Msc.1 – Année 2016-2017

Rapport technique Infrastructure

Qwirk++

SUPINFO International University

Groupe
FRANÇOIS A., TOM B., ANTHONY R.

Sommaire

Introduction

L'objectif de l'application Qwirk++ (aussi abrégé « Qwirk ») est de pouvoir communiquer entre une ou plusieurs personnes sur un plan personnel ou professionnel.

L'application s'appuie sur l'utilisation d'une plateforme 100% open source : Rocket Chat. Cette application est une source personnalisable qui permet d'adapter une solution générique à des besoins spécifiques.

Notre application se démarque des autres concurrents (Slack, Skype…) par son innovation, sa rapidité et son ergonomie.

L'application a été entièrement développée en anglais.

Elle est également « cross-plateform », c'est-à-dire qu'elle fonctionne sous différents systèmes d'exploitation : Microsoft Windows, Mac OSX ou encore Linux.

De nos jours, il est nécessaire d'avoir une infrastructure solide et stable pour avoir une qualité à la hauteur. Celle-ci doit être bien définie dès le départ, ce qui permet d'assurer une continuité des services.

A travers ce rapport, nous commencerons par vous présenter l'architecture de manière générale. Puis, nous évoquerons les outils techniques utilisés et enfin, la mise en place de cette infrastructure en vous montrant la démarche que nous avons suivie.

Précisons qu'en annexes, vous trouverez un glossaire et un lexique expliquant les termes les plus techniques (identifiés par une « * »).

1. Présentation de l'infrastructure

L'architecture réseau se compose de :

- 8 serveurs physiques :
  - 1 serveur contrôleur de domaine*
  - 4 serveurs avec le rôle Hyper-V gérant deux clusters* distincts
  - 3 serveurs faisant office de baie de SAN* iSCSI*
- 1 serveur virtualisé dans chacun des deux clusters :
  - 1 serveur web et base de données
- 1 pare-feu/routeur

Cette architecture se base sur 3 réseaux différents :

- 1 réseau LAN* gérant l'administration
- 1 réseau LAN destiné à la connexion à la baie de stockage SAN
- 1 réseau WAN* pour la connexion à Internet

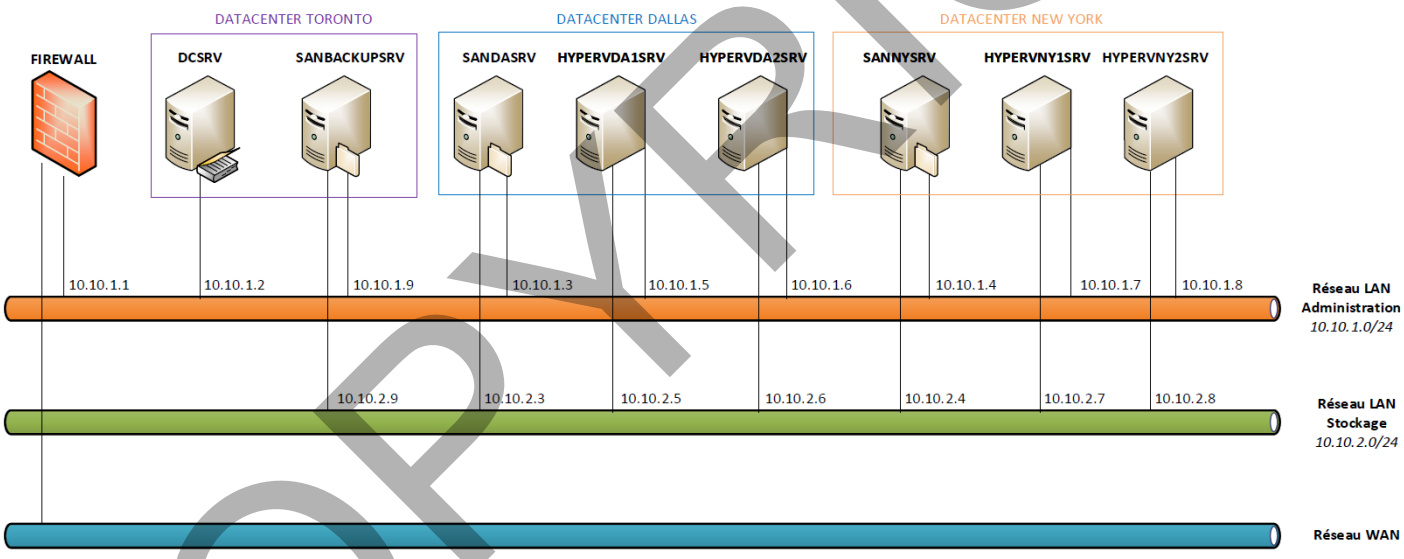


Figure 1. Schéma de l'infrastructure

Ci-dessous un récapitulatif des différentes machines avec leurs adressages IP* :

Serveur	Descriptif	Adresse IP	Passerelle	DNS
FIREWALL	Pare-feu pfSense	10.10.1.1 X.X.X.X	X X.X.X.X	10.10.1.2 X.X.X.X
DCSRV	Contrôleur de domaine, DNS, Active Directory, Superviseur des 2 clusters Hyper-V	10.10.1.2	10.10.1.1	127.0.0.1
SANNYSRV	Baie de stockage SAN iSCSI pour le datacenter de New York	10.10.1.4 10.10.2.4	10.10.1.1 X	10.10.1.2 10.10.1.2
SANDASRV	Baie de stockage SAN iSCSI pour le datacenter de Dallas	10.10.1.3 10.10.2.3	10.10.1.1 X	10.10.1.2 10.10.1.2
SANBACKUPSRV	Baie de stockage SAN iSCSI backup au datacenter de Toronto	10.10.1.9 10.10.2.9	10.10.1.1 X	10.10.1.2 10.10.1.2
HYPERVDA1SRV	Premier nœud de l'hyperviseur dans le datacenter de Dallas	10.10.1.5 10.10.2.5	10.10.1.1 X	10.10.1.2 10.10.1.2
HYPERVDA2SRV	Deuxième nœud de l'hyperviseur dans le datacenter de Dallas	10.10.1.6 10.10.2.6	10.10.1.1 X	10.10.1.2 10.10.1.2
HYPERVNY1SRV	Premier nœud de l'hyperviseur dans le datacenter de New York	10.10.1.7 10.10.2.7	10.10.1.1 X	10.10.1.2 10.10.1.2
HYPERVNY2SRV	Deuxième nœud de l'hyperviseur dans le datacenter de New York	10.10.1.8 10.10.2.8	10.10.1.1 X	10.10.1.2 10.10.1.2
WEBDBDASRV	Serveur web + base de données du datacenter de Dallas	10.10.1.11 10.10.2.11	10.10.1.1 X	10.10.1.2 X
WEBDBNYSRV	Serveur web + base de données du datacenter de New York	10.10.1.12 10.10.2.12	10.10.1.1 X	10.10.1.2 X

Figure 2. Tableau récapitulatif de l'infrastructure réseau (en vert : les serveurs virtualisés ; en gris : les serveurs physiques)

2. Présentation des outils utilisés

Durant notre projet, nous avons dû utiliser les outils suivants :

- **Microsoft Windows Server 2016 Standard :** système d'exploitation Microsoft. Principale édition offrant toutes les fonctionnalités du produit (serveur IIS, Active Directory*, Hyper-V, serveur de fichier…). Cependant, le nombre de machines virtuelles* couvertes par la licence est limité à deux.
    - ➔ Utilisé par DCSRV, SANNYSRV, SANDASRV, SANBACKUPSRV, HYPERVDA1SRV, HYPERVDA2SRV, HYPERVNY1SRV, HYPERVNY2SRV

- **Ubuntu 16.04 LTS** : système d'exploitation open source de type Unix développé par la société *Canonical* sur la base de la distribution Linux Debian.
    - ➔ Utilisé par WEBDBDASRV et WEBDBNYSRV

- **PfSense 2.3 :** Routeur/pare-feu open source basé sur le système d'exploitation FreeBSD. Il utilise le pare-feu à états Packet Filter et permet de créer des fonctions de routage et de NAT* lui permettant de connecter plusieurs réseaux informatiques. Il se comporte comme l'équivalent libre des outils et services utilisés sur des routeurs professionnels propriétaires. L'administration est simple et se configure soit en ligne de commande ou via une interface graphique web.
    - ➔ Utilisé par FIREWALL

3. Mise en place de l'infrastructure

3.1. Le contrôleur de domaine

Le serveur « DCSRV » est le contrôleur de domaine (DC) de « qwirl.lan ». Sur celui-ci, nous avons installé le rôle « Active Directory Domain Services » qui va nous créer notre Annuaire Active Directory. C'est sur ce serveur que nous allons pouvoir créer des objets comme des utilisateurs, des groupes, des GPO*…

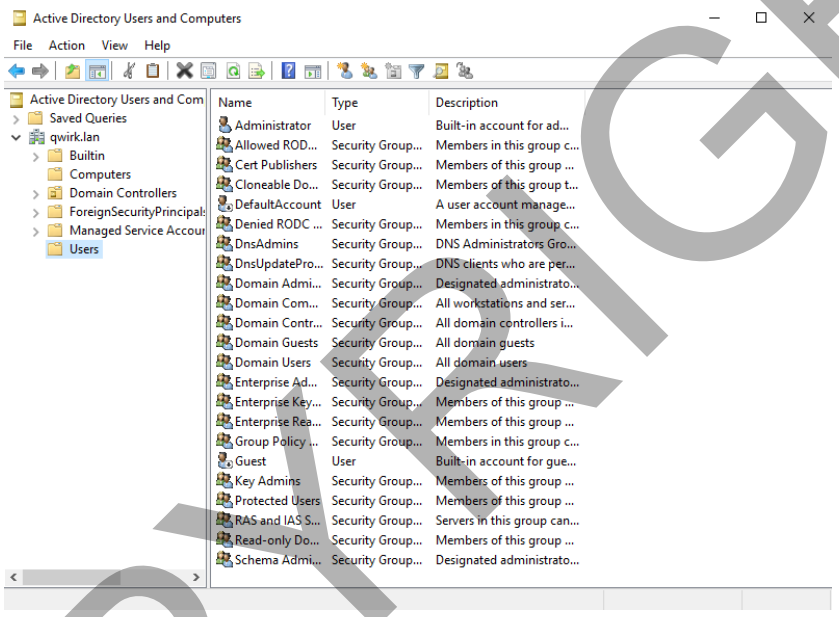


Figure 3. Interface Active Directory Users and Computers

Egalement, il fait office de serveur DNS*. Celui-ci contient 2 zones :

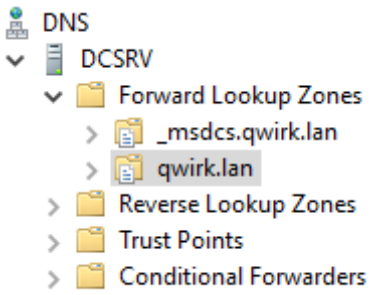


Figure 4. Zones présentes dans le serveur DNS

La zone primaire « qwirk.lan » contient tous les pointages d'un nom d'une VM (dcsrv) vers son adresse IP locale (10.10.1.2).

Name	Type	Data	Timestamp
_msdcs			
_sites			
_tcp			
_udp			
DomainDnsZones			
ForestDnsZones			
(same as parent folder)	Start of Authority (SOA)	[71], dcsrv.qwirk.lan., host...	static
(same as parent folder)	Name Server (NS)	dcsrv.qwirk.lan.	static
(same as parent folder)	Host (A)	10.10.1.2	6/4/2017 8:00:00 PM
DACLUSTER	Host (A)	10.10.1.9	6/4/2017 9:00:00 PM
dcsrv	Host (A)	10.10.1.2	static
firewall	Host (A)	10.10.1.1	static
HYPERVDA1SRV	Host (A)	10.10.1.5	6/4/2017 9:00:00 PM
HYPERVDA2SRV	Host (A)	10.10.1.6	6/4/2017 9:00:00 PM
HYPERVNY1SRV	Host (A)	10.10.1.7	6/4/2017 9:00:00 PM
HYPERVNY2SRV	Host (A)	10.10.1.8	6/4/2017 9:00:00 PM
NYCLUSTER	Host (A)	10.10.1.10	6/4/2017 9:00:00 PM
SANBACKUPSRV	Host (A)	10.10.1.9	6/4/2017 9:00:00 PM
SANBACKUPSRV	Host (A)	10.10.2.9	6/4/2017 9:00:00 PM
SANDASRV	Host (A)	10.10.1.3	6/4/2017 9:00:00 PM
SANNYSRV	Host (A)	10.10.1.4	6/4/2017 9:00:00 PM
WEBDBDASRV	Host (A)	10.10.1.11	static
WEBDBNYSRV	Host (A)	10.10.1.12	static

Figure 5. Zone « qwirk.lan » du serveur DNS

Pour rappel, il dispose aussi de l'interface permettant de gérer le cluster à basculement entre les deux hyperviseurs (cf. §4.2.1).

A noter, qu'il est important qu'il reste toujours allumé car sans lui, les autres serveurs ne pourront pas communiquer entre eux, ni s'authentifier depuis la page de login. A ce titre, dans le cadre du POC* nous n'avons mis en place qu'un seul contrôleur de domaine pour l'ensemble de nos datacenters. En cas pratique, il faudrait un DC (répliquant les données en lecture seule du DCSRV) sur les datacenters de Dallas et de New York.

4.1. Gestion du stockage

4.1.1. Une baie de stockage SAN iSCSI présent sur chaque datacenter

Lors de la mise en place d'une infrastructure, il est important de séparer les données d'un côté et les « services » de l'autre. C'est pourquoi, sur chaque datacenter (Dallas et New York), nous avons créé une baie de stockage en SAN iSCSI grâce au rôle « Microsoft iSCSI Software Initiator Windows ». Sur celui-ci, nous avons installé en plus du disque dur où est stocké le système d'exploitation (OS*), deux autres permettant d'assurer la redondance des données. Ces deux disques durs forment un « pool » de stockage. C'est sur ce dernier, que l'on a créé un disque dur dit « virtuel » ayant la même fonctionnalité qu'un RAID1* (copie simultanément des données sur les deux disques durs). En cas de dysfonctionnement d'un des disques durs, le second permettra la restauration des données sur le nouveau.

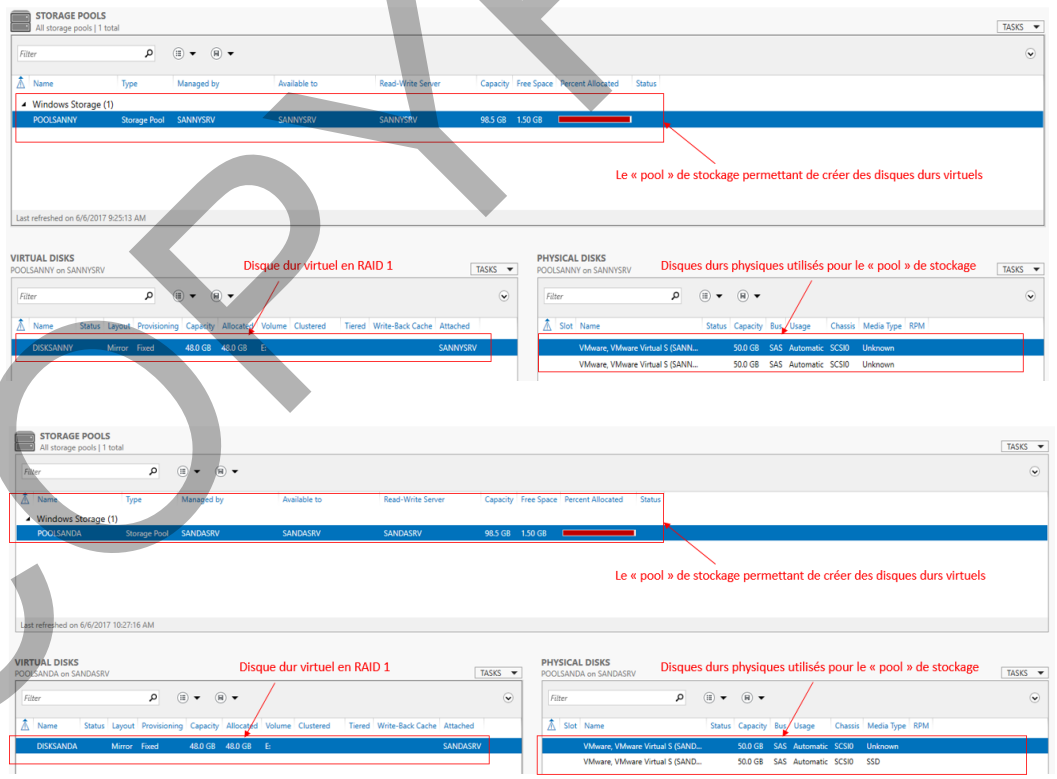


Figure 6. Présentation du serveur « SANNYSRV » (en haut) et du serveur « SANDASRV » (en bas)

Nous avons créé sur chaque baie de SAN (« SANNYSRV » et « SANDASRV) des disques virtuels iSCSI :

- « NY_VM » et « DA_VM » : permet de stocker la machine virtuelle (« WEBDBNYSRV » et « WEBDBDASRV » respectivement.
- « NY_QUORUM » et « DA_QUORUM » : joue le rôle de témoin pour le cluster « NYCLUSTER » et « DACLUSTER » respectivement.
- « NY_DATA » et « DA_DATA » : héberge les données de l'application Qwirk++ (web et base de données).

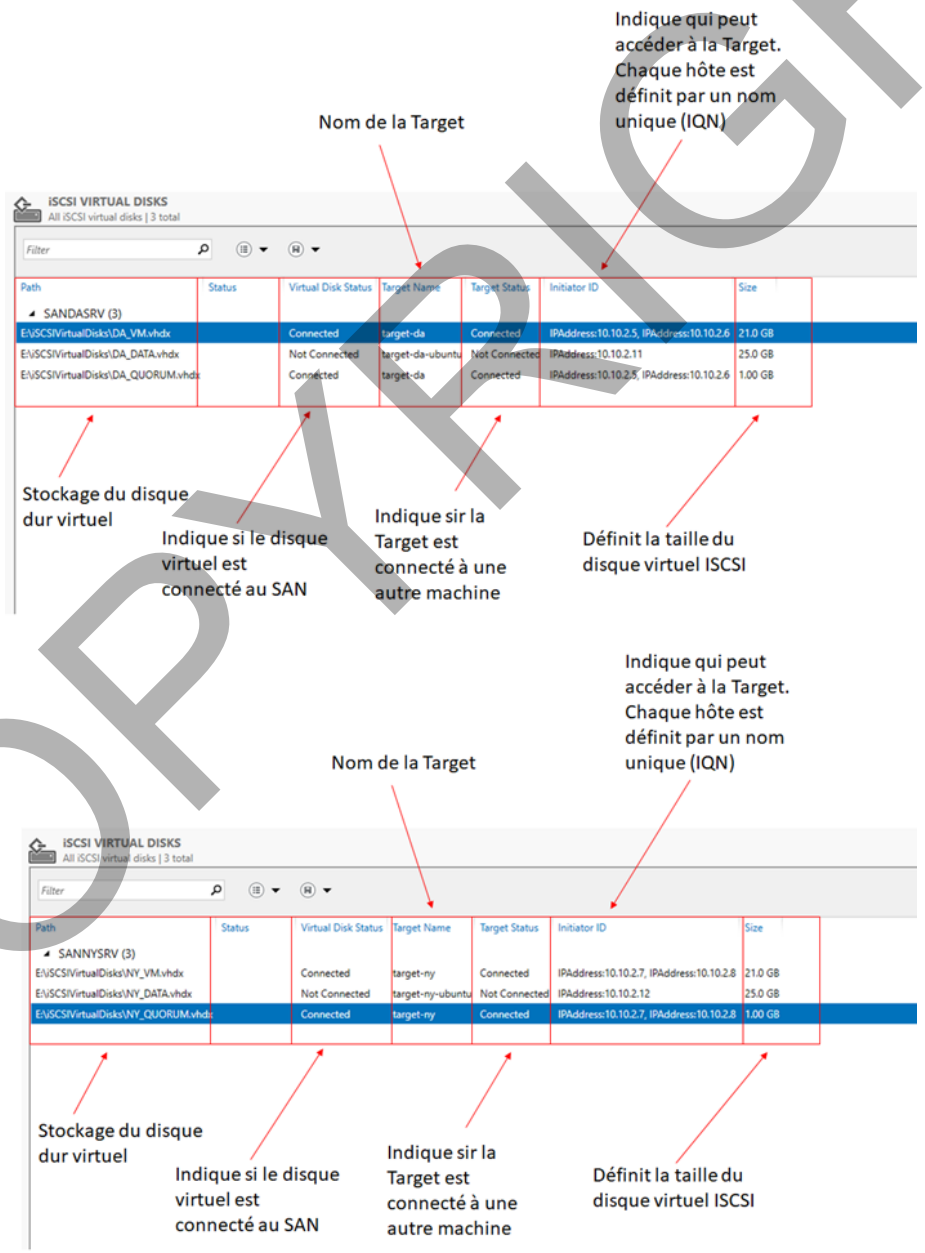


Figure 7. Présentation des disques virtuels iSCSI sur le SANNYSRV (en haut) et sur el SANDASRV (en bas)

Chaque disque virtuel est associé à une Target* :

- « target-ny » et « target-da » : permet de connecter aux deux hyperviseurs de New York et de Dallas, les disques virtuels « NY_QUORUM » et « DA_QUORUM » ainsi que « NY_VM » et « DA_VM ».
- « target-ny-ubuntu » et « target-da-ubuntu » : permet de connecter aux deux serveurs web de New York et de Dallas, les disques virtuels « NY_DATA » et « DA_DATA ».

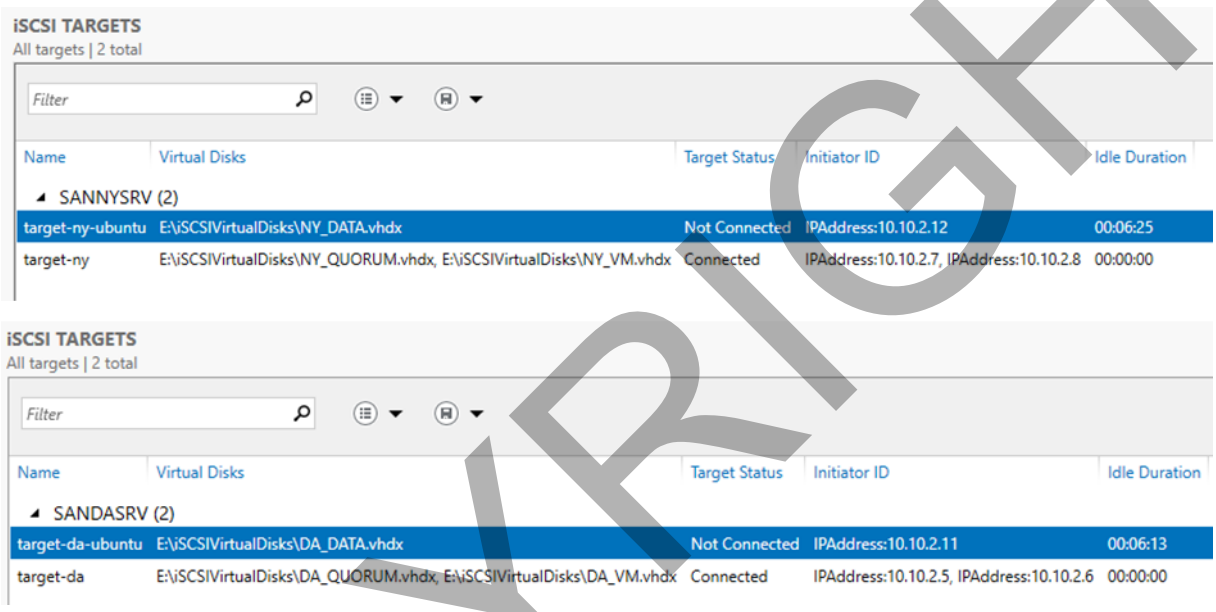


Figure 8. Liste des Targets présentes sur le SANNYSRV (en haut) et sur le SANDASRV (en bas)

Ensuite, les deux hyperviseurs de chaque datacenter se connectent à leur Target via l'interface Initiator* iSCSI.

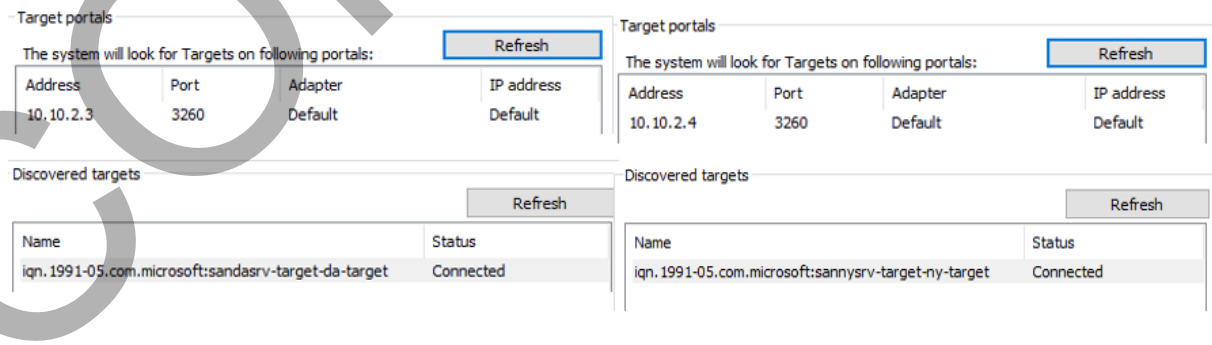


Figure 9. Connexion aux Target permettant de récupérer les disques durs virtuels iSCSI (SANDASRV à gauche ; SANNYSRV à droite)

Nous avons activé le multi-path* sur chacun de ces disques pour que les deux hyperviseurs puissent se connecter simultanément. Il suffit ensuite en

se rendant dans le gestionnaire de disques, de les initialiser puis de créer un nouveau volume et enfin les ajouter dans le cluster (cf. §4.2.1) pour que les VM puissent être stockés dessus.

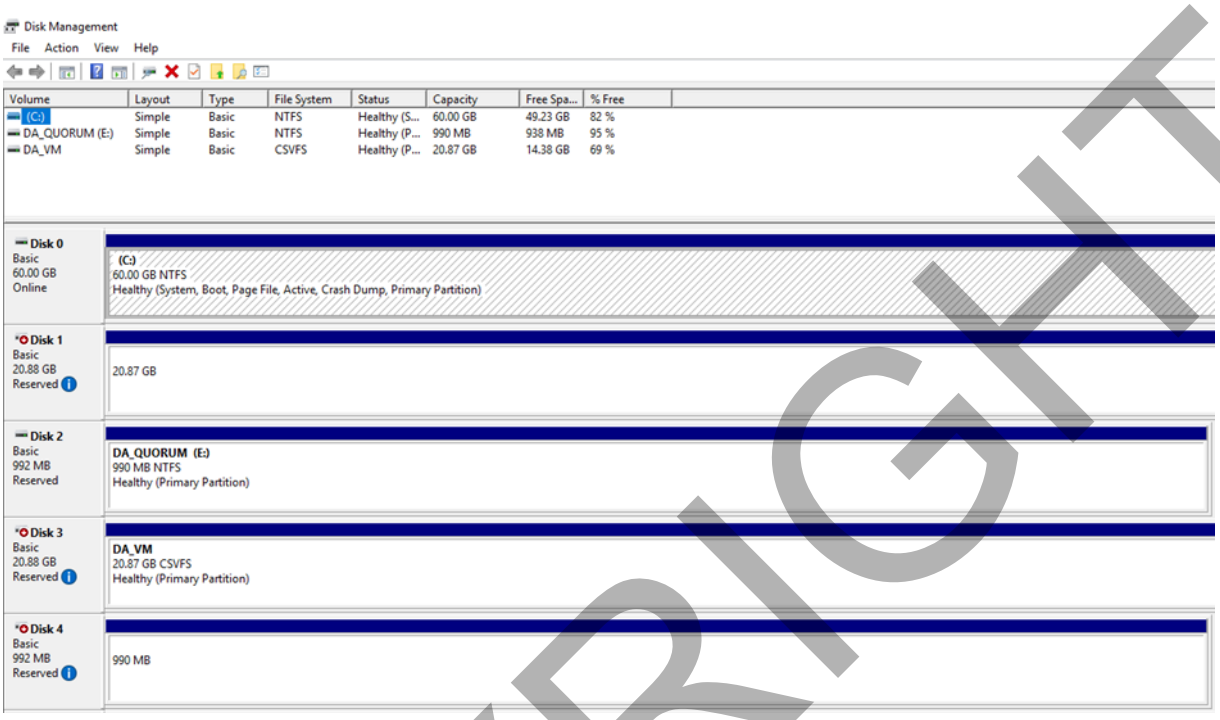
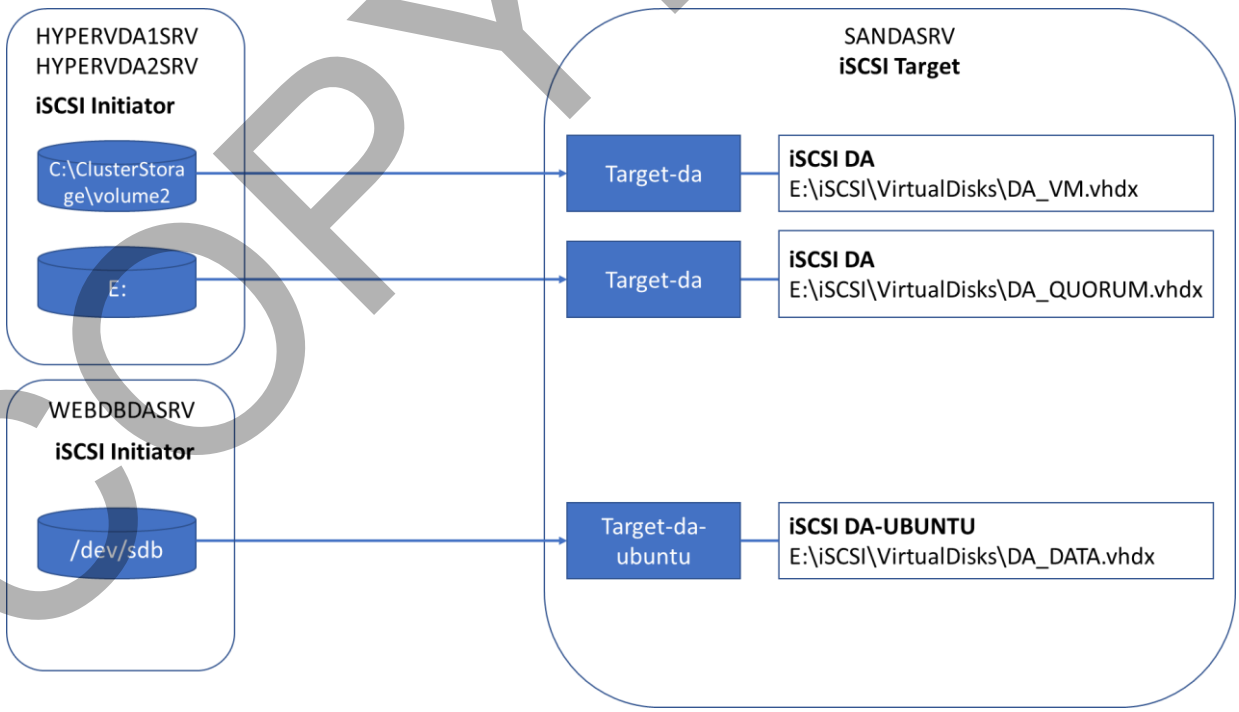


Figure 10. Nouveaux volumes créés sur un des hyperviseurs du datacenter de Dallas



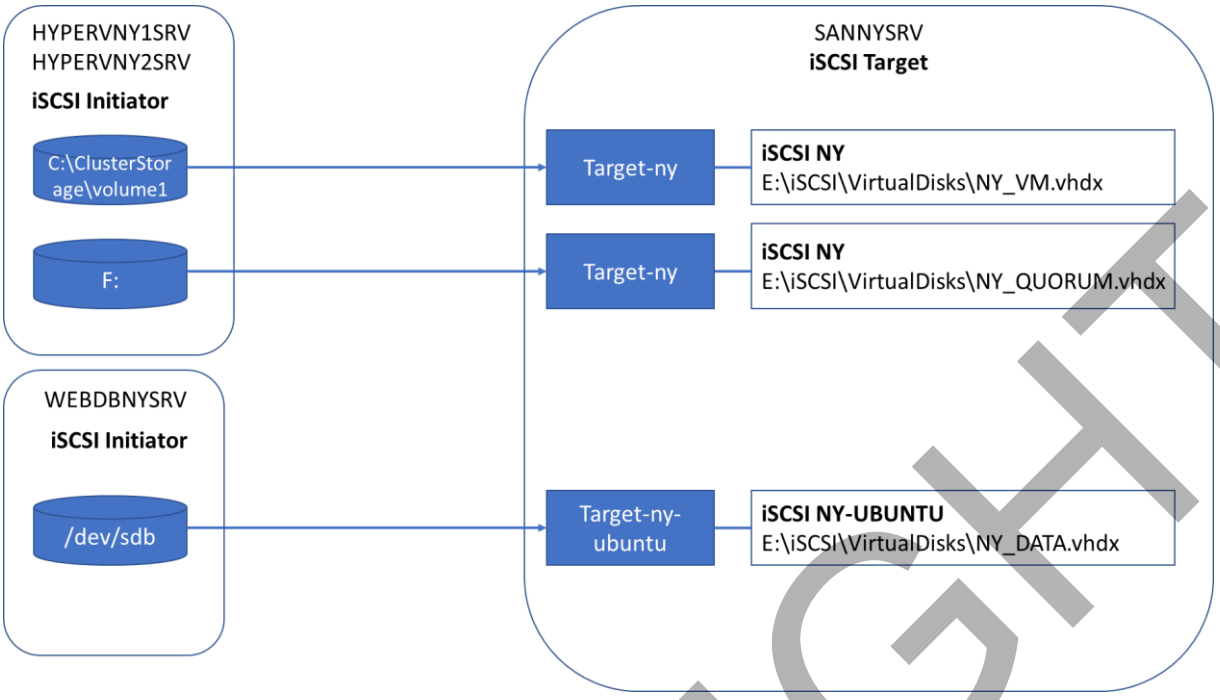


Figure 11. Schémas récapitulatifs représentant une connexion entre une Target et un Initiator

4.1.2. Le serveur de Backup présent sur le datacenter de Toronto

Dans le but d'assurer une continuité des services, nous avons mis en place sur le datacenter de Toronto, un serveur SAN iSCSI de Backup. Fonctionnant sous Microsoft Windows Server 2016, il est relié au même nom de domaine (« qwirk.lan ») que les autres serveurs.

Pour rappel, les données de l'application web ainsi que ceux de la base de données sont stockées sur un disque dur iSCSI (« NY_DATA » et « DA_DATA ») présentes sur chaque serveur SAN de chaque datacenter. Une réplication synchrone est établie entre les deux permettant d'avoir les données de l'application « Qwirk++ » synchronisées en temps réel. Il est primordial de sauvegarder ces données chaque jour.

Ainsi, nous avons installé sur le SANBACKUPSRV, le logiciel de sauvegarde Acronis Backup Standard[1] et sur les deux autres baies de

[1] Version 12.5.7048

stockage, l'agent Backup. Ce dernier permet au serveur de backup de configurer à distance la sauvegarde.

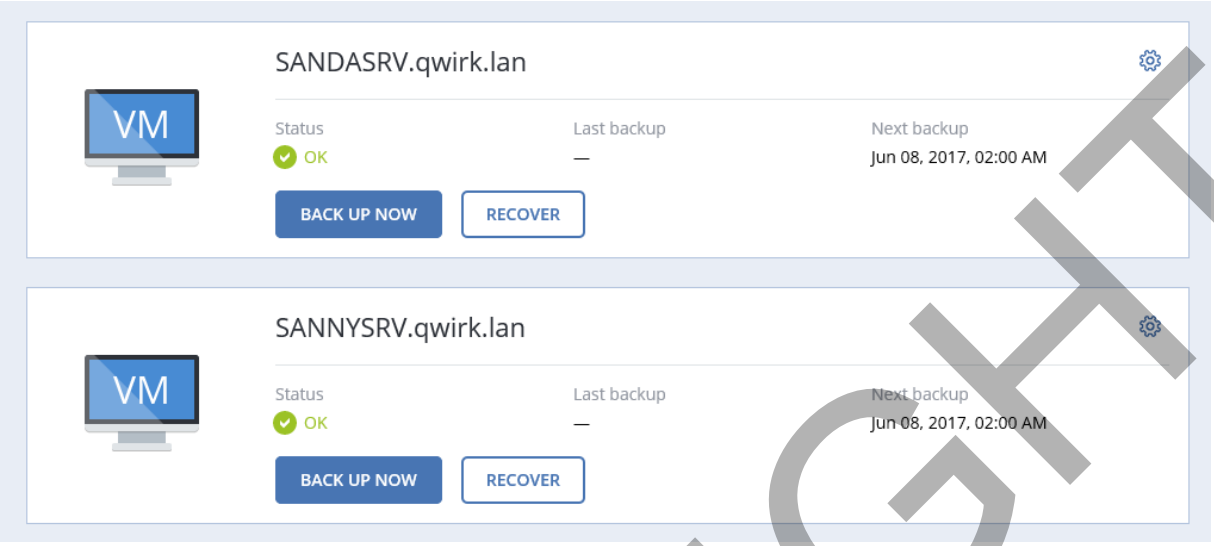


Figure 12. Etat des agents

Une sauvegarde complète des données contenues dans les disques virtuels « NY_DATA » et « DA_DATA » a été programmé tous les jours à 2h du matin.

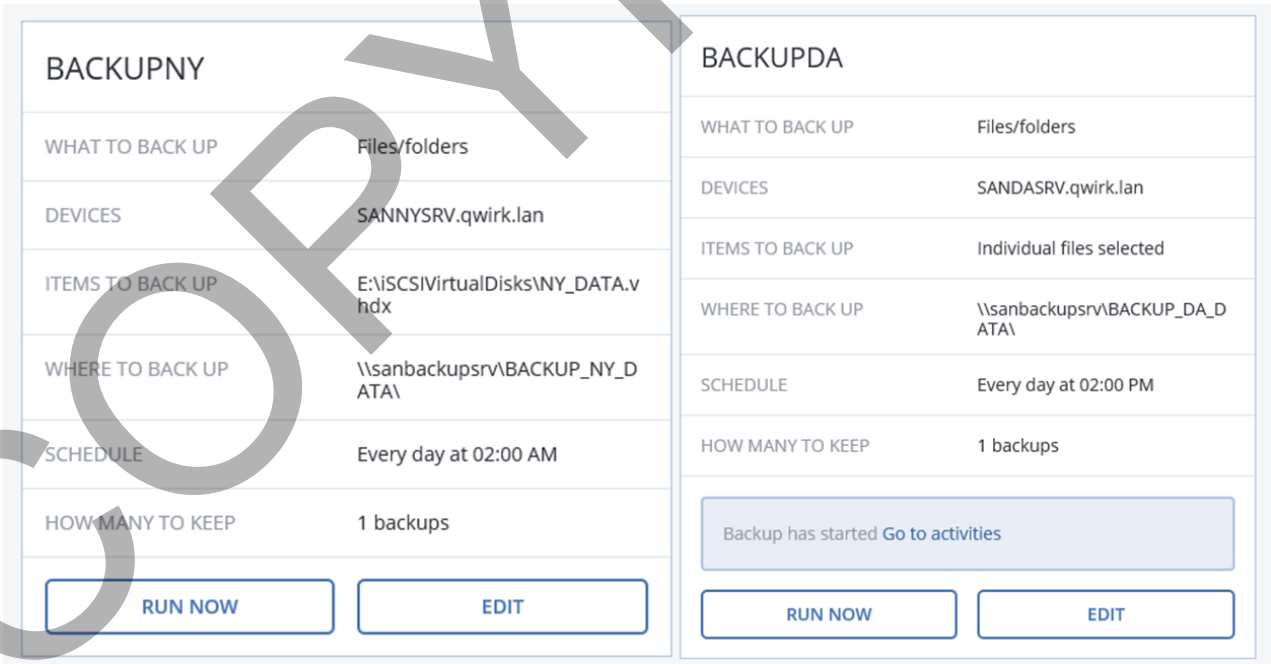


Figure 13. Paramétrage des sauvegardes

A noter que ces sauvegardes sont stockées dans un dossier partagé différent présent sur le serveur SANBACKUPSRV :

- « \\SANBACKUPSRV\BACKUP_DA_DATA » pour le SANDASRV
- « \\SANBACKUPSRV\BACKUP_NY_DATA » pour le SANNYSRV

Ces dossiers sont stockés sur un disque dur monté en RAID1 permettant d'assurer une redondance des données.

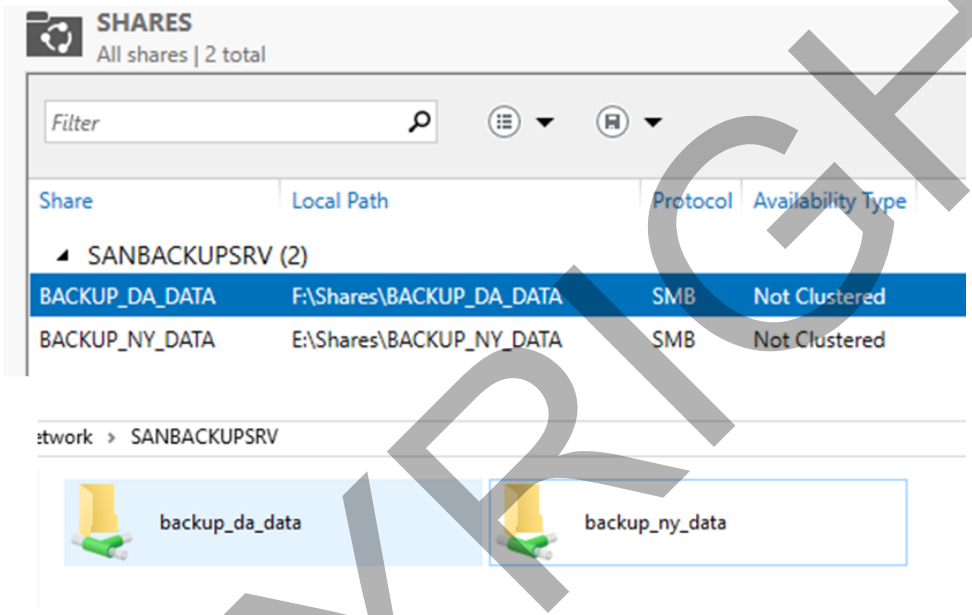


Figure 14. Disques durs partagés

Il est également possible de faire grâce au logiciel Acronis des sauvegardes sur des « tape » (bande magnétique) ou encore sur le cloud.

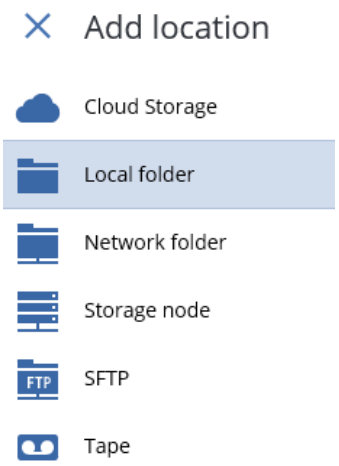


Figure 15. Emplacements disponibles pour la sauvegarde

4.2. Gestion de la haute disponibilité

4.2.1. Le cluster à basculement

Aujourd'hui, la disponibilité est un enjeu important dans l'infrastructure informatique. Elle permet d'assurer la continuité d'un service en cas de dysfonctionnement. Nous avons mis en place un cluster à basculement pour chaque datacenter de Dallas et de New York. Chaque cluster est constitué de la même façon, à savoir : deux hyperviseurs (Microsoft Hyper V).

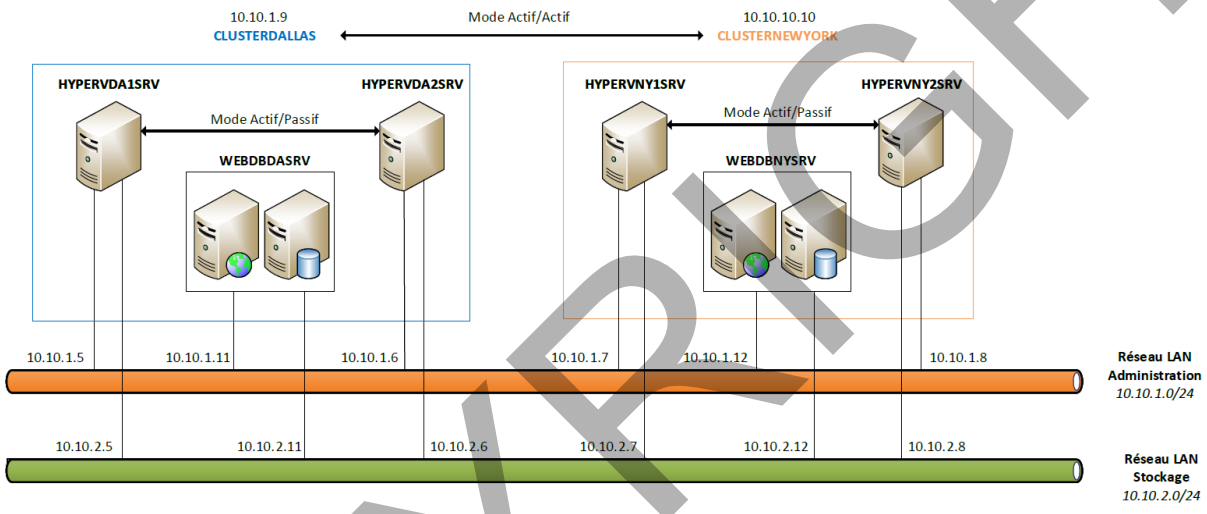


Figure 16. Schéma représentant le cluster à basculement entre le datacenter de Dallas et de New York

Dans la suite, je prendrais l'exemple du cluster « NYCLUSTER » présent sur le datacenter de Dallas, mais il s'agit de la même configuration que l'on peut retrouver à Dallas.

Sur le serveur qui gère l'Active Directory du domaine (DCRV), nous retrouvons l'interface « Failover Cluster Manager » permettant de contrôler les clusters :

- « dacluster.qwirk.lan » : contrôle le cluster du datacenter de Dallas
- « nycluster.qwirk.lan » : contrôle le cluster du datacenter de New York.

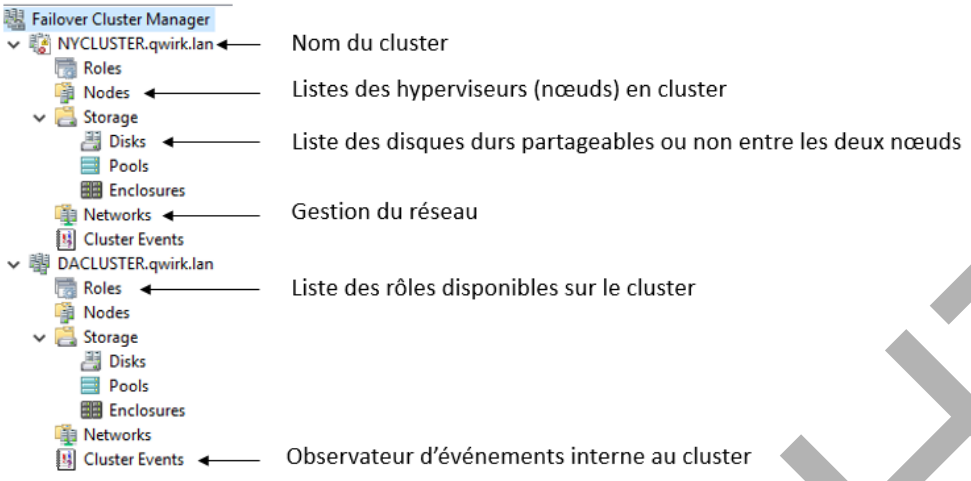


Figure 17. Description des onglets dans l'interface du cluster

Concernant la VM, on peut connaître son nom, son état (en cours, arrêté, en pause…) et sa priorité (ordre de passage en cas de reboot ou de changement d'hyperviseur en cas de panne).

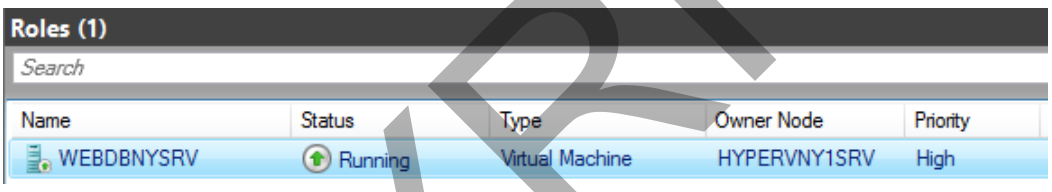


Figure 18. Machine virtuelle présent sur un des hyperviseur

Nous pouvons également, définir plusieurs paramètres sur la VM tels que :

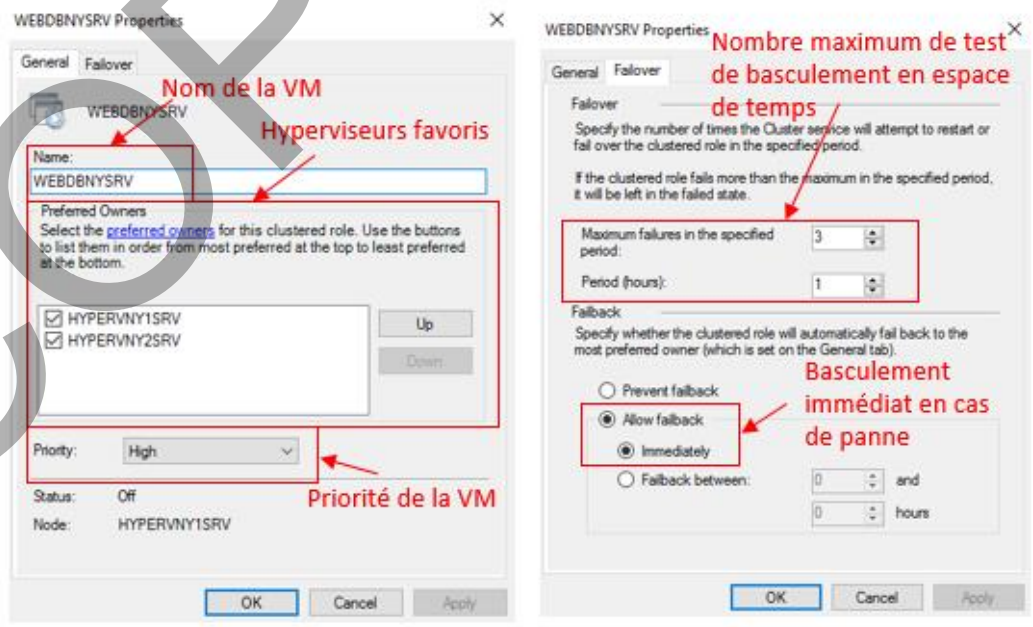


Figure 19. Propriété d'une machine virtuelle

Le cluster fonctionne comme ci-dessous :

HYPERVNY2SRV « A » et HYPERVNY2SRV « B » sont des hyperviseurs. « A » contient une machine virtuelle (VM) : WEBNYSRV « A1 »

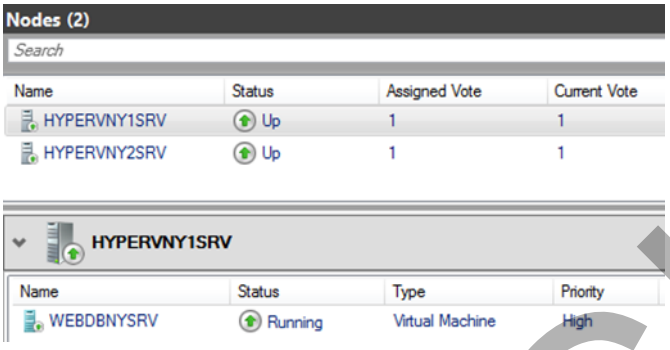


Figure 20. Machine virtuelle présent dans le cluster NYCLUSTER

Imaginons que le serveur « A » tombe en panne. Son état passe de « Up » à « Isolated » et la VM « A1 » a comme statut « Unmonitored ».

Le statut « Isolated » (Isolé) signifie que le nœud n'est plus un membre actif mais qu'il continue d'héberger le rôle VM.

Le statut « Unmonitored » signifie que la VM « A1 » n'est plus surveillé par le service du cluster.

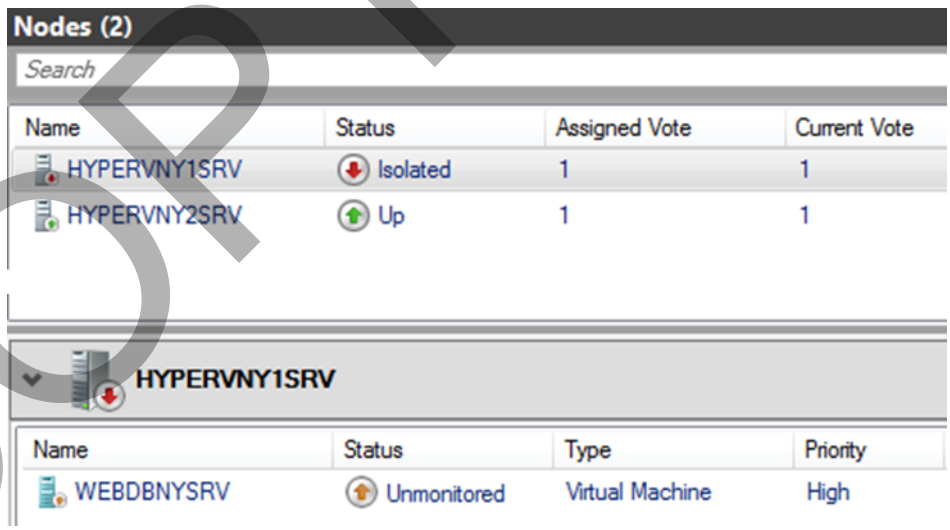


Figure 21. Simulation d'une panne sur le serveur HYPERVNY1SRV

L'hyperviseur « A » passe en état de « Down » au bout de 4 minutes. On remarque que la machine virtuelle « A1 » n'est plus présente sur le serveur « A ».

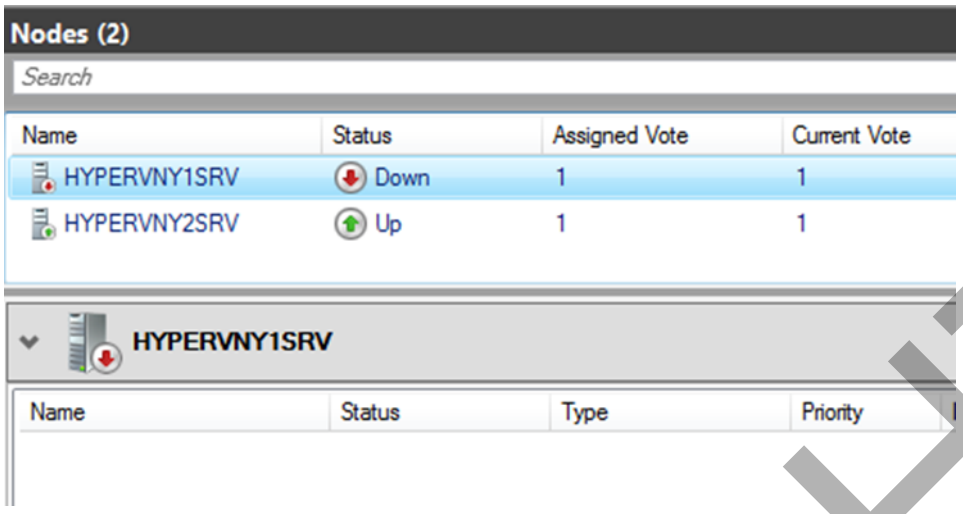


Figure 22. Etat « down » du serveur HYPERVNY1SRV

Cas particulier, si « A » est déjà passé à l'état « Isolated » trois fois en l'espace d'une heure, son état devient « Quarantined» (Quarantaine) pendant 7200 secondes soit 2 heures. Si « A » est de nouveau disponible, son état reste le même durant la période prédéfinie avant de repasser « Up ». Toutefois, en cas de nécessiter, nous pouvons forcer le passage de « Quarantined » à « Up ».

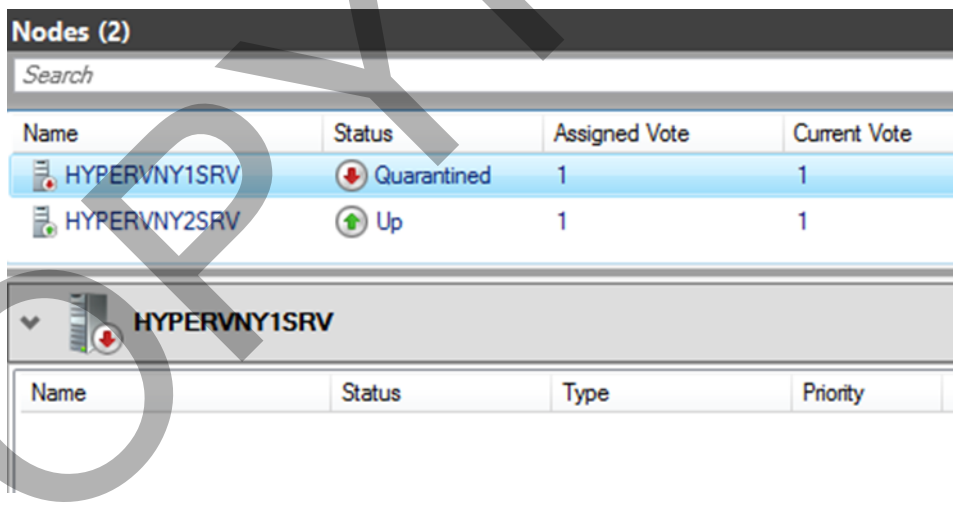


Figure 23. Etat « Quarantined » du serveur HYPERVNY1SRV

A noter que tous ces paramètres peuvent être modifiables en ligne de commandes PowerShell.

```
PlumbAllCrossSubnetRoutes          : 0
PreferredSite                      :
PreventQuorum                      : 0
QuarantineDuration                 : 7200
QuarantineThreshold                : 3
QuorumArbitrationTimeMax           : 20
RecentEventsResetTime              : 6/4/2017 8:28:55 PM
RequestReplyTimeout                : 60
ResiliencyDefaultPeriod            : 240
ResiliencyLevel                    : AlwaysIsolate
RouteHistoryLength                 : 10
S2DBusTypes                        : 0
S2DCacheBehavior                   : Default
S2DCacheDesiredState               : Enabled
S2DCacheMetadataReserveBytes       : 34359738368
S2DCachePageSizeKBytes             : 16
S2DEnabled                         : 0
S2DIOLatencyThreshold              : 10000
S2DOptimizations                   : 0
SameSubnetDelay                    : 1000
SameSubnetThreshold                : 10
SecurityLevel                      : 1
SharedVolumeCompatibleFilters      : {}
SharedVolumeIncompatibleFilters    : {}
SharedVolumeSecurityDescriptor     : {1, 0, 4, 128...}
SharedVolumesRoot                  : C:\ClusterStorage
SharedVolumeVssWriterOperationTimeout : 1800
ShutdownTimeoutInMinutes           : 20
UseClientAccessNetworksForSharedVolumes : 2
WitnessDatabaseWriteTimeout        : 300
```

Figure 24. Extrait de la commande PowerShell « Get-Cluster »

Etant donné que le serveur « A » est en quarantaine, c'est le serveur « B » qui a récupéré la machine virtuelle « A1 ».

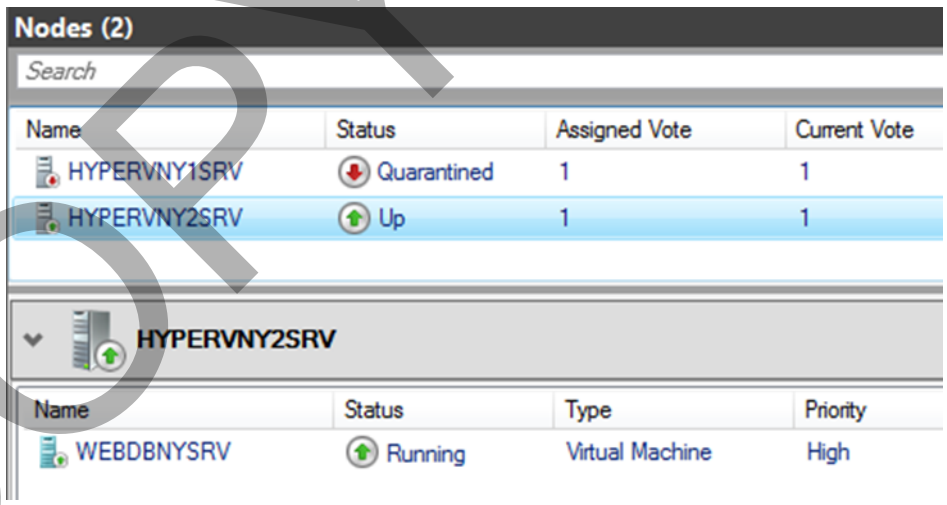


Figure 25. Le serveur HYPERVNY2SRV récupère la VM de l'hyperviseur HYPERVNY1SRV

Ce dispositif est possible étant donné que les disques durs de ces VM sont stockés dans une baie de SAN. En effet, si les disques durs étaient stockés sur les hyperviseurs en cas de panne de l'un des deux, les données ne seraient plus accessibles par l'autre.

Depuis l'onglet « disk » présent dans l'interface permettant de gérer le cluster, nous avons accès aux disques durs qui seront utilisés pour les VM. Il faut les mettre en « Cluster Shared Volume » pour que ce volume soit partagé dans le cluster.

Egalement, nous avons ajouté un disque dur « QUORUM » qui est important pour la gestion du cluster. Le quorum détermine le nombre d'échecs que le cluster peut supporter (dans notre cas, il est paramétré à 3 échecs en l'espace d'une heure). S'il y a un échec supplémentaire, le cluster doit cesser de s'exécuter. Notre disque dur se comporte comme un témoin de disque, c'est-à-dire qu'il contient une copie de la configuration du cluster.

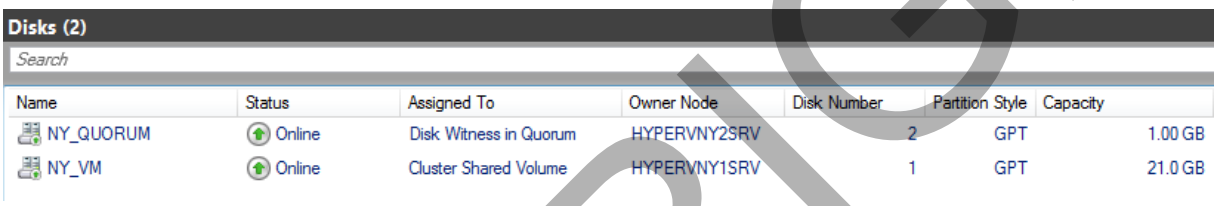


Figure 26. Listes des disques durs présents dans le cluster de New York

4.2.2. La réplication en temps réel des données applicatives

Chaque cluster fait tourner une machine virtuelle (VM) qui héberge l'application « Qwirk++ ». Etant donné que chaque datacenter doit fonctionner en mode Actif/Actif entre eux, il faut que les données de la base de données soient répliquées en temps réel.

En effet, prenons l'exemple suivant : on a un ami « A » qui habite à New York et un collègue « B » de Dallas. Les deux personnes seront connectées au bon datacenter grâce à la fonctionnalité du CDN (cf. §4.5). Quand « A » va envoyer un message à « B », il faut que « B » le reçoive instantanément.

Pour mettre en place cette fonctionnalité, nous avons établi une réplication de données entre les serveurs web grâce au logiciel DRBD*.

Dans le schéma ci-dessous, ce sont les disques durs correspondant à « VM_DATA » qui seront utilisés pour faire cette réplication.

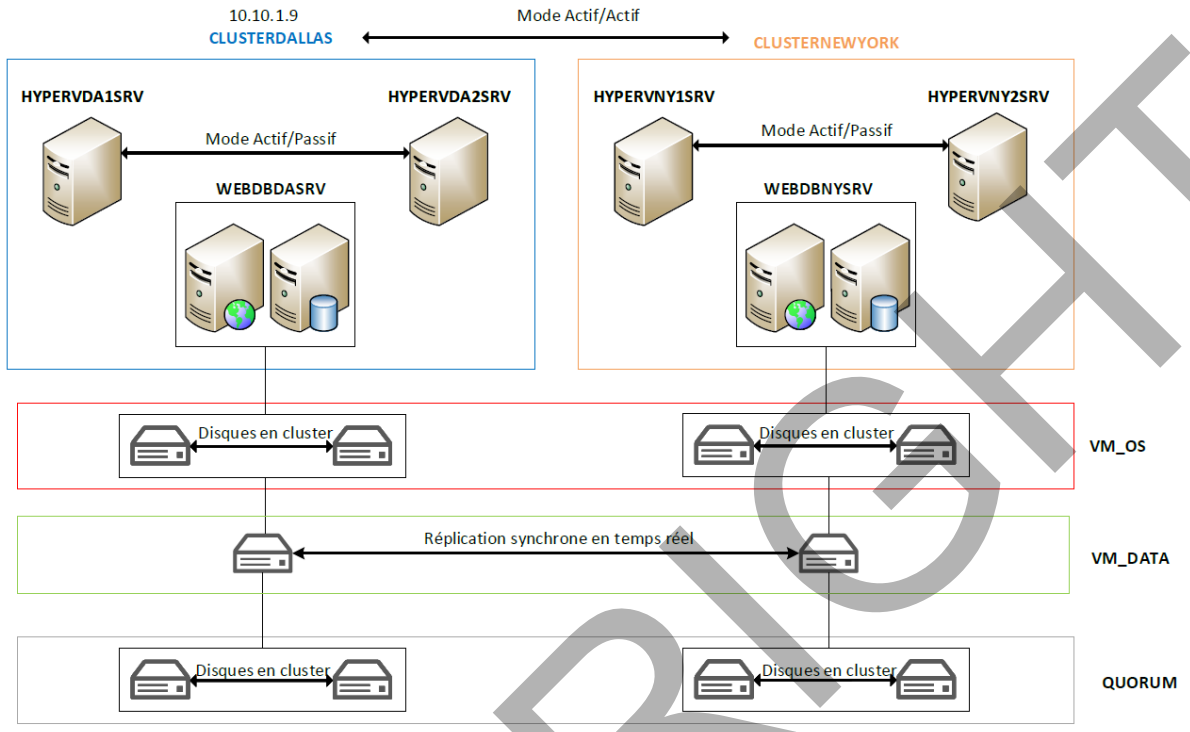


Figure 27. Schéma représentant la réplication synchrone des données

Installation des disques durs iSCSI

D'abord, nous avons connecté le disque dur iSCSI :

- « DA_DATA » provenant du SAN de Dallas vers le serveur WEBDBDASRV (appelé « node1 »).
- « NY_DATA » provenant du SAN de New York vers le serveur WEBDBNYSRV (appelé « node2).

Node1	Node2
root@node1:~# apt-get install open-iscsi **root@node1:~# sudo iscsiadm --mode discovery --type sendtargets --portal 10.10.2.3** 10.10.2.3:3260,1 iqn.1991-05.com.microsoft:sandasrv-target-da-ubuntu-target 10.10.1.3:3260,1 iqn.1991-05.com.microsoft:sandasrv-target-da-ubuntu-target **root@node1:~# iscsiadm --mode node --**	**root@node2:~# apt-get install open-iscsi** **root@node2:~# sudo iscsiadm --mode discovery --type sendtargets --portal 10.10.2.4** 10.10.2.4:3260,1 iqn.1991-05.com.microsoft:sannysrv-target-ny-ubuntu-target 10.10.1.4:3260,1 iqn.1991-05.com.microsoft:sannysrv-target-ny-ubuntu-target **root@node2:~# iscsiadm --mode node --**

targetname iqn.1991-05.com.microsoft:sandasrv-target-da-ubuntu-target \ --portal 10.10.2.3 --login Logging in to [iface: default, target: iqn.1991-05.com.microsoft:sandasrv-target-da-ubuntu-target, portal: 10.10.1.3,3260] (multiple) Logging in to [iface: default, target: iqn.1991-05.com.microsoft:sandasrv-target-da-ubuntu-target, portal: 10.10.2.3,3260] (multiple) iscsiadm: Could not login to [iface: default, target: iqn.1991-05.com.microsoft:sandasrv-target-da-ubuntu-target, portal: 10.10.1.3,3260]. iscsiadm: initiator reported error (19 - encountered non-retryable iSCSI login failure) Login to [iface: default, target: iqn.1991-05.com.microsoft:sandasrv-target-da-ubuntu-target, portal: 10.10.2.3,3260] successful. iscsiadm: Could not log into all portals **root@node1:~# nano /etc/iscsi/iscsid.conf** #Changer : node.startup = automatic	**targetname iqn.1991-05.com.microsoft:sannysrv-target-ny-ubuntu-target \ --portal 10.10.2.4 --login** Logging in to [iface: default, target: iqn.1991-05.com.microsoft:sannysrv-target-ny-ubuntu-target, portal: 10.10.1.4,3260] (multiple) Logging in to [iface: default, target: iqn.1991-05.com.microsoft:sannysrv-target-ny-ubuntu-target, portal: 10.10.2.4,3260] (multiple) iscsiadm: Could not login to [iface: default, target: iqn.1991-05.com.microsoft:sannysrv-target-ny-ubuntu-target, portal: 10.10.1.4,3260]. iscsiadm: initiator reported error (19 - encountered non-retryable iSCSI login failure) Login to [iface: default, target: iqn.1991-05.com.microsoft:sannysrv-target-ny-ubuntu-target, portal: 10.10.2.4,3260] successful. iscsiadm: Could not log into all portals **root@node2:~# nano /etc/iscsi/iscsid.conf** #Changer : node.startup = automatic

Configuration du DRBD

Node1	Node2
#Installation du packet **root@node1:~# apt-get install drbd8-utils** #Charger le module et le mettre en persistant **root@node1:~# modprobe drbd** **root@node1:~# echo "drbd" >> /etc/modules**	#Installation du packet **root@node2:~# apt-get install drbd8-utils** #Charger le module et le mettre en persistant **root@node2:~# modprobe drbd** **root@node2:~# echo "drbd" >> /etc/modules**

#Copier dans les deux nodes le même fichier dans **/etc/drbd.d/global_common.conf**

```
# Global configuration
global {
    # Do not report statistics usage to LinBit
    usage-count no;
}
# All resources inherit the options set in this section
common {
```

```
    # C (Synchronous replication protocol)
    protocol C;
    startup {
        # Wait for connection timeout (in seconds)
        wfc-timeout 1 ;
        # Wait for connection timeout, if this node was a degraded cluster (in seconds)
        degr-wfc-timeout 1 ;
    }
    net {
        # Maximum number of requests to be allocated by DRBD
        max-buffers 8192;
        # The highest number of data blocks between two write barriers
        max-epoch-size 8192;
        # The size of the TCP socket send buffer
        sndbuf-size 512k;
        # How often the I/O subsystem's controller is forced to process pending I/O
requests
        unplug-watermark 8192;
        # The HMAC algorithm to enable peer authentication at all
        cram-hmac-alg sha1;
        # The shared secret used in peer authentication
        shared-secret "xxx";
        # Split brains
        # Split brain, resource is not in the Primary role on any host
        after-sb-0pri disconnect;
        # Split brain, resource is in the Primary role on one host
        after-sb-1pri disconnect;
        # Split brain, resource is in the Primary role on both host
        after-sb-2pri disconnect;
        # Helps to solve the cases when the outcome of the resync decision is incompatible
with the current role assignment
        rr-conflict disconnect;
    }
    handlers {
        # If the node is primary, degraded and if the local copy of the data is
inconsistent
        pri-on-incon-degr "echo Current node is primary, degraded and the local copy of
the data is inconsistent | wall ";
    }
    disk {
        # The node downgrades the disk status to inconsistent on io errors
        on-io-error pass_on;
        # Disable protecting data if power failure (done by hardware)
```

```
        no-disk-barrier;
        # Disable the backing device to support disk flushes
        no-disk-flushes;
        # Do not let write requests drain before write requests of a new reordering domain
are issued
        no-disk-drain;
        # Disables the use of disk flushes and barrier BIOs when accessing the meta data
device
        no-md-flushes;
    }
    syncer {
        # The maximum bandwidth a resource uses for background re-synchronization
        rate 500M;
        # Control how big the hot area (= active set) can get
        al-extents 3833;
    }
}
#Copier dans les deux nodes le même fichier dans /etc/drbd.d/r0.res
resource r0 {
    startup {
      become-primary-on both;
    }
    net {
      protocol C;
      allow-two-primaries yes;
    }
    # WEBDBDASRV
    on node1 {
        device      /dev/drbd0;
        # Disk containing the drbd partition
        disk        /dev/sdb;
        # IP address of this host
        address     10.10.2.11:7788;
        # Store metadata on the same device
        meta-disk    internal;
    }
    # WEBDBNYSRV
    on node2 {
        device      /dev/drbd0;
        disk        /dev/sdb;
        address     10.10.2.12:7788;
        meta-disk   internal;
    }
```

```
}
#Lancer sur les deux nodes les commandes suivantes :
drbdadm create-md r0
drbdadm up r0
```

#Première réplication bloc par bloc **root@node1:~# drbdadm -- --overwrite-data-of-peer primary r0** #Attendre la fin de la syncho avant de passer à la suite **root@node1:~# cat /proc/drbd** version: 8.4.7 (api:1/proto:86-101) srcversion: 0904DF2CCF7283ACE07D07A 0: cs:Connected ro:Primary/Secondary ds:UpToDate/Diskless C r----- ns:0 nr:0 dw:0 dr:0 al:8 bm:0 lo:0 pe:0 ua:0 ap:0 ep:1 wo:n oos:15693308	
	root@node2:~# drbdadm secondary r0 **root@node2:~# drbdadm primary r0** **root@node2:~# drbdadm attach r0** **root@node2:~# drbdadm -- --overwrite-data-of-peer primary r0** **root@node2:~# cat /proc/drbd** version: 8.4.7 (api:1/proto:86-101) srcversion: 0904DF2CCF7283ACE07D07A 0: cs:Connected ro:Primary/Primary ds:UpToDate/UpToDate C r----- ns:0 nr:15693308 dw:15693308 dr:0 al:8 bm:0 lo:0 pe:0 ua:0 ap:0 ep:1 wo:n oos:0
root@node1:~# drbd-overview 0:r0/0 Connected Primary/Primary UpToDate/UpToDate	

Formatage du disque dur DRBD en OCFS

Node1	Node2
root@node1:~#apt-get install ocfs2-tools	**root@node2:~#apt-get install ocfs2-tools**

#Configuration sur les deux nodes du cluster dans **/etc/ocfs2/cluster.conf**

```
node:
    ip_port = 7777
    ip_address = 10.10.2.11
    number = 0
    name = node1
```

```
    cluster = ocfs2

node:
    ip_port = 7777
    ip_address = 10.10.2.12
    number = 1
    name = node2
    cluster = ocfs2

cluster:
    node_count = 2
    name = ocfs2
```

```
#On formate les disque DRBD
root@node1:~# mkfs.ocfs2 /dev/drbd0
mkfs.ocfs2 1.6.4
Cluster stack: classic o2cb
Label:
Features: sparse backup-super unwritten
inline-data strict-journal-super xattr
Block size: 4096 (12 bits)
Cluster size: 4096 (12 bits)
Volume size: 16069947392 (3923327 clusters)
(3923327 blocks)
Cluster groups: 122 (tail covers 20351
clusters, rest cover 32256 clusters)
Extent allocator size: 4194304 (1 groups)
Journal size: 100433920
Node slots: 8
Creating bitmaps: done
Initializing superblock: done
Writing system files: done
Writing superblock: done
Writing backup superblock: 2 block(s)
Formatting Journals: done
Growing extent allocator: done
Formatting slot map: done
Formatting quota files: done
Writing lost+found: done
mkfs.ocfs2 successful
```

Monter le disque dur DRDB

Node1	Node2
#On démarre le service de cluster	#On démarre le service de cluster
root@node1:~# /etc/init.d/o2cb online	**root@node2:~# /etc/init.d/o2cb online**
Loading stack plugin "o2cb": OK	Loading stack plugin "o2cb": OK
Loading filesystem "ocfs2_dlmfs": OK	Loading filesystem "ocfs2_dlmfs": OK
Creating directory '/dlm': OK	Creating directory '/dlm': OK
Mounting ocfs2_dlmfs filesystem at /dlm: OK	Mounting ocfs2_dlmfs filesystem at /dlm: OK
Setting cluster stack "o2cb": OK	Setting cluster stack "o2cb": OK
Starting O2CB cluster ocfs2: OK	Starting O2CB cluster ocfs2: OK
root@node1:~# cd /mnt	**root@node2:~# cd /mnt**
root@node1:/mnt# mkdir data	**root@node2:/mnt# mkdir data**
root@node1:/# cd ~	**root@node2:/# cd ~**
root@node1:~# mount /dev/drbd0 /mnt/data	**root@node2:~# mount /dev/drbd0 /mnt/data**

```
root@node1:/mnt/data# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            943M     0  943M   0% /dev
tmpfs           193M  6,0M  187M   4% /run
/dev/sda1        18G  4,2G   13G  25% /
tmpfs           964M  192K  963M   1% /dev/shm
tmpfs           5,0M     0  5,0M   0% /run/lock
tmpfs           964M     0  964M   0% /sys/fs/cgroup
tmpfs           193M   52K  193M   1% /run/user/1000
/dev/drbd0       25G  1,3G   24G   6% /mnt/data
```

```
root@node2:/mnt/data# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            943M     0  943M   0% /dev
tmpfs           193M  6,0M  187M   4% /run
/dev/sda1        18G  4,2G   13G  25% /
tmpfs           964M  192K  963M   1% /dev/shm
tmpfs           5,0M     0  5,0M   0% /run/lock
tmpfs           964M     0  964M   0% /sys/fs/cgroup
tmpfs           193M   56K  193M   1% /run/user/1000
/dev/drbd0       25G  1,3G   24G   6% /mnt/data
```

Figure 28. Listes des disques durs présents sur les deux serveurs web

Test de la réplication

Node1	Node2
root@node1:~# cd /mnt/data/ **root@node1:/mnt/data# ls** lost+found **root@node1:/mnt/data# nano test**	
	root@node2:~# cd /mnt/data/ **root@node2:/mnt/data# ls** lost+found test **root@node2:/mnt/data# mkdir testo**
root@node1:/mnt/data# ls lost+found test testo	

4.3. Gestion de la virtualisation

Afin de mutualiser les capacités de chaque serveur, permettant de réaliser des économies et de réduire les investissements en infrastructures physiques, nous avons décidé de virtualiser des serveurs au sein de notre cluster. Pour rappel, chaque disque dur de ces serveurs virtualisés est stocké dans une baie de SAN. Avantage de cette virtualisation : on peut faire fonctionner différents systèmes d'exploitation sur une même machine physique. Plusieurs solutions de virtualisation existent tels que : VMware vSphere, Citrix Xen Server, Oracle VM, Microsoft Hyper-V… C'est cette dernière que nous avons mise en place. Il s'agit dans notre cas d'un hyperviseur de type 2. C'est-à-dire qu'on exécute sur le matériel physique, un OS (Windows Server 2016) avec le rôle « Hyper-V » d'installé et c'est ce dernier qui va virtualiser un autre OS. Notons que cette solution existe aussi en hyperviseur de type 1.

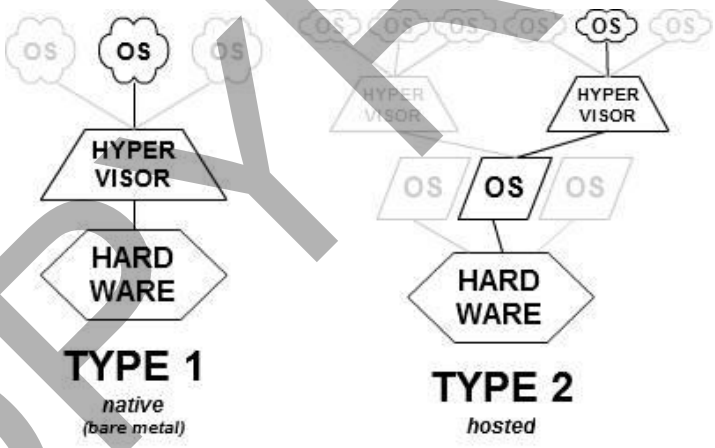


Figure 29. Différences entre les types d'hyperviseurs

Pour gérer les VM dans chaque hyperviseur, nous devons nous rendre dans l'interface « Hyper-V Manager », puis sélectionner la VM souhaitée. En faisant un clic-droit dessus, nous pouvons la démarrer, l'arrêter, la mettre en pause, et se connecter à la VM. A noter que cela peut également être fait en passant par le gestionnaire de cluster.

Figure 30. Gestionnaire de l'Hyper-V (ici sur HYPERVDA2SRV)

4.3.1. Le serveur web

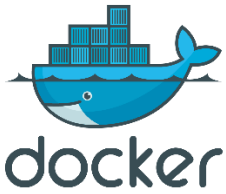
Les serveurs « WEBDBDASRV » et « WEBDBNYSRV » hébergent le back-end* et le front-end* de l'application de chat « Qwirk++ ». Ces deux serveurs possèdent la même configuration matérielle et logicielle. Ils fonctionnent sous Ubuntu[2].

Pour pouvoir faire fonctionner notre application, nous avons dû installer le container « Docker »[3], le Framework Open-Source basé sur Node.js « Meteor »[4] ainsi que la base de données orientée documents « MongoDB »[5]

Les données de l'application et de la base de données sont stockées sur un autre disque dur et sont répliquées en temps réel (cf. §4.2.2).

[2] Version 16.04 LTS
[3] Version 1.13.0
[4] Version 1.4
[5] Version 3.4.4

4.3.1.1. Docker



Pour installer le logiciel « Docker », nous avons dû rentrer les lignes de commandes suivantes en mode super admin (root) sur les deux serveurs web :

```
apt-get update && apt-get upgrade
apt-get install curl
curl -ssl https://get.docker.com/ | sh
curl -l https://github.com/docker/compose/releases/download/1.13.0/docker-compose-
`uname -s`-`uname -m` > /usr/local/bin/docker-compose
chmod +x /usr/local/bin/docker-compose
```

4.3.1.2. Meteor et MongoDB

Pour installer l'application « Meteor », nous avons dû rentrer les lignes de commandes suivantes en mode super admin (root) sur les deux serveurs web :

```
curl https://install.meteor.com/ | sh
apt-get install nodejs npm
ln -sT /usr/bin/nodejs /usr/bin/node
chown user:user .meteor/ -R
#On se place dans le dossier Qwirk.Chat se trouvant sur le disque dur DRBD
cd /mnt/data/Qwirk.Chat
chown -R /mnt/data/Qwirk.Chat
npm install npm -g
meteor npm install --save-dev meteor-desktop
```

La ligne de commande suivante permet de lancer le processus « Meteor » :

```
#On lance le serveur web sur le port 3000
meteor --mobile-server=127.0.0.1:3000
```

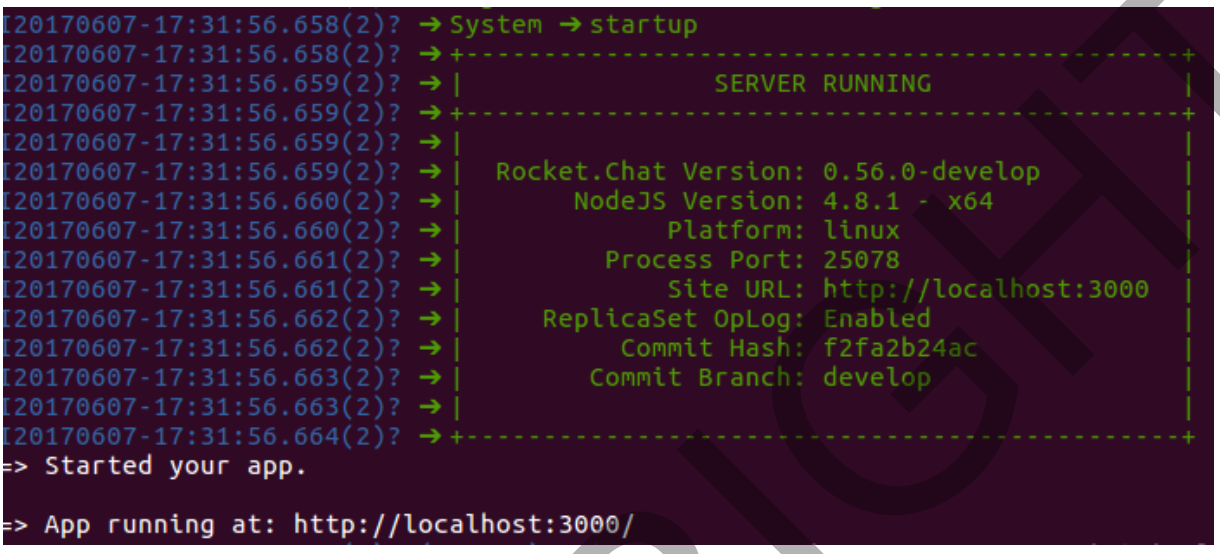


Figure 31. Application lancée

Quand l'utilisateur est connecté au réseau interne, il peut accéder à l'application via l'adresse URL* « chat.qwirk.lan ». Celle-ci redirige automatiquement vers l'adresse IP d'un des serveur web.

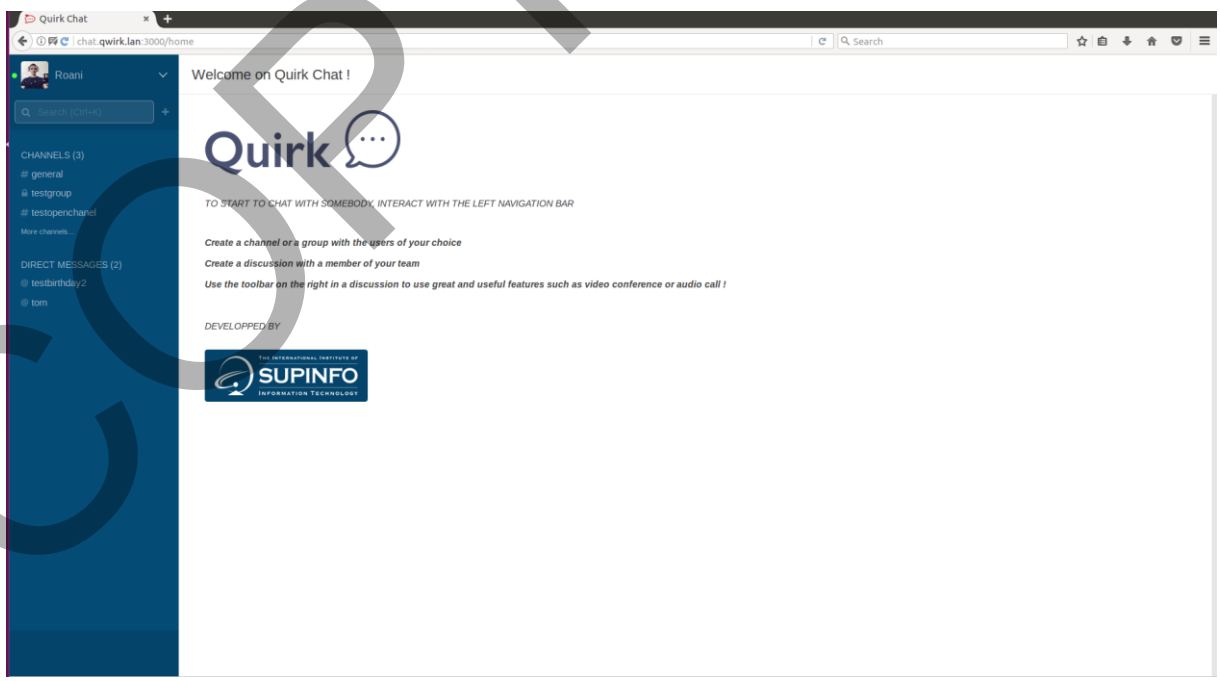


Figure 32. Page principale de l'application depuis un navigateur

A l'inverse, quand l'utilisateur est à l'extérieur du réseau « qwirk.lan », il doit se connecter sur l'adresse IP publique du pare-feu. Ce dernier redirige en interne la requête vers un des serveurs web.

En quelque sorte, il faut renseigner dans la barre URL : « http://[IP_Publique_ou_nom_de_domaine]:3000 ». Le « 3000 » correspond au port logiciel* que l'application utilise.

L'application étant multi-plateforme, elle fonctionne également sur d'autres appareils tels que les téléphones mobiles ou encore les tablettes.

Figure 33. Ecrans de l'application sur mobile (Android)

Pour lancer l'application Qwirk++ en mode bureau, il faut exécuter la commande suivante dans un nouveau terminal :

```
npm run desktop --init
npm run desktop
```

4.4. Gestion de la sécurité

4.4.1. L'utilisation d'un pare-feu

La sécurité est d'or dans une infrastructure. C'est pourquoi l'utilisation d'un routeur et pare-feu est fortement recommandée. Notre choix s'est porté vers « PfSense[6] ».

Lorsqu'un des serveurs tente d'accéder à Internet, sa requête transite vers le pare-feu et c'est ce dernier qui l'autorise ou non. De même lorsqu'un appareil tente d'y accéder depuis l'extérieur.

C'est à travers l'administration web que nous pouvons créer des règles pour le WAN ou le LAN. Pour faciliter l'organisation, nous avons créé des « aliases » pour chacun de nos serveurs.

Firewall Aliases IP

Name	Values	Description
DCSRV	10.10.1.2	Contrôleur de domaine
FIREWALL	10.10.1.1	Routeur/Pare-Feu
HYPERV1NYSRV	10.10.1.7	Serveur HYPERV1 sur le datacenter de New York
HYPERV2DASRV	10.10.1.6	Serveur HYPERV2 sur le datacenter de Dallas
HYPERV2NYSRV	10.10.1.8	Serveur HYPERV2 sur le datacenter de New York
HYPERVDA1SRV	10.10.1.5	Serveur HYPERV1 sur le datacenter de Dallas
SANBACKUPSRV	10.10.1.9	Serveur de backup SAN sur le datacenter de Toronto
SANDASRV	10.10.1.3	Serveur SAN sur le datacenter de Dallas
SANNYSRV	10.10.1.4	Serveur SAN sur le datacenter de New York
WEBDBDASRV	10.10.1.11	Serveur web + BD sur le datacenter de Dallas
WEBDBNYSRV	10.10.1.12	Serveur web + BD sur le datacenter de New York

Figure 34. Tableau d'aliases correspondant à une machine spécifique

[6] Version 2.3.3_1

Rules (Drag to Change Order)

	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description
✖	0 /212 B	*	RFC 1918 networks	*	*	*	*	*		Block private networks
✖	0 /0 B	*	Reserved Not assigned by IANA	*	*	*	*	*		Block bogon networks
☐ ✔	0 /0 B	IPv4+6 TCP/UDP	*	*	WAN address	*	*	none		
☐ ✔	0 /0 B	IPv4+6 TCP/UDP	*	*	WAN net	*	*	none		
☐ ✔	0 /0 B	IPv4+6 ICMP any	*	*	WAN address	*	*	none		
☐ ✔	0 /0 B	IPv4 TCP	*	*	WEBDBDASRV	3000 (HBCI)	*	none		NAT Acces serveur web et BD du datacenter de Dallas
☐ ✔	0 /0 B	IPv4 TCP	*	*	WEBDBNYSRV	3000 (HBCI)	*	none		NAT Access serveur web et BD du datacenter de New York
☐ ✔	0 /0 B	IPv4 TCP	*	*	WEBDBDASRV	3000 (HBCI)	*	none		NAT Redirection vers l'application Qwirk++
☐ ✔	0 /0 B	IPv4 TCP	*	*	FIREWALL	80 (HTTP)	*	none		NAT Acces GUI Firewall depuis l'exterieur

Figure 35. Règles appliquées pour le WAN

Rules (Drag to Change Order)

	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
✔	1 /616 KiB	*	*	*	LAN Address	80	*	*		Anti-Lockout Rule	⚙
☐ ✔	1 /95.75 MiB	IPv4 *	LAN net	*	*	*	*	none		Default allow LAN to any rule	⚓ ✎ ⧉ ⊘ 🗑
☐ ✔	0 /0 B	IPv6 *	LAN net	*	*	*	*	none		Default allow LAN IPv6 to any rule	⚓ ✎ ⧉ ⊘ 🗑

Figure 36. Règles appliquées pour le LAN

4.4.2. Le bannissement en cas d'échec de connexion

Dans le cadre d'assurer une sécurité, une solution a été retenue : lorsqu'un utilisateur se trompe 10 fois dans le processus de connexion, celui-ci sera banni 5 minutes dans un premier temps. Après ce délai, s'il se trompe à nouveau, il se verra banni pendant une durée d'une semaine. Enfin, après cette semaine de bannissement, si celui-ci se trompe à nouveau, il sera banni pour une durée de 2 mois.

A noter que le bannissement est basé sur l'adresse IP de l'utilisateur.

4.4.3. Le cryptage des données

Le cryptage des données permet de garder une confidentialité et une intégrité de celle-ci lors de l'exécution de l'application à travers tout le réseau. Etant donné que nous fournissons un POC, toute notre architecture est sur le même réseau. Dans la réalité, il faudrait que chaque datacenter possède des plages d'adresses IP différentes (pour le LAN et le LAN STORAGE) mais également dispose d'un pare-feu. Ensuite, une solution de VPN pourrait être implémenté entre chaque site (Dallas, New York et Toronto) afin de créer des tunnels sécurisés dans lesquels les données circuleraient de façon anonymes et cryptées.

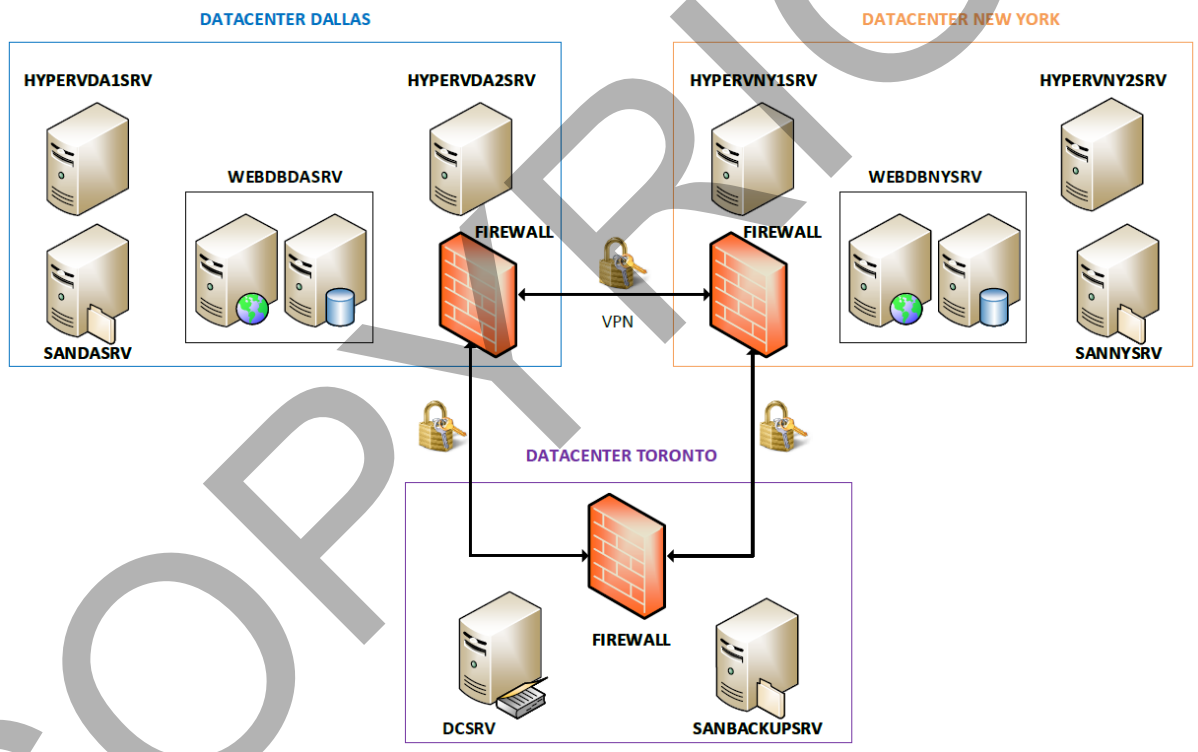


Figure 37. Fonctionnement du VPN inter-sites

4.5. Gestion du Content Delivery Network (CDN)

Le Content Delivery Network (CDN) est recommandé lorsqu'on met en place des sites internet. Son but : rediriger la requête du client vers un serveur le plus proche de chez lui lorsqu'il veut accéder à notre site internet. En quelque sorte, le serveur contient une copie en cache des éléments statiques (image, vidéo, texte…). Ce serveur communique avec le serveur dit « origine » qui contient, tous les éléments du site internet y compris ceux dynamiques (connexion à une base de données…). Dès qu'un changement au niveau statistique est modifié, le serveur « origine » envoie aux autres serveurs CDN répartis dans le monde, une copie de ses données. En faisant cela, nous permettons au client de se connecter plus rapidement au site internet et donc de gagner du temps.

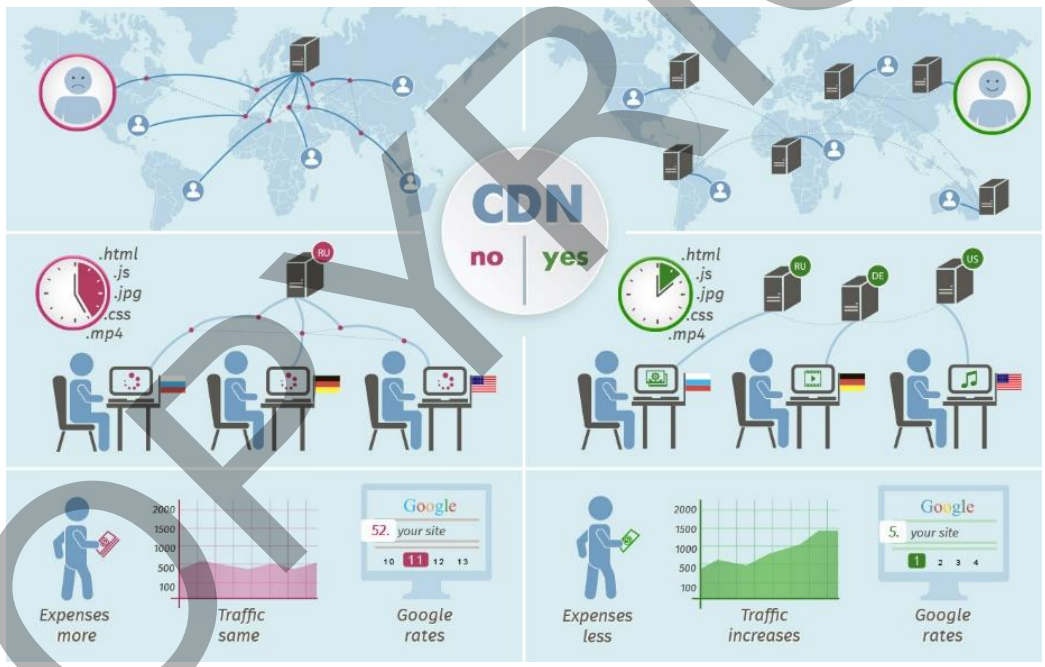


Figure 38. Différence entre une architecture avec et sans CDN

Une solution a été retenue, le choix de *Cloudflare* permettrait de mettre en cache tous le contenu statique du site afin de permettre aux utilisateurs un accès plus rapide à l'application. Pour ce faire, il faudrait que nos sites disposent chacun d'une IP publique que l'on rattacherait à notre solution *Cloudfare*. Cette solution se charge de réorienter toutes les requêtes vers le serveur le plus proche de l'utilisateur afin d'avoir le minimum de temps

d'accès à l'application. De plus, elle offre une fonctionnalité supplémentaire : le service de protection avancé Anti-DDOS.

Figure 39. Listes des serveurs CDN dans le monde

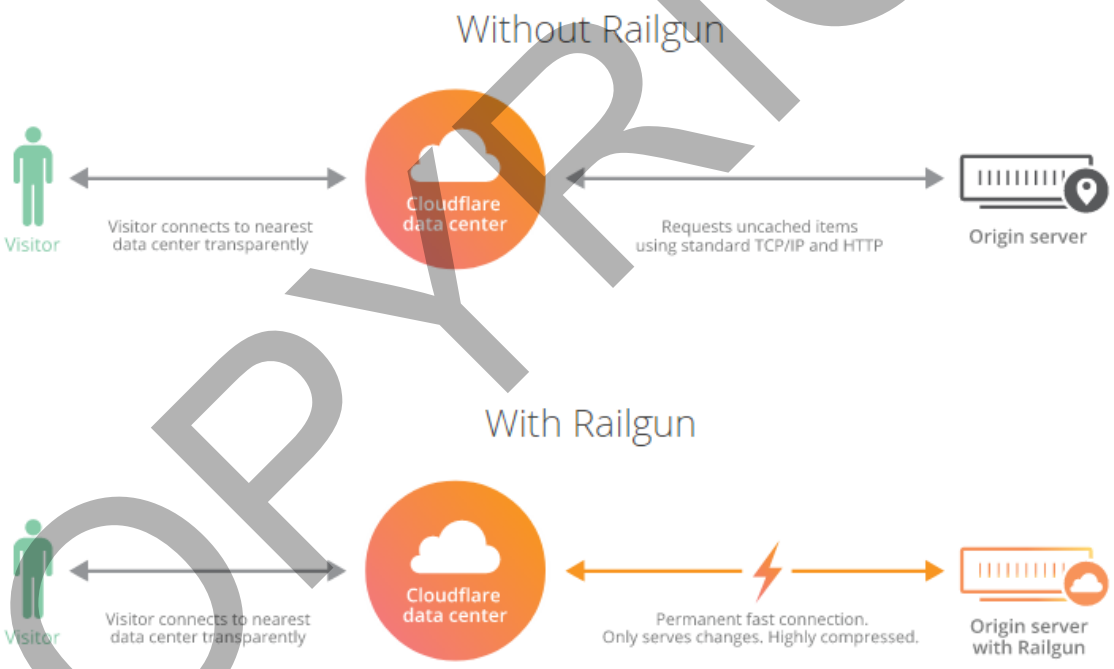


Figure 40. Fonctionnement du CDN de Cloudflare

Conclusion

Au travers de ce rapport, nous avons pu voir que mettre en place une infrastructure pour Qwirk++ pouvait paraître simple au premier abord, mais reste en fait assez complexe. En effet, il faut bien définir avec le développeur, les ressources auxquelles il a besoin. Dans notre cas, l'application tourne avec le Framework Meteor et une base de données MongoDB. Il était plus approprié de s'orienter vers une architecture Linux plutôt que Microsoft.

Le choix de la virtualisation de nos serveurs est primordial. D'un côté, cela permet d'économiser des ressources matérielles et d'un autre côté, nous pouvons gérer les pannes avec la mise en place de clusters à basculement et des baies de stockages SAN externes. Notons aussi l'importance de faire des sauvegardes des données vers un serveur de backup tous les jours.

La mise en place d'un pare-feu est important étant donné qu'il permet de rendre l'infrastructure plus résistante en cas d'attaque depuis l'extérieur.

Notons que cette infrastructure sera présentée le jour de la démonstration. Cependant, si nous avions eu plus de ressources (financières), nous nous serions orientés vers l'externalisation des serveurs (cloud), une solution moins coûteuse et plus sûre pour l'avenir.

Vous trouverez en annexe 3, le coût de l'architecture réseau (hors main d'œuvre).

Bibliographie

https://fr.wikipedia.org/wiki/Hyperviseur#/media/File:Hyperviseur.png

http://ahcdn.com/images/info-cdn-en.jpg

https://www.cloudflare.com/website-optimization/railgun/

https://www.freepixel.net/wp-content/uploads/2016/09/cloudflare-network-map.jpg

https://blog.netapsys.fr/wp-content/uploads/2016/06/meteor-logo.png

https://webassets.mongodb.com/_com_assets/cms/MongoDB-Logo-5c3a7405a85675366beb3a5ec4c032348c390b3f142f5e6dddf1d78e2df5cb5c.png

http://www.duchess-france.org/wp-content/uploads/2015/06/docker.png

Table des illustrations

Annexes

Annexe 1 : glossaire

NAT : Network Address Translation, traduction d'adresse réseau

IP : Internet Protocol, protocole utilisé pour les adresses IP.

LAN : Local Area Network, réseau local.

WAN : Wide Area Network, réseau Internet.

SAN : Storage Area Network, réseau de stockage.

ISCSI : Internet Small Computer System Interface, protocole de stockage en réseau basé sur le protocole IP.

OS : Operating System (Système d'exploitation), par exemple Microsoft Windows 10.

GPO : Group Policy Object, permet de configurer des restrictions d'utilisation de Windows ou des paramètres à appliquer, soit sur un ordinateur donné soit sur un compte utilisateur donné.

DNS : Domain Name System, permet de faire de pointer un nom vers une adresse IP.

URL : Uniform Resource Locator, adresse web.

DRBD : Distributed Replicated Block Device, périphérique en mode bloc répliqué et distribué.

POC : Proof Of Concept, preuve de concept (démonstration devant le client)

Annexe 2 : Lexique

Adresse IP : numéro unique permettant à un ordinateur de communiquer dans un réseau. Elle est représentée sous la forme suivante : xxx.xxx.xxx.xxx (avec xxx un nombre compris entre 0 et 255). Existe en norme IPv4 et IPv6.

RAID1 : Redundant Array of Independent Disks, un RAID1 correspond à deux disques en miroir.

Target : périphérique qui reçoit et traite les commandes.

Initiator : composant logiciel côté serveur, comportant un pilote permettant de gérer et transporter les blocs de commandes sur le réseau IP.

Multi-path : permet à plusieurs PC d'accéder à la même ressource en même temps.

Machine virtuelle : illusion d'un appareil informatique créée par un logiciel d'émulation.

Cluster : grappe de serveurs sur un réseau.

Domaine : définit un ensemble de machines partageant des informations annuaire.

Annuaire Active Directory : permet de gérer les ressources du réseau : données utilisateur, imprimantes, serveurs, bases de données, groupe, ordinateurs et stratégies de sécurité.

Back-end : terme désignant un étage de sortie d'un logiciel devant produire un résultat.

Front-end : correspond à la partie visible et accessible par l'utilisateur.

Port logiciel : permet de recevoir ou d'émettre des informations.

Annexe 3 : Coût de l'hébergement de l'infrastructure physique chez le client vs l'hébergement sur le cloud (pour 3 ans)

Infrastructure physique chez le client :

Catégorie	Description	Quantité	PU HT	Remise	Total HT	Taxe	Total TTC
Serveur	DELL Serveur rack PowerEdge R730 | Good pour le DCSRV	1	6 315,00 €	1 926,08 €	4 388,92 €	20%	5 266,70 €
	DELL Serveur rack PowerEdge R730 | Better pour les 4 hyperviseurs	4	7 987,00 €	2 436,04 €	29 511,96 €	20%	35 414,35 €
Onduleur	Dell Smart-UPS 8000VA	3	4 685,52 €		14 056,56 €	20%	16 867,87 €
	Multiprise 3 prises pour onduleur	3	5,52 €		16,56 €	20%	19,87 €
	Windows Server 2016 CAL	8	39,06 €		312,49 €	20%	374,99 €
	Windows Server 2016 Stantard	8	784,00 €		6 272,00 €	20%	7 526,40 €
KVM	Alten CL1000	3	649,96 €		1 949,88 €	20%	2 339,86 €
RACK	Amoire Rack 42U Dell powerRack 4210	3	790,00 €		2 370,00 €	20%	2 844,00 €
Sécurité	pfSense SG-2440	3	488,00 €		1 464,00 €	20%	1 756,80 €
	Kaspersky Endpoint Security for Business - Advanced (10 postes) - Licence 1 an	3	790,00 €		2 370,00 €	20%	2 844,00 €
Stockage	DELL Powervault MD3420	3	11 841,00 €	5 091,63 €	30 431,37 €	20%	36 517,64 €
	Logiciel Acronis BACKUP 12.5 ADVANCED - 36 mois	1	1 359,00 €		1 359,00 €	20%	1 630,80 €
Switch	CISCO SMALL BUSINESS SG350XG-2F10	3	1 449,96 €		4 349,88 €	20%	5 219,86 €

Total HT	94 463,70 €
Total TTC	113 356,44 €

Cloud :

Catégorie	Description	Quantité	PU HT	Remise	Total HT	Taxe	Total TTC
Serveurs	OVH Serveur Enterprise SP-128 - 12 mois - Pour les 2 hyperviseurs	6	1 679,88 €		10 079,28 €	20%	12 095,14 €
	OVH Serveur Stockage FS-30T - 12 mois - Pour 1 SAN	3	1 679,88 €		5 039,64 €	20%	6 047,57 €
	OVH Serveur Infrastructure EG-16 - 12 mois - Pour le DCSRV	3	935,88 €		2 807,64 €	20%	3 369,17 €
OS	Windows Server 2016 - Standard Edition 1 CPU - 1 mois	144	20,00 €		2 880,00 €	20%	3 456,00 €
Sauvegarde	Logiciel Acronis BACKUP 12.5 ADVANCED + 250Go CLOUD - 36 mois	1	1 858,00 €		1 858,00 €	20%	2 229,60 €
CDN	CDN - 1 mois	36	9,99 €		359,64 €	20%	431,57 €

Total HT	23 024,20 €
Total TTC	27 629,04 €

Nous constatons qu'il y a une différence de 71 439,50 € HT soit **85 727,40€** TTC. Il est donc plus rentable de prendre l'hébergement sur le cloud.

Cette différence s'explique par le fait que tout ce qui est externe aux serveurs (onduleur, switch, pare-feu, électricité, climatisation, salle dédiée…) est directement géré par l'hébergeur.