capalogic

Mythos Proxy

Eine kritische Betrachtung

Stand: 22.05.2012

Capalogic GmbH
Angerring 4
38446 Wolfsburg

Telefon 05363 811156
Fax 0911 60 10 99
E-Mail info@capalogic.de
Web www.capalogic.de

Inhaltsverzeichnis

1. Einleitung

Viele Unternehmen setzen Web Proxy (auch Forward Proxy) wie SQUID ein, um Zugriffe aus dem Intranet in das Internet zu reglementieren und bündeln.

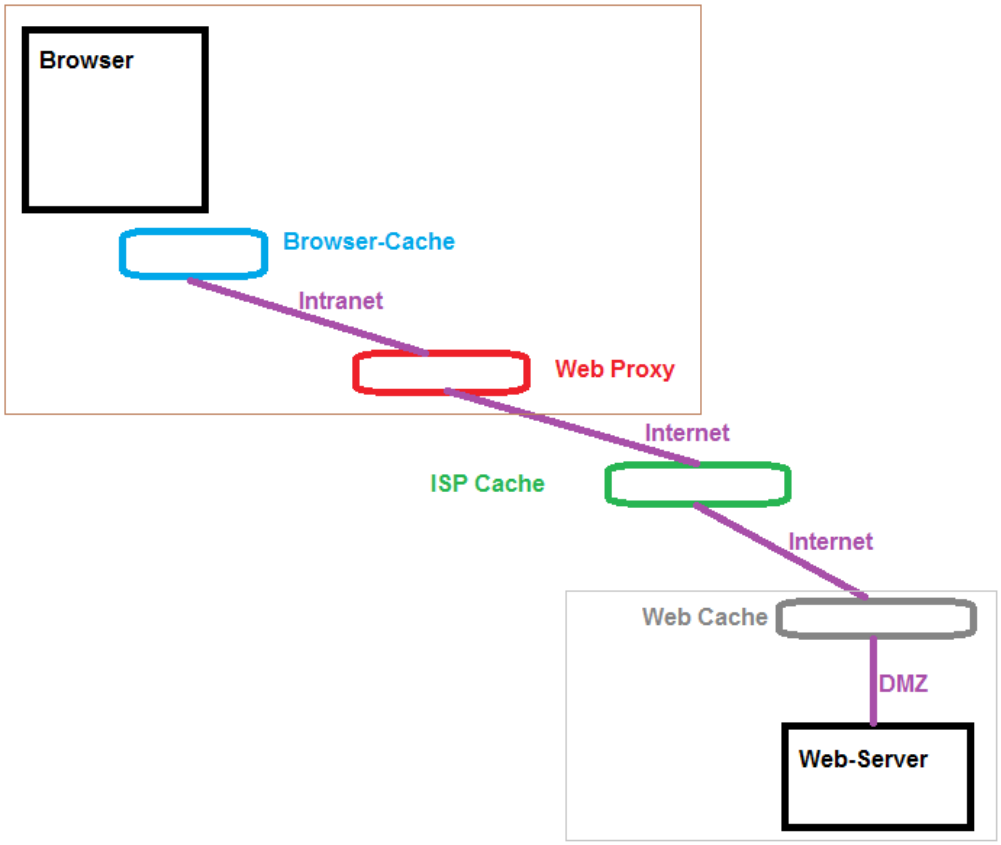
Betreiber von Web-Diensten, Portalen, etc. setzen oft Proxies in der DMZ als sogenannten Reverse Cache ein, um die Last auf die Webserver in der DMZ zu verringern.

Weiterhin nutzen manche Netzbetreiber wie z. b. T-Mobile sogenannte Forward Caches, die den Kunden Webseiten möglichst aus dem Cache liefern sollen, so dass der Verkehr durch das Backbone-Netz verringert wird.

Erschwerend kommt hinzu, dass die Browser selbst einen lokalen Cache unterhalten und jeder Browser hat hier andere Strategien und legt Standards anders aus.

Hier findet sich eine brauchbare Begriffsdefinition:
http://en.wikipedia.org/wiki/Web_cache

Wenn es ganz dumm läuft, addieren sich viele Caches beim Abruf einer Webseite:



2. Historischer Ballast

Die Motivation für solche Caches war das langsame und leistungsschwache Netzwerk, welches das Internet in den 80er und 90er Jahren war. So limitierten Browser die Anzahl Verbindungen zu einer Domain auf 2. Auch lange nachdem die Netze und Webserver leistungsfähiger waren, gab es diese Limitierungen noch in den Browsern und Web-Programmierer wie bei Google umgingen die Begrenzung durch Verwendung von Subdomains (z. B. Maps.google.de).

Früher waren Webseiten vorwiegend statisch. Heute sind sie zunehmend dynamisch. Selbst CSS- und Javascript-Dateien werden teilweise dynamisch generiert (je nach User-Agent-String, spring Browser). Daher wurden im Laufe der Zeit mehrere Mechanismen zur Cache-Kontrolle eingeführt (Pragma No-chache, Etag, expires-Header). Oftmals tricksen Anwendungen die Caches auch dadurch ab, dass eindeutige URL-Parameter angehängt werden.

3. Enorme Komplexität

Nachdem also gezeigt wurde, dass die Netze sich längst zu Autobahnen entwickelt haben, stellt sich die Frage, welchen Nutzen all diese Caches noch haben. Man muss es deutlich sagen, keinen. Aber im Gegenteil, durch die unbekannte Reihe von Caches, die hintereinander geschaltet sind, wird die Entwicklung von Web-Anwendungen immer komplexer. So kann es passieren, dass sich eine Anwendung über UTMS-Stick anders darstellt, als über das DSL-Netz, weil T-Mobile einen zusätzlichen Forward Cache betreibt und der sich nicht an Standards wie Etag oder expires-Header hält (ein Fall aus der Praxis).

Daher müssen Web-Entwickler immer höllisch aufpassen, mit welchen Cache-Steuerungsinformationen ausgeliefert werden, weil sie sonst ggfs. Viel länger in Caches überleben, als gewünscht.

Wussten Sie, das man in den meisten Browsern mit Strg-F5 einen Header

Cache-Control: `no-cache`

mit dem Request sendet, der Caches unterwegs dazu bewegen soll, dass sie die Information nicht aus dem Cache holen? Aber noch besser: machen Sie nun wieder einen „normalen“ F5 (Reload), dann bekommen Sie nun doch wieder die veraltete Version der Seite aus einem dieser verflixten Caches, da das no-cache ja nicht hies: lösche die Datei aus deinem Cache, sondern nur: geh am Cache vorbei und hol die Datei von dem dahinter liegenden System (könnte schon wieder ein Cache sein oder endlich mal der Web-Server).

Noch schöner: Proxies wandeln häufig immer noch HTTP1.1-Anfragen in HTTP1.0-Anfragen um und spiegeln dann dem Browser wieder eine HTTP1.1-Response vor. Da kann ich mich browserseitig mächtig freuen, dass ich mit HTTP1.1 arbeite, hinter dem Web Proxy geht es dann doch wieder gemächlich zu. Wer seine Apache-Logs eines Live-Systems mal genauer ansieht, wird sehen, dass dort die meisten Anfragen immer noch HTTP1.0 sind, während alle modernen Browser längst HTTP1.1 nutzen. Auch ein Weg den Fortschritt zunichte zu machen.

4. Auswirkung auf die Anwendungen

Der Anwendungsentwickler muss sich bei der Implementierung Gedanken machen, welche Ressourcen dynamisch sind und welche nicht. Man möchte meinen, dass Icons recht statisch sind, aber dann wechselt der Anwender das Theme und die Icons aktualisieren nicht. Also gehört in URL der Theme-Name:

http://meinserver.de/img/arrow_up.png?theme=skyblue

Verändert der Administrator aber die Icons in diesem Theme, so werde die zunächst nicht sichtbar, da lokale und Proxy-Caches diese noch cachen. Also müsste eigentlich eine eindeutige Theme-Id in die URL:

http://meinserver.de/img/arrow_up.png?themeId=23m45m67hg3

Ein Beispiel aus der Praxis: Ein Portal für Music-Downloads zeigt das Album-Cover an. Keine besonderen Caching-Vorkehrungen, im Gegenteil, zwecks Suchmaschinenoptimierung sogar ein sprechender Name:

http://server.de/covers/nirvana/teen_spirit_cover.png

Wird sich ja nicht mehr ändern, oder? Leider kam das doch ab und zu vor. Nur reichte es nicht, das Bild auszutauschen, da es in vielen Caches noch variablen lange gespeichert ist. Die Konsequenz: die URL erhält die Image-Id und die ändert sich beim Upload des neuen Covers.

Javascript-Dateien sind recht dynamisch. Eigentlich wäre eine Versionsnummer im Dateinamen das Mindeste. Sieht man aber oft anders.

Die Steuerung über expires- oder ETag-Header ist auch möglich, jedoch ist nicht garantiert, dass sich jeder Cache (in gleicher Weise) daran hält. Die Holzhammer-methode besteht darin, an jede URL eine eindeutige ID (z. B. Zeit in msec) anzuhängen. Nicht wirklich elegant.

5. Web Proxy als Gatekeeper

Viele Administratoren möchten nicht, dass jeder Browser im Intranet direkt ins Internet gehen kann. Entweder wird dazu im Browser der entsprechende Proxy fest gesetzt (z. B. durch Group Policies) oder, noch perfider, es wird ein transparenter Proxy durch Port Forwarding eingerichtet (Anfragen nach außerhalb an Port 80/443 werden an Web Proxy umgeleitet).

Die Motivation ist unterschiedlich:

- Bündelung der Zugriffe
- Sperren unerwünschter Inhalte (Facebook und andere Social Networks, E-Mail-Dienste, etc.)
- Sperren unerwünschter Protokolle (z. B. Videostreams)
- Virenschutz

Die Nachteile sind mannigfaltig:

- Webseiten auf andere Ports wie 80/443 werden nicht erreicht
- Funktionsumfang wird eingeschränkt
- Funktionalität wird behindert

Bei einem Kunden hat der Web Proxy die https-Verbindungen terminiert und mit einem eigenen Zertifikat ausgewiesen (das war in jedem Browser als vertrauenswürdig importiert). Die Absicht war es, den Virenscanner auch auf https-Verbindungen einsetzen zu können. Aber leider wurden auch signierte Applets ausgepackt und neu signiert und waren damit nicht arbeitsfähig. Außerdem konnte https-Verkehr so vom Admin im Klartext mitgelesen werden.

All diese Maßnahmen werden nur deshalb unternommen, weil die Browser auf den meisten Systemen mit viel zu vielen Rechten laufen. Dabei wäre es trivial die Browser in eine Sandbox zu packen. Das c't-Magazin hat das schon vor Jahren demonstriert.

5.1. Proxy-Ausnahmen

Oftmals sollen Intranet-Anwendungen direkt ohne Proxy ausgeführt werden. Das verringert einerseits Latenz und Netzwerkverkehr und ermöglicht andererseits durchgängige Protokolle (HTTP 1.1). Häufig reagieren Intranet-Anwendungen sogar allergisch auf Proxies, genauer gesagt, sie sind auf deren Gemeinheiten nicht vorbereitet. Daher werden häufig Proxy-Ausnahmen, z. B. über Windows-Gruppen-Richtlinien, verwaltet. Allerdings ist die Pflege von expliziten Listen aufwändig und fehleranfällig und resultiert oft darin, dass niemand mehr weiß, warum welche Einträge aufgeführt sind. Letztlich wird jede Anpassung der Ausnahmeliste zum organisatorischen Drama.

Sofern Ausnahmen gerechtfertigt sind, sollten Sie immer nach Domänen oder Subnetzen organisiert werden, niemals auf explizite Server.

6. Lösungsansatz: Proxies abbauen

Da Proxies mehr Probleme bereiten als sie Nutzen stiften, sollten wir uns vom Mythos Proxy verabschieden. Die Browser cachen schon genug und hier hat der Benutzer noch die meiste Kontrolle. Notfalls löscht der den Verlauf und weg ist der Cache. Alle weiteren Proxies wie Web Proxy, Reverse Cache und Forward Cache gehören endlich eingemottet, zumindest in den Industriestaaten mit ihren schnellen Netzen.

Ja, es gibt sich noch die Anwender mit ISDN und 384-KBit-DSL, aber hier reicht der Browser-Cache völlig aus. Nur in Entwicklungsländern mit schwachen Backbones scheinen Revers und Forward Caches sinnvoll.

7. Zusammenfassung

Proxies hatten einst ihre Berechtigung und in manchen Fällen haben Sie das heute noch. Wegen der unkoordinierten Evolution und mangelnden Standardisierung führen Sie in der Praxis aber häufig zu Problemen, insbesondere, wenn Lösungen aus den Intranet in die Cloud verlagert werden. Anwendungsentwicklern bleibt bislang keine andere Möglichkeit, als sich intensiv mit dem Thema zu beschäftigen und ihre Anwendung entsprechend zu implementieren.