het ziekenhuis worden opgehaald en daar lokaal worden uitgevoerd.
- **Azure Blob Storage** Voor cloud opslag van grote bestanden die niet in de containers passen zoals getrainde modellen en referentie tabellen.

Inhoudsopgave

Technische specificaties huidige oplossing

Azure Container Registry

Het data science teams publiceert alle code en applicaties als docker images via de Azure Container registry. Dit is een private registry waarin onze images opgeslagen worden en

gecheckt worden op vulnerabilities. De Docker images kunnen via docker cli en docker compose bij de ziekenhuizen worden opgehaald en uitgevoerd worden. Hiervoor zijn credentials en toegang nodig die we regelen via Azure Service Principals.

Azure Blob Storage

Azure Blob Storage is een service die grote hoeveelheden ongestructureerde data kan opslaan. Het is een veilige en schaalbare opslagplaats voor objecten, zoals afbeeldingen, video's, audio en documenten. De data wordt als blobs opgeslagen in storage containers, die weer in storage accounts zitten. Storage containers zijn vergelijkbaar met mappen en kunnen worden gebruikt om blobs te organiseren. Zowel containers als blobs kunnen worden benaderd via een URL. Doordat er geen eisen gesteld worden aan het data format kunnen wij complexe data zoals getrainde modellen opslaan in efficiënte en goed werkbare formats. Grote bestanden zoals getrainde modellen en referentie tabellen kunnen niet mee in de docker containers door de bestandsgrootte restricties door github bij het bouwen met onze CI/CD pipeline en worden daarom opgeslagen in de Azure Blob Storage. Daarnaast geeft blob storage ons ook de mogelijkheid om bijvoorbeeld centraal logs van inferentie te kunnen opslaan.

Azure Service Principals

Op dit moment gebruiken we Azure Service Principals om de toegang tot de Azure Blob Storage en Azure Container Registry te regelen. Een Service Principal is een identiteit die gebruikt kan worden om resources te authenticeren en autoriseren. We hebben op dit moment twee service principals:

- **Intern gebruik (binnen DHD)**: push en pull rechten op de Azure Container Registry en admin rechten op de Azure Blob Storage
- **Extern gebruik (bij de ziekenhuizen)**: pull rechten op de Azure Container Registry en read/write rechten op de Azure Blob Storage

Gebruik Azure binnen DHD

Team Data Science maakt binnen de muren van DHD gebruik van Azure via de interne service principal om de volgende taken uit te voeren:

Azure Container Registry

- Pushen en pullen van applicatie Docker images de Azure Container Registry
  - automatische built en push van images via github actions CI/CD pipeline
  - handmatig pushen van images voor testen en ontwikkeling via docker CLI
  - ophalen van images via terminal (docker login en docker push/pull) of via vantage6 node
- Container registry beheren (bijv. verwijderen van oude Docker images) via de azure CLI
  - Via de Azure portal
  - Via de Azure CLI

Azure Blob Storage

- Uploaden en downloaden van data via azure python sdk

    - ophalen van getrainde modellen voor inferentie
    - wegschrijven van data evaluatie rapporten en inferentie samenvattingen
    - backend voor data controle applicatie voor medisch codeurs voor het controleren van traindata
- Beheer van de storage containers en blobs in de Azure Blob Storage
    - Via de Azure CLI
    - Via de Azure portal
    - Via azure python sdk

Gebruik Azure buiten DHD

Buiten DHD maken we gebruik van de externe service principal om de volgende taken uit te voeren:

Azure Container Registry

- Docker images ophalen via vantage6 node
- Docker images ophalen vanuit de ziekenhuis server door in te loggen op de ziekenhuis server zelf

Azure Blob Storage

- AIOC inferentie
    - Wegschrijven van ETL data kwaliteit rapport vanuit ziekenhuizen
    - Wegschrijven van de samenvatting van uitgevoerde inferentie (hoeveel % handmatig / hoeveel % automatisch)
    - ophalen van getrainde modellen vanuit de ziekenhuis server
    - ophalen referentie tabellen
- AIOC training
    - Ophalen en wegschrijven werklog van validatie dashboard bij traindata (alleen opname_id, icd10 code en comment van onze medisch codeur, geen patienten gegevens)

Vantage6

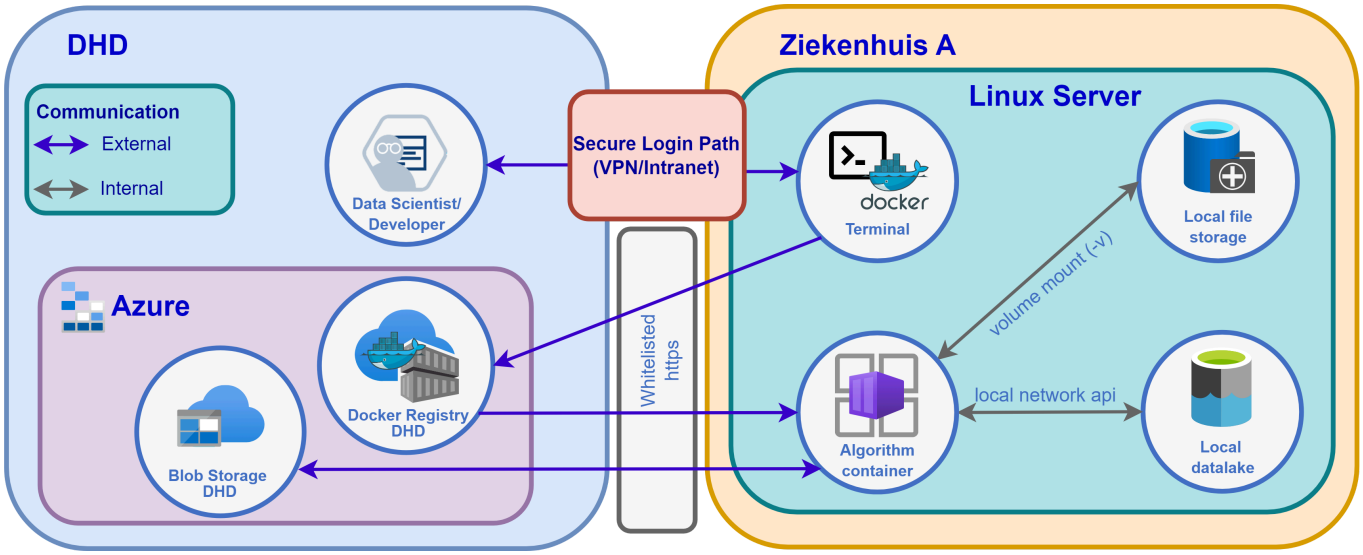
Om algoritmes te kunnen uitvoeren op de ziekenhuis servers zonder te hoeven inloggen op de servers zelf gebruiken wij vantage6. Gebruikmakend van docker images kan er bij de centrale vantage6 server binnen het DHD netwerk opdrachten klaargezet worden voor de ziekenhuizen. De ziekenhuizen hebben een vantage6 node geinstalleerd op hun server die de docker images ophaalt en uitvoert. De resultaten worden teruggestuurd naar de centrale vantage6 server en kunnen via de vantage6 python client opgehaald worden. Op dit moment worden alle resultaten op de centrale vantage6 server opgeslagen in een Azure postgres database als json. Dit zorgt voor problemen bij het opslaan van grote datasets zoals getrainde modellen waarbij we zowel tegen de limitaties van de toegestane grootte van een datapunt in de postgres database als de snelheid van het ophalen van de data aanlopen.

Huidige architectuur

Op dit moment gebruiken we twee verschillende routes om Docker containers te kunnen uitvoeren bij ziekenhuizen. De direct access route is de huidige methode voor inferentie met het huidige v1 model van aioc. De nieuwe infrastructuur voor v2 van aioc voor zowel trainen als inferentie zal gebruik maken van de vantage6 route. De direct access route zal dan alleen nog gebruikt worden voor installatie en onderhoud van de vantage6 nodes bij de ziekenhuizen.

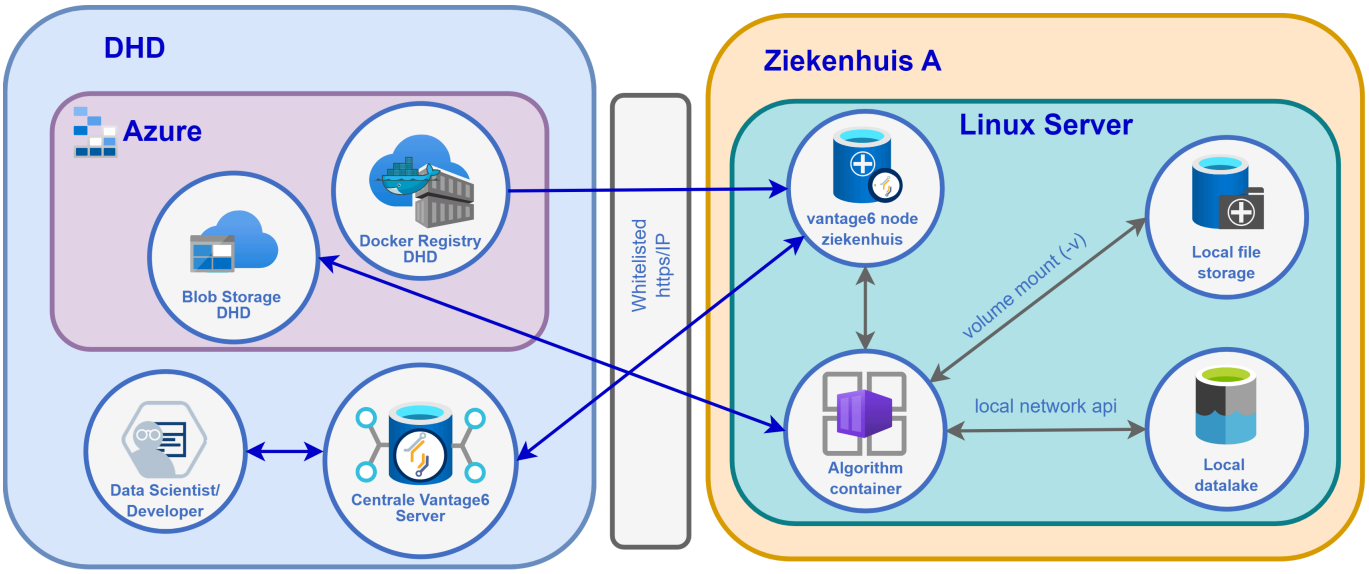
Direct access route

Deze route is in gebruik genomen in de eerste versie van AIOC. De data scientist van DHD logt in bij de ziekenhuis server ofwel via een VPN client of Remote access platforms. deze route wordt er direct op de server ingelogd via een door het ziekenhuis aangeleverde methode. De data scientist kan dan via de terminal de docker images ophalen en uitvoeren via de Docker CLI of docker-compose. Via service principal credentials die lokaal op de ziekenhuis server worden opgeslagen wordt er toegang verkregen tot de Azure Blob Storage en Azure Container Registry.



Vantage6 route

Via deze route kan de data scientist een opdracht klaarzetten voor een of meerdere ziekenhuizen vanuit zijn eigen systeem via bijvoorbeeld de vantage6 python client. De communicatie loopt altijd via de centrale vantage6 server die binnen een netwerk van DHD staat. De vantage6 nodes die bij ziekenhuizen geinstalleerd zijn halen de Algorithm images op vanuit de Azure Container Registry en voeren deze uit op de ziekenhuis server. Ziekenhuizen hebben op hun server een standaard mappenstructuur met de patienten data die via volume mount tijdelijk aan de algorithm container gemount wordt. De resultaten worden lokaal opgeslagen en/of teruggestuurd naar de centrale server en kunnen via de vantage6 python client opgehaald worden. De service principal credentials om met de Azure Container Registry te communiceren worden meegegeven in de vantage6 node configuratie.



Verbeterpunten huidige oplossing

Principal service account per ziekenhuis

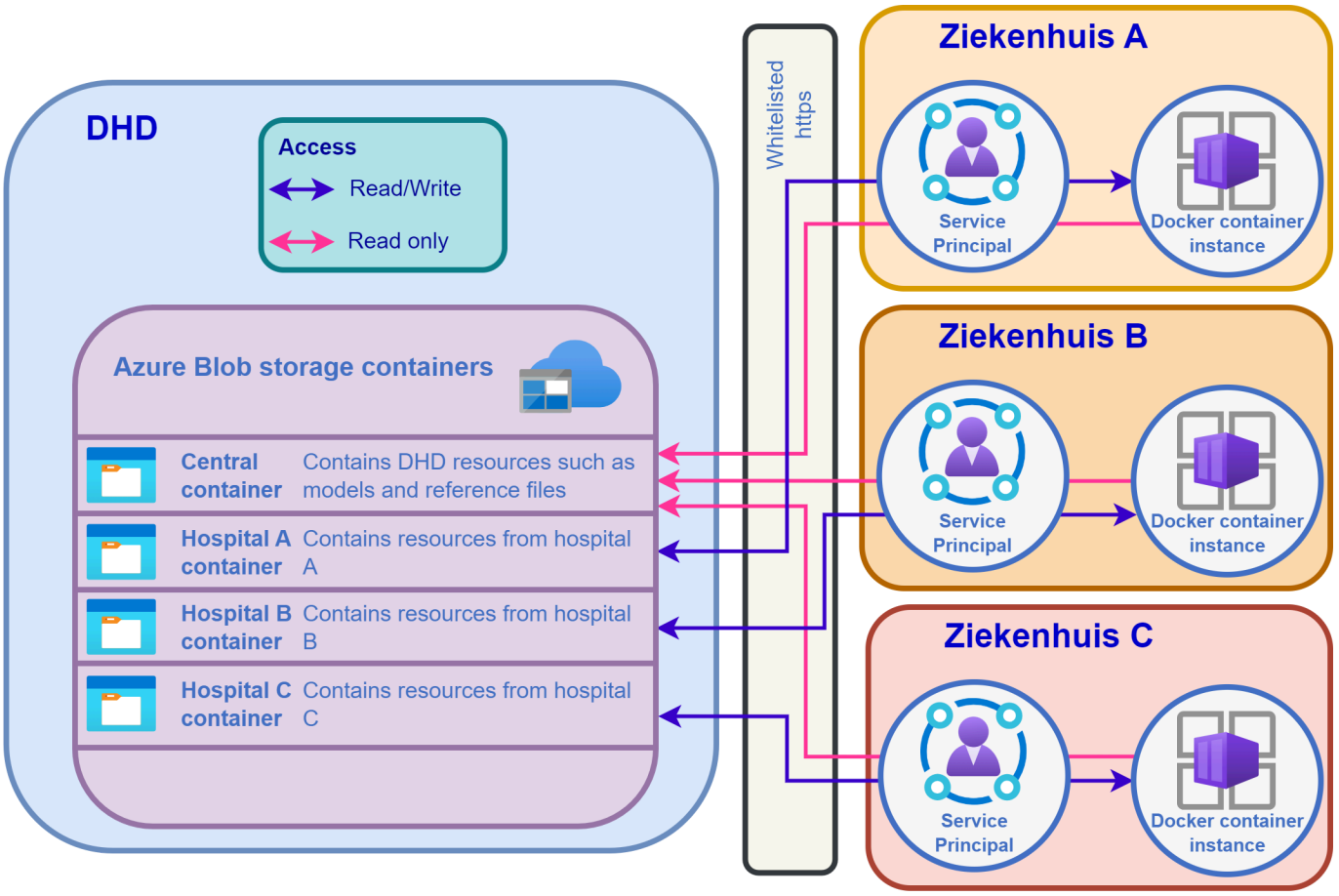
Op dit moment gebruikt elk ziekenhuis hetzelfde externe principal service account. Dit zorgt ervoor dat je ziekenhuizen niet individueel kan afschermen en toegang kan geven tot specifieke storage containers. Omdat we niet willen dat ziekenhuizen elkaars gegevens kunnen inzien wil je het liefst dat elk ziekenhuis zijn eigen container binnen onze blob storage heeft met een eigen service principal. Zo kan je vanuit DHD makkelijk toegangs wijzigingen doorvoeren en is de ziekenhuis data bij design al afgeschermd van andere ziekenhuizen. Daarnaast zorgt dit ook voor betere bescherming waarbij een ziekenhuis makkelijk toegang kan worden ontzegd zonder dat dit effect heeft op de andere ziekenhuizen.

Fine grained toegang tot storage containers

Om de toegang tot de storage container inhoud vanaf buiten af zo minimaal mogelijk te houden willen we graag dat de ziekenhuizen alleen toegang hebben tot de storage containers die ze nodig hebben. Dit betekent dat we per ziekenhuis een storage container moeten aanmaken met alle ziekenhuis specifieke gegevens en op de algemene storage containers een read only toegang te geven zodat ziekenhuis per definitie nooit data kunnen wegschrijven. Dit is nu niet mogelijk omdat we maar 1 service principal hebben voor alle ziekenhuizen.

Principal service accounts gelijk trekken met Vantage6 organizations

Door per organisatie een service principal aan te maken net als een vantage6 organization account kunnen we beheer hiervan makkelijk samen voegen en de toegang tot de Azure services beter afstemmen op de verdere organisatie indeling binnen plugin. Daarnaast kan via collaboration storage containers bijvoorbeeld geaggregeerde modellen beschikbaar gesteld worden voor de gehele collaboratie.



White listing van IP adressen

Om de toegang tot de Azure Blob Storage en Azure Container Registry te beperken willen we graag dat alleen de IP adressen van de ziekenhuizen toegang hebben tot de Azure services. Indien mogelijk kan er gekeken worden of er ip restricties kunnen worden ingesteld aan de hand van de principal service accounts.

Blob storage voor het opslaan van grote vantage6 resultaten

Op dit moment wordt alles wat teruggestuurd wordt vanuit de ziekenhuis vantage6 nodes naar de centrale server opgeslagen in de postgres database als json. Dit zorgt voor problemen bij de ontwikkeling van grote taalmodellen zoals bij AIOC wegens de limitaties van de opslag grootte van een datapunt in de postgres database. Vanuit vantage6 is dit een bekend probleem. Om dit probleem op te lossen kan er voor gekozen worden om blob storage te gebruiken voor de daadwerkelijke opslag van deze grote bestanden en alleen de url of een referentie naar het bestand in de postgres database op te slaan. Hierdoor maak je niet alleen gebruik van het schaalbare en goedkope opslag van Azure Blob Storage maar is het ook mogelijk om complexe data zoals tussentijdse modellen in een ander data format dan json op te slaan. Hierdoor is het mogelijk om bijvoorbeeld gebruik te maken van het efficiënte parquet format voor grote datasets of pytorch eigen formats voor het efficient opslaan van getrainde modellen.

Beheer service principal credentials

Alleen de Docker containers hebben toegang nodig tot de blob storage credentials. Het liefst willen we daarom de Docker containers toegang kunnen geven tot de credentials zonder dat de ziekenhuizen deze credentials kunnen inzien door ze bijvoorbeeld lokaal op hun server te

moeten zetten. Met behulp van services zoals Azure Key Vault kunnen we mogelijk de credentials veilig opslaan en alleen toegang geven aan de Docker containers.

Hoe gaan we dit realiseren?

Om de verbeterpunten te realiseren, moeten we een eenvoudige manier implementeren om per ziekenhuis een service principal aan te maken en deze te koppelen aan een storage container in Azure Blob Storage en Azure Container Registry. Door een kleine module te bouwen die automatisch, op basis van een template, een service principal aanmaakt en koppelt aan de juiste storage container, kunnen we dit proces automatiseren en eenvoudig beheren. Terraform is een veelgebruikte tool waarmee je templates kunt maken om Azure-resources te beheren. Door een Terraform-template te ontwikkelen voor het aanmaken van service principals en het koppelen aan de juiste storage containers, kunnen we dit proces verder automatiseren.

Voordelen van terraform vs. handmatig beheer

- In bulk service principals aanmaken en koppelen aan storage containers
- Terraform werkt via een centrale state file waardoor je makkelijk de huidige status van de infrastructuur kan inzien
- Je kan meerdere cloud acties in 1 keer uitvoeren (denk daarbij aan toegang regelen accounts aanmaken en storage containers koppelen)
- Je kan de infrastructuur als code beheren en versiebeheer toepassen
- Je kan de infrastructuur makkelijk terugzetten, aanpassen of dupliceren
- Terraform is niet cloud specifiek en kan ook gebruikt worden voor andere cloud providers
- Terraform is open source en heeft een grote community waardoor er veel documentatie en hulp beschikbaar is

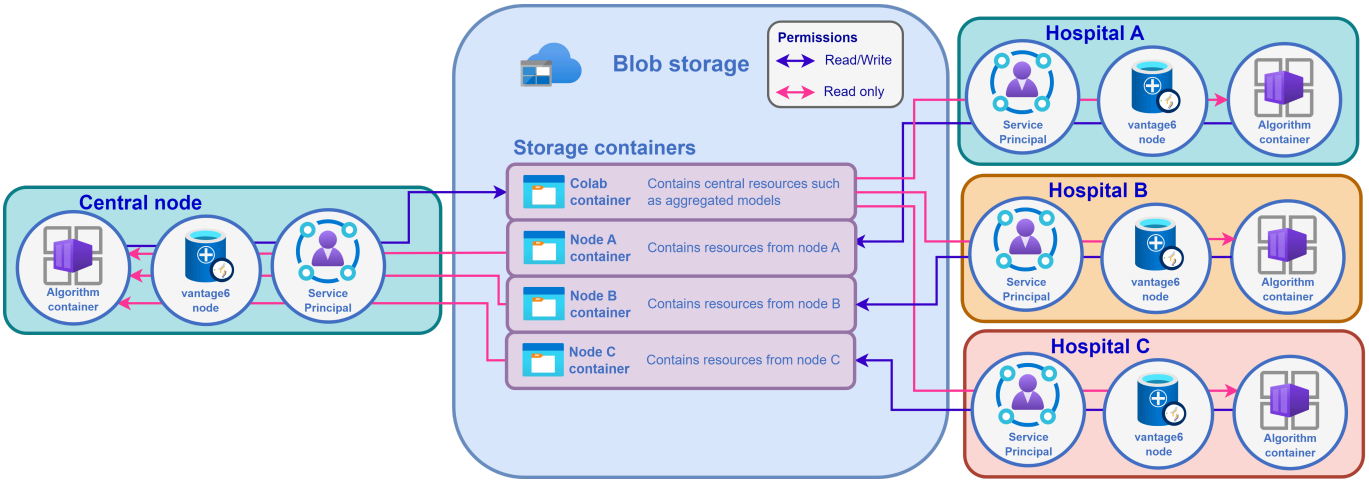
Vantage6 en Azure Blob Storage

Bij implementatie van blob storage als large file exchange voor vantage6 kunnen vanuit de nodes dezelfde service principal accounts gebruikt worden zoals eerder beschreven voor de ziekenhuizen. Vooral voor de server side implementatie zal gekeken moeten worden naar de rol van de server in blob storage container en blob beheer. Belangrijke punten om rekening mee te houden zijn:

- Mag de server zelfstandig blob storage toegang aan de nodes geven?
- Moet de server storage container beheer kunnen uitvoeren voor beheer van collaborations en nodes
- Worden blobs weggeschreven vanuit de server of vanuit de nodes
- Secret distributie en management bij zowel nodes als server voor toegang tot blob storage

Omdat niet iedereen die vantage6 gebruikt gebruik wil maken van blob storage is het belangrijk dat er een algemene implementatie komt die ook gebruik kan maken van andere (cloud) storage oplossingen. Het is daarom waarschijnlijk onwenselijk om vantage6 zelf toegang tot containers te laten regelen of te veel afhankelijk te maken van de blob storage implementatie. De minst ingrijpende oplossing is waarschijnlijk om elke node een vaste container te geven die als config parameter wordt meegegeven en waar de node zelfstandig toegang toe heeft. Bij het uitvoeren van de opdracht bij meerdere instaties wordt er altijd een node aangewezen als de centrale node die verantwoordelijk is voor het uitvoeren van het aggregatie algoritme en het

wegschrijven van de resultaten. Deze node zal in dat geval als enige wel toegang moeten hebben tot de node specifieke containers om de lokale modellen/resultaten op te halen. Het is daarom belangrijk dat dit bijvoorbeeld een vaste node is binnen een collaboratie zodat de toegang tot de storage containers makkelijk te beheren is en beperkt blijft. In het geval van DHD zal er dus een aparte node binnen het netwerk van dhd worden opgezet die als centrale node fungeert zodat geen van de ziekenhuizen toegang heeft tot de storage containers van de andere ziekenhuizen.



Tussentijdse oplossing voor trainen met grote modellen

Omdat we op dit moment tegen de limitaties van de postgres database van de vantage6 server aanlopen bij grote taal modellen is het belangrijk dat we zo snel mogelijk een werkbare situatie creeeren voor het trainen. Omdat de voorgestelde aanpassingen tijd kosten om goed te implementeren zal er een tussentijdse oplossing moeten komen. Gezien we op dit moment met 5 ziekenhuizen trainen is het voorstel om in onze blob storage alvast voor de train ziekenhuizen een storage container op te zetten waar ziekenhuizen alleen hun lokaal getrainde model naar toe kunnen schrijven en een container waar de centrale node het geaggregeerde model naar toe kan schrijven. Samen gevat ziet dit er als volgt uit:

- **Storage containers**
  - **local-models**: write only voor de ziekenhuizen en read only voor de centrale node
  - **aggregated-models**: write only voor de centrale node en read only voor de ziekenhuizen