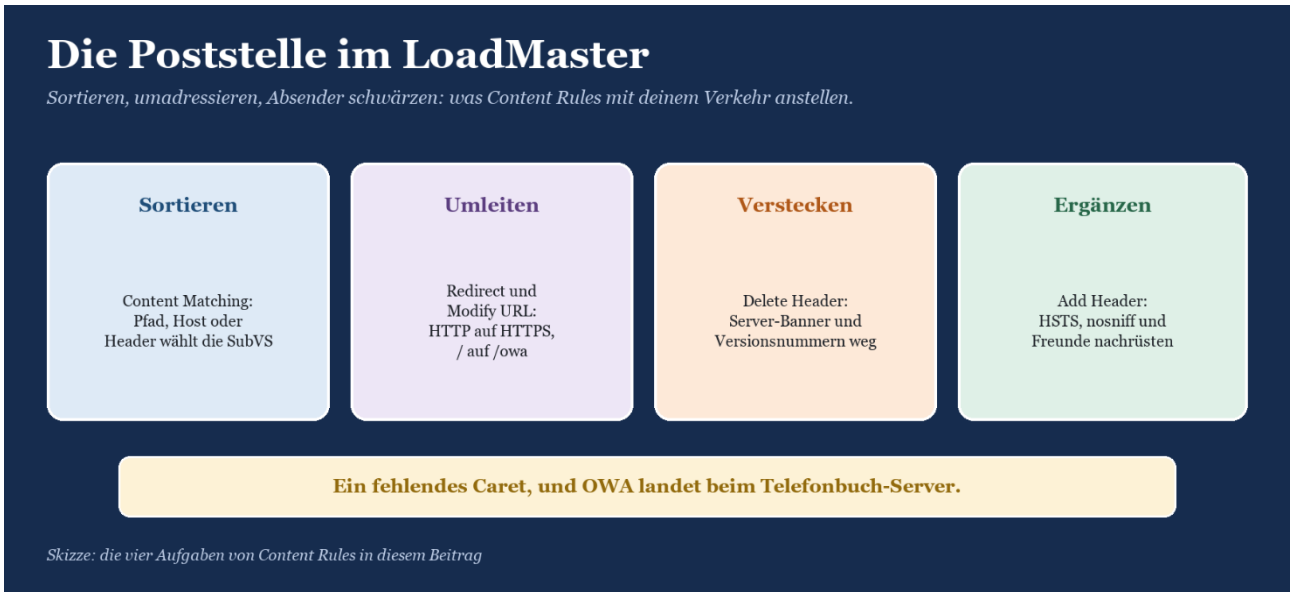
Content Rules: Header umschreiben, Pfade umleiten, Dinge verstecken



WISSEN

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

› **Kemp LoadMaster**

BERATUNG

Content Rules, die seit drei Firmware-Generationen niemand mehr versteht? Wir räumen auf und dokumentieren, was wirklich gebraucht wird.

› **Beratung zum Kemp LoadMaster**

SCHULUNG

Regex, Flags und Rangfolge im Labor üben, statt am produktiven OWA herauszufinden, was ein fehlendes Caret anrichtet.

› **Kemp-LoadMaster-Schulung**

Jede LoadMaster-Installation, die älter als zwei Jahre ist, hat sie: eine Liste von Content Rules mit Namen wie „test2“, „owaNEU“ und „bitteNichtAnfassenKlaus“. Niemand weiß genau, was sie tun, aber alle wissen, dass beim letzten Aufräumversuch das Telefon geklingelt hat. Content Rules sind das Schweizer Taschenmesser des LoadMasters, und wie beim echten Taschenmesser schneidet man sich am häufigsten mit der Klinge, die man für harmlos hielt.

Dabei steckt hinter den Content Rules kein Hexenwerk, sondern ein überschaubares Baukastensystem: Regeln, die einen Request anhand von Pfad, Host, Header, Quelladresse oder Body einer SubVS zuordnen, und Regeln, die Header auf dem Hin- oder Rückweg hinzufügen, ersetzen oder löschen. Dazu ein paar Flags, mit denen du Regeln verketten kannst, und eine Rangfolge, die festlegt, wer zuerst zum Zug kommt. Genau an diesen drei Stellen, Regex, Flags und Reihenfolge, knallt es in der Praxis.

In diesem Beitrag geht es um die Aufgaben, die vor Exchange, SharePoint, ADFS und Eigenentwicklungen wirklich anfallen: Pfade an die richtige SubVS verteilen, HTTP auf HTTPS umleiten, die nackte Domäne nach /owa schicken, Server-Banner entfernen und Security-Header nachrüsten. Er gehört zur Serie rund um den Kemp LoadMaster und erzählt nicht die Herstellerdoku nach, sondern erklärt, warum die Dinge so sind und welche Microsoft-Eigenheiten dir dabei in die Quere kommen.

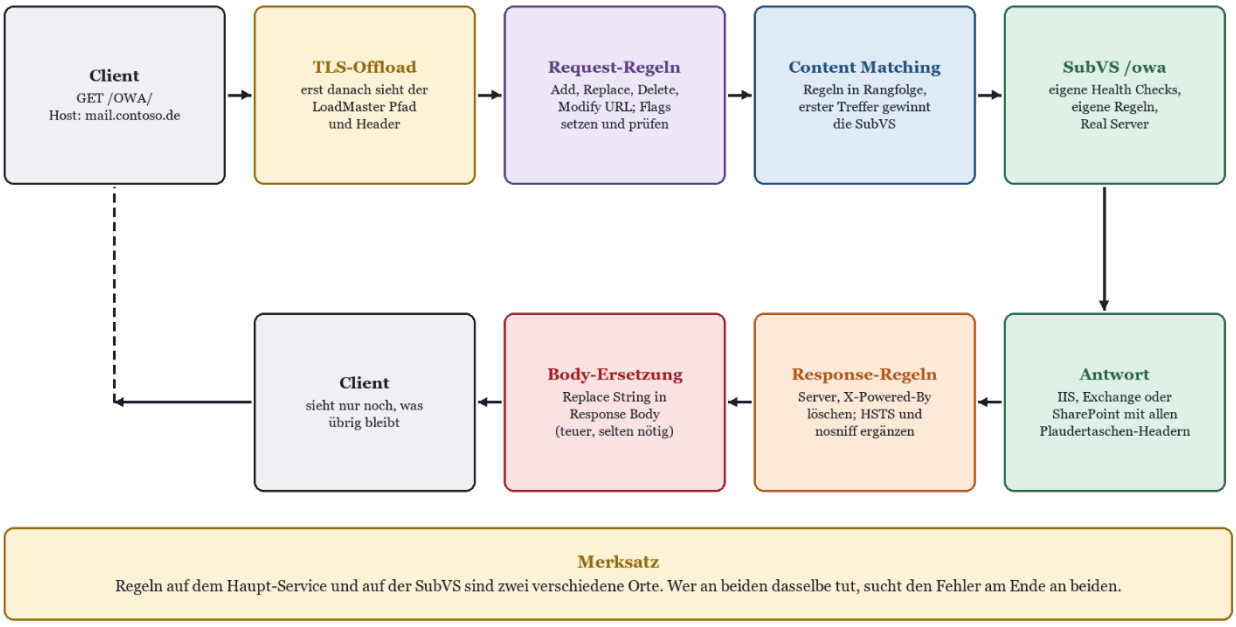
Der Baukasten: was Content Rules können und wo sie wirken

Content Rules legst du zentral an, unabhängig von einem Virtual Service. Erst die Zuordnung entscheidet, wo sie wirken: als Content-Matching-Regel an einem Real Server oder einer SubVS, als Request- oder Response-Regel bei den Header-Modifikationen eines Virtual Service. Eine Regel kann an vielen Stellen hängen. Die Übersicht der Content Rules zeigt dir in der Spalte „In Use“, ob und wie oft eine Regel zugeordnet ist. Ein Stern bedeutet: nirgends benutzt. Das ist der erste Ort, an dem du beim Aufräumen nachsiehst.

Für Regelnamen gelten ein paar harte Grenzen, die man spätestens beim Import über die REST-API bemerkt: nur alphanumerische Zeichen, keine Leerzeichen, eindeutig, und der Name „default“ ist reserviert. Groß- und Kleinschreibung unterscheidet der LoadMaster, „Rule1“ und „rule1“ dürfen also nebeneinander existieren. Das darfst du, solltest du aber nie tun, es sei denn, du willst deinem Nachfolger oder deiner Nachfolgerin einen sehr langen Nachmittag schenken.

Wo welche Regel greift: der Weg eines Requests durch den Virtual Service

Hinweg oben, Rückweg unten; Flags aus dem Hinweg sind auf dem Rückweg noch sichtbar



Skizze: vereinfachtes Modell zur Orientierung, keine Firmware-Interna

Die sechs Regeltypen im Überblick

Regeltyp	Was er tut	Typischer Einsatz vor Microsoft-Diensten	Stolperstein
Content Matching	Prüft URL, Header, Quelladresse, Methode oder Body und wählt damit Real Server oder SubVS	Exchange-Pfade auf SubVS verteilen, ECP von außen sperren	Rangfolge: Der erste Treffer gewinnt
Add Header	Fügt einen Header hinzu	HSTS, X-Content-Type-Options, Kennungs-Header für das Backend	Doppelte Header, wenn das Backend ihn schon setzt
Delete Header	Entfernt einen Header, wenn Name und Wert zum Muster passen	Server, X-Powered-By, X-AspNet-Version, X-OWA-Version	Muster passt nicht vollständig, Header bleibt still stehen
Replace Header	Ersetzt den Wert eines Headers per Regex, mit Rückverweisen \1, \2	Location-Header von http auf https, Set-Cookie um Secure ergänzen	Header-Name ohne Doppelpunkt eintragen

Modify URL	Schreibt den Pfad im Request um, der Browser merkt nichts	Altlasten-Pfade einer Eigenentwicklung, Root intern umschreiben	Relative Links und Redirects des Backends
Replace String in Response Body	Ersetzt Text im Antwort-Body	Interne Hostnamen in Altanwendungen ersetzen	Kostet Leistung, scheitert an komprimierten Antworten

Was eine Content-Matching-Regel prüfen kann

Ohne Angabe im Feld „Header Field“ prüft eine Content-Matching-Regel die URL, genauer den Pfad. Trägst du einen Headernamen ein, etwa Host oder User-Agent, prüft sie dessen Wert. Dazu kommen drei Sonderwerte: „src-ip“ für die Quelladresse des Clients, „method“ für die HTTP-Methode und „body“ für den Request-Body, wobei der LoadMaster nur die ersten 50 KB liest. Die Optionen „Include Host in URL“ und „Include Query in URL“ erweitern den geprüften Text um Hostnamen beziehungsweise Query-String. „Negation“ dreht das Ergebnis um, „Ignore Case“ ignoriert Groß- und Kleinschreibung, und „Fail on Match“ lässt die Verbindung bei einem Treffer scheitern, auf Wunsch mit Fehlercode oder Fehlerseite.

FAKTEN · Content Rules in Zahlen

Match String: höchstens 250 Zeichen, Shell-Muster und PCRE sind erlaubt, Progress empfiehlt PCRE.

Regex-Schreibweise: zwischen Schrägstrichen, etwa /^\/owa.*/, Schrägstriche im Muster maskierst du mit \/.

Match Type: Regular Expression, Prefix oder Postfix; Prefix und Postfix erwarten einfachen Text, keine Regex.

Body-Prüfung mit Header Field „body“: nur die ersten 50 KB des Requests.

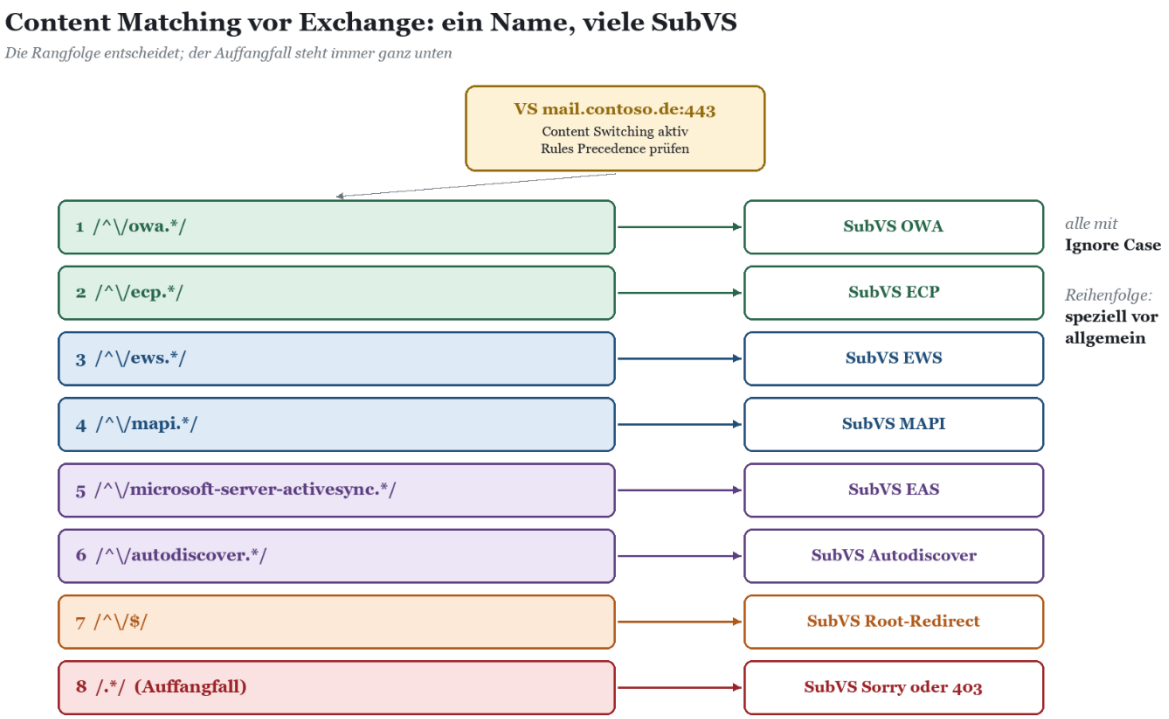
Flags: neun Stück, gesetzt per „Set Flag If Matched“, abgefragt per „Perform If Flag Set“ oder „Perform If Flag is NOT Set“. Im Request gesetzte Flags gelten auch für Response-Regeln.

Regeln duplizieren kannst du in der Oberfläche seit Firmware 7.2.52.

Content Matching: Pfade an die richtige SubVS

Der Klassiker ist Exchange: ein Name, ein Zertifikat, eine VIP, aber OWA, ECP, EWS, MAPI over HTTP, ActiveSync, Autodiscover und das Offline-Adressbuch verhalten sich völlig unterschiedlich. Sie brauchen eigene Health Checks, teils eigene Persistenz und eigene Sicherheitsregeln. Die Lösung heißt SubVS, und die Verteilung übernimmt Content Matching. Wenn dir die Begriffe Virtual Service, Real Server und SubVS noch nicht in Fleisch und Blut übergegangen sind, lohnt vorher ein Blick in Virtual Service, Real Server, SubVS: das Grundvokabular des LoadMasters.

Damit Content Matching überhaupt greift, aktivierst du in den erweiterten Eigenschaften des Virtual Service das Content Switching. Die Schaltfläche erscheint erst, wenn mindestens ein Real Server oder eine SubVS existiert. Danach weist du jeder SubVS ihre Regel zu. Die Kemp-Templates für Exchange erledigen genau das für dich, inklusive der passenden Regeln. Was die Templates im Detail anlegen und wo du nachschärfen solltest, steht in Exchange Server hinter dem LoadMaster: Templates, Namespaces, Health Checks.



Rangfolge: Der erste Treffer gewinnt

Sobald Regeln zugeordnet sind, zeigt dir die Schaltfläche „Rules Precedence“ die Reihenfolge, in der der LoadMaster sie prüft, und dort kannst du Regeln nach oben verschieben. Das ist kein Kosmetikdialog. Eine Regel /.*/ als Auffangfall ganz oben frisst jeden Request, und alle sorgfältig gebauten Pfadregeln darunter werden nie gefragt. Das Prinzip ist dasselbe wie bei Firewall-Regeln und Transportregeln in Exchange: speziell vor allgemein, Auffangfall nach unten.

Besonders tückisch wird es bei Pfaden, die sich gegenseitig enthalten. SharePoint mit /sites und /sites/projekte auf unterschiedlichen Backends, oder eine Eigenentwicklung mit /api und /api/v2. Steht die kurze Regel oben, landet die lange nie. Die Reihenfolge, in der du die Regeln angelegt hast, ist dabei kein verlässlicher Hinweis. Verlass dich auf die Precedence-Ansicht und auf einen Test, nicht auf dein Gedächtnis.

WARNUNG · Kein Treffer ist auch ein Ergebnis

Passt kein einziger Request auf eine Regel, verteilt der LoadMaster nicht einfach irgendwohin, der Client bekommt einen Fehler. Das ist oft gewollt, denn nicht jeder Pfad soll von außen erreichbar sein, aber es muss eine bewusste Entscheidung sein. Lege lieber eine eigene Auffang-SubVS an, die mit 403 oder einer freundlichen Hinweisseite antwortet, als dich auf das Standardverhalten zu verlassen, das beim nächsten Firmware-Update niemand mehr im Kopf hat.

Die Regex-Fallen, in die alle einmal treten

PCRE ist mächtig, und die Content-Rules-Oberfläche ist ehrlich: Sie macht genau das, was du hineinschreibst, nicht das, was du gemeint hast. Die folgenden Fallen sehen wir in fast jeder Bestandsaufnahme, meist mehrere gleichzeitig.

Ein Muster, viele Opfer: was /owa wirklich trifft

Links das naive Muster, rechts die verankerte Variante mit Ignore Case

Angefragter Pfad	/owa/ ohne Anker	/^\/owa(\/|$)/ + Ignore Case
/owa/	Treffer	Treffer
/OWA/	kein Treffer (Groß)	Treffer
/owa/auth/logon.aspx	Treffer	Treffer
/owapublic/telefonbuch	Treffer (falsch!)	kein Treffer
/api/flow?ziel=/owa	Treffer (falsch!)	kein Treffer
/owa?ReturnUrl=x	Treffer	Treffer

Achtung: Die Query zählt nur mit, wenn Include Query in URL aktiv ist. Ohne das Häkchen sieht die Regel bei der fünften Zeile nur /api/flow.

Skizze: Beispielpfade fiktiv; das Verhalten hängt zusätzlich an den Flags der Regel

Falle	Was passiert	Besser
Fehlender Anker	/owa/ trifft auch /owapublic oder einen Query-Parameter, der /owa enthält	/^\/owa(\/|$)/ mit Anker am Anfang
Groß- und Kleinschreibung	Outlook und Mobilgeräte schicken /OWA, /Autodiscover oder /Microsoft-Server-ActiveSync in wechselnder Schreibweise	Ignore Case bei allen Pfadregeln
Unmaskierte Punkte	/contoso.de/ im Host trifft auch contosoXde	Punkte als \. maskieren: /^mail\.contoso\.de$/
Gieriges .*	Rückverweise in Replace Header schnappen sich mehr als gedacht	Genauer formulieren, etwa [^/]* statt .*
Query vergessen	Regel auf ?ReturnUrl= greift nie, weil die Query nicht geprüft wird	Include Query in URL bewusst setzen
Host vergessen	Zwei Namen auf derselben VIP, Regel unterscheidet nur den Pfad	Include Host in URL oder eigene Regel auf Header Host
Prefix mit Regex	Bei Match Type Prefix wird ^/owa wörtlich gesucht	Prefix nur mit einfachem Text, sonst Regular Expression
Shell-Muster gemischt	Ein * meint in Shell und PCRE etwas anderes	Konsequent PCRE verwenden, wie von Progress empfohlen

TIPP · Regeln vorher außerhalb des LoadMasters testen

Jeder PCRE-fähige Tester taugt für einen ersten Check, auch die PowerShell: '/OWA/auth' -match '^/owa(/|$)' liefert True, weil -match in PowerShell standardmäßig Groß- und Kleinschreibung ignoriert. Das ist bequem, kann aber täuschen. Teste deshalb mit -cmatch gegen, wenn du wissen willst, wie sich die Regel ohne Ignore Case verhält. Und teste nicht nur die Pfade, die treffen sollen, sondern vor allem die, die nicht treffen dürfen.

Umleiten: HTTP auf HTTPS und die Wurzel auf /owa

Umleitungen sind die Content Rules, die am häufigsten falsch gebaut werden, weil zwei völlig verschiedene Dinge gleich klingen. Ein Redirect sagt dem Browser: Frag bitte woanders. Die Adresszeile ändert sich, Lesezeichen und Suchmaschinen lernen den neuen Ort. Ein Rewrite mit Modify URL ändert dagegen nur, was der Server sieht. Der Browser glaubt weiterhin, er habe / angefragt. Für den Server ist das ein anderer Request, für die Anwendung manchmal ein Rätsel.

HTTP auf HTTPS: ein Virtual Service ohne Real Server

Für die Umleitung von Port 80 auf 443 brauchst du keine Content Rule, sondern einen eigenen Virtual Service auf Port 80 mit derselben Adresse, der keine Real Server hat. Weil er nie einen verfügbaren Server findet, greift die Fehlerbehandlung „Not Available Redirection Handling“ in den erweiterten Eigenschaften: Fehlercode „302 Found“ und als Redirect URL https://%h%s. Dabei steht %h für den angefragten Host und %s für den Pfad. Am HTTPS-Service gibt es zusätzlich eine Komfortfunktion, die genau diesen Port-80-Redirector auf Knopfdruck anlegt. Danach verschwindet das Feld, also nicht wundern.

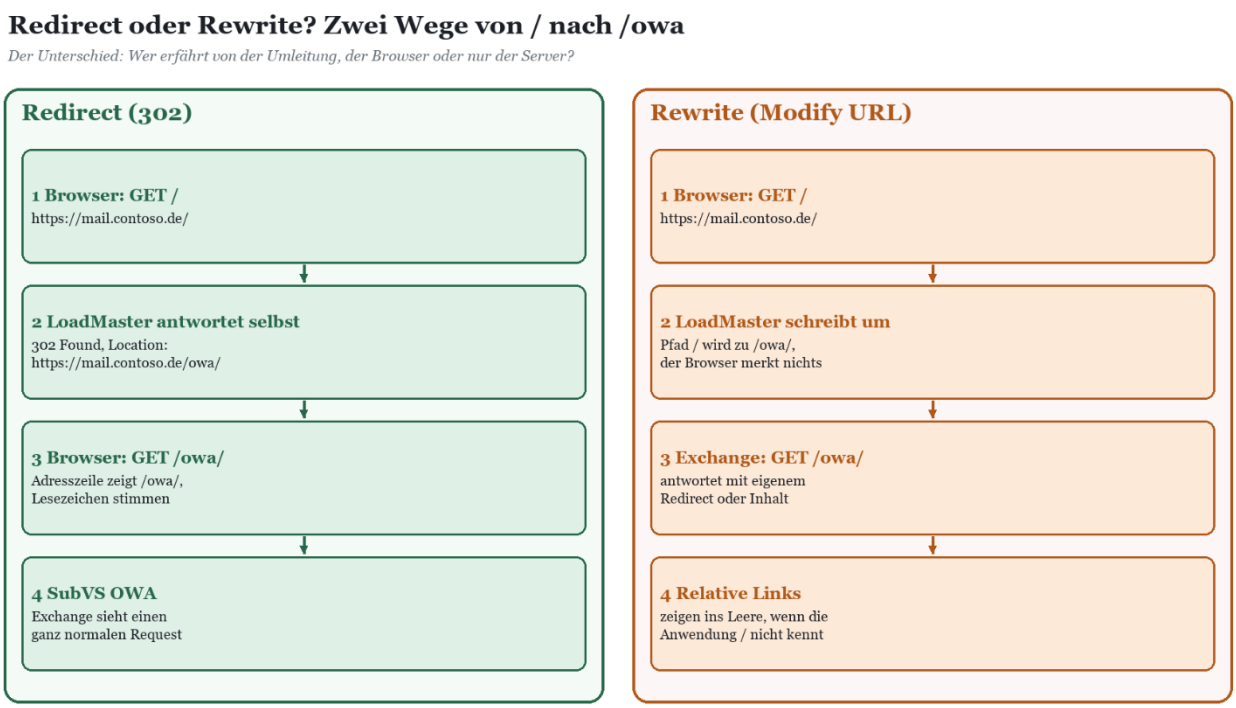
Zwei Microsoft-spezifische Hinweise dazu. Erstens: Leite nur um, was ein Browser anfragt. Autodiscover über HTTP-Redirect auf Port 80 ist ein eigener, von Outlook bewusst genutzter Mechanismus, und wer den Port-80-Service für den Autodiscover-Namen versehentlich mit 302 auf einen falschen Ort zeigen lässt, erzeugt Zertifikatswarnungen in Outlook, die monatelang niemand zuordnen kann. Zweitens: 302 ist vorübergehend, 301 ist dauerhaft und wird von Browsern hartnäckig zwischengespeichert. Für den Start ist 302 die gnädigere Wahl, weil sich ein Irrtum wieder einfangen lässt.

> **WICHTIG · Redirect auf Port 80 ersetzt kein HSTS**
>
> Der erste Request läuft beim Redirect weiterhin unverschlüsselt über die Leitung, inklusive Cookies, falls diese nicht als Secure markiert sind. Erst der Header Strict-Transport-Security sorgt dafür, dass der Browser beim nächsten Mal direkt HTTPS benutzt. Redirect und HSTS gehören deshalb zusammen. Wie Zertifikate, Offload und Re-Encryption dafür zusammenspielen, steht in TLS-Offload, Re-Encryption und Zertifikate: PKI trifft LoadMaster.

Die Wurzel auf /owa: Redirect statt Rewrite

Anwenderinnen und Anwender tippen mail.contoso.de und erwarten OWA. IIS auf dem Exchange-Server liefert für / meist nur die Standardseite oder einen Fehler, und eine HTTP-Weiterleitung im IIS ist bei jedem kumulativen Update ein Kandidat für Überraschungen. Auf dem LoadMaster baust du das sauberer: eine Content-Matching-Regel mit dem Muster /^\/$/, die ausschließlich den nackten Schrägstrich trifft, zeigt auf eine eigene kleine SubVS ohne Real Server. Deren Not Available Redirection Handling antwortet mit 302 und der Redirect URL https://%h/owa/. Der Browser landet sichtbar auf /owa/, alles andere bleibt unberührt.

Die Alternative mit Modify URL und demselben Muster funktioniert auf den ersten Blick auch, weil Exchange selbst weiterleitet. Sie hat aber einen Haken: Der Browser kennt den neuen Ort nicht. Relative Verweise, Cookies mit Pfadangabe und manche Anmeldeflüsse, besonders mit Vorauthentifizierung am LoadMaster, kommen damit durcheinander. Rewrite ist das Werkzeug für Fälle, in denen das Backend einen anderen Pfad erwartet als der Client kennt, etwa nach einer Migration von IIS ARR. Das Thema vertieft Migration von Windows NLB, IIS ARR oder TMG-Erbe auf den LoadMaster.



Skizze: Empfehlung für Root auf OWA ist der Redirect; Rewrite nur, wenn du weißt warum

Kriterium	Redirect (302 am LoadMaster)	Rewrite (Modify URL)
Adresszeile im Browser	zeigt das neue Ziel	bleibt beim Original
Zusätzlicher Roundtrip	ja, einer	nein
Für die Anwendung sichtbar	nein, sie sieht nur den neuen Request	ja, sie sieht den umgeschriebenen Pfad
Risiko mit relativen Links	gering	hoch, wenn die Anwendung ihren Pfad kennt
Typischer Einsatz	Root auf /owa, alter Name auf neuen Namen, HTTP auf HTTPS	Backend erwartet anderen Pfad als der Client
Fehlersuche	im Browser-Entwicklerwerkzeug sofort sichtbar	nur im L7-Debug oder Backend-Log sichtbar

Verstecken und ergänzen: Antwort-Header unter Kontrolle

IIS, ASP.NET, Exchange und SharePoint sind ausgesprochen mitteilsam. Ein einfacher Request an OWA verrät oft die IIS-Version, das ASP.NET-Framework, die genaue Exchange-Build-Nummer und den Namen des Servers, der geantwortet hat. Für die Fehlersuche ist das Gold, für Angreifer auch: Wer die Build-Nummer kennt, weiß, welche Sicherheitsupdates fehlen. Das Entfernen dieser Header ersetzt kein Patchen, aber es zwingt den Gegner, zu raten, statt nachzulesen.

Welche Header weg dürfen und welche bleiben müssen

Header	Wer ihn setzt	Verrät	Empfehlung
Server	IIS, http.sys	Webserver und Version, etwa Microsoft-IIS/10.0	Löschen am LoadMaster oder direkt im IIS
X-Powered-By	IIS-Konfiguration	ASP.NET	Löschen

X-AspNet-Version	ASP.NET	Framework-Version	Löschen
X-OWA-Version	Exchange	Genaue Exchange-Build-Nummer	Löschen, aber im internen Monitoring weiter auswerten
X-FEServer, X-BEServer, X-CalculatedBETarget	Exchange	Servernamen und Topologie	Extern löschen, intern behalten: wertvoll für die Fehlersuche
MicrosoftSharePointTeamServices	SharePoint	SharePoint-Build	Löschen, Office-Clients vorher testen
SPRequestGuid, request-id	SharePoint, Exchange	Korrelations-IDs, keine Versionen	Behalten: ohne sie findest du nichts im ULS- oder Exchange-Log
X-MS-*-Header allgemein	Exchange, SharePoint	unterschiedlich	Nie pauschal per Muster löschen, Clients werten einige aus

Die Delete-Header-Regel arbeitet mit einem Muster über Header und Wert und löscht nur, wenn das Muster vollständig passt. Das ist bewusst streng, damit du nicht aus Versehen den falschen Header erwischst, führt aber dazu, dass eine zu eng formulierte Regel einfach nichts tut. Prüfe nach jeder Änderung mit dem Entwicklerwerkzeug des Browsers oder Invoke-WebRequest in der PowerShell, ob der Header wirklich weg ist. Hänge die Regeln als Response-Regeln an den Virtual Service, nicht an die Request-Seite. Die Request-Seite ist ein beliebter Ort für Delete-Regeln, die seit Jahren nichts tun.

TIPP · Microsoft-Bordmittel nicht vergessen

Seit IIS 10 kennt die Request Filtering-Konfiguration das Attribut removeServerHeader, mit dem IIS den Server-Header gar nicht erst sendet; X-Powered-By entfernst du über die HTTP-Antwortheader des Servers. Der LoadMaster ist trotzdem der bessere zentrale Ort, wenn mehrere Anwendungen hinter einem Namen stehen oder Exchange-Updates die IIS-Konfiguration zurücksetzen. Ideal ist beides: Bordmittel für die Grundhygiene, LoadMaster als Sicherheitsnetz.

Security-Header nachrüsten, ohne OWA zu zerlegen

Mit Add-Header-Regeln auf der Response-Seite rüstest du Header nach, die Exchange und SharePoint nicht oder nicht überall setzen. Das ist der Teil, bei dem der Pentest-Bericht gerne zehn Punkte auflistet und bei dem du genau zwei davon ohne Risiko umsetzen kannst. Die anderen brauchen Tests, und einer davon ist ein Minenfeld.

Header	Beispielwert	Risiko vor Exchange und SharePoint
Strict-Transport-Security	max-age=31536000	Gering, sobald alles unter dem Namen per HTTPS läuft. includeSubDomains erst, wenn wirklich jede Subdomäne HTTPS kann
X-Content-Type-Options	nosniff	Gering, prüfe nur, ob das Backend ihn schon sendet
Referrer-Policy	strict-origin-when-cross-origin	Gering
X-Frame-Options	SAMEORIGIN	Mittel: OWA, ECP und SharePoint betten eigene Seiten in Frames ein. SAMEORIGIN geht in aller Regel, DENY zerlegt Funktionen
Content-Security-	anwendungsspezifisch	Hoch: OWA und SharePoint laden Skripte und

Policy		Ressourcen aus vielen Quellen. Ohne wochenlangen Report-Only-Betrieb nicht anfassen
Permissions-Policy	camera=(), microphone=()	Gering bis mittel, Teams-Integrationen und Web-Add-Ins testen

WARNUNG · Doppelte Header sind kein Sicherheitsgewinn

Setzt das Backend einen Header bereits und fügt der LoadMaster ihn noch einmal hinzu, steht er zweimal in der Antwort. Browser behandeln das je nach Header unterschiedlich, bei X-Frame-Options mit widersprüchlichen Werten im Zweifel restriktiv, bei Content-Security-Policy gelten beide Richtlinien gleichzeitig. Das robuste Muster: erst eine Delete-Regel für den Header, dann die Add-Regel mit dem gewünschten Wert, beide in dieser Reihenfolge auf der Response-Seite. Dann ist egal, was das Backend nach dem nächsten Update tut.

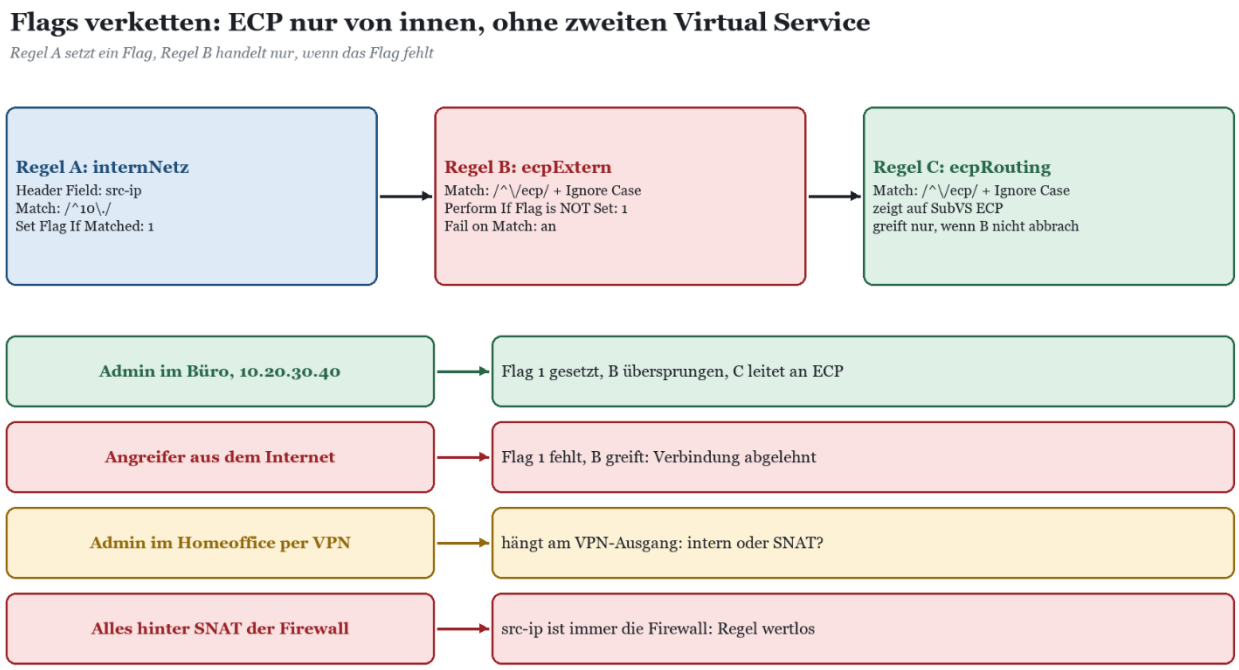
Replace Header: Location und Cookies geradebiegen

Bei TLS-Offload spricht der LoadMaster mit dem Backend oft per HTTP, und manche Anwendungen antworten dann mit einem Location-Header, der auf http:// zeigt. Eine Replace-Header-Regel auf den Header Location mit dem Muster /^http:\/\/(.*)$/ und dem Ersatz https://\1 biegt das zurecht. Wichtig: Im Feld für den Headernamen steht nur Location, ohne Doppelpunkt. Ähnlich ergänzt du bei Altanwendungen das Secure- oder HttpOnly-Attribut an Set-Cookie. Exchange und SharePoint brauchen das bei sauberer Konfiguration nicht, Eigenentwicklungen aus dem Jahr 2011 dagegen fast immer.

Header lassen sich natürlich auch auf dem Hinweg ergänzen, etwa ein Header mit der echten Client-Adresse für das Backend. Dafür hat der LoadMaster allerdings eigene Optionen, die du einer handgebauten Add-Header-Regel vorziehen solltest. Warum, und wie IIS und Exchange den Header auswerten, erklärt Client-IP sichtbar machen: Transparenz, SNAT und X-Forwarded-For.

Flags, Reihenfolge und Praxisrezepte

Mit Flags baust du Logik, die eine einzelne Regel nicht abbilden kann. Eine Regel setzt bei Treffer eines der neun Flags, eine andere läuft nur, wenn das Flag gesetzt oder eben nicht gesetzt ist. So entsteht ein kleines Und und Nicht, ohne dass du einen zweiten Virtual Service brauchst. Das bekannteste Beispiel vor Exchange ist das Exchange Admin Center, das von außen niemand braucht und das trotzdem in vielen Umgebungen offen im Internet steht.



Skizze: Adressen und Regelnamen fiktiv; Reihenfolge der Regeln ist Teil der Logik

Die Kette funktioniert nur, wenn die Regel, die das Flag setzt, vor der Regel ausgewertet wird, die es abfragt. Und sie funktioniert nur, wenn der LoadMaster die echte Quelladresse sieht. Steht vor ihm eine Firewall mit Source-NAT, ist src-ip für jeden Client gleich, und deine schöne Regel sperrt entweder alle oder niemanden. Wer zusätzlich eine Vorauthentifizierung oder eine WAF auf denselben Pfaden betreibt, sollte festhalten, welche Prüfung zuerst greift. Für die WAF-Seite lohnt Web Application Firewall und Geoblocking am LoadMaster: Schutz ohne Fehlalarm-Hölle, für die Vorauthentifizierung Edge Security Pack: Vorauthentifizierung, SSO-Domänen und der TMG-Ersatz.

AUS DER PRAXIS · Die ECP-Sperre, die keine war

In einer Verwaltung mit rund 600 Postfächern war ECP laut Dokumentation von außen gesperrt. Ein Pentest fand es trotzdem. Die Fail-on-Match-Regel lautete /^\/ecp/ ohne Ignore Case, der Pentester rief /ECP/ auf. Die Routing-Regel für die ECP-SubVS hatte dagegen Ignore Case, das Template hatte sie so angelegt. Ein Häkchen, ein Befund im Bericht, und eine lange Diskussion darüber, warum die Dokumentation seit drei Jahren falsch war.

Praxisrezepte auf einen Blick

Ziel	Regeltyp und Ort	Muster und Werte	Achte auf
Exchange-Pfad auf SubVS	Content Matching an der SubVS	/^\/owa(\/|$)/, Ignore Case	Rangfolge, Auffangfall nach unten
HTTP auf HTTPS	VS Port 80 ohne Real Server, Not Available Redirection Handling	302 Found, https://%h%s	Autodiscover-Namen, erst 302, später 301
Wurzel auf /owa	Content Matching an Redirect-SubVS ohne Real Server	/^\/$/, Redirect https://%h/owa/	Nur der nackte Schrägstrich, nicht /.*
ECP nur intern	zwei Content-Matching-Regeln mit Flag	src-ip /^10\./ setzt Flag 1; /^\/ecp/ mit Fail on Match,	SNAT, VPN-Ausgang, Ignore Case

		wenn Flag 1 fehlt	
Server-Banner weg	Delete Header, Response-Regel	Header Server	Wirklich prüfen, ob er weg ist
Exchange-Build verstecken	Delete Header, Response-Regel	X-OWA-Version	Monitoring liest die Version anders aus
HSTS ergänzen	Delete Header, dann Add Header, Response	Strict-Transport-Security: max-age=31536000	includeSubDomains erst nach Inventur
nosniff ergänzen	Add Header, Response	X-Content-Type-Options: nosniff	Doppelte Header vermeiden
Location auf https	Replace Header, Response	/^http:\/\/(.*)$/ zu https://\1	Headername ohne Doppelpunkt
Altlasten-Pfad umbiegen	Modify URL, Request	/^\/altapp\/(.*)/ zu /neuapp/\1	Relative Links, Cookies mit Pfad

Testen, dokumentieren, aufräumen

- **Benennen:** Regelnamen beschreiben Zweck und Ort, etwa exOwaRoute oder respDelServer. „test2“ ist kein Name, sondern ein Geständnis.
- **Inventur:** Unbenutzte Regeln mit Stern in der Spalte „In Use“ exportieren und dann löschen. Was niemand benutzt, kann auch niemand vermissen.
- **Testmatrix:** Für jede Regel eine Liste von Pfaden, die treffen müssen, und eine, die nicht treffen dürfen, inklusive Großschreibung und Query.
- **Beobachten:** Nach Änderungen das L7-Logging und Debug-Werkzeuge nutzen, um zu sehen, welche SubVS ein Request tatsächlich bekommen hat.
- **Versionieren:** Regeln per REST-API exportieren und im Repository ablegen. Dann ist „bitteNichtAnfassenKlaus“ wenigstens nachvollziehbar.

Für die Fehlersuche bei Regeln, die nicht tun, was sie sollen, hilft Troubleshooting am LoadMaster: Logs, L7-Debug und TCP-Dump. Und wenn SharePoint statt Exchange hinter dem LoadMaster steht, gelten dieselben Prinzipien mit anderen Pfaden und eigenen Fallen bei Office-Clients, nachzulesen in SharePoint Server Subscription Edition hinter dem LoadMaster.

FAQ: häufige Fragen zu Kemp Content Rules

Wirken Content Rules auch ohne TLS-Offload?

Nur eingeschränkt. Ohne Entschlüsselung sieht der LoadMaster weder Pfad noch Header, Content Matching und Header-Modifikationen haben dann nichts zum Arbeiten. Für Layer-7-Regeln brauchst du Offload oder Re-Encryption.

Darf ich Regeln am Haupt-Service und an der SubVS gleichzeitig anhängen?

Technisch ja, aber halte es übersichtlich. Header-Regeln, die für alle Dienste gelten, gehören an eine Stelle, dienstspezifische an die SubVS. Wer dieselbe Delete-Regel an beiden Orten pflegt, ändert beim nächsten Mal garantiert nur eine.

Warum trifft meine Regel auf /OWA nicht?

Weil Ignore Case fehlt. Exchange behandelt Pfade ohne Rücksicht auf Groß- und Kleinschreibung, der LoadMaster ohne dieses Häkchen nicht.

Ist ein Redirect auf /owa ein Sicherheitsproblem?

Nein, er verrät nichts, was ein Angreifer nicht ohnehin durch Ausprobieren findet. Er ist reiner Komfort. Sicherheitsrelevant ist, was von außen unter den übrigen Pfaden erreichbar ist.

Kann ich mit Content Rules eine WAF ersetzen?

Nein. Content Rules prüfen, was du explizit beschreibst, eine WAF prüft auf bekannte Angriffsmuster. Fail on Match ist gut, um Pfade zu sperren, aber keine Abwehr gegen Injection.

Bremst Replace String in Response Body den LoadMaster aus?

Ja, spürbar, weil jede Antwort gelesen und umgeschrieben werden muss. Bei komprimierten Antworten findet die Regel zudem oft nichts. Verwende sie nur für Altanwendungen, die du nicht anders reparieren kannst, und begrenze sie per Flag auf die betroffenen Pfade.

Gehen meine Regeln bei einem Template-Update verloren?

Templates legen Regeln an, löschen aber deine eigenen nicht. Gefährlicher ist der umgekehrte Fall: Ein neu angewendetes Template ergänzt Regeln mit ähnlichen Namen, und du hast plötzlich zwei Varianten. Nach jedem Template-Einsatz deshalb die Rangfolge prüfen.

Fazit: wenige Regeln, scharf formuliert

Content Rules sind kein Ort für Kreativität, sondern für Präzision. Verankerte Muster mit Ignore Case, eine bewusst gepflegte Rangfolge mit dem Auffangfall ganz unten, Redirects statt Rewrites, wo der Browser es wissen soll, und Header-Regeln auf der Response-Seite, die du nach jeder Änderung tatsächlich nachprüfst. Mehr braucht es vor Exchange, SharePoint und ADFS in den meisten Fällen nicht.

Das Verstecken von Server-Bannern und das Ergänzen von Security-Headern ist schnelle, sichtbare Hygiene, die jeder Pentest-Bericht honoriert. Es ersetzt aber weder Updates noch eine WAF noch eine saubere Vorauthentifizierung. Und wenn du eine Regel nicht in einem Satz erklären kannst, dann wird sie die nächste sein, die irgendwer „bitteNichtAnfassen“ nennt.

TIPP · Unterstützung bei Bestandsaufnahme und Umbau

Wenn deine Content Rules über Jahre gewachsen sind und niemand mehr sicher sagen kann, welche wirken, hilft die Beratung zum Kemp LoadMaster beim Inventarisieren, Testen und Aufräumen. Wer Regex, Flags und Rangfolge mit dem eigenen Team im Labor durchspielen möchte, findet das in der Kemp-LoadMaster-Schulung.

WEITERLESEN · Weiter im Thema

› Virtual Service, Real Server, SubVS: das Grundvokabular des LoadMasters
› Exchange Server hinter dem LoadMaster: Templates, Namespaces, Health Checks
› Web Application Firewall und Geoblocking am LoadMaster: Schutz ohne Fehlalarm-Hölle
› TLS-Offload, Re-Encryption und Zertifikate: PKI trifft LoadMaster
› Client-IP sichtbar machen: Transparenz, SNAT und X-Forwarded-For
› Troubleshooting am LoadMaster: Logs, L7-Debug und TCP-Dump