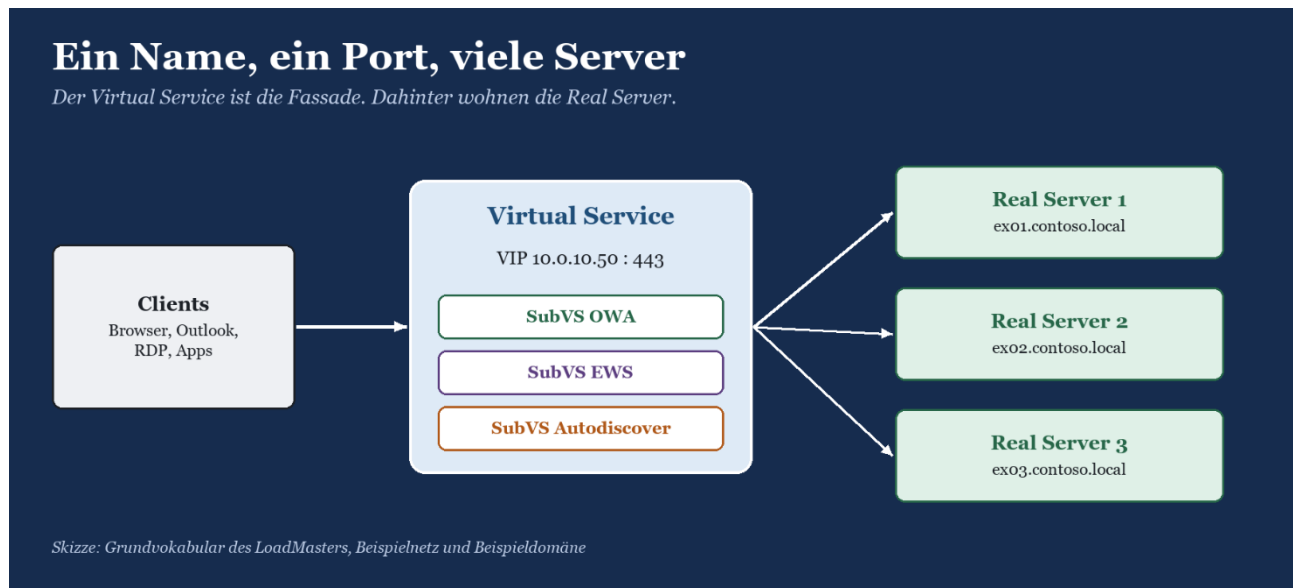


Virtual Service, Real Server, SubVS: das Grundvokabular des LoadMasters



WISSEN

Alles zum Kemp LoadMaster mit Microsoft-Brille: Exchange, SharePoint, ADFS, RDS und Kerberos an einem Ort.

› [Kemp LoadMaster](#)

BERATUNG

Du planst einen neuen LoadMaster oder willst einen gewachsenen aufräumen? Wir schauen gemeinsam auf Netz, Services und Rückweg.

› [Beratung zum Kemp LoadMaster](#)

SCHULUNG

Grundlagen und Praxis für dein Admin-Team: vom ersten Virtual Service bis zur Kerberos-Delegation.

› [Kemp-LoadMaster-Schulung](#)

Jeder LoadMaster beginnt sein Leben gleich: Jemand klickt auf „Add New“, tippt eine IP-Adresse und einen Port ein, hängt zwei Server dahinter und freut sich, dass OWA lädt. Drei Wochen später wird ein Server gepatcht, die Hälfte der Outlook-Clients fragt nach dem Kennwort, und im IIS-Log stehen nur noch Anfragen von einer einzigen Adresse. An diesem Punkt merkst du, dass der Load Balancer kein Verteilerkasten mit Weboberfläche ist, sondern ein Netzwerkgerät mit eigener Meinung über Pakete.

Dieser Beitrag ist der Einstieg in die Serie und klärt das Grundvokabular: Was ein Kemp virtual service eigentlich ist, was ein Real Server und ein SubVS davon unterscheidet, warum Layer 4 und Layer 7 zwei völlig verschiedene Tiere sind und weshalb das Default Gateway auf deinen Windows-Servern über Leben und Tod einer Veröffentlichung entscheidet. Die Herstellerdokumentation erklärt dir, wo die Häkchen sitzen. Hier geht es darum, warum sie dort sitzen und was passiert, wenn du sie falsch setzt.

Den Überblick über alle Themen der Serie, von Health Checks über Kerberos-Delegation bis zur Migration von Windows NLB, findest du im Wissensbereich [Kemp LoadMaster: der Wissensbereich](#). Dort hängen auch die vertiefenden Beiträge, auf die dieser Artikel an den passenden Stellen verweist.

FAKTEN • Das Grundvokabular in einem Atemzug

Ein Virtual Service ist die Kombination aus virtueller IP-Adresse (VIP), Port und Protokoll. Clients

sprechen nur mit ihm.

Real Server sind die tatsächlichen Server dahinter, etwa deine Exchange- oder IIS-Knoten.

Ein SubVS hängt unter einem Virtual Service, teilt sich dessen VIP und Port und hat eigene Real Server, Health Checks und Persistenz.

Layer-4-Services sind beim LoadMaster immer transparent. Bei Layer 7 ist Transparenz eine Option, und genau dort entscheidet sich, ob der Rückweg über den LoadMaster laufen muss.

Virtual Service, Real Server, SubVS: wer hier wer ist

Die Begriffe klingen harmlos, und genau deshalb werden sie im Alltag durcheinandergeworfen.

„Der VS ist down“ kann heißen, dass die VIP nicht antwortet, dass alle Real Server aus der Verteilung gefallen sind oder dass ein einzelner SubVS seine Prüfung nicht besteht. Für die Fehlersuche ist das ein gewaltiger Unterschied.

Virtual Service: VIP plus Port plus Protokoll

Ein Virtual Service ist die Adresse, die du im DNS veröffentlichst. Er besteht aus einer virtuellen IP-Adresse, einem Port und dem Protokoll TCP oder UDP. 10.0.10.50 auf TCP 443 ist ein Virtual Service, 10.0.10.50 auf TCP 25 ist ein anderer, auch wenn beide dieselbe VIP benutzen. Der Name mail.contoso.de zeigt im DNS auf die VIP, und ab dort übernimmt der LoadMaster.

Wichtig für Windows-Admins: Die VIP gehört keinem Server. Sie lebt auf dem LoadMaster, im HA-Paar auf dem jeweils aktiven Knoten. Wer aus NLB-Zeiten gewohnt ist, dass die Cluster-IP auf jedem Knoten konfiguriert wird, darf diese Gewohnheit jetzt feierlich beerdigen. Auf den Real Servern hat die VIP nichts verloren, es sei denn, du baust bewusst Direct Server Return, und das ist ein Thema für Leute mit sehr guten Nerven.

Real Server: der, der die Arbeit macht

Ein Real Server ist ein Ziel aus IP-Adresse und Port, an das der LoadMaster Verbindungen weitergibt. Der Port muss nicht derselbe sein wie am Virtual Service: Vorne 443, hinten 8443 ist völlig legitim. Jeder Real Server bekommt ein Gewicht, einen Zustand und eine Prüfung. Fällt die Prüfung durch, nimmt der LoadMaster den Server aus der Verteilung, ohne ihn zu fragen.

Standardmäßig erwartet der LoadMaster seine Real Server in einem direkt angeschlossenen Netz. Server hinter einem Router, sogenannte nicht-lokale Real Server, musst du erst global zulassen, und dann funktionieren sie nur ohne Transparenz. Das klingt nach einer Fußnote, ist aber in Umgebungen mit getrennten Server-VLANs der häufigste Grund, warum ein frisch angelegter Real Server sich weigert, überhaupt hinzugefügt zu werden.

SubVS: der Untermieter mit eigener Klingel

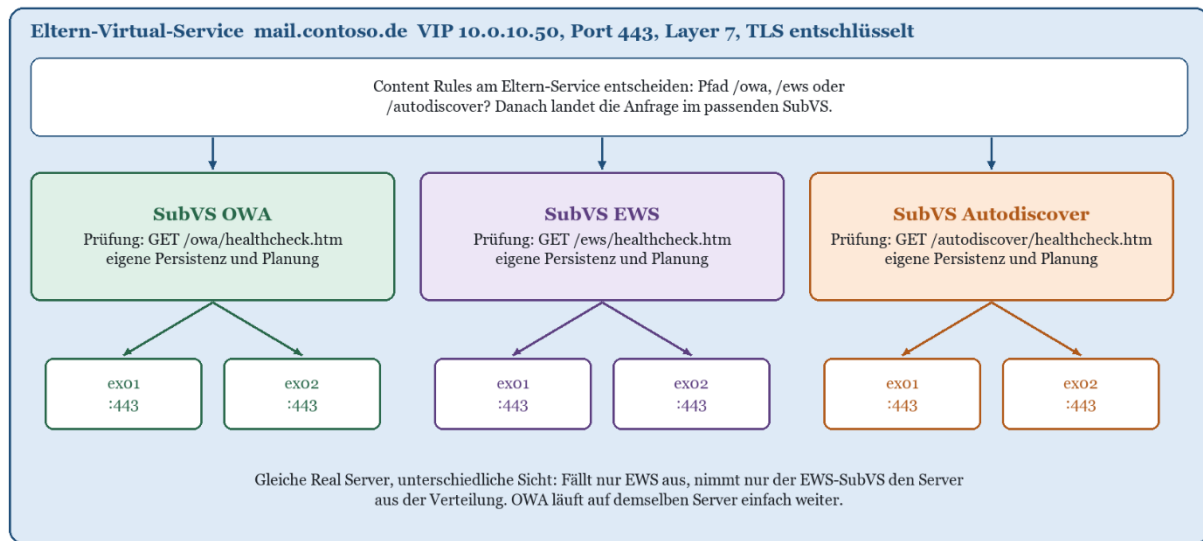
Ein SubVS ist ein Virtual Service im Virtual Service. Er wird innerhalb eines Eltern-Services angelegt, nutzt dessen VIP und Port und bekommt eigene Real Server, eigene Health Checks, eigene Persistenz und eigene Planungsmethode. Welche Anfrage in welchem SubVS landet, entscheiden Content Rules am Eltern-Service, typischerweise anhand des Pfads oder des Host-Headers.

Der Klassiker ist Exchange: Ein einziger Name, ein einziger Port 443, aber OWA, EWS, ActiveSync, MAPI und Autodiscover sind getrennte Anwendungen mit eigenen Application Pools. Fällt nur der EWS-Pool aus, soll nur EWS den Server meiden, während OWA dort fröhlich weiterläuft. Mit einem flachen Virtual Service und einer einzigen Prüfung auf /owa bekommst du das nicht hin. Die Exchange-Templates von Kemp bauen deshalb genau diese Struktur aus Eltern-Service und SubVS

je Dienst. Wie das im Detail aussieht, zeigt [Exchange Server hinter dem LoadMaster: Templates, Namespaces, Health Checks](#).

Ein Virtual Service, mehrere Untermieter

Der SubVS teilt sich VIP und Port mit dem Eltern-Service, hat aber eigene Server und Prüfungen



Skizze: SubVS-Struktur am Beispiel Exchange, Pfade und Namen beispielhaft

Damit Content Rules einen Pfad lesen können, muss der LoadMaster den Inhalt sehen. Bei HTTPS heißt das: TLS am Eltern-Service entschlüsseln. Ohne Offload sieht der LoadMaster nur verschlüsselten Datenstrom, und jede Pfadregel greift ins Leere. Was das für Zertifikate und Re-Encryption bedeutet, steht in [TLS-Offload, Re-Encryption und Zertifikate: PKI trifft LoadMaster](#).

Begriff	Was er ist	Typischer Denkfehler
Virtual Service	VIP plus Port plus Protokoll, der Einstiegspunkt für Clients	„Ein VS pro Anwendung.“ Tatsächlich ist es ein VS pro VIP und Port, die Anwendung kann mehrere brauchen, etwa HTTPS und SMTP.
VIP	Virtuelle IP-Adresse auf dem LoadMaster, im HA-Paar auf dem aktiven Knoten	„Die VIP muss auch auf den Servern konfiguriert sein.“ Nur bei Direct Server Return, sonst nie.
Real Server	Ziel aus IP und Port, an das Verbindungen gehen	„Gleicher Port wie vorne.“ Vorne und hinten dürfen unterschiedliche Ports haben.
SubVS	Untergeordneter Service mit eigener Serverliste und Prüfung unter gleicher VIP	„SubVS geht immer.“ Pfadbasierte Auswahl braucht Layer 7 und bei HTTPS entschlüsselten Verkehr.
Content Rule	Regel, die Anfragen anhand von Pfad, Header oder Host auswählt oder ändert	„Regeln wirken auch bei Layer 4.“ Ohne Layer 7 gibt es nichts zu lesen.
Health Check	Prüfung, ob ein Real Server Anfragen annehmen soll	„Port offen heißt gesund.“ Ein offener Port sagt nichts über die Anwendung.
Transparenz	Real Server sieht die echte Client-IP statt der LoadMaster-Adresse	„Schalte ich einfach ein.“ Dann muss der Rückweg zwingend über den LoadMaster laufen.

TIPP · Namen, die dir nachts um drei helfen

Benenne Virtual Services nach Dienst und Name, nicht nach Projekt: „Exchange HTTPS mail.contoso.de“ statt „Test neu 2“. SubVS nennst du nach dem virtuellen Verzeichnis. Wer beim

Störfall erst raten muss, welcher Service gemeint ist, hat schon verloren, nur noch nicht offiziell.

Layer 4 oder Layer 7: Paketschubser oder Proxy

Die wichtigste Entscheidung an einem Virtual Service ist nicht der Scheduling-Algorithmus, sondern die Schicht, auf der er arbeitet. Davon hängt ab, was der LoadMaster sehen kann, was er ändern darf und wie die Pakete zurückkommen müssen.

Layer 4: schnell, schlicht und immer transparent

Auf Layer 4 schaut der LoadMaster nur auf IP-Adressen und Ports. Er nimmt das Paket an, schreibt die Zieladresse von der VIP auf den gewählten Real Server um und schickt es weiter. Die Quelladresse bleibt die des Clients. Layer-4-Services sind beim LoadMaster immer transparent, das lässt sich nicht abschalten. Daraus folgt zwingend: Die Antwort des Servers muss wieder durch den LoadMaster, damit er die Absenderadresse zurück auf die VIP setzen kann.

Im Gegenzug kann Layer 4 keine Pfade lesen, keine Header einfügen, keine Cookies setzen und kein TLS entschlüsseln. Persistenz funktioniert nur anhand der Quell-IP. Für Protokolle, die kein HTTP sprechen, ist das oft genau richtig, für Webanwendungen meistens zu wenig.

Layer 7: zwei Verbindungen und ein neugieriger Vermittler

Auf Layer 7 ist der LoadMaster ein Reverse Proxy. Er nimmt die TCP-Verbindung des Clients an, liest bei Bedarf HTTP mit, wählt einen Real Server und baut zu diesem eine zweite, eigene Verbindung auf. Das ist die Grundlage für alles, was den LoadMaster für Microsoft-Workloads interessant macht: SubVS mit Pfadregeln, Cookie-Persistenz, Header wie X-Forwarded-For, TLS-Offload, Vorauthentifizierung mit dem Edge Security Pack und Kerberos-Delegation.

Weil der LoadMaster die zweite Verbindung selbst aufbaut, ist der Rückweg bei Layer 7 ohne Transparenz trivial: Der Server antwortet dem LoadMaster, denn der hat ihn ja angesprochen. Der Preis ist, dass im IIS-Log nicht mehr der Client steht, sondern eine Adresse des LoadMasters. Das ist kein Fehler, sondern Physik.

Layer 4 schiebt Pakete, Layer 7 führt Gespräche

Bei Layer 4 gibt es eine TCP-Verbindung, bei Layer 7 zwei

Layer 4



Layer 7



Skizze: Verbindungsmodell Layer 4 gegenüber Layer 7, Adressen beispielhaft

Eigenschaft	Layer 4	Layer 7
TCP-Verbindungen	eine, durchgereicht	zwei, Client zum LoadMaster und LoadMaster zum Server

Sicht auf Inhalte	nur IP und Port	HTTP-Header, Pfad, Host, Cookies (bei HTTPS nach Offload)
Transparenz	immer an	wählbar
Rückweg	muss über den LoadMaster	ohne Transparenz automatisch über den LoadMaster
Persistenz	Quell-IP	zusätzlich Cookie- und Header-basierte Verfahren
SubVS mit Pfadregeln	nein	ja
Typische Einsätze	SMTP-Sonderfälle, RDP-Durchleitung, UDP	Exchange, SharePoint, ADFS, IIS

WARNUNG · Force L4 ist kein Turbo-Knopf

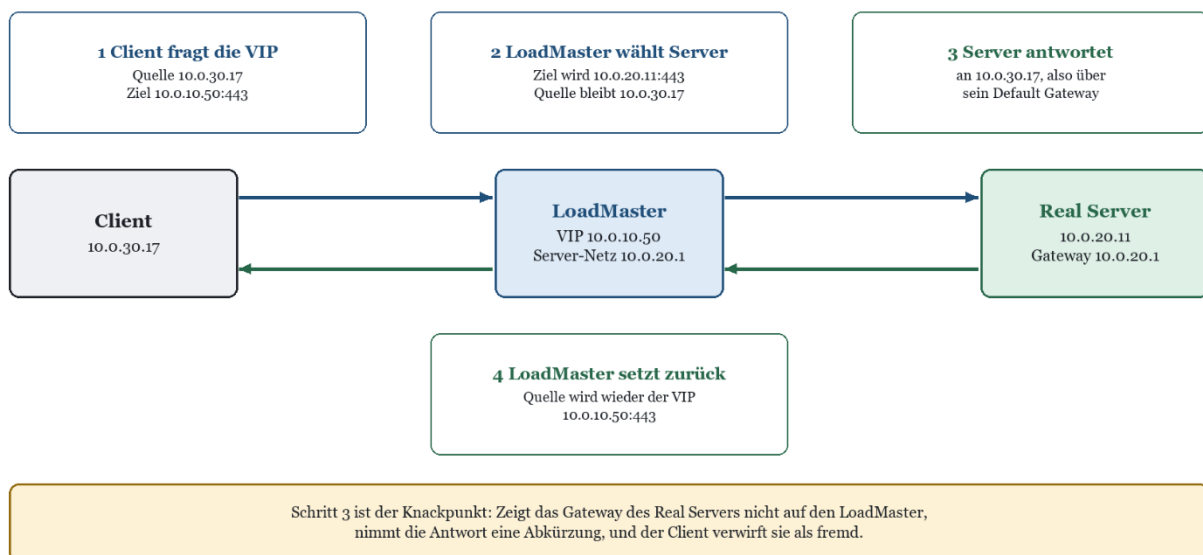
Die Option, einen Service auf Layer 4 zu zwingen, ist für Sonderfälle gedacht. Die Dokumentation selbst rät: Im Zweifel nicht anhängen. Wer sie setzt, weil „Layer 4 schneller ist“, verliert Pfadregeln, Cookie-Persistenz und Header, macht den Service automatisch transparent und hat damit plötzlich ein Routing-Problem, das vorher nicht existierte. Schneller ist dann vor allem der Anruf beim Helpdesk.

Der Datenfluss: hin ist leicht, zurück ist die Kunst

Fast jeder Fehler beim ersten Virtual Service hat dieselbe Wurzel: Der Hinweg wird geplant, der Rückweg wird angenommen. Dabei entscheidet TCP am Rückweg, ob eine Verbindung zustande kommt. Der Client hat sein SYN an die VIP geschickt und erwartet das SYN-ACK von genau dieser Adresse. Kommt die Antwort von irgendwoher sonst, ist sie für ihn Rauschen.

Hin und zurück: der Datenfluss durch den Virtual Service

Transparent konfiguriert: Der Real Server sieht die echte Client-Adresse



Skizze: Datenfluss Client, VIP, Real Server und Rückweg, Adressen beispielhaft

Die Skizze zeigt den sauberen Fall mit Transparenz. Der Client 10.0.30.17 spricht die VIP an. Der LoadMaster wählt einen Real Server und schreibt das Ziel um, die Quelle bleibt der Client. Der Real Server sieht also einen Fremden aus einem anderen Netz und schickt seine Antwort dorthin,

wohin er alles schickt, was er nicht kennt: an sein Default Gateway. Ist das der LoadMaster, setzt dieser die Absenderadresse zurück auf die VIP, und der Client ist glücklich.

Transparenz, SNAT und die Frage, wen der Server eigentlich sieht

Bei Layer 7 ohne Transparenz verwendet der LoadMaster als Quelladresse zum Real Server entweder die VIP oder, wenn die Option Subnet Originating Requests am Service aktiv ist, seine eigene Adresse im Netz des Real Servers. Die Dokumentation empfiehlt, diese Option pro Virtual Service einzuschalten, und im Two-Arm-Betrieb wirst du gleich sehen, warum. Ist Transparenz aktiv, hat Subnet Originating Requests keine Wirkung.

Eine Eigenheit, die regelmäßig für Verwirrung sorgt: Sitzt ein Client im selben Netz wie VIP und Real Server, schaltet der LoadMaster für diese Verbindung automatisch auf NAT, auch wenn Transparenz eingeschaltet ist. Sonst würde der Server direkt an den Nachbarn antworten, und die Verbindung käme nie zustande. Dein Test vom Admin-PC im Servernetz funktioniert also, die Anmeldung aus dem Clientnetz nicht. Willkommen in der Hölle der Tests, die das Falsche beweisen.

Konfiguration	Quelle am Real Server	Anforderung an den Rückweg	Wenn sie fehlt
Layer 4 (immer transparent)	echte Client-IP	Gateway des Servers ist der LoadMaster	Timeouts, halboffene Verbindungen
Layer 7 mit Transparenz	echte Client-IP	Gateway des Servers ist der LoadMaster, Client nicht im Servernetz	Timeouts aus fremden Netzen, lokal scheint alles zu gehen
Layer 7 ohne Transparenz, Subnet Originating Requests an	Adresse des LoadMasters im Servernetz	keine besondere, Server antwortet im eigenen Netz	nicht zutreffend, Client-IP nur per Header
Layer 7 ohne Transparenz, Subnet Originating Requests aus	VIP	Route vom Server zur VIP muss über den LoadMaster führen	sporadische Abbrüche, Firewall verwirft asymmetrischen Verkehr

FAKTEN · Was die Herstellerdokumentation für Transparenz verlangt

Der Real Server braucht den LoadMaster als Default Gateway, im HA-Paar die gemeinsame Adresse. Clients dürfen nicht im selben Subnetz wie der Real Server sitzen, sonst wird automatisch genattet.

Mit nicht-lokalen Real Servern hinter einem Router funktioniert Transparenz nicht.

Verkehr vom Eltern-Service zum SubVS ist immer transparent, vom SubVS zum Real Server optional. Transparenz und Re-Encryption schließen sich aus.

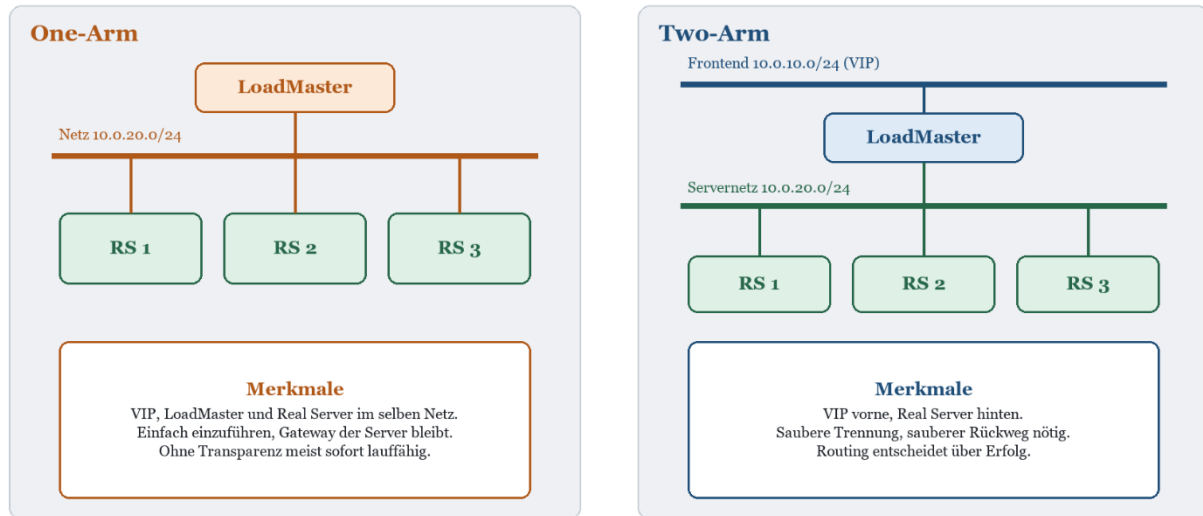
Brauchst du die echte Client-Adresse, etwa für Empfangsconnectors in Exchange oder für Auditing in einer Eigenentwicklung, gibt es bei HTTP den eleganteren Weg über einen Header wie X-Forwarded-For. Bei SMTP gibt es ihn nicht, und dort wird die Frage richtig unangenehm. Beide Fälle vertiefen [Client-IP sichtbar machen: Transparenz, SNAT und X-Forwarded-For](#) und [SMTP über den LoadMaster: Relay, Empfangsconnectors und die echte Absender-IP](#).

One-Arm, Two-Arm und das Default Gateway

Wie viele Beine der LoadMaster im Netz hat, ist keine Geschmacksfrage, sondern legt fest, welche der Kombinationen aus der Tabelle oben überhaupt funktionieren. Beide Varianten sind legitim. Gefährlich wird es erst, wenn jemand Two-Arm baut und One-Arm denkt.

One-Arm und Two-Arm

Ein Bein im Netz oder ein Bein in jedem Netz



Skizze: Netzlayouts One-Arm und Two-Arm, Adressbereiche beispielhaft

One-Arm: ein Kabel, wenig Drama

Beim One-Arm-Betrieb hängt der LoadMaster mit einer Schnittstelle im selben Netz wie die Real Server, die VIP liegt ebenfalls dort. Die Server behalten ihr bisheriges Default Gateway, der LoadMaster arbeitet bei Layer 7 ohne Transparenz, und der Rückweg ist automatisch korrekt, weil die Server einen Nachbarn im eigenen Netz antworten. Für den Einstieg und für viele Umgebungen im Mittelstand ist das die entspannteste Variante.

Der Haken: Layer-4-Services oder transparente Layer-7-Services funktionieren im One-Arm-Betrieb nur, wenn die Server trotzdem den LoadMaster als Gateway benutzen. Und dann routet der LoadMaster plötzlich den gesamten ausgehenden Verkehr deiner Exchange-Server, inklusive Updates und Domänencontroller-Kommunikation. Das will gut überlegt sein.

Two-Arm: sauber getrennt, sauber geroutet oder gar nicht

Beim Two-Arm-Betrieb hat der LoadMaster eine Schnittstelle im Frontend-Netz, in dem die VIPs liegen, und eine zweite im Servernetz. Das gefällt Netzwerkern und Sicherheitsleuten, weil Clients die Real Server nicht direkt erreichen müssen. Es funktioniert aber nur, wenn der Rückweg wieder über das zweite Bein läuft.

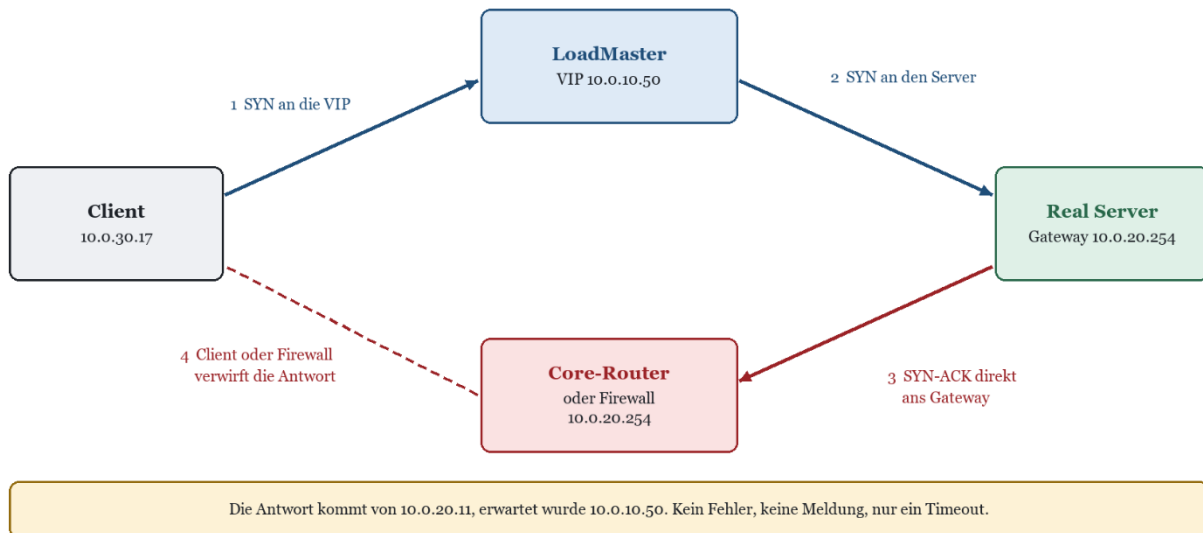
Kriterium	One-Arm	Two-Arm
Schnittstellen	eine, VIP und Server im selben Netz	zwei, VIP vorne, Server hinten
Default Gateway der Server	bleibt, solange ohne Transparenz	LoadMaster oder passende Routen
Layer 4 und Transparenz	nur mit Gateway auf dem LoadMaster	naheliegend, wenn Gateway passt
Typischer Fehler	Tests aus dem Servernetz beweisen nichts	Antwort umgeht den LoadMaster
Einführung	schnell, kaum Netzänderungen	Planung mit Netz- und Firewall-Team nötig

Warum Two-Arm ohne passendes Routing still scheitert

Jetzt kommt der Teil, der in keinem Wizard steht. Du baust Two-Arm, die Server stehen im Servernetz, ihr Default Gateway ist seit zehn Jahren der Core-Router. Du legst einen transparenten Service an, oder einen Layer-4-Service, der ohnehin transparent ist. Der Client schickt sein SYN an die VIP, der LoadMaster reicht es an den Server weiter, die Quelle ist der Client. Der Server antwortet an den Client, und zwar über sein Gateway, also am LoadMaster vorbei.

Two-Arm ohne passendes Routing: der stille Tod

Hinweg über den LoadMaster, Rückweg am LoadMaster vorbei



Skizze: asymmetrischer Rückweg bei transparentem Virtual Service, Adressen beispielhaft

Beim Client kommt nun ein SYN-ACK von 10.0.20.11 an, obwohl er mit 10.0.10.50 sprechen wollte. Er verwirft es. Liegt eine zustandsbehaftete Firewall dazwischen, verwirft sie es schon vorher, weil sie nie ein passendes SYN gesehen hat. Es gibt keine Fehlermeldung, kein Ereignis im Windows-Log, keinen roten Eintrag im LoadMaster. Der Health Check ist grün, denn der läuft vom LoadMaster aus und bekommt seine Antwort korrekt. Nur der Benutzer sieht eine Sanduhr, die sich dreht wie ein Bürokrat im Ruhestand.

Die nicht transparente Variante ist hinterhältiger. Mit Subnet Originating Requests ausgeschaltet spricht der LoadMaster den Server mit der VIP als Quelle an. Die VIP liegt im Frontend-Netz, der Server antwortet also über sein Gateway, und ob die Antwort beim LoadMaster ankommt, hängt davon ab, wie der Router das Frontend-Netz erreicht. Manchmal klappt das, manchmal verwirft eine Firewall den asymmetrischen Verkehr, manchmal nur unter Last. Genau diese Fehlerbilder beißen sich später in jedem Ticket fest.

AUS DER PRAXIS · Das Ticket „OWA geht nur manchmal“

Eine Verwaltung hatte ihren LoadMaster in Two-Arm-Bauweise vor zwei Exchange-Servern. Intern lief alles, von außen brachen Anmeldungen sporadisch ab. Die Health Checks waren grün, die Logs leer. Ursache: Ein Service war auf Layer 4 gezwungen worden, weil jemand Probleme mit großen Anhängen vermutete. Damit war er transparent, das Gateway der Server zeigte aber auf die Firewall. Interne Tests liefen aus dem Servernetz und wurden automatisch genattet. Die Lösung bestand aus einem entfernten Häkchen und einem Mitschnitt, der in fünf Minuten zeigte, was zwei Wochen lang vermutet worden war.

Default Gateway auf Windows-Real-Servern: was du wirklich prüfen musst

Windows kennt pro Schnittstelle ein Default Gateway, und bei mehreren Schnittstellen mit Gateway entscheidet die Metrik, welches gewinnt. Das ist der Klassiker bei Servern, die neben dem Produktivnetz noch ein Backup- oder Management-Netz haben: Die Antwort an den Client geht über die Schnittstelle mit der niedrigeren Metrik, und das ist nicht immer die, an die du gedacht hast.

- **Gateway prüfen:** Auf jedem Real Server die tatsächliche Standardroute ansehen, nicht die Doku. Mehrere Standardrouten sind ein Warnsignal.
- **HA beachten:** Zeigt das Gateway auf den LoadMaster, dann auf die gemeinsame Adresse des HA-Paars, nie auf die Adresse eines einzelnen Knotens. Sonst endet der Failover in der Stille.
- **Management-Netze per Route:** Wenn der LoadMaster Gateway wird, aber Verwaltungs- oder Backup-Netze direkt erreichbar bleiben sollen, gehören dafür gezielte statische Routen auf den Server.
- **Lieber ohne Transparenz:** Wenn du die echte Client-IP nur in Logs brauchst, ist Layer 7 ohne Transparenz mit Subnet Originating Requests plus X-Forwarded-For fast immer die robustere Wahl.

Zum Prüfen und Ergänzen reichen die Bordmittel aus dem NetTCPIP-Modul. Die Adressen sind Beispiele aus dem Netz der Skizzen:

```
Get-NetRoute -DestinationPrefix 0.0.0.0/0 | Format-Table ifIndex, InterfaceAlias,
NextHop, RouteMetric
Get-NetIPInterface -AddressFamily IPv4 | Sort-Object InterfaceMetric
New-NetRoute -DestinationPrefix 10.0.40.0/24 -InterfaceAlias "Ethernet" -NextHop
10.0.20.254
```

New-NetRoute legt die Route ohne weitere Angaben dauerhaft an, sie überlebt also den nächsten Neustart. Das ist gut, solange du sie dokumentierst, und schlecht, wenn dein Nachfolger in drei Jahren rätselt, warum ein einzelnes Netz einen Sonderweg nimmt.

WICHTIG · Gateway im HA-Paar

Steht ein HA-Paar vor deinen Servern, ist die gemeinsame Adresse im Servernetz das einzig richtige Gateway für transparente Services. Wie sich die Knoten die Adressen teilen und was beim Failover passiert, erklärt [HA-Paar, Failover und Konfig-Sync: zwei LoadMaster, ein Gedanke](#).

Der erste Virtual Service: Reihenfolge, die Ärger spart

Mit dem Vokabular im Kopf ist die Reihenfolge beim Anlegen eines neuen Services kein Geheimnis mehr. Sie ist nur unbequem, weil das Netz vor dem Klicken kommt.

- **Netzlayout festhalten:** One-Arm oder Two-Arm, welche Netze, welche Gateways, welche Firewalls dazwischen. Eine Skizze auf Papier reicht, solange sie existiert.
- **Schicht wählen:** Für HTTP-Anwendungen Layer 7. Layer 4 nur mit Begründung und mit geklärtem Rückweg.
- **Transparenz entscheiden:** Brauchst du die Client-IP wirklich auf Netzebene, oder reicht ein Header? Im Zweifel ohne Transparenz und mit Subnet Originating Requests.
- **Struktur planen:** Ein flacher Service oder Eltern-Service mit SubVS je Anwendung? Bei Exchange und SharePoint mit mehreren Diensten fast immer die zweite Variante.
- **Prüfungen definieren:** Was heißt gesund für diese Anwendung? Ein offener Port ist es nicht.

- **Persistenz festlegen:** Braucht die Anwendung überhaupt Sitzungsbindung, und wenn ja, woran?
- **Aus dem richtigen Netz testen:** Vom Clientnetz aus, nicht vom Admin-PC im Servernetz, und notfalls mit einem Mitschnitt auf dem LoadMaster.

Für die beiden Punkte Prüfung und Persistenz gibt es eigene Beiträge: [Health Checks, die wirklich prüfen: warum „Port 443 offen“ nicht reicht](#) erklärt, warum ein offener Port keine Gesundheit beweist, und [Persistence und Scheduling: wer hängt an welchem Server und warum](#) zeigt, wann Sitzungsbindung hilft und wann sie nur Last ungleich verteilt. Wenn ein Service trotz allem schweigt, hilft [Troubleshooting am LoadMaster: Logs, L7-Debug und TCP-Dump](#).

TIPP · Erst messen, dann glauben

Bevor du am Routing drehst: Ein kurzer Mitschnitt auf dem LoadMaster, gefiltert auf die Adresse eines Testclients, zeigt dir in Sekunden, ob das SYN-ACK zurückkommt. Kommt das SYN beim Server an, aber keine Antwort beim LoadMaster, ist es fast immer der Rückweg.

Wenn du das Netzlayout lieber einmal mit jemandem durchgehen willst, der diese Fehlerbilder schon oft gesehen hat, hilft dir die [Beratung zum Kemp LoadMaster](#). Wer das Grundvokabular mit dem ganzen Team praktisch üben will, findet in der [Kemp-LoadMaster-Schulung](#) ein Labor mit genau diesen Fallen.

FAQ zum Kemp Virtual Service

Was ist ein Kemp Virtual Service in einem Satz?

Die Kombination aus virtueller IP-Adresse, Port und Protokoll auf dem LoadMaster, die Clients anstelle der echten Server ansprechen.

Brauche ich für HTTPS und HTTP zwei Virtual Services?

Ja, denn es sind zwei Ports. Üblich ist ein Service auf Port 80, der nur auf HTTPS umleitet, und der eigentliche Service auf Port 443.

Wann nehme ich einen SubVS statt eines eigenen Virtual Service?

Wenn mehrere Anwendungen unter derselben VIP und demselben Port laufen, aber getrennt geprüft oder auf unterschiedliche Server verteilt werden sollen. Ein eigener Service lohnt sich, wenn die Anwendung einen eigenen Namen und eine eigene VIP bekommt.

Warum sehe ich im IIS-Log nur die Adresse des LoadMasters?

Weil der Service auf Layer 7 ohne Transparenz arbeitet. Das ist die robuste Standardvariante. Die echte Client-Adresse bekommst du über einen X-Forwarded-For-Header und ein passendes IIS-Logfeld.

Muss das Default Gateway der Real Server immer der LoadMaster sein?

Nein. Nur bei Layer-4-Services und transparenten Layer-7-Services. Bei Layer 7 ohne Transparenz und mit Subnet Originating Requests bleibt das bestehende Gateway der Server unangetastet.

Mein Health Check ist grün, aber Clients bekommen Timeouts. Wie kann das sein?

Der Health Check läuft vom LoadMaster aus und bekommt seine Antwort auf direktem Weg. Clientverkehr bei transparenten Services muss dagegen über das Gateway zurück. Ein grüner Check und tote Verbindungen sind das Lehrbuchsymptom eines falschen Rückwegs.

Kann ich Real Server in einem anderen Subnetz hinter einem Router nutzen?

Ja, wenn du nicht-lokale Real Server global zulässt. Transparenz steht dann für diesen Service nicht zur Verfügung.

Fazit: Erst der Rückweg, dann das Häkchen

Virtual Service, Real Server und SubVS sind schnell erklärt: Fassade, Arbeiter, Untermieter. Die eigentliche Arbeit steckt in den Fragen dahinter. Auf welcher Schicht arbeitet der Service? Sieht der Server den Client oder den LoadMaster? Und führt der Weg der Antwort zuverlässig dorthin zurück, wo die Frage herkam?

Wenn du diese drei Fragen vor dem ersten Klick beantwortest, ersparst du dir die unangenehmste Fehlerklasse im Load Balancing: die, die keine Fehlermeldung erzeugt. One-Arm ohne Transparenz ist der gutmütige Einstieg, Two-Arm ist sauberer, verlangt aber ein Routing, das du verstanden hast. Und Force L4 bleibt im Werkzeugkasten, bis du einen Grund hast, den du auch deinem Netzwerkteam erklären kannst. Pakete haben nämlich kein Mitleid. Benutzer übrigens auch nicht.

WEITERLESEN · Weiter im Thema

- › [Health Checks, die wirklich prüfen: warum „Port 443 offen“ nicht reicht](#)
- › [Persistence und Scheduling: wer hängt an welchem Server und warum](#)
- › [Exchange Server hinter dem LoadMaster: Templates, Namespaces, Health Checks](#)
- › [TLS-Offload, Re-Encryption und Zertifikate: PKI trifft LoadMaster](#)
- › [Client-IP sichtbar machen: Transparenz, SNAT und X-Forwarded-For](#)
- › [HA-Paar, Failover und Konfig-Sync: zwei LoadMaster, ein Gedanke](#)
- › [Content Rules: Header umschreiben, Pfade umleiten, Dinge verstecken](#)
- › [Troubleshooting am LoadMaster: Logs, L7-Debug und TCP-Dump](#)
- › [Migration von Windows NLB, IIS ARR oder TMG-Erbe auf den LoadMaster](#)