JUST DO IT
HÖHERE ABTEILUNG FÜR
INFORMATIONSTECHNOLOGIE

tgm
Die Schule der Technik

Ausarbeitung

Loadbalancing

DEZSYS - Dezentrale Systeme
5YHITM 2016/17

Maximilian Seidl

Version 0.3
Note:
Begonnen am 29. November 2016
Betreuer: Th.Micheler, M.Schabel
Beendet am 29. November 2016

Inhaltsverzeichnis

1 Was ist Load Balancing?

[1] Der Begriff Load Balancing (zu deutsch Lastverteilung) beschreibt umfangreiche Berechnungen oder große Mengen von Anfragen, welche auf mehrere parallel arbeitende System verteilt werden. Es gibt verschiedene Ausprägungen, welche mit dem Umfang der Kommunikation skalieren. Diese reichen von Lastverteilung auf einem System mit mehreren Prozessoren, bis hin zu ganzen Computer-Clustern, welche die Arbeit auf mehrere Rechner aufteilen.

2 Serverlastverteilung

Serverlasterverteilung (englisch Server Load Balancing "SLB") kommt zum Einsatz, wenn sehr viele Clients eine hohe Anfragedichte erzeugen und damit einen einzelnen Server-Rechner überlasten würden.

Typische Kriterien zur Ermittlung der Notwendigkeit von SLB sind die **Datenrate**, die **Anzahl der Clients** und die **Anfragerate**.

Die Zunahme der Datenverfügbarkeit wird durch SLB gefördert. Reduntate Datenhaltung wird durch den Einsetzen von mehreren Systemen ermöglicht. Die Aufgabe des SLB ist in diesem Fall die Vermittlung der Clients an die einzelnen Server.

3 Load Balancing Algorithmen

[2] Der eingehende Datenfluß eines Load Balancers kann mittels bestimmter Algorithmen effizient aufgeteilt werden. Hierbei wird von primitiven und komplexen Algorithmen unterschieden.

- 3.1 Round Robin
- 3.2 Fastest
- 3.3 Least Connections
- 3.4 Observed
- 3.5 Predictive

3.1 Round Robin

Round Robin übergibt jede neue Verbindungsanforderung an den nächsten Server, der in seinem internen Array an der Reihe ist. Dieser Algorithmus arbeitet gut in den meisten Netzwerkkonfigurationen, dennoch ist der optimale Anwendungsgebrauch bei unterschiedlich schneller Hardware. Dies bedeutet, dass die Hardware mit der Load Balancing betrieben wird, unterschiedlich in der Verarbeitungsgeschwindigkeit, der Verbindungsgeschwindigkeit und dem Speicher sein sollte.

Folgende Zitate eines Entwicklers bei *DevCentral* beschreibt den Umfang von Round Robin in kurzer Form.

> *The system builds a standard circular queue and walks through it, sending one request to each machine before getting to the start of the queue and doing it again. While I've never seen the code (or actual load balancer code for any of these for that matter), we've all written this queue with the modulus function before. In school if nowhere else.*

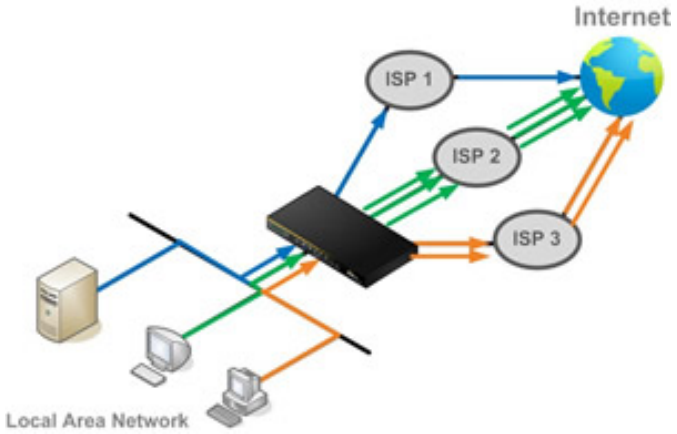

Abbildung 1: Round Robin Algorithmus

3.1.1 Weighted Round Robin

Bei dieser speziellen Art von Round Robin, ist die Anzahl der Verbindungen, die jede Maschine über die Zeit erhält, proportional zu einem Verhältnisgewicht. Diese Gewichtung kann für jede Maschine manuell eingerichtet werden. Es ist beispielsweise möglich die Last auf die Maschinen folgendermaßen aufzuteilen: **Maschine 3 kann die 2-fache Last von Maschine 1 und 2 übernehmen.**

3.1.2 Dynamic Round Robin

Ist ähnlich dem Weighted Round Robin, dennoch besteht neben der Gewichtung auch noch eine kontinuierliche Überwachung der Maschinen/Server. Diese Art von Load Balancing wird als dynamischer Algorithmus definiert, bei dem die Verbindungen abhängig von Echtzeit-Analysen aufgeteilt werden.

> *If you think of Weighted Round Robin where the circular queue is rebuilt with new (dynamic) weights whenever it has been fully traversed, you'll be dead-on.*

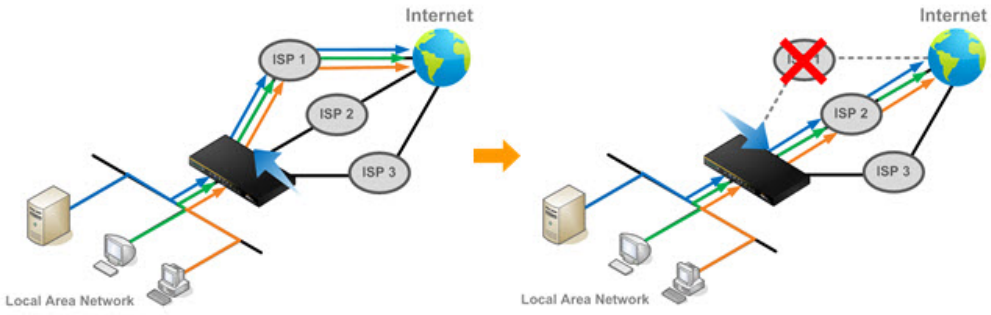

Abbildung 2: Dynamic Round Robin Algorithmus

3.2 Fastest

Die Methode Fastest übergibt die Verbindung basierend auf der schnellsten Reaktionszeit aller Server. Diese Methode kann besonders nützlich in Umgebungen sein, in denen Server über verschiedene **logische** Netzwerke verteilt sind.

> *The load balancer looks at the response time of each attached server and chooses the one with the best response time. This is pretty straight-forward, but can lead to congestion because response time right now won't necessarily be response time in 1 second or two seconds. Since connections are generally going through the load balancer, this algorithm is a lot easier to implement than you might think, as long as the numbers are kept up to date whenever a response comes through.*

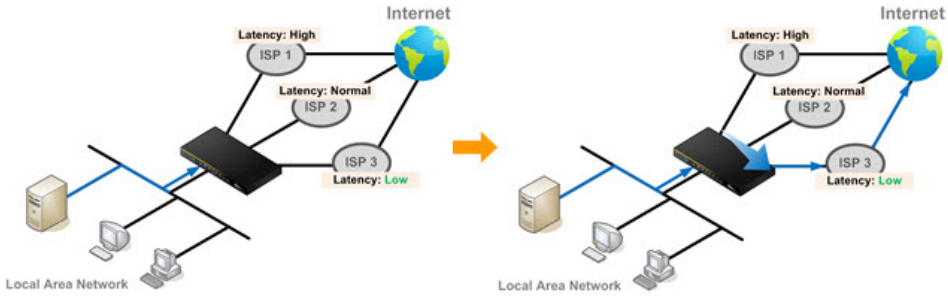

Abbildung 3: Fastest Algorithmus

3.3 Least Connections

Bei diesem Algorithmus werden neue Verbindungen zu dem Server weitergeleitet, welcher die wenigsten aktuellen Verbindungen hat. Am effizientesten arbeitet diese Methode in einem Netzwerk, bei dem alle Komponenten technisch eine gleiche Performance liefern. Es ist ebenfalls ein dynamischer Algorithmus, bei dem Echtzeit-Analysen das Verfahren beeinflussen können.

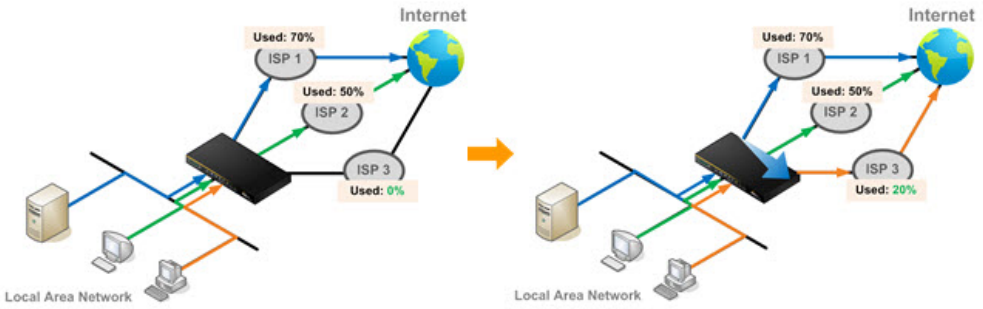

Abbildung 4: Least Connections Algorithmus

3.4 Observed

Das Observed-Verfahren verwendet eine Kombination aus der Logik, die in **Least** und **Fastest**-Connection Algorithmen enthalten sind, um die Verbindungen zu verteilen. In diesem Verfahren bekommen die Server ein *Ranking*, abhängig von der Response-Time sowie die Dauer der letzten Verbindungen. Lange Verbindungen könnten hier eine "unfaire" Einstufung eines Servers führen und sind somit ein großer Nachteil.

> *This algorithm tries to merge Fastest and Least Connections, which does make it more appealing than either one of the above than alone. In this case, an array is built with the information indicated and the element with the highest value is chosen to receive the connection. This somewhat counters the weaknesses of both of the original algorithms*

3.5 Predictive

Der Predictive Algorithmus verwendet das Ranking von **Observed**. Zusätzlich analysiert diese Methode auch noch das Verhalten der Server und bildet einen Trend, der das Ranking über die Zeit bestimmt. Server mit guter Performance befinden sich mit anderen Servern der selben Performance in einem Pool und bekommen somit mehr Verbindungen als andere Geräte. Diese prädiktive Methode lässt sich in jeder Architektur integrieren.

> *This method attempts to fix the one problem with Observed by watching what is happening with the server. If its response time has started going down, it is less likely to receive the packet. Again, no idea what the weightings are, but an array is built and the most desirable is chosen.*

4 Verfahren von SLB

Es werden einige verschiedene Verfahren geboten, welche bei Server Load Balancing zum Einsatz kommen können. Ein mögliches Verfahren wäre das Vorschalten eines Systems (**Load Balancer, Frontend Server**), der Anfragen aufteilt, oder die Verwendung von DNS (Domain-Name-System) mit dem sogenannten **Round-Robin-Verfahren**. Bei Webservern darf eine Serverlastverteilung nicht fehlen, da ein einzelner Host nur eine begrenzte Anzahl an HTTP-Anfragen auf einmal beantworten kann. Es besteht auch die Möglichkeit, dass der Load Balancer die Verschlüsselung zum Client übernimmt und intern die Anfragen auf seine Art und Weise verarbeitet.

Es stehen folgende Verfahren zur Verfügung:

- 4.1 DNS Round Robin
- 4.2 Flat based SLB
- 4.3 NAT based SLB
- 4.4 Anycast SLB

4.1 DNS Round Robin

[3]

Lastverteilung per DNS (englisch Round robin DNS) ist die einfachste Methode zur SLB auf Basis des DNS. Da es allerdings zu einem Caching der DNS-Antworten kommt, ist dieses Verfahren nur manchmal wirklich sinnvoll.

4.1.1 Aufbau und Funktionsweise

[4] DNS lässt es zu, dass einem Namen mehrere IP-Adressen zugewiesen werden können. Dies bedeutet, dass mehrere sogenannte **Resource Records** mit gleichem Label, gleicher Klasse und gleichem Typ entstehen. Eine deratige Anordnung wird als *Resource Record Set* bezeichnet.

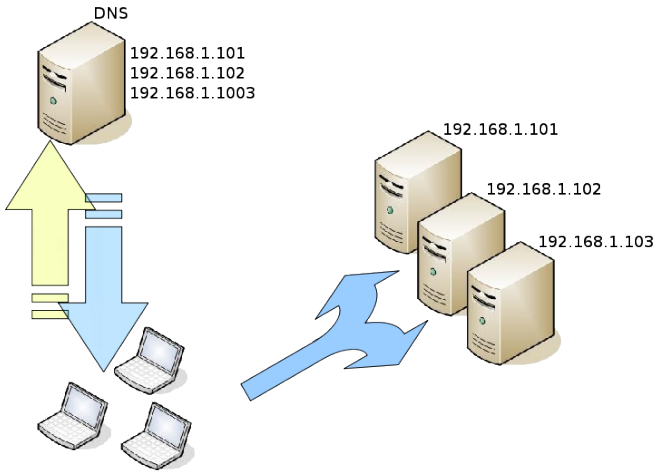

Abbildung 5: Grafik eines Aufbaus von DNS Round Robin

```
server.example.com.   1800  IN  A  192.168.1.101
server.example.com.   1800  IN  A  192.168.1.102
server.example.com.   1800  IN  A  192.168.1.103
```

Listing 1: DNS Round Robin Example

Wird ein derartiger Name von einem Resolver abgefragt, so liefert der DNS-Server grundsätzlich alle bekannten IP-Adressen zurück, allerdings in wechselnder Reihenfolge. Der erste Request wird dann beispielsweise mit [10.0.2.70, 10.0.2.71, 10.0.2.72] beantwortet und der zweite mit [10.0.2.71, 10.0.2.72, 10.0.2.70]. Der Resolver legt dann fest welche IP-Adresse verwendet werden soll.

Nach welcher Vorgangsweise ein DNS die Reihenfolge vorgibt, kann bei *Bind*-kompatiblen Nameservern konfiguriert werden. Bei **BIND** sind drei Varianten möglich: **zyklisch**, **zufällig** oder **fest**. Die Reihenfolge bei zyklisch und zufällig ist selbsterklärend, jedoch bei fest werden die IP-Adressen in der Reihenfolge zurückgegeben, in der sie im DNS abgelegt worden sind.

4.1.2 Technischer Fortschritt

[5] Mittels **SRV** (Service Resource Records) kann per DNS erkannt werden, welche IP-basierenden Dienste in einer Domain angeboten werden. So wird zu jedem Dienst weitere Information geliefert, wie zum Beispiel der Server-Name, der diesen Dienst bereitstellt.

```
_ldap._tcp.example.com. 3600 IN SRV 10 0 389 ldap01.example.com.
```

Listing 2: SRV - (Service) Resource Records

Bei **NAPTR**, ausgeschrieben *Naming Authority Pointer Resource Records* werden DNS-Namen und Adressen von Servern, sowie weitere Informationen zugeordnet. Diese Records liefern die zusätzliche Informationen auf flexible Art und Weise. Außerdem wird das Protokoll angegeben, das der Server verwendet. Falls mehrere NAPTR-Records zu einem Namen existieren, kann eine **Priorisierung** festgelegt werden. Auch wenn mehrere Records gleicher Priorität zu einem Namen existieren, kann eine Lastverteilung erreicht werden.

Bei moderneren Resource-Record-Typen lässt sich außerdem noch eine Gewichtung definieren, die festlegt, welche Server-IP-Adressen am häufigsten an erster Stelle stehen. Die entsprechenden Server werden damit häufiger angesprochen.

Außerdem gibt es die Möglichkeit, aus einem Pool von möglichen Servern nur einige zurückzuliefern. So werden beispielsweise vom Google-Nameserver immer drei IP-Adressen zurückgeliefert, die teilweise wechseln. Sinnvoll ist auch eine standortbezogene Rücklieferung von IP-Adressen, wenn mehrere verteilte Rechenzentren zur Verfügung stehen.

4.1.3 Nachteile

Die Lastverteilung durch DNS ist natürlich nur in dem Sinn gleichmäßig, was die Zuteilung betrifft. Über die danach entstehende tatsächliche Belastung weiß DNS nichts. Auch wird nicht überprüft, ob die Zielserver überhaupt ansprechbar sind. Vorgeschaltete Skripts können aber die Verfügbarkeit prüfen und nur diejenigen Server im Nameserver eintragen, die aktuell tatsächlich zur Verfügung stehen. Damit lassen sich Lastverteilung und Ausfallsicherung verbinden.

4.2 Flat based SLB

Bei diesem Verfahren wird nur ein Netzwerk benötigt. Die Server und der Load Balancer müssen über einen Switch miteinander verbunden sein. Sendet der Client eine Anfrage (an den Load Balancer), wird der entsprechende Ethernet-Frame so manipuliert, dass es eine direkte Anfrage des Clients an den Server darstellt.

4.2.1 Aufbau und Funktionsweise

Bei einer **flat based SLB** tauscht der Load Balancer bei Anfragen seine eigene **MAC-Adresse** gegen die des zu vermittelnden Servers aus und sendet dann das Paket weiter. Die **IP-Adressen** bleiben hierbei **unverändert**. Bei einem derartigen Vorgehen spricht man auch von **MAT (MAC Address Translation)**. Der Server, welcher die Anfrage bekommen hat, schickt seine Antwort direkt an die IP-Adresse des Absenders, also die des Clients.

Der Client hat damit den Eindruck, er kommuniziere nur mit einem einzigen Rechner, wobei es sich um den Load Balancer handelt. Der Server selbst kommuniziert nur mit einem einzigen Client. Dieses Verfahren wird als **DSR (Direct Server Return)** bezeichnet.

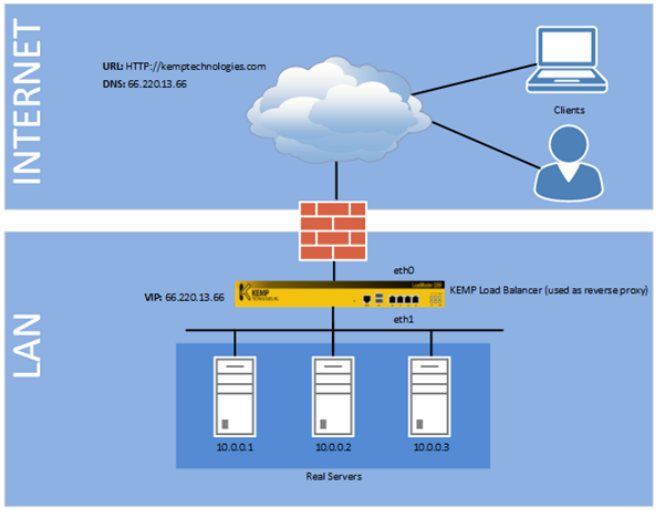

Abbildung 6: Flat based SLB Architektur

4.2.2 Vorteile

Der Vorteil bei Flat based SLB ist die Entlastung des Load Balancers, da der meist datenreiche Rückverkehr auf direktem Weg stattfindet.

4.3 NAT based SLB

[6] NAT basierte SLB-Netzwerkarchitekturen sind definitionsgemäß jene Verfahren, bei denen sich die IPs der **virtuellen IPs** und der **realen Server** in verschiedenen Subnets befinden. Diese Vorgänge heißen NAT, weil der Load Balancer NAT-Pakete zwischen zwei Subnets, wie eine **Firewall** oder ein **Router**, transferiert.

4.3.1 Implementierung

Der Hauptunterschied zwischen NAT- und Flat-basierten Architekturen besteht darin, dass der Load Balancer ein NAT zwischen zwei Netzwerken ausführt. Die wohl einfachste und typischste Art, ein NAT based SLB zu implementieren, ist eine **route-path, two-armed** Konfiguration. In der **nachfolgenden Abbildung** wird der Ablauf so einer Konfiguration beschrieben. Es zeigt wie der Load Balancer die normalen **Routing-IP-Adressen** in **nonrouted** IPs, auf denen die realen Server sitzen, übersetzt.

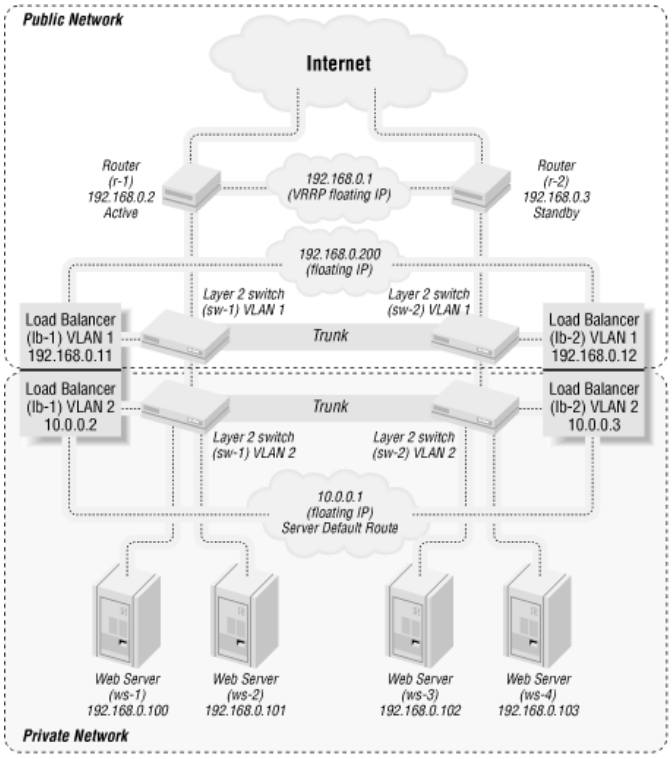

Abbildung 7: route-path, two-armed Architektur

In dieser Konfiguration befinden sich die Server in einem separaten VLAN, welche die VIP-Adressen (Virtuelle IP-Adressen) des Load Balancers besitzen. Die einzigen **floating IPs** befinden sich im *public* Netzwerk, welche zwischen den **aktiven** und **standby** Load Balancern angelegt sind. Ein Floating-Standard-Gateway wird im *public*-Bereich nicht benötigt, da die Load Balancer nicht die Rolle eines Gateways übernehmen. Das Floating-Gateway ist in das private Netzwerk integriert. Die Load Balancer übernehmen bei diesem Verfahren die Rolle einer **Firewall**, weil diese eine enge Kontrolle über den Netzwerkverkehr haben.

Außerdem gibt es noch eine andere Methode bei der sich alle Geräte nur **ein LAN** teilen. Die Load Balancer sind hierbei für mehrere Netzwerke im selben LAN konfiguriert und übernehmen somit das NAT. In der **folgenden Abbildung** wird der Aufbau eines **route-path, one-armed** SLB dargestellt.

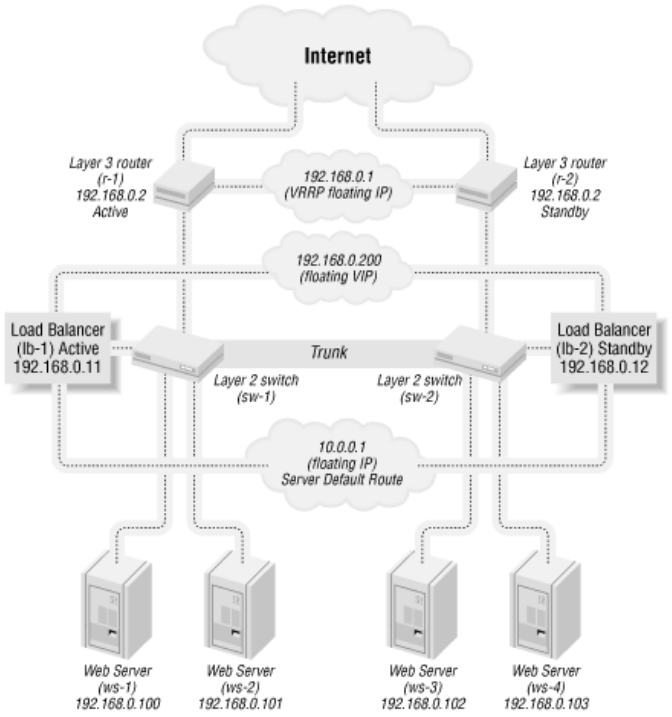

Abbildung 8: route-path, one-armed Architektur

Die Load Balancer sind in dieser Architektur für zwei Subnetze im gleichen LAN konfiguriert, einer für die öffentlichen Schnittstellen der virtuellen IPs und der andere für das private Subnet der Server. Obwohl sich alles noch in einem LAN befindet, übernimmt der Load Balancer das NAT.

Sowohl aus Sicherheitsgründen als auch aus architektonischer Sicht ist es **besser**, eine **two-armed** Architektur mit zwei seperaten LANs (oder VLANs) zu verwenden. Wenn alles in einem LAN untergebracht wird, werden Sicherheitsziele und Vorteile einer NAT-basierten Konfiguration vernachlässigt. Das tatsächliche Bestehen einer Schranke zwischen öffentlichem und privatem Netzwerk, verstärkt die Gesamtsicherheit eines Netzwerks. Der Datenfluss ist mit zwei VLANs einfach zu verwalten, da es klare Abgrenzungspunkte für die beiden getrennten Netze gibt, wodurch die Fehlersuche in den meisten Fällen extrem erleichtert wird.

4.3.2 Bridge-Path and Direct Server Return

Da NAT von einem Netzwerk zu einem anderen eine Layer 3 Funktion im OSI-Modell ist, ist die Variante **bridge-path** keine Option für NAT-basiertes SLB. Damit NAT funktioniert muss der Load Balancer Interfaces auf zwei Netzwerken haben, allerdings baut bridge-path normalerweise auf nur einem Netz auf.

DSR (Direct Server Return) tritt bei NAT-basierenden Szenarien, im Gegensatz zu flat-basierenden Architekturen nicht häufig auf, ist aber trotzdem möglich. Hierbei wird aber zusätzlich zum Load Balancer und einer Layer 2 Infrastruktur ein Layer 3 Gerät benötigt. Bei Direct Server Return werden Pakete bereits modifiziert versendet und das Layer 3 Element leitet diese einfach von einem Netzwerk zu einem anderen. Das Modifizieren benötigt zwar mehr Rechenleistung für die Server, ist aber eine Entlastung für den Load Balancer. In der **folgenden Abbildung** wird ein Beispiel einer solchen Konfiguration dargestellt.

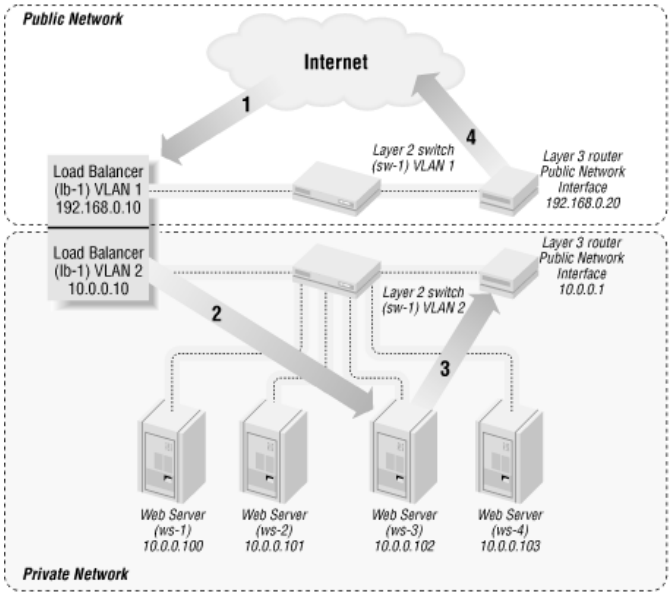

Abbildung 9: bridge-path NAT based SLB mit DSR

- **Schritt 1:**

  Ein Paket kommt zum Load Balancer

- **Schritt 2:**

  Es zu einem Webserver weitergeleitet

- **Schritt 3:**

  Der Webserver versendet das Paket bereits umgeschrieben

- **Schritt 4:**

  Es muss nun noch zum *public* Netzwerk weitergeleitet werden, damit es in das Internet gelangt. Das Layer 3 Element leitet somit das Paket unverändert an das öffentliche Netzwerk.

Hierbei ist die eigentliche Belastung des Load Balancers sehr gering, den die einzige Aufgabe besteht darin, die Pakete durch zu leiten ohne weiteren Aufwand.

4.3.3 Warum NAT-basierend?

Es gibt mehrere Vorteile für NAT-basierte SLB. Einer davon ist die extra Sicherheit, welche durch eine NAT-Struktur gewährleistet wird. Im Umgang mit Servern auf einem **nonrouting** IP-Adressen-Pool hat man eine bessere Kontrolle über die tatsächliche Sichtbarkeit nach außen, Besser gesagt: Wie der Server von außen gesehen wird.

Die NAT based SLB eignet sich gut für Websiten, bei denen die Mehrheit des Datenverkehrs HTTP (oder SSL) ist. Mit der zusätzlichen Sicherheit der NAT-IPs und der relativ niedrigen Abhängigkeit nach außen, bietet eine NAT based Architektur ein zusätzliches Maß an Sicherheit und Zuverlässigkeit.

4.4 Anycast SLB

[7] Bei der Lastverteilung über **Anycast** wird eine ganze Gruppe von Rechnern/Servern über eine Adresse angesprochen. Es antwortet derjenige, der über die kürzeste Route erreichbar ist. Im Internet wird dieses mit **BGP (Border Gateway Protocol)** realisiert.

4.4.1 Anycast

[8] Anycast ist eine Addressierungsart in Computernetzen, bei der man über eine Adresse einen einzelnen Rechner aus einer ganzen Gruppe von Rechnern ansprechen kann. Es antwortet derjenige, der über die kürzeste Route erreichbar ist. Diese Technik befinden sich gemäß **OSI-Modell** in der **Vermittlungsschicht**.

Realisiert wird Anycast durch eine Verteilung mehrerer gleichartiger Server auf geografisch getrennte IP-Netze. In der Praxis wird oft auf jedem Kontinent oder in jedem Land einer Region mindestens ein Server installiert. Jeder dieser Rechner erhält dieselbe IP-Adresse und propagiert eine entsprechende Route über ein Routing-Protokoll (**BGP**). Bei Ausfall oder Unerreichbarkeit verschwindet die Route und alle folgenden Pakete werden zu einem anderen Server geleitet. Der gewünschte Service bleibt somit auch bei Ausfall eines Servers verfügbar. Damit erhöht sich die Verfügbarkeit und die Ausfallsicherheit. Zur Administration muss ein Server direkt angesprochen werden können. Anycast-Server besitzen daher in fast allen Fällen zusätzlich eine eigene **Unicast-Adresse**.

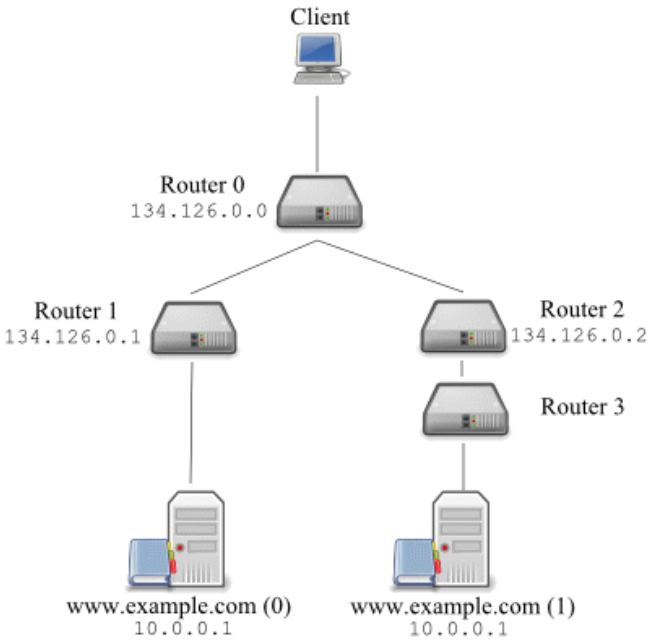

Abbildung 10: Beispiel eines Netzwerks aufgebaut mit Anycast

4.4.2 BGP Border Gateway Protocol

[9] Dieses Protokoll wird hauptsächlich im Internet eingesetzt und verbindet **Autonome Systeme (AS)** miteinander. Ein AS kann man als eine große Anzahl von IP-Adressen, betrachtet als eine Einheit, ansehen. Das Protokoll verwendet für Routing-Entscheidungen sowohl strategische, wie auch technisch-metrische Kriterien, wobei in der Praxis zusätzlich betriebswirtschaftliche Aspekte berücksichtigt werden.

4.4.3 Vorteile

Der Vorteil bei diesem Verfahren liegt in der geographisch nahe Auswahl eines Servers mit entsprechender Verringerung der Latenz (Ping). Die Umsetzung erfordert allerdings die Instandhaltung eines eigenen Autonomen Systems

Literatur

[1] Lastverteilung (informatik), definition und aufbau. 2016. https://de.wikipedia.org/wiki/Lastverteilung_(Informatik),.

[2] Don MacVittie. Beschreibung der algorithmen, mit grafiken. 2016. https://devcentral.f5.com/articles/intro-to-load-balancing-for-developers-ndash-the-algorithms.

[3] Lastverteilung per dns, beschreibung der funktion und umsetzung. 2016. https://de.wikipedia.org/wiki/Lastverteilung_per_DNS,.

[4] Thomas Sanchez. Round robin dns. 2015. http://www.webopedia.com/TERM/R/Round_Robin_DNS.htm,.

[5] Definition und funktionsweise von srv. 2016. https://de.wikipedia.org/wiki/SRV_Resource_Record,.

[6] Tony Bourke. *Server Load Balancing*. O´REILLY, 2001. https://web.archive.org/web/20040721090101/http://www.oreilly.com/catalog/serverload/chapter/ch07.html.

[7] Autonome systeme, mit definition und beschreibung. 2016. https://de.wikipedia.org/wiki/Autonomes_System.

[8] Anycast, adressierungsmethode mit beschreibung. 2016. https://de.wikipedia.org/wiki/Anycast.

[9] Border gateway protokoll, definition aufbau und beschreibung. 2016. https://de.wikipedia.org/wiki/Border_Gateway_Protocol,.

Listings

Abbildungsverzeichnis

