„Sposoby klastrowania aplikacji webowych w oparciu o rozwiązania OpenSource”

Piotr Klimek

piko@piko.homelinux.net

Agenda

- Wstęp
- Po co to wszystko?
- Warstwa WWW
- Warstwa SQL
- Warstwa zasobów dyskowych
- Podsumowanie

Wstęp

- Ilość systemów typu LAMP (Linux, Apache, PHP, MySQL) rośnie z roku na rok, część z nich w początkowej fazie swojego rozwoju nie jest projektowana pod kątem dużych obciążeń, a co za tym idzie nie są one zoptymalizowane i nie wyczerpują w pełni możliwości drzemiących w środowiskach tego typu.
- Większość administratorów wcześniej czy później staje przed koniecznością zoptymalizowania swojego środowiska, aby było ono w stanie podołać zwiększającemu się obciążeniu.

Problem

Jest to zadanie trudne, gdyż najczęściej wymaga się od administratorów, aby zmiany które wprowadzają nie zakłócały pracy środowiska i nie wiązały się ze rewolucyjnymi zmianami w aplikacji. Ponadto przy odpowiednio dużym obciążeniu, lub wymaganiach stawianych co do dostępności środowiska nie istnieje możliwość dokonania jego optymalizacji w oparciu o pojedynczą maszynę i należy wprowadzić rozwiązanie oparte o klaster serwerów.

Rozwiązanie

Rozwiązaniem problemów z wydajnością i awaryjnością jest stworzenie klastra serwerów dla aplikacji webowej, który będzie ukierunkowany na wydajność, skalowalność oraz zapewnienie bezawaryjnej pracy całego środowiska niezależnie od tego, czy wszystkie maszyny wchodzące w jego skład działają, czy nie.

Duży nacisk powinien być położony na wyeliminowanie pojedynczych punktów awarii (single point of failure - SPOF), dlatego też wszystkie krytyczne punkty klastra powinny być redundantne. Ponadto klaster powinien być w granicach rozsądku zaprojektowany pod kątem zmniejszenia kosztów potrzebnych do jego budowy.

Podział środowiska

Problem (i rozwiązania) zostały podzielone na logiczne warstwy, z których składa się środowisko działania aplikacji webowej. Każda z warstw tworzy osobną logiczną całość, z której korzysta aplikacja.

Warstwy to:

* Sieć – LVS + VRRP
* SQL – MySQL+ Dual Master Replication
* Storage – NFS + DRBD

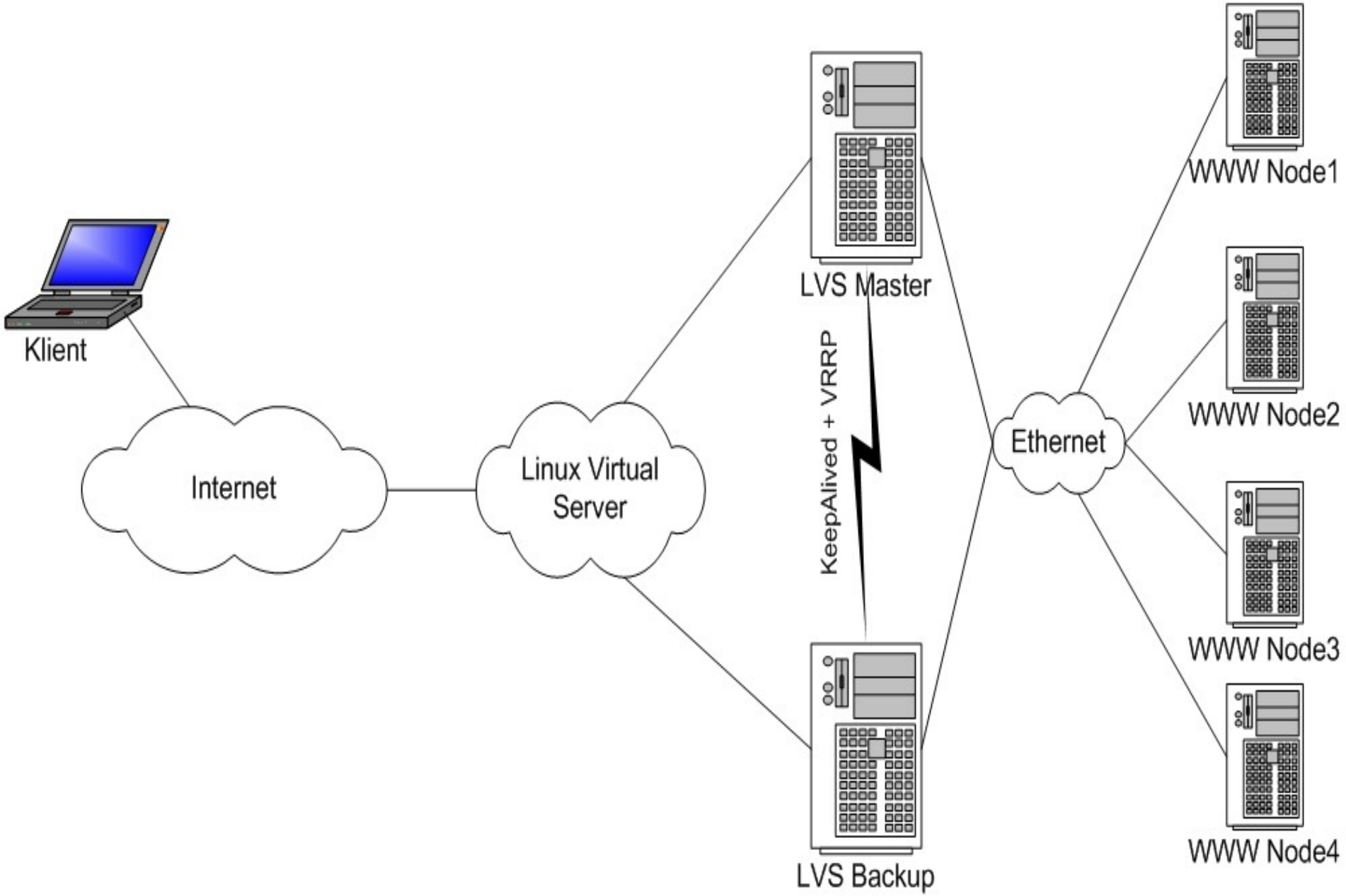
Każda z warstw ponadto jest skalowalna i odporna na awarie.

Warstwa sieciowa

Pierwszą z warstw jest sieć, do realizacji której użyty został projekt LVS. Pakiety pochodzące od klientów nie trafiają w takiej konfiguracji bezpośrednio do serwera aplikacji (dzięki odpowiednio ustawionemu routingowi/adresacji) lecz do serwera odpowiedzialnego za dalsze jego przekierowanie z uwzględnieniem szeregu warunków.

W tym miejscu, aby zapewnić bezawaryjne działanie klastra użyte są dwie maszyny, z których jedna odpowiada na żądania klientów, a druga jest dla niej zapasem (hot backup), który uaktywnia się tylko w przypadku awarii pierwszej. Ponadto maszyna odpowiedzialna za rozdzielanie ruchu pomiędzy poszczególnymi serwerami WWW jest w stanie przy pomocy odpowiednich algorytmów rozdzielać ruch w taki sposób jaki jest wymagany, by optymalnie wykorzystać posiadane zasoby.

LVS + KeepAlived

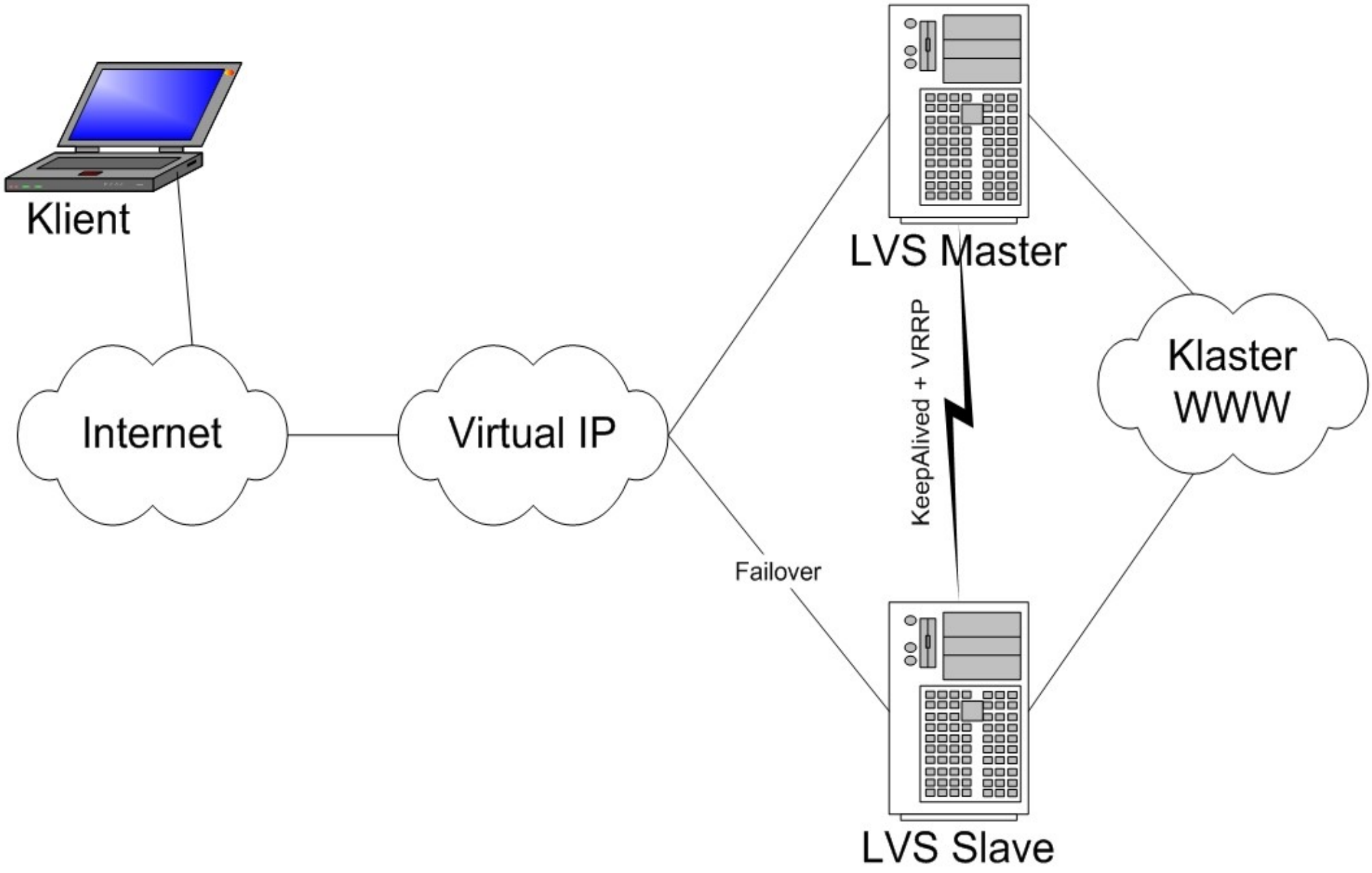


VRRP

Aby zapewnić redundancje kluczowych punktów klastra (SPOF) został użyty protokół VRRP (Virtual Router Redundancy Protocol), który sam w sobie pozwala na współdzielenie przez urządzenia adresów IP w trybie master-standby. Aby efektywnie z niego korzystać zastosowany został projekt KeepAlived, który zajmuje się badaniem dostępności drugiej maszyny w parze i w przypadku wykrycia awarii przejmuje jej adres IP.

Do administratora zaś należy napisanie odpowiednich skryptów, które pozwolą na przezroczyste przełączanie usług, tak aby zapewnić nieprzerwane działanie aplikacji. Tandem VRRP i KeepAlived został użyty w każdej z warstw by zapewnić nieprzerwany dostęp do najważniejszych usług.

KeepAlived + VRRP



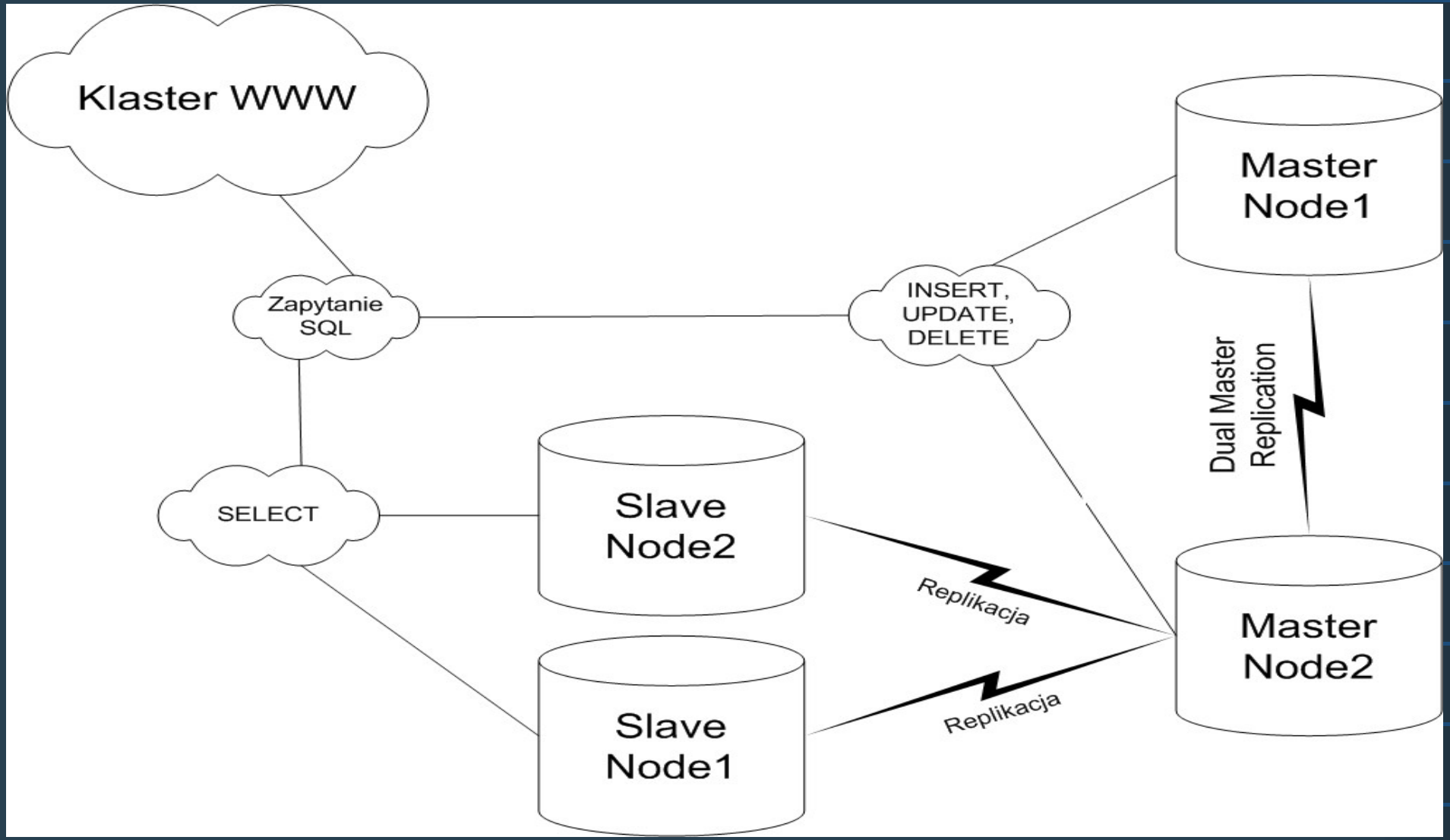
Warstwa SQL

Druga warstwa zapewnia komunikacje aplikacji z bazami danych opartymi o MySQL, w którym w celu zapewnienia niezawodności i aktualności danych użyta została technika replikacji zapytań pomiędzy serwerami.

Domyślna konfiguracja replikacji w MySQL opiera się o jedną maszynę typu master, do której trafiają zapytania modyfikujące zawartość danych (INSERT, UPDATE, DELETE), dopiero z tej maszyny dane są replikowane na serwery typu slave.

W moim projekcie, aby uniknąć awarii całego klastra w przypadku awarii serwera master znajdują się dwa takie serwery, technika ta nazywa się 'dual master replication' i polega na tym, że para serwerów pełni wobec siebie jednocześnie rolę master i slave, dzięki czemu zapytania modyfikujące dane mogą trafiać zarówno do jednego jak i drugiego serwera. Natomiast w przypadku awarii jednego z nich na drugim znajduje się aktualna kopia danych.

Dual Master Repliaction – schemat działania



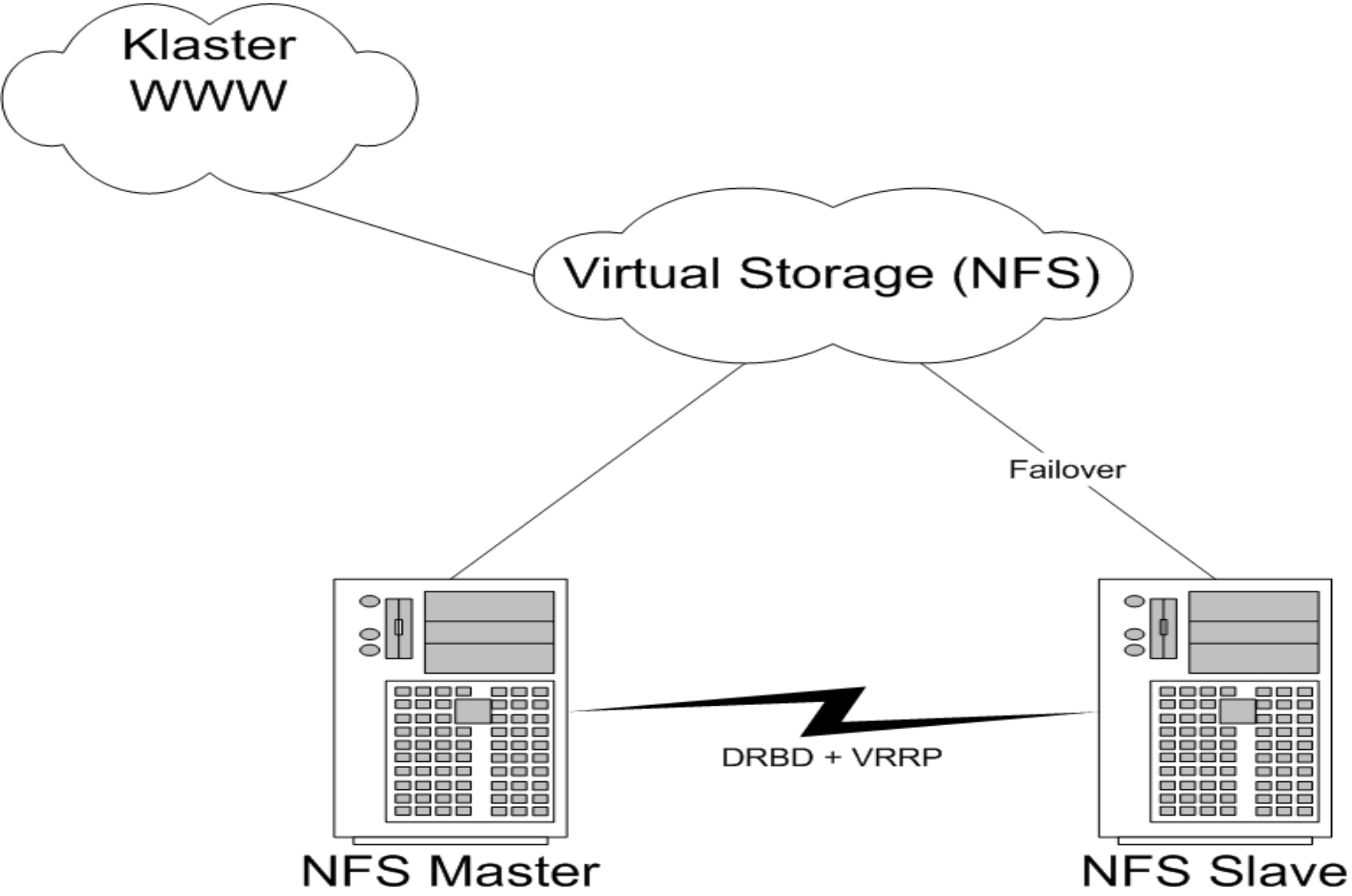
Warstwa zasobów dyskowych

Zadaniem trzeciej warstwy jest zapewnienie aplikacji dostępu do zasobów dyskowych, przy pomocy znanego od lat sieciowego systemu plików NFS, oraz dedykowanych serwerów dla niego. Aby uniknąć tworzenia SPOFa serwery NFS są zabezpieczone poprzez RAID over NET z użyciem rozwiązania DRBD.

Rozwiązanie to jest alternatywą dla bardzo drogiej dedykowanej macierzy. Obydwa serwery NFS realizują między swoimi systemami plików RAID, przy czym jeden z nich pracuje w trybie standby i jedynie replikuje zmiany pojawiające się na pierwszej maszynie.

W przypadku awarii tak samo jak we wcześniej opisanych warstwach następuje przezroczyste dla aplikacji przełączenie serwerów. W momencie, gdy pierwszy z serwerów wróci do macierzy po awarii replikowane są na niego dane, które zostały w czasie jego nieobecności zapisane na drugiej maszynie i przejmuje on ponownie rolę mastera.

NFS + DRBD

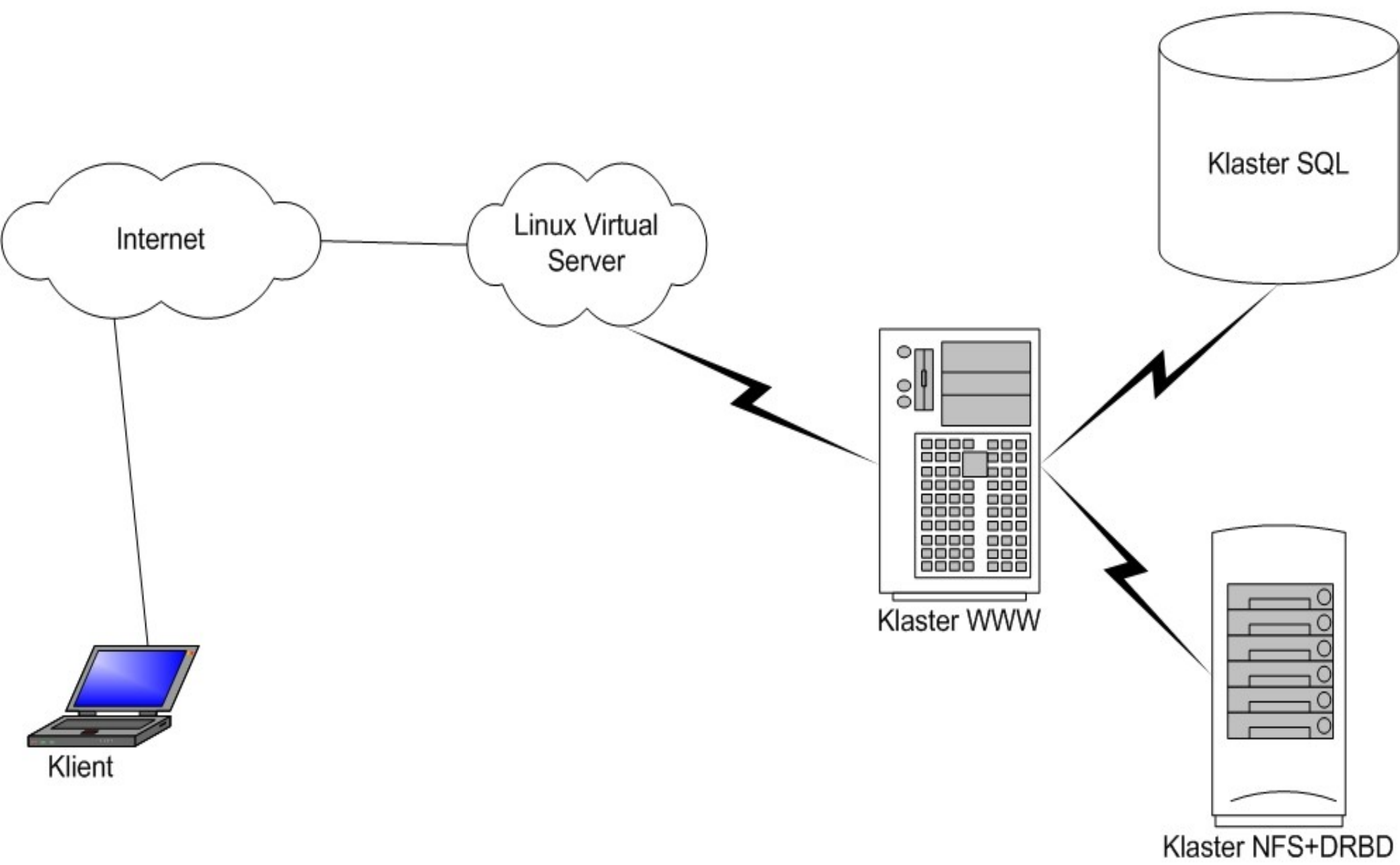

Klaster
WWW
Virtual Storage (NFS)
Failover
DRBD + VRRP
NFS Master
NFS Slave

Podsumowanie

W rezultacie relatywnie tanim kosztem otrzymujemy klaster, który od strony softwarowej jest odporny na awarie w kluczowych punktach. Celowo pominąłem aspekty zabezpieczeń sprzętowych takich jak macierze RAID, redundantne zasilanie, redundancje łącz, system monitoringu, system backupów. Nie wspomniałem też o nadzorowaniu rozwoju aplikacji, oraz jej bezpieczeństwie. Są to tematy do osobnych rozważań.

Przedstawione środowisko zapewnia dość dużą skalowalność, oraz może zostać zbudowane za równowartość macierzy sprzętowej dobrej klasy. W newralgicznych punktach takich jak serwery SQL pełniące role master, oraz serwery NFS należy liczyć się z koniecznością zakupu sprzętu z wyższej półki. Jednak wszystkie maszyny typu slave (WWW i SQL), oraz te które realizują LVS mogą być budowane na bazie komputerów, jakie każdy z nas posiada w domu, gdyż ich bezawaryjność, czy wydajność jest zapewniana poprzez ich ilość a nie klasę sprzętu.

Gotowy klaster dla aplikacji webowej




Pytania?