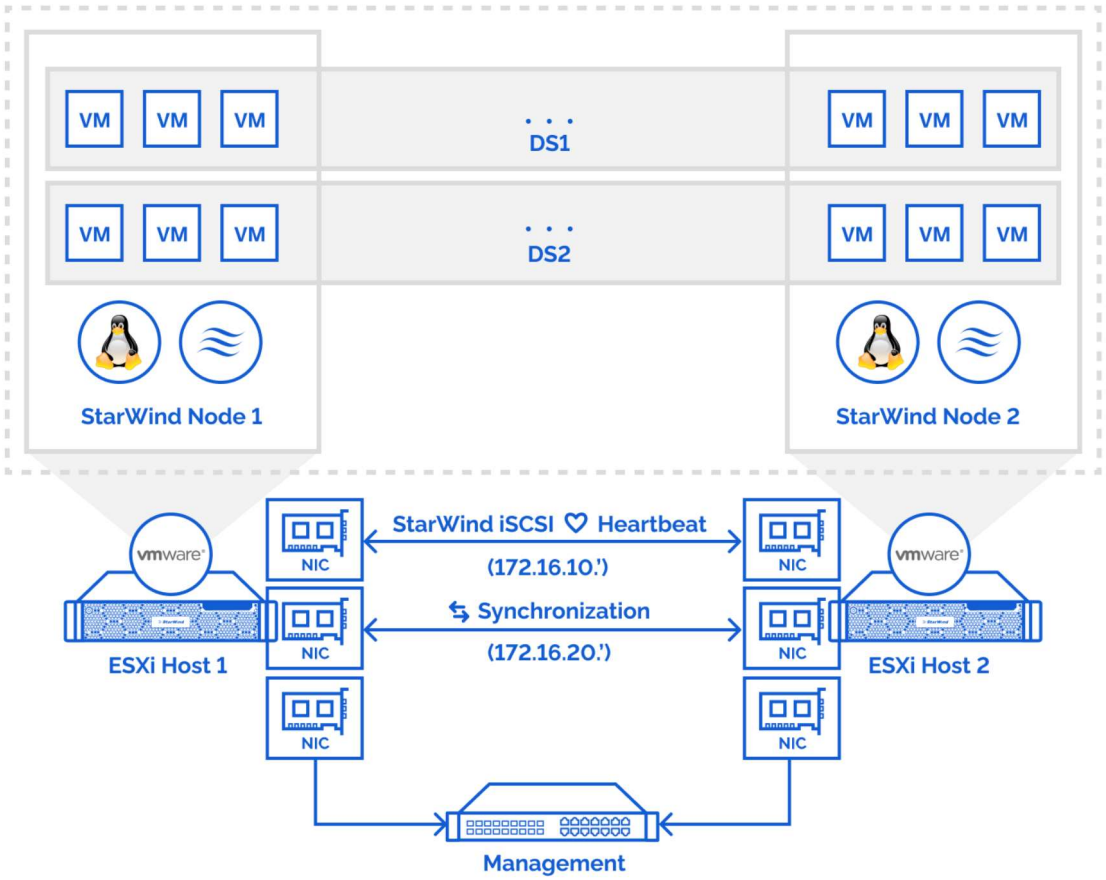
Test Starwind vSAN

Starwind vSAN is een bekende speler op de markt van Hyper Converged Infrastructure. Hun oplossing voor vSAN zou ideaal zijn voor Dataline aangezien er geen dure VMWare licenties nodig zijn. Maar de vraag is natuurlijk hoe goed is Starwind vSAN? Als eens gekeken wordt naar de reviews dan zijn er veel bedrijven tevreden met Starwind vSAN. Om te zijn van ons stuk wordt een test opstelling opgezet, hier wordt de performance, efficiëntie en kwaliteit van Starwind vSAN onder de loep genomen.

Opstelling

Als opstelling worden 2 gebruikte desktop computers die Dataline nog liggen had gebruikt. Die breiden worden dan uitgebreid met een gloed nieuwe snelle netwerkkaart (NIC) met 2 poorten (25 Gbps) en een 350 GB SSD.

Starwind heeft een 30 day free trial waarmee er een 2 node setup kan getest worden. Twee nodes is het absolute minimum dat nodig is om een vSAN te gebruiken. Met een node wordt hier een server bedoeld die de Starwind virtuele machine aan het draaien is. De setup die zal gevolgd worden is de volgende.



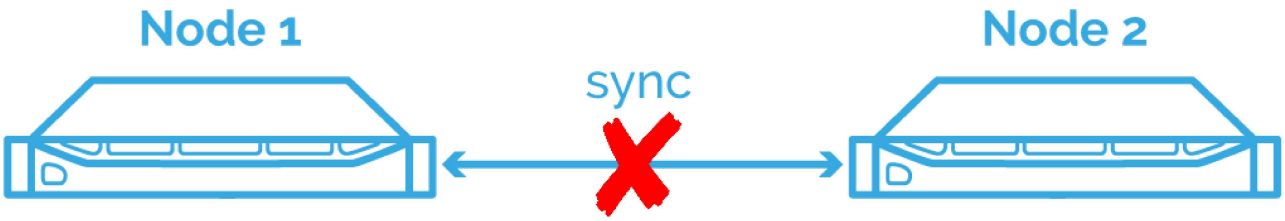
In totaal zullen er 3 verbindingen nodig zijn op de nodes:

- iSCSI/Heartbeat
- Synchronisatie
- Management

Het iSCSI/heartbeat en synchronisatie kanaal wordt gebruikt om data en commando's door te geven tussen de hosts. Deze verbindingen gebeuren best met de nieuwe netwerkkaart zodat de data zo snel mogelijk kan gesynchroniseerd worden. Management is bedoeld als aanspreekpunt voor het beheren van de ESXi host en Virtuele machines.

Failover strategie

Bij het geval dat de verbinding tussen 2 nodes zou weg vallen door een netwerk problemen dan hebben is er een zogenaamde netwerk partitie, dit zorgt dat de nodes niet meer kunnen communiceren met elkaar en kan leiden tot een **split brain** scenario. Split brain kan zeer erge gevolgen hebben zoals het verliezen van data en het moet absoluut vermeden worden.

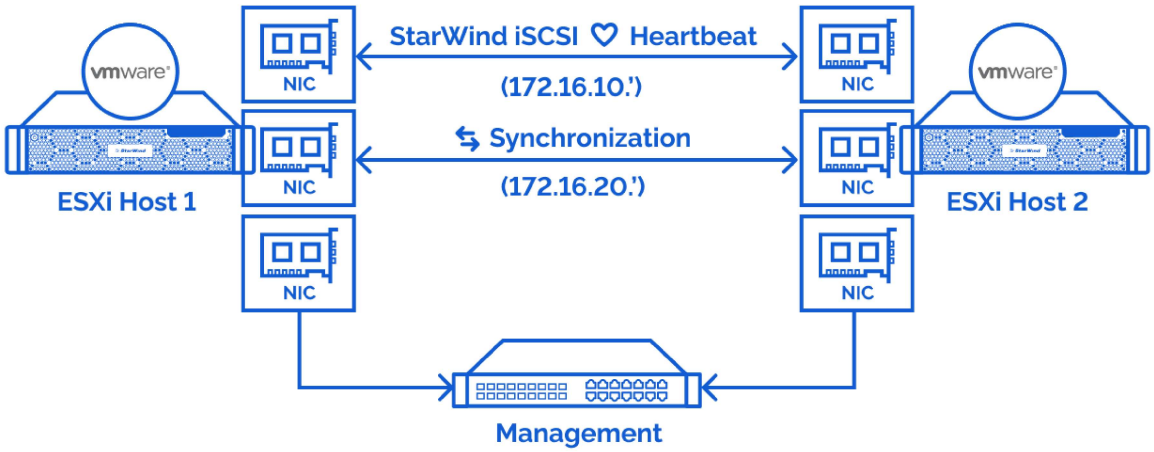


In deze situatie zal elke node naar zijn eigen storage schrijven. Zo ontstaan er data inconsistenties tussen de 2 nodes waardoor ze steeds meer van elkaar gaan verschillen. Uiteindelijk gaan ze zoveel met elkaar verschillen dat de data niet meer gesynchroniseerd kan worden.

Als moet hersteld worden dan zal een node moeten gekozen worden om verder op te werken. Alle wijzigingen van de andere node worden daardoor ongedaan gemaakt. Het risico van een split brain scenario moet daarom zo laag mogelijk zijn. Starwind vSAN geeft ons 2 manieren om zo'n scenario te vermijden.

Heartbeat

Er wordt een zogenaamde heartbeat verbinding opgesteld tussen nodes. Wanneer een node merkt dat de synchronisatie verbinding niet meer werkt tussen een partner node dan zal er een ping gestuurd worden via het heartbeat kanaal. Als de partner node antwoord dan zal Starwind de node blokkeren met een lagere prioriteit tot de synchronisatie verbinding terugkomt. Als de partner node niet antwoord dan gaat Starwind vSAN er vanuit dat die offline is. Starwind vSAN markeert de partner node dan als niet gesynchroniseerd.



Stel bijvoorbeeld dat er 2 nodes zijn zoals in de figuur en de data kan niet verzonden worden via de synchronisatie verbinding. Dan kunnen de 2 nodes niet gaan synchroniseren met elkaar. Via het heartbeat kanaal checkt de primary node de status van de 2 systemen. Het ziet dat beide systemen nog in sync zijn en zal de secundaire node blokkeren zodat het niet meer reageert op requests. De secundaire node check regelmatig de verbinding van het synchronisatie kanaal. Vanaf dat er terug verbinding is, zal het synchronisatie process terug starten en wordt de secundaire node weer actief.

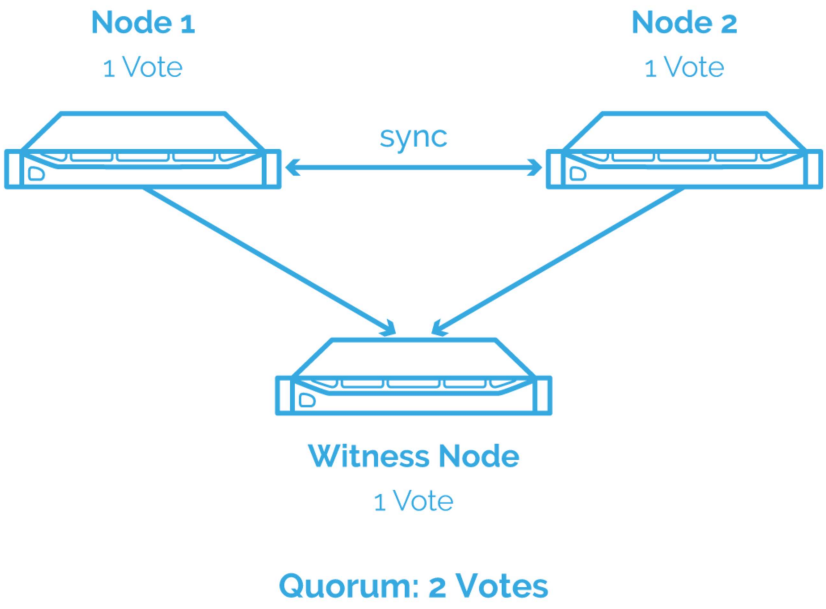
Nadeel:

Indien alle verbindingen (heartbeat + synchronisatie) verbroken worden tussen de 2 hosts dan zal er een split brain situatie ontstaan. Starwind vSAN probeert dit probleem te minimaliseren door toe te laten dat de heartbeat verbinding over het management netwerk gaat. Starwind raad deze aanpak enkel aan wanneer er meerdere heartbeat verbindingen zijn.

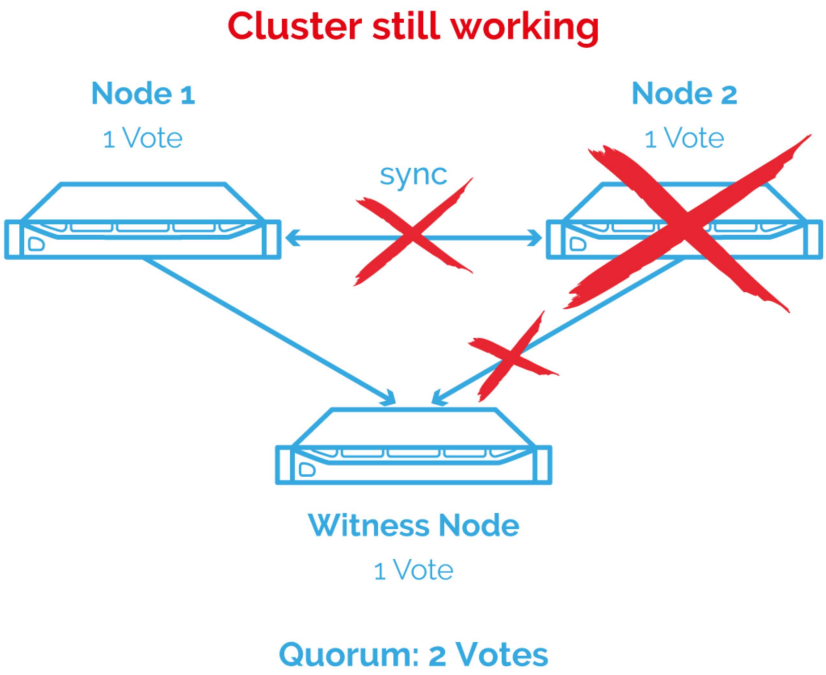
> To summarize, this kind of strategy is mostly applicable to the systems where you have enough network links that can be used as the additional heartbeat channels and are physically separated from the primary ones. **Starwind**

Node majority

Een andere manier om split brain te gaan vermijden is door een **Witness node** te gaan toevoegen. Het is een gekende strategie en wordt ook door VMWare gebruikt. De witness houd informatie bij van beide nodes en wanneer de synchronisatie zou weg vallen dan zal de witness mee beslissen over welke node de primaire node is.



Om te beslissen wie de primaire node wordt zal er een stemming gedaan worden. De node met de meeste stemmen wordt dan de primaire node. Elke node zal voor zichzelf stemmen waardoor en geen meerderheid zal zijn, daarom moet er een derde node om de knoop door te hakken. De witness zelf houd gewoon meta data bij en heeft dus zelf niet veel storage en resources nodig.



Stel bovenstaande configuratie wordt gebruikt en de synchronisatie valt weg. Dan zal de witness node die verbonden is met beide nodes, een meerderheid vormen met de node die de meest relevante data heeft (in dit voorbeeld node 1). Daardoor zal node 2 gemarkeerd worden als niet gesynchroniseerd en zal die niet meer luisteren naar requests. Het grootte voordeel van deze aanpak is dat split brain zich niet meer kan voordoen, maar er is wel een extra node nodig.

Voordelen

- Het split brain scenario is volledig uitgesloten
- Een extra heartbeat verbinding is niet nodig

Nadelen

- Bij 2 node setup is een derde witness node nodig
- met 3 nodes mag er maar 1 failure voorkomen

Synchronisatie test

Om te kijken of de synchronisatie tussen de nodes gebeurt moeten er eens gecontroleerd worden als de er degelijk gerepliceerd wordt over de nodes. Met een simpele test wordt dit gecontroleerd.

1. Maak een bestand aan op één node.
2. Sluiten de vm af
3. Start dezelfde VM op de andere node
4. Kijk of dit bestand ook op de andere node zich bevindt

Dit lukt zonder problemen en alles werkt zoals verwacht.

Fio

De manier waarop een workload zal gesimuleerd worden is door gebruik te maken van een Disk IO test tool genaamd **fio**. Het is een command line tool die zeer veel parameters heeft om te gaan testen. De gebruikte parameters zijn gebaseerd op een artikel van Oracle over het testen van Block storage **Oracle**.

Met fio worden IO operaties uitgevoerd om statistieken te verzamelen zoals:

- Het aantal IOPS (Input/Output operaties per seconde)
- De bandbreedte van de data stroom (in MB/s of GB/s)
- Hoe snel de storage antwoord op IO requests (latency)

Het aantal IOPS is een zeer belangrijke statistiek. Databases zullen zeer veel kleine reads en writes gaan uitvoeren op een storage device. Daarom moet een database server best zo hoog mogelijk aantal IO operaties kunnen uitvoeren om efficient vele requests te kunnen vervolledigen.

Om de verschillende scenario's van de storage te controleren gaan er 4 soorten testen uitgevoerd worden:

- Random reads
- File random reads/writes
- Random read/writes
- Sequentiële reads

Alle testen worden rechtstreeks op de storage uitgevoerd behalve de file random reads/writes. Deze test zal IO operaties doen op een file. Het verschil met rechtstreeks werken is dat hier het besturingssysteem nog tussen komt. Er wordt dus verwacht dat deze test iets trager zullen zijn.

Performance testen

Storage is de traagste factor van elke computer, daarom is de performance van storage zeer belangrijk voor virtuele machines. Om te kijken of Starwind vSAN een goede optie zou zijn, moeten er gecontroleerd worden hoe efficient Starwind omgaat met storage. Een heleboel factoren hebben invloed hebben op de performance van vSAN, dus het is belangrijk om verschillende opstellingen te testen en te kijken wat het beste optie is

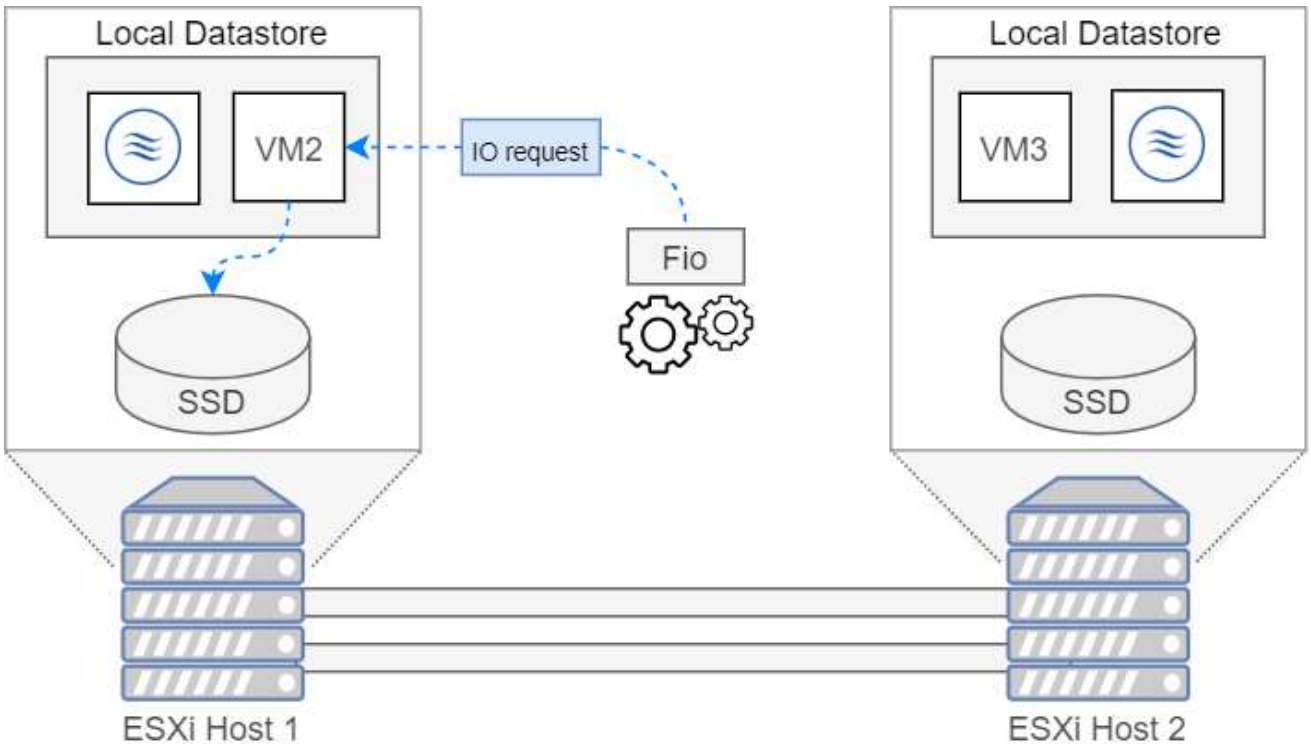
Er zijn 3 scenario's die interessant zijn om te testen voor performance:

- VM op lokale datastore
- VM op vSAN datastore
- VM moet werken via iSCSI omdat lokale host niet gesynchroniseerd is.

De laatste test is belangrijk want deze stelt het scenario voor dat een lokaal storage device niet meer beschikbaar zou zijn. De virtuele machine zal dan over het netwerk IO operaties doen met behulp van iSCSI. Op deze manier zal de virtuele machine nooit down komen te staan.

Lokale Datastore

Er wordt een vm op de lokale datastore opgestart. Met deze opstelling zal de bandbreedte van IO operaties gelijk moeten zijn aan de snelheid van een SATA SSD wat ongeveer 350MB/s zou moeten zijn. In onderstaande figuur wordt het pad dat een IO request volgt eens gevisualiseerd.



Resultaten

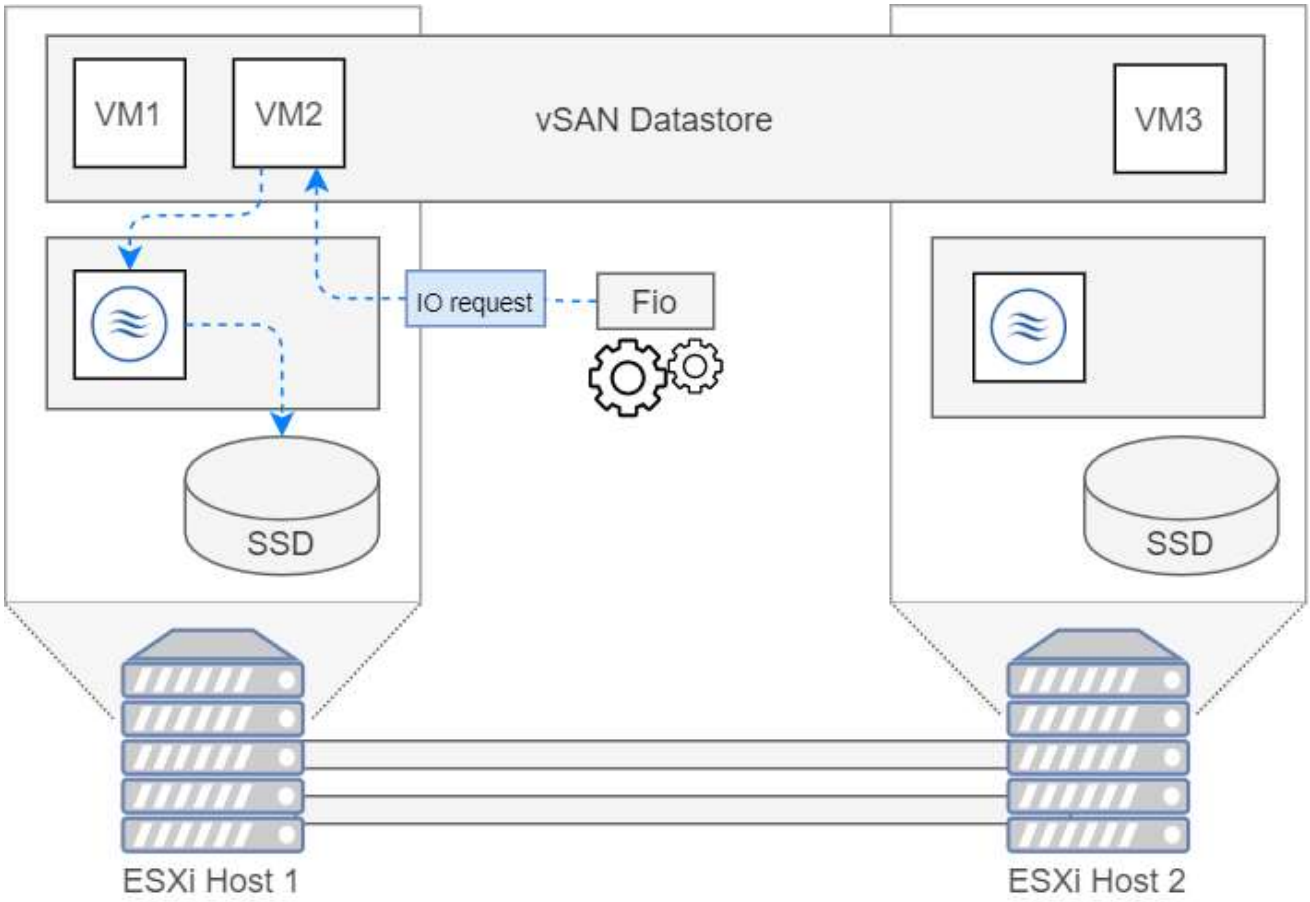
IOPS performance test	IOPS (reads)	IOPS (writes)
Random Reads	285k	/
Random reads/writes	70,9k	70,9k
File random reads/writes	69,4k	69,3k
Sequential reads	292k	/

Throughput Performance Tests	BW (Read)	BW (Write)
Random Reads	2468 MB/s	/
Random reads/writes	347 MB/s	348 MB/s
File random reads/writes	319 MB/s	319 MB/s
Sequential reads	2698 MB/s	/

Latency Performance Tests	Tijd (µs)	
Random Reads	45,6 µs	/
Random reads/writes	109,69 µs	51,20 µs

vSAN Datastore

Er wordt een vm op de vSAN datastore opgestart. Met deze opstelling zal nog steeds gebruikt gemaakt worden van de lokale SSD maar dan via de vSAN datastore. Dit zou enige overhead moeten creëren waardoor er wat lagere resultaten zullen behaalt worden. In onderstaande figuur wordt het pad dat een IO request volgt eens gevisualiseerd.



Resultaten

IOPS performance test	IOPS (reads)	IOPS (writes)
Random Reads	202k	/
Random reads/writes	26,3k	26,3k
File random reads/writes	25,7k	25,7k
Sequential reads	251k	/

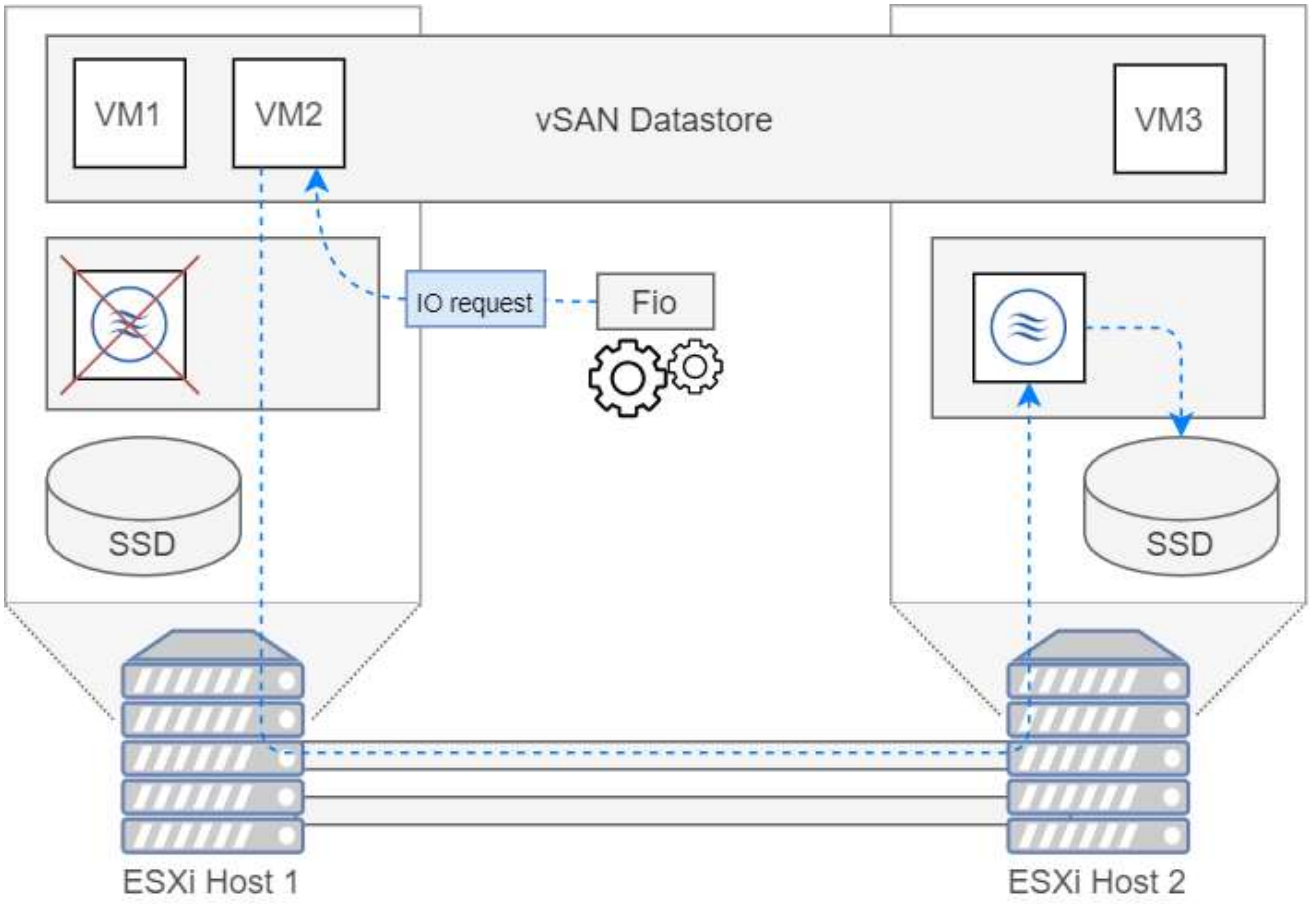
Throughput Performance Tests	BW (Read)	BW (Write)
Random Reads	2622 MB/s	/
Random reads/writes	289 MB/s	290 MB/s

Throughput Performance Tests	BW (Read)	BW (Write)
File random reads/writes	271 MB/s	300 MB/s
Sequential reads	2210 MB/s	/

Latency Performance Tests	Tijd (Read)	Tijd (Write)
Random Reads	109,12 µs	/
Random reads/writes	266,03 µs	406,39 µs

Via iSCSI verbinding

Deze keer wordt de Starwind vSAN node afgesloten op de host. Dit zorgt dat de vm geen toegang meer zal hebben tot de lokale SSD. De vm moet nu via de vSAN datastore IO operaties gaan uitvoeren op de andere Starwind node. Het zal dit doet doen aan de hand van iSCSI commando's. De iSCSI verbinding gaat over de nieuwe snelle netwerk kaart (25 Gbps), hierdoor zou er niet zo'n groot verschil mogen zijn ten opzichte van de vSAN datastore.



Resultaten

IOPS performance test	IOPS (reads)	IOPS (writes)
Random Reads	222k	/
Random reads/writes	38,8k	38,8k
File random reads/writes	35,2k	35,2k
Sequential reads	265k	/

Throughput Performance Tests	BW (Read)	BW (Write)
Random Reads	3226 MB/s	/
Random reads/writes	276 MB/s	277 MB/s
File random reads/writes	329 MB/s	330 MB/s
Sequential reads	2662 MB/s	/

Latency Performance Tests	Tijd (µs)	
Random Reads	110,83 µs	/
Random reads/writes	267,58 µs	212,2 µs

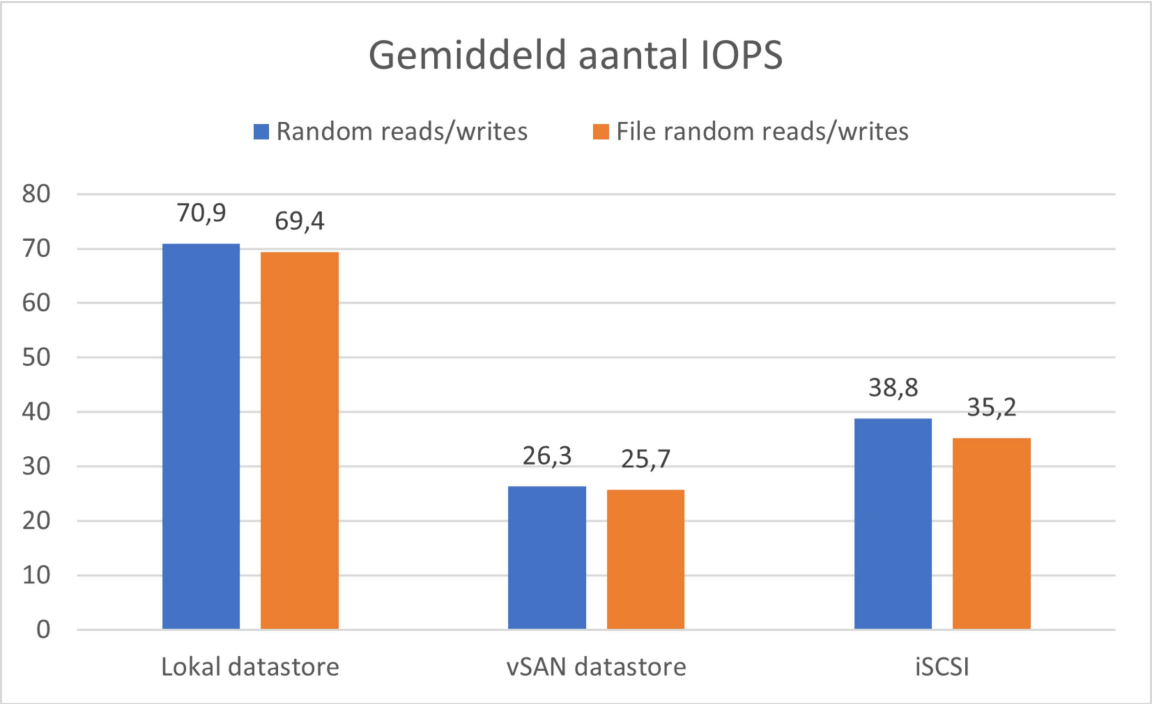
Conclusie

Om uit deze testen een conclusie te halen worden de resultaten eens op een rij gezet.

> **Opmerking**
>
> In de testen hierboven zijn de resultaten van de **random reads** en **sequential reads**enorm groot. Zo groot zelf dat ze sneller zijn dan de SSD. Dit is mogelijks te wijten aan caching van de storage. Deze resultaten zullen om die reden niet meegerekend worden in de conclusie.

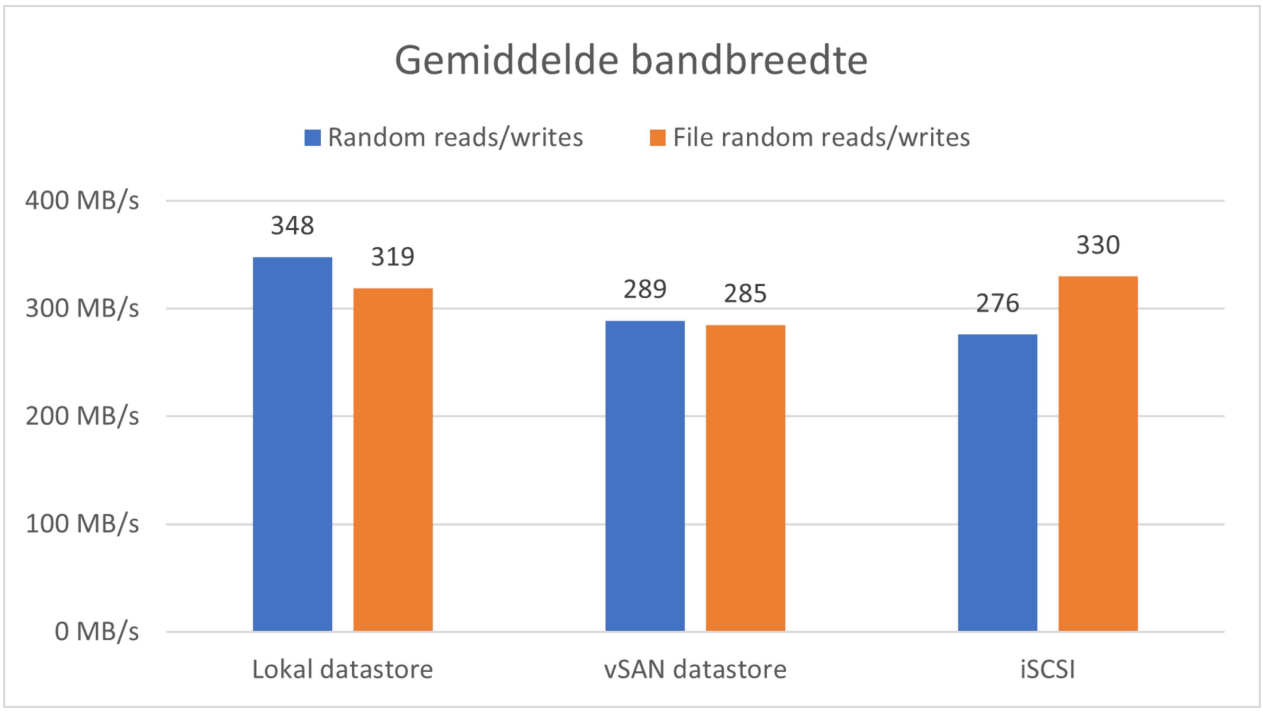
IOPS

In de onderstaande grafiek wordt gekeken naar het gemiddelde aantal IOPS voor elk scenario. Hier kan er gezien worden dat het aantal IOPS duidelijk hoger zal zijn wanneer een lokale datastore wordt gebruikt. Dit is logisch aangezien dat met een lokale datastore de IO requests sneller de SSD zullen bereiken.



Bandbreedte

Hier wordt gekeken naar de gemiddelde bandbreedte voor elk scenario. Het valt op dat er weinig verschil is tussen de verschillende methoden. Dit is te wijten aan het feit dat voor de bandbreedte test grotere blokken data worden opgevraagd. Dit zorgt dat er minder IO operaties moeten gebeuren waardoor het verschil tussen de 3 minder merkbaar is.



Latency

In de onderstaande grafiek wordt gekeken naar de tijd dat de SSD erover doet om een IO request te voldoen. Er is een duidelijk verschil namelijk dat requests op de lokale datastore sneller worden beantwoord. Net zoals bij de IOPS zullen requests sneller de SSD bereiken met een lokale datastore.