Web Caching

Με την ραγδαία ανάπτυξη της Web κίνησης, το caching ήταν η πρώτη βασική τεχνική που στόχευε στην ελαχιστοποίηση της υστέρησης που αντιλαμβάνονται οι web χρήστες καθώς και στον περιορισμό της μετάδοσης πλεονάσουζας πληροφορίας στο δίκτυο. Επίσης διαπιστώθηκε ότι πολλαπλοί χρήστες σε ένα οργανισμό επισκέπτονται το ίδιο σύνολο λίγων σχετικά δικτυακών τόπων. Το RFC 2616 καθορίζει ότι μία cache είναι ένας τοπικός χώρος αποθήκευσης μηνυμάτων απάντησης. Ενας λιγότερο αυστηρός ορισμός του caching είναι η μετακίνηση web περιεχομένου πλησιέστερα στους τελικούς χρήστες. Το Web caching είναι η πλέον μελετημένη εφαρμογή του WWW. Σήμερα υπάρχουν πολλαπλά προϊόντα για caching είτε στην μορφή λογισμικού είτε στην μορφή υλικού (hardware).

Ο πρώτος web server που κατασκευάστηκε στο CERN είχε ένα συσχετιζόμενο proxy server που συμπεριλάμβανε μία cache. Ενα από τα πρώτα έργα πάνω στην τεχνολογία του caching ήταν το harvest το οποίο δεικτοδοτούσε (index) πληροφορία στο πλαίσιο του διαδικτύου. Το harvest επεδίωκε το caching και την διατήρηση αντιγράφων (replicate) πληροφορίας η οποία συγκεντρώνονταν με διάφορα εργαλεία. Μία ιεραρχική cache αντικειμένων ήταν το βασικό τμήμα της αρχιτεκτονικής του harvest. Ενας μηχανισμός εγγραφής των servers (registry) επέτρεπε την συλλογή πληροφοριών για άλλες caches. Το registry ανταποκρίνονταν σε αιτήματα για την τοποθεσία των caches. Οι caches που βρίσκονταν στην ίδια τοποθεσία (collocated) μπορούσαν να οργανωθούν σε δεξαμενές (pools) πόρων.

Το caching αποτέλεσε σημαντικότατο συστατικό του WWW από την περίοδο που δημοσιεύτηκε η προδιαγραφή HTTP/1.0. Τα πρώτα πειράματα στην τεχνολογία caching επέδειξαν σημαντικές μειώσεις στον όγκο της ανταλλασόμενης πληροφορίας. Οι caches άρχισαν να διατηρούν μία πληθώρα πληροφοριών σχετικά τα έγγραφα που είχαν αποθηκευμένα όπως π.χ., τύπος περιεχομένου (content type), μέγεθος και μέσος χρόνος μεταξύ αλλαγών-μεταβολών. Θέματα συνέπειας καθώς και η εξουσιοδοτημένη πρόσβαση στους πόρους της cache άρχισαν να απασχολούν την κοινότητα του WWW. Ομάδες caches οργανώθηκαν σε ιεραρχίες οι οποίες κάλυπταν ολόκληρες περιοχές ή ακόμη και κράτη.

Αναλυτικά οι στόχοι του caching είναι οι εξής:

- Να μειώσει τον χρόνο που αντιλαμβάνεται ο τελικός χρήστης από την υποβολή μίας αίτησης μέχρι την παρουσίαση του ζητούμενου πόρου στο τερματικό του,
- Να μειώσει τον φόρτο στο δίκτυο αποφεύγοντας την επανάληψη της μετάδοσης του ιδίου πόρου,
- Να μειώσει τον φόρτο στον origin server με την παρεμβολή ενός μεσάζοντα μεταξύ του πελάτη και του origin server που θα χειρίζεται αιτήσεις.

Οι λόγοι για τους οποίους αναπτύχθηκε η τεχνολογία caching

Οι εταιρίες WWW hosting πρέπει να πληρώσουν για το εύρος ζώνης το οποίο χρησιμοποιούν και θέλουν να αυξήσουν την δυνατότητα caching για να μειώσουν τα εξοδά τους. Ολοι οι εμπλεκόμενοι στην αλυσίδα του WWW οφελούνται από την εισαγωγή της τεχνολογίας caching. Οι τελικοί χρήστες οφελούνται γιατί μειώνεται ο χρόνος απόκρισης του συστήματος. Ενα σημαντικό ποσοστό των εγκαταλελημένων προσπαθειών (aborts) κατά την διάρκεια της συνόδου ενός χρηστή οφείλεται στην δυσαρέσκεια του χρήστη για την βραδύτητα άντλησης δεδομένων. Το εύρος ζώνης το οποίο δαπανάται σαν αποτέλεσμα της επαναλαμβανόμενης μετάδοσης πλεονάζουσας πληροφορίας είναι ιδιαίτερα μεγάλο. Δεδομένου του προβλήματος συμφόρησης (congestion) στο διαδίκτυο, η ελάτωση της κίνησης ή η μετακίνηση του φόρτου προς τα άκρα του δικτύου είναι ιδιαίτερα ωφέλιμη. Τα πλεονεκτήματα για το δίκτυο είναι δύο: μόνο τα αναγκαία δεδομένα διασχίζουν το δίκτυο και υπάρχει διαθέσιμο εύρος ζώνης και για άλλα δεδομένα.

Ποιοί πόροι επιδέχονται caching

Μία cache μπορεί να αποφασίσει το αν κάποια απάντηση μπορεί να υποστεί caching ή όχι βάσει δύο παραγόντων: απαιτήσεις που σχετίζονται με το πρωτόκολλο HTTP καθώς και το υπο εξέταση περιεχόμενο. Σχετικά με το HTTP πρωτόκολλο απαιτείται οι caches να λαμβάνουν υπόψη κάποιες ντιρεκτίβες σχετικά με την δυνατότητα αποθήκευσης κάποιου συγκεκριμένου μηνύματος. Οι απαιτήσεις οι οποίες εξαρτώνται από το περιεχόμενο επηρεάζονται από τις επιχειρηματικές απαιτήσεις μίας cache καθώς και τις πολιτικές που επηρεάζουν την συχνότητα ελέγχου της ορθότητας των περιεχομένων της cache (revalidation). Οι πολιτικές αυτές, με την σειρά τους, επηρεάζονται από τα χαρακτηριστικά όπως π.χ., το μέγεθος και τον τύπο.

Το HTTP/1.1 καθορίζει απλούς κανόνες για το ποιοί πόροι επιδέχονται caching. Η μέθοδος αίτησης, τα πεδία επικεφαλίδας της αίτησης, ο κωδικός κατάστασης της απάντησης (response

status) καθώς και τα πεδία επικεφαλίδας της απάντησης πρέπει να υποδεικνύουν εάν ο πόρος επιδέχεται caching. Οι απαντήσεις στα αιτήματα OPTIONS, PUT και DELETE δεν επιδέχονται caching. Οι απαντήσεις στα αιτήματα POST δεν επιδέχονται caching εκτός εάν η απάντηση φέρει τις κατάλληλες Cache-Control και Expires επικεφαλίδες. Εάν η cache δεν υποστηρίζει το πεδίο επικεφαλίδα Range, όλες οι απαντήσεις που φέρουν κωδικό κατάστασης 206 (Partial Content) δεν μπορούν να γίνουν cached.

Ορισμένες απαντήσεις περιέχουν πληροφορία από τον origin server που αποκλείει την αποθήκευση του μηνύματος σε caches. Υπάρχουν 2 είδη τέτοιας πληροφορίας: πληροφορίες σχετικά με την δυνατότητα αποθήκευσης και ντιρεκτίβες προς τις cache. Εάν η απάντηση περιέχει το πρώτο είδος πληροφορίας, η απόφαση για caching θα πρέπει να βασίζεται σε αυτήν. Για παράδειγμα, ο origin server μπορεί να καθορίσει ακριβώς το διάστημα για το οποίο ο πόρος θα πρέπει να θεωρείται έγκυρος μέσω του πεδίου Expires. Η ντιρεκτίβα cache-control μπορεί να αποκλείσει το caching ορισμένων πόρων. Για παράδειγμα η cache-control: private καθορίζει ότι μία διαμοιραζόμενη cache δεν μπορεί να αποθηκεύσει τον πόρο. Ενα μήνυμα απάντησης που περιέχει την ντιρεκτίβα cache-control: no-store δεν θα πρέπει να αποθηκευτεί καθόλου. Η ντιρεκτίβα cache-contol: no-cache περιορίζει την πιθανότητα αποθήκευσης του αντικειμένου στη cache γιατί θα πρέπει να ελέγχεται η ορθότητα του αντικειμένου πρίν από κάθε φορά που επιστρέφεται ως cache hit. Οι ντιρεκτίβες δεν περιορίζονται μόνο στις απαντήσεις από τον origin server. Μπορεί να ενσωματώνονται και στις ερωτήσεις από τον user agent. Για παράδειγμα η cache-control: no-store μπορεί να εμφανιστεί τόσο σε απάντηση όσο και σε ερώτηση. Το πρωτόκολλο διαθέτει αυτές τις ντιρεκτίβες για να προστατεύει τον προσωπικό χαρακτήρα μίας απάντησης (privacy) και για υποδείξει την πτητικότητα του πόρου (volatile resource) δηλαδή ότι μπορεί να αλλάξει αμέσως μετά την αποστολή του.

Η παρουσία πεδίων επικεφαλίδας όπως τα Authorization και Vary ελαχιστοποιούν τις πιθανότητες αποθήκευσης του πόρου σε cache. Η επικεφαλίδα αίτησης Authorization υποδεικνύει ότι ο ζητούμενος πόρος δεν είναι διαθέσιμος για όλους. Επίσης, η παρουσία του Vary δεικνύει ότι μία αποδεκτή απάντηση για cache θα πρέπει να περιορίζεται από τις τιμές που ορίζονται στο Vary πεδίο.

Μία cache μπορεί να έχει και τους δικούς της κανόνες για τον έλεγχο της δυνατότητας caching κάποιου συγκεκριμένου πόρου, άσχετα από τους περιορισμούς που επιβάλλονται

μέσω του πρωτοκόλλου. Δηλαδή, αν και ο πόρος επιδέχεται caching δεν σημαίνει, κατ'ανάγκη ότι θα γίνει και cache. Τα μηνύματα μπορεί να είναι μεγάλα σε όγκο, να παράγονται δυναμικά, να περιέχουν cookies, παράμετροι που επηρεάζουν σημαντικά την δυνατότητα αποθήκευσης του πόρου σε caches. Οι πολιτικές caching επηρεάζονται από χαρακτηριστικές του μηνύματος και όχι από τους περιορισμούς του πρωτοκόλλου.

Οι επιχειρήσεις μπορεί να θέλουν να περιορίσουν το κόστος μεταφοράς αγνοώντας ορισμένους περιορισμούς που σχετίζονται με την cache. Για παράδειγμα μπορεί να αποθηκεύουν πόρους που δεν θα πρέπει να αποθηκεύουν (π.χ., cache-control: private). Επίσης, οι caches μπορεί να λαμβάνουν υπόψη την επιβάρυνση αποθηκευτικού χώρου (storage overhead) και να μην αποθηκεύουν ορισμένα μεγάλα αρχεία παρά το γεγονός ότι επιδέχονται caching. Εάν το αντικείμενο είναι μεγάλο, πολλοί πόροι θα πρέπει να εκτοπιστούν από την cache. Αναφορικά με την χρονική υστέρηση, το κόστος ανάκτησης, από τους origin servers, μικρών σε όγκο πόρων είναι μεγαλύτερο από το κόστος ανάκτησης τους από caches. Ετσι, μία cache μπορεί να αποφύγει την αποθήκευση μεγάλων πόρων. Από την άλλη πλευρά, μία μεγάλη cached απάντηση, εάν ζητηθεί ορισμένες φορές από τους clients επιφέρει σημαντικά οφέλη εύρους ζώνης.

Πολλές caches δεν αποθηκεύουν τις απαντήσεις από scripts με το σκεπτικό ότι οι παράμετροι των σχετικών ερώτησεων δεν πρόκειται να ξαναχρησιμοποιηθούν. Η παρουσία πρόσθετης πληροφορίας σχετικά με το caching στις επικεφαλίδες μίας δυναμικά διαμορφούμενης απάντησης (π.χ., Expires ή ETag) μπορεί να σημαίνει ο πόρος μπορεί να γίνει cache (π.χ., ένα CGI script το οποίο επιστρέφει το n-ιοστό ψηφίο του αριθμού π). Πολλά WWW ερωτήματα συχνά καταλήγουν στις ίδιες απαντήσεις και υπάρχουν σήμερα caches που το λαμβάνουν αυτό υπόψη. Μία άλλη κατηγορία απαντήσεων που θεωρούνται ως μη επιδεχόμενες cache είναι οι απαντήσεις που εξατομικεύονται (tailored). Για παράδειγμα, οι απαντήσεις που συνοδεύονται από cookies θεωρούνται uncacheable, γιατί εκτιμάται ότι διαφέρουν από χρήστη σε χρήστη.

Η απόφαση για την αποθήκευση κάποιου πόρου σε cache εξαρτάται από τον ρυθμό αλλαγών στο συγκεκριμένο πόρο. Ορισμένοι πόροι μεταβάλλονται σπάνια, ίσως και καθόλου (π.χ., ηλεκτρονικές εκδόσεις βιβλίων). Μία παλιά ευρεστική τεχνική για τον προσδιορισμό της δυνατότητας caching ενός πόρου ήταν η ημερομηνία τελευταίας αλλαγής. Υιοθετούνταν η υπόθεση ότι εάν ο πόρος δεν έχει μεταβληθεί για μεγάλο χρονικό διάστημα είναι χαμηλή η

πιθανότητα αλλαγής του στο άμεσο μέλλον. Αυτός ο πόρος θεωρούνταν υποψήφιος για αποθήκευση στην cache. Σε περίπτωση που ο πόρος αποθηκεύονταν στην cache, ο χρόνος τελευταίας αλλαγής υποδεικνύει το διάστημα στο οποίο θα έπρεπε να επανελεγχθεί η ορθότητα (revalidate) του πόρου. Αντίστροφα, η λογική αυτή υποθέτει ότι αν ο πόρος έχει αλλαχθεί πρόσφατα είναι υψηλή η πιθανότητα αλλαγής του σε σύντομο χρονικό διάστημα. Ο πόρος αυτός δεν θα μείνει επίκαιρος (fresh) στην cache για μεγάλο διάστημα. Ενας άλλος προβληματισμός που λαμβάνεται επίσης υπόψη για το caching ενός πόρου είναι ότι οι πόροι που αλλάζουν συχνά είναι δημοφιλείς και, κατ' επέκταση, χαρακτηρίζονται από υψηλό ρυθμό προσπέλασης (rate of access). Αυτοί οι πόροι θα πρέπει να υποστούν caching.

Που υλοποιείται το caching

Οι Caches εντοπίζονται στους browsers και σε όλους τους ενδιάμεσους (intermediaries) μεταξύ του user agent και του origin server. Συνήθως, η cache εντοπίζεται σε ένα proxy ή στους browsers. Συχνά προτιμάται η αποθήκευση πόρων σε περισσότερες από μία τοποθεσίες (locations) γιατί:

- Η browser cache μπορεί να αποφύγει να επαναμεταφέρει πόρους που ζήτησε ο χρήστης κατά την διάρκεια της ιδίας συνόδου. Ο browser όμως δεν εκμεταλλεύεται τους πόρους που ζητούνται ιδιαίτερα συχνά από άλλους χρήστες στο ίδιο περιβάλλον.
- Ενα caching proxy μπορεί να υποστηρίξει δεκάδες ίσως και εκατοντάδες χρήστες. Μία browser cache μπορεί να αποθηκεύσει ένα λογικό σύνολο από απαντήσεις που μεταδόθηκαν πρόσφατα για μεγαλύτερο διάστημα απ'ότι ένα caching proxy. Ενα caching proxy, όντας πόρος διαμοιραζόμενος από πολλούς, θα πρέπει να εκτοπίσει ορισμένες απαντήσεις από την cache ταχύτερα από τον browser.
- Μία περιφερειακή cache μπορεί να υποστηρίξει πολλαπλές caches που χαρακτηρίζονται από μία γεωγραφική εγγύτητα και ανήκουν σε μία ή περισσότερες διαχειριστικές οντότητες. Μία εθνική cache μπορεί να ομαδοποιήσει ένα σύνολο από περιφερειακές caches και να ελλατώσει το κόστος σε χώρες όπου η διεθνής κίνηση χαρακτηρίζεται από υψηλό κόστος.

Αλλες μορφές caches είναι οι reverse και interception caches. Ενα reverse proxy συμπεριφέρεται ως ένα front-end σε ένα ή περισσότερους origin servers και μπορεί να περιέχει και cache. Σε ένα reverse proxy, το caching υλοποιείται για λογαριασμό των origin servers και όχι των user agents. Στόχος του reverse proxy είναι η μείωση του φόρτου στους origin servers. Οι πιο δημοφιλείς πόροι που ζητούνται από ένα origin server είναι πολύ

πιθανόν να εντοπίζονται στο reverse proxy. Ενα reverse proxy προωθεί την αίτηση στον origin server εάν η απάντηση δεν εντοπιστεί στην cache. Η προώθηση πραγματοποιείται μέσω ενός tunnel, δηλαδή το proxy συμπεριφέρεται ως ένας απλός μεταγωγέας (blind relay).

Οι interception proxies μπορεί να βρίσκονται οπουδήποτε σε ένα δίκτυο και να ελέγχουν τα επίπεδα δικτύου και μεταφοράς. Τα interception proxies αναχαιτίζουν την HTTP αίτηση και λαμβάνουν ρητά την απάντηση. Συνήθως τοποθετούνται κοντά στους clients. Τα interception proxies δεν χρειάζεται να βρίσκονται πάνω στην διαδρομή των πακέτων. Μία συσκευή που θα μπορούσε να ελέγξει τα πακέτα σε επίπεδο μεταφοράς θα μπορούσε να ανακατευθύνει την κίνηση σε ένα interception proxy. Μετά την αποθυλάκωση των πακέτων, η αίτηση θα μπορούσε προαιρετικά να ανακατευθυνθεί σε κοντινές caches κάτω από τον ίδιο διαχειριστικό έλεγχο όπως το interception proxy. Αυτή η αναχαίτιση είναι γενικά διαφανής στον χρήστη, ο οποίος αντιλαμβάνεται μειωμένη καθυστέρηση, αν ο πόρος εντοπιστεί σε στην cache.

Υλοποίηση του caching

Η υλοποίηση του μηχανισμού του caching βασίζεται σε συγκεκριμένα βήματα: η cache πρέπει, αρχικά, να αποφασίσει αν το μήνυμα επιδέχεται caching, μετά να διαγνώσει αν υπάρχει διαθέσιμος χώρος για την αποθήκευση του, και, στην περίπτωση που δεν υπάρχει χώρος, να αποφασίσει ποιά cached αντικείμενα θα αντικατασταθούν. Η cache, με την λήψη κάποιας αίτησης θα πρέπει να αποφασίσει αν είναι σε θέση να την εξυπηρετήσει. Εάν είναι όντως σε θέση να διεκπεραιώσει την αίτηση, θα πρέπει να επιστρέψει τον πόρο στον αιτούμενο client και να ενημερώσει κάποια τοπική πληροφορία. Η cache θα πρέπει να έχει μία πολιτική συνέπειας (συνάφειας) για να διατηρεί την κατάσταση επικαιρότητας (freshness information) του πόρου.

Διαφορετικές caches υλοποιούν διαφορετικές προσεγγίσεις για την απόφαση caching του πόρου. Οπως έχει ήδη επισημανθεί, τα διαφορετικά κριτήρια για την απόφαση αυτή είναι:

- Υπάρχουν απαιτήσεις του πρωτοκόλλου που καθορίζουν ότι ο συγκεκριμένος πόρος δεν μπορεί να αποθηκευτεί στην cache.
- Επιδέχεται συνήθως το περιεχόμενο caching?
- Είναι πιθανή η επαναχρησιμοποίηση του πόρου?

- Μπορεί η απόφαση για αποθήκευση της συγκεκριμένης απάντησης να οδηγήσει σε αντικατάσταση ενός ή περισσοτέρων πόρων?

Μία cache χρησιμοποιεί κάποια ή όλα τα παραπάνω κριτήρια για να αποφασίσει αν πρέπει να εφαρμόσει caching σε κάποιο πόρο.

Αντικάτασταση και Αποθήκευση Πόρων στην Cache

Μετά την απόφαση για αποθήκευση ενός πόρου, η cache ελέγχει αν μπορεί να προχωρήσει στην αποθήκευση χωρίς να διαγράψει κάποια από τα αντικείμενα που ήδη βρίσκονται στην cache. Εάν όχι, ενεργοποιείται ο μηχανισμός cache replacement (αντικατάστασης cache). Ο μηχανισμός αυτός εισάγει κάποια επιβάρυνση, ειδικά αν μικρότερα αντικείμενα που είναι ήδη cached πρέπει να διαγραφούν. Πρόσθετη επιβάρυνση προκαλείται όταν λαμβάνονται, μελλοντικά, αιτήσεις για αντικείμενα τα οποία έχουν διαγραφεί. Σε αυτήν την περίπτωση πρέπει να εγκατασταθούν νέες συνδέσεις για την άντληση τους από τους origin servers. Συχνά, πόροι οι οποίοι θεωρούνται ότι δεν είναι επίκαιροι (stale) διαγράφονται από την cache ακόμη και στην περίπτωση που αυτή δεν είναι πλήρης. Ετσι περιορίζεται η ανάγκη για ενεργοποίηση του μηχανισμού cache replacement την στιγμή διεκπεραίωσης μίας αίτησης (μείωση της υστέρησης που εκλαμβάνεται ο χρήστης).

Μόλις προκύπτει ελεύθερος χώρος, η cache εξάγει πληροφορία για το μήνυμα όπως το χρόνο τελευταίας μεταβολής καθώς και πληροφορία για την λήξη του πόρου. Επικεφαλίδες όπως οι Expire και Cache-control: max-stale μεταφέρουν πληροφορία σχετική με την λήξη (expiration) του πόρου. Αυτά τα πεδία συντελούν στο να είναι η cache συμβατή με τους περιορισμούς του πρωτοκόλλου HTTP για το βάθος χρόνου στο οποίο η απάντηση μπορεί να επιστραφεί ως σημασιολογικά έγκυρη. Μία cache η οποία είναι συμβατή με το πρωτόκολλο είναι υποχρεωμένη να εξασφαλίζει ότι οι απαντήσεις που επιστρέφει θεωρούνται από τον origin server ως επίκαιρες (fresh). Αν απουσιάζουν πληροφορίες λήξης του πόρου, η cache προσδιορίζει ευρεστικά ένα χρόνο λήξεως για να αποφασίσει πότε καθίσταται μη-εγκυρος (stale). Ο αλγόριθμος μπορεί να βασίζεται στη τιμή του πεδίου Last-Modified που είναι συνιφασμένη με τον πόρο. Για παράδειγμα, μία cache μπορεί να προσθέσει ένα καθορισμένο χρονικό διάστημα π.χ. 10’ στην τιμή Last-modified και να χρησιμοποιήσει την νέα τιμή σαν διάστημα ελέγχου επικαιρότητας (freshness interval). Επίσης, παράγεται ένα κλειδί για χρήση σε μελλοντικές αναφορές (lookups). Το κλειδί αυτό είναι μία τιμή κατακερματισμού (hash value) η οποία βασίζεται στο URL του πόρου.

Επιστροφή αποθηκευμένου πόρου

Εάν ένας πόρος που αντιπροσωπεύει κάποιο κλειδί που αναζητείται στην cache εντοπιστεί, θεωρείται ότι συνέβει ένα cache hit. Τότε, ανάλογα με την πολιτική της cache και ενδεχόμενους περιορισμούς που επιβάλλονται από πεδία επικεφαλίδας, ένας επανέλεγχος ορθότητας μπορεί να εκτελεστεί για να διαπιστωθεί εάν ο πόρος είναι επίκαιρος. Εάν ο έλεγχος αυτός αποβεί θετικός, η αίτηση ικανοποιείται από την cache. Διαφορετικά, η cache ανακτά ένα νέο αντίγραφο του πόρου και εφαρμόζει την πολιτική της για να αποφανθεί αν ο πόρος θα πρέπει να αποθηκευτεί παράλληλα με την προώθηση του στον αιτούμενο client. Εάν ο πόρος δεν εντοπιστεί στην cache, (περίπτωση cache miss) η αίτηση προωθείται.

Συντήρηση cache

Περιοδικά, μία cache μπορεί να ελέγξει εάν τα αντικείμενα που είναι αποθηκευμένα σε αυτή είναι επίκαιρα και να πυροδοτήσει την διαγραφή των «παλιών» αντικειμένων. Μία cache μπορεί επίσης να ελέγξει τον ρυθμό αιτήσεων για cached αντικείμενα για να αποφασίσει ποιοί πόροι είναι δημοφιλείς και να προβεί σε ειδικές ενέργειες για λογαριασμό τους. Για παράδειγμα, μία cache μπορεί να προελέγχει την ορθότητα – εγκυρότητα (pre-validation) για να διαπιστώσει αν τα αντικείμενα που ζητούνται περισσότερο είναι επίκαιρα. Αυτός ο προέλεγχος μπορεί να υλοποιηθεί με την HTTP Head μέθοδο που αντλεί μόνο τα μετα-δεδομένα για τον υπο συζήτηση πόρο. Μία cache μπορεί και προ-δραστικά, να επικοινωνήσει με τον origin server και να ελέγξει εάν ο πόρος έχει μεταβληθεί. Εάν έχει όντως μεταβληθεί μπορεί να εκκινήσει την διαδικασία του prefetching για να ενημερωθεί η cache.

Αντικατάσταση περιεχομένων cache

Μόλις η cache είναι πλήρης, αντικείμενα πρέπει να διαγραφούν για να δημιουργήσουν χώρο για την αποθήκευση νέων απαντήσεων. Πολλές στρατηγικές για την αντικατάσταση αντικειμένων έχουν προταθεί. Ορισμένες προέρχονται από το παραδοσιακό χώρο του cache management σε συστήματα αρχείων, ενώ άλλες είναι εξειδικευμένες στο Web περιβάλλον. Μία ιδιαίτερα γνωστή προσέγγιση είναι το Least Recently Used (LRU) – αντικατάσταση του αντικειμένου που χρησιμοποιήθηκε λιγότερο. Οι στόχοι του caching, δηλαδή η μείωση του όγκου της πληροφορίας που ανταλλάσεται στο δίκτυο καθώς και της υστέρησης που αντιλαμβάνεται ο χρήστης, οδηγούν σε σύνθετες αποφάσεις για την αντικατάσταση περιεχομένου cache. Οι σύνθετες αποφάσεις αποτελούν ένα συνδιασμό μετρικών που

περιλαμβάνουν το μέγεθος των απαντήσεων που αποθηκεύονται, τον τύπο αντικειμένου ακόμη και την έννοια της απόστασης προς τον origin server.

Η χρησιμότητα διατήρησης ενός πόρου στην cache μπορεί να υπολογιστεί από πολλούς παράγοντες όπως:

- Το κόστος ανάκτησης του πόρου: το κόστος ανάκτησης ενός πόρου από ένα origin server προσδιορίζεται από την διασυνδεσιμότητα της cache και την απόσταση που πρέπει να διανύσει ο πόρος μέχρι να καταχωρηθεί στην cache. Αντικαθιστώντας ένα πόρο του οποίου η ανάκτηση ήταν "ακριβή", το ίδιο κόστος θα πρέπει να αντιμετωπιστεί στην περίπτωση που ο πόρος ζητηθεί πάλι στο μέλλον.
- Το κόστος αποθήκευσης του πόρου: Μία cache έχει σταθερό μέγεθος και η αποθήκευση ενός αντικειμένου σημαίνει λιγότερο χώρο για άλλα αντικείμενα. Ενας μεγάλος σε όγκο πόρος καταλαμβάνει σημαντικό χώρο αλλά ενδεχόμενη αντικατάσταση του σημαίνει ότι η ανάκτηση του πάλι θα κοστίσει σημαντικά.
- Ο αριθμός των προσβάσεων στο πόρο κατά το παρελθόν: ένα αντικείμενο που έχει προσπελαστεί πολλές φορές στο παρελθόν είναι πολύ πιθανόν να προσπελαστεί και στο μέλλον και, κατά συνέπεια, είναι επωφελές να παραμείνει στην cache για μεγαλύτερο διάστημα.
- Η πιθανότητα προσπέλασης του πόρου στο άμεσο μέλλον: Εάν ο πόρος είναι πιθανόν να ανακτηθεί στο άμεσο μέλλον δεν ενδείκνυται η απόρριψη του από την cache. Η πιθανότητα πρόσβασης σε ένα πόρο θα μπορεί να είναι γνωστή a priori ή να προσδιορίζεται βάσει της ιστορικότητας προσπέλασης (access patterns).
- Ο χρόνος από την τελευταία μεταβολή του πόρου: Ενας πόρος που δεν έχει μεταβληθεί για μεγάλο διάστημα είναι λιγότερο πιθανό να αλλάξει στο κοντινό μέλλον. Ενας πόρος που παρείχθει πρόσφατα μπορεί να είναι δυναμικός ή να υπάρχει μεγάλη πιθανότητα να αλλάξει πάλι στο μέλλον. Οι πόροι που υπάρχει μεγάλη πιθανότητα να αλλάξουν είναι συνήθως δημοφιλείς. Αυτοί οι πόροι μπορούν να μεταβληθούν σαν αποτέλεσμα του δημοφιλούς τους χαρακτήρα και είναι έτσι καλοί υποψήφιοι για caching. Η αποθηκευμένη απάντηση μπορεί όπως να πρέπει να αντικατασταθεί συχνά με τον μεταβαλλόμενο πόρο. Ο χρόνος τελευταίας μεταβολής ενός πόρου μπορεί έτσι να χρησιμοποιηθεί για να προσδιοριστούν οι πιθανοί υποψήφιοι για αντικατάσταση.
- Ο χρόνος λήξης που προσδιορίζεται ευρεστικά: Εάν δεν υπάρχει χρόνος λήξης προσδιορισμένος από τον server, η cache προσδιορίζει ευρεστικά ένα χρόνο λήξης. Εάν

δεν υπάρχουν πόροι για τους οποίους έχει παρέλθει ο χρόνος λήξης, τότε αυτοί που βρίσκονται κοντινότερα στην λήξη τους αποτελούν υποψήφιους για αντικατάσταση.

Οι αλγόριθμοι που χρησιμοποιούνται ως επί το πλείστον για cache replacement είναι οι ακόλουθοι:

- Least Recently Used (LRU)
- Least Frequently Used (LFU)
- Size of object (SIZE)
- Hyper-G (LRU/LFU/SIZE): Το σύστημα Hyper-G συνδιάζει τις πολιτικές LRU/LFU και Size. Η πρώτη απόφαση για αντικατάσταση (replacement) βασίζεται στο LFU. Εάν υπάρχουν περισσότεροι από ένας πόροι που πληρούν το παραπάνω κριτήριο, εφαρμόζεται η πολιτική LRU. Εάν πάλι, δεν προσδιορίζεται ένας πόρος προς αντικατάσταση, επιλέγεται ο μεγαλύτερος σε όγκο πόρος.
- GreedyDual-Size: Ο αλγόριθμος αυτός είχε προταθεί για αντικατάσταση σελίδων στην μνήμη υπολογιστικών συστημάτων. Ο αρχικός προσανατολισμός του δηλαδή αφορούσε ένα σύστημα με σταθερό μέγεθος πόρων ενώ το κόστος ανάκτησης από την δευτερεύουσα αποθήκευση (συστήματα δίσκων) ήταν βασικότατος παράγοντας. Ο αλγόριθμος επεκτάθηκε για να καλύψει την περίπτωση την ποικιλότητα των μεγεθών των www πόρων. Ο μετασχηματισμένος αλγόριθμος συσχετίζει μία τιμή χρησιμότητας (utility value) και αντικαθιστά τον πόρο με την χαμηλότερη τιμή χρησιμότητας. Εκτός από το κόστος μεταφοράς του πόρου στην cache και το μέγεθος του, η τιμή χρησιμότητας επηρεάζεται από τον παράγοντα παλαίωσης (age factor) που ενημερώνονται καθώς πόροι απομακρύνονται από την cache.

Συνάφεια cache

Ο origin server αποφασίζει την διάρκεια για την οποία ο πόρος θα πρέπει να θεωρείται έγκυρος (freshness duration). Μία cache θα πρέπει να εξασφαλίσει ότι μία αποθηκευμένη απάντηση είναι ακόμη έγκυρη πριν να απαντήσει σε κάποιον client που αιτείται τον πόρο. Η συνάφεια της cache από ένα σημαντικό πρόβλημα. Πόλλοι σχετικοί αλγόριθμοι έχουν προταθεί κατά τα τελευταία έτη για το πρόβλημα της συνάφειας των web caches. Η ανάγκη για cache συνάφεια εξαρτάται στους πόρους και στις πολιτικές οι οποίες έχουν επιβληθεί στην cache. Οι caches μπορεί απλά να επιστρέφουν μία παλιά αποθηκευμένη τιμή μαζί με μία αιτία για την απαξίωση του πόρου. Μεταξύ των αιτίων είναι η αδυναμία εγκατάστασης

σύνδεσης προς τον origin server ή ο υψηλός φόρτος της cache. Η επικεφαλίδα Warning του HTTP/1.1 μπορεί να χρησιμοποιηθεί για να υποδηλώσει ότι επιστρέφεται μία μη-επίκαιρη απάντηση.

Το πρωτόκολλο HTTP/1.1 παρέχει πολλούς τρόπους για την διατήρηση της συνάφειας των caches. Εάν ο origin server θέσει ένα συγκεκριμένο χρόνο λήξης για ένα πόρο, ο proxy που παρέχει caching ανεξάρτητα από την σημασιολογία του πόρου, είναι υποχρεωμένος να υιοθετήσει τον ίδιο χρόνο λήξης. Η μόνη διαφοροποίηση σε αυτό είναι ο περιορισμός που μπορεί να τεθεί στο αίτημα του client μέσω επικεφαλίδας Cache-control: only-if-cached που αναγκάζει τον Proxy να επιστρέψει μία ήδη αποθηκευμένη απάντηση χωρίς να ελέγξει την ορθότητα της στον origin server. Εάν ο origin server δεν θέσει ένα χρόνο λήξης, ο Proxy μπορεί να προσδιορίσει ευρεστικά ένα χρόνο λήξης. Ο πλέον συνήθης τρόπος ελέγχου της συνάφειας στο www είναι η αποστολή ενός GET ή HEAD αιτήματος με μία επικεφαλίδα if-modified-since. Η επικεφαλίδα μεταφέρει μία χρονοσφραγίδα που δεικνύει το χρόνο τελευταίας μεταβολής του πόρου όπως αυτός υποδεικνύεται από τον origin server. Σε ορισμένες περιπτώσεις, ο χρόνος παραγωγής της απάντησης μπορεί να είναι ο χρόνος τελευταίας μεταβολής του πόρου. Οι ετικέτες οντότητας (entity tags) του HTTP/1.1 σε συνδυασμό με την επικεφαλίδα if-modified-since μπορούν να χρησιμοποιηθούν για την πραγματοποίηση ελέγχων συνάφειας. Ο origin server μπορεί να απαντήσει με ένα πλήρες αντίγραφο του πόρου ή με την απάντηση 304 Not Modified (και χωρίς σώμα στην απάντηση). Παρόλα αυτά ένας έλεγχος συνάφειας προυποθέτει πλήρη διαδοχή μηνυμάτων HTTP request/response.

Εάν το caching proxy στέλνει ένα revalidation αίτημα κάθε φορά που συμβαίνει ένα cache hit, η πολιτική καλείται strong concistency. Εάν το proxy χρησιμοποιεί ένα ευρεστικό αλγόριθμο για να αποφανθεί ένα ο πόρος είναι επίκαιρος, χωρίς να συμβουλευτεί τον origin server, η πολιτική καλείται weak consistency. Οι δύο ευρεστικοί αλγόριθμοι για έλεγχο συνάφειας είναι οι: leased-based και time-to-live.

Leased-based προσέγγιση: Η cache συμφωνεί να αποθηκεύσει ένα πόρο για συγκεκριμένο χρονικό διαστημα (περίοδος χρονομίσθωσης – lease) χωρίς να ελέγχει την ορθότητα του (revalidation). Ο server «υπόσχεται» να ειδοποιήσει την cache για ενδεχόμενες αλλαγές στον αποθηκευμένο πόρο κατά την διάρκεια της περιόδου χρονομίσθωσης. Εάν η περίοδος παρέλθει η cache μπορεί να ελέγξει την ορθότητα του πόρου ή να ανανεώσει την

χρονομίσθωση. Αυτή η προσέγγιση μεταφέρει το κόστος του revalidation στον origin server ο οποίος θα πρέπει να γνωρίζει και παρακολουθεί όλους τους proxies στους οποίους έχει υποσχεθεί ενημερώσεις. Η προσέγγιση δεν μπορεί να κλιμακωθεί αν ο origin server είναι υποχρεωμένος να ειδοποιήσει εκατοντάδες – χιλιάδες proxies.

Time-to-live προσέγγιση: Οι πόροι έχουν συσχετιστεί με ένα χρόνο λήξης αποθήκευσης. Οταν παρέλθει αυτό το χρονικό διάστημα, οι πόροι παύουν να θεωρούνται επίκαιροι. Κατά την διάρκεια της περιόδου time-to-live (TTL), η cache δεν επικυρώνει τις απαντήσεις διασώζοντας εύρος ζώνης. Η απόδοση τιμής στον χρόνο TTL μπορεί να επηρεαστεί ένα πλήθος παραμέτρων όπως ο χρόνος λήξης που αναφέρεται στην επικεφαλίδα της απάντησης, η συχνότητα αναφοράς στην απάντηση, ο χρόνος τελευταίας αλλαγής του πόρου.

Πρωτόκολλα caching

Ανάλογα με το πως οργανώνονται οι caches, μπορούν να δέχονται και να λαμβάνουν πληροφορίες για τους πόρους για τους οποίους ενδιαφέρονται. Αυτή η επικοινωνία είναι εξωτερική, ανεξάρτητη από τα request/response μηνύματα που ρέουν μεταξύ clients και origin servers. Η inter-cache επικοινωνία μπορεί να βασίζεται στο HTTP αλλά συνήθως χρησιμοποιεί εξειδικευμένα, light-weight πρωτόκολλα. Εάν ένα σύνολο από caches οργανώνεται σε ιεραρχία, μία cache μπορεί να επικοινωνήσει με τις υπόλοιπες caches στο ίδιο επίπεδο να διαπιστώσει εάν ο ζητούμενος πόρος είναι διαθέσιμος σε αυτές. Ενα ερώτημα για κάποιο πόρο μπορεί να απαντηθεί από μία ή περισσότερες caches που τυχαίνει να έχουν τον πόρο. Συχνά, η ανάκτηση ενός πόρου από μία τοπική cache είναι προτιμότερη από την ανάκτηση από τον origin server. Η αναμονή για την απάντηση από όλες τις caches στην ιεραρχία μπορεί να αυξήσει σημαντικά την υστέρηση στον χρήστη. Για την υποβοήθηση της επικοινωνίας μεταξύ των caches εξετάζονται τέσσερα πρωτόκολλα: τα Internet Cache Protocol (ICP), Cache Array Resolution Protocol (CARP), Cache Digest Protocol (CDP) και Web Cache Coordination Protocol (WCCP).

Internet Cache Protocol (ICP)

Μία cache που δεν διαθέτει τον ζητούμενο πόρο μπορεί να θέλει να ελέγξει την διαθεσιμότητα του πόρου σε μία άλλη γειτονική cache. Αυτή η επικοινωνία είναι διαφορετική από την παραδοσιακή αίτηση για ένα πόρο από τον origin server. Σε αυτή την περίπτωση οι caches αποτελούν την πηγή καθώς και τον προορισμό των ανταλλασόμενων μηνυμάτων. Ενα διαφορετικό πρωτόκολλο απαιτείται για την επικοινωνία μεταξύ των caches. Ενα από τα

πρώτα πρωτόκολλα που καθιερώθηκαν σε αυτήν την επικοινωνία είναι το Internet Cache Protocol (ICP). Το ICP είναι ένα πρωτόκολλο ερωταποκρίσεων. Το μήνυμα που στέλνεται από μία client cache είναι μία ερώτηση για το αν ο ομότιμος κόμβος έχει ένα αντίγραφο από τον πόρο που χρειάζεται η συγκεκριμένη cache. Ο δημοφιλής χαρακτήρας του ICP οφείλεται στο γεγονός ότι χρησιμοποιείται από το Squid.

Το ICP χρησιμοποιείται σε ιεραρχίες από caches, σύνολα caches που συνδέονται μεταξύ τους και κάτω από ένα κοινό γονέα. Η διαδικασία αυτή επαναλαμβάνεται, και η μετακίνηση προς τα ανώτερα επίπεδα της ιεραρχίας σημαίνει μετακίνηση προς μία περισσότερο κεντρική cache. Οι κεντρικές caches μπορεί να έχουν μια περιφερειακή (regional) cache ως άμεσο πρόγονο ενώ οι περιφερειακές cache μπορεί να έχουν μία εθνική cache ώς πρόγονο. Αν υποτεθεί ότι η cache Original Cache δεν έχει κάποιο ζητούμενο πόρο, θα σταλλούν ICP αιτήσεις (πάνω από UDP) σε όλους τους ομότιμους κόμβους ταυτόχρονα. Εάν κάποιος από τους ομότιμους κόμβους διαθέτει τον αιτούμενο πόρο, η OriginalCache θα ενημερωθεί σχετικά και θα ζητήσει την ανάκτηση του χρησιμοποιώντας HTTP. Εάν κανείς από τους ομότιμους κόμβους δεν διαθέτει τον πόρο, η OriginalCache θα προωθήσει την αίτηση στον γονέα της. Ο γονέας της OriginalCache επαναλαμβάνει την διαδικασία. Εάν καμμία από τις caches δεν διαθέτει τον πόρο, η OriginalCache θα πρέπει να προωθήσει την αίτηση στον origin server. Η φιλοσοφία βάσει της οποίας λειτουργεί το ICP είναι ότι η αποστολή των ICP queries, ακόμη και αν αυτή επαναληφθεί πολλές φορές σε διάφορα επίπεδα της ιεραρχίας, είναι σημαντικά γρηγορότερη την επικοινωνία με τον Origin Server.

Cache Array Resolution Protocol (CARP)

Το CARP καθορίζει ένα μηχανισμό μέσω του οποίου ένα σύνολο από caching proxies μπορούν να λειτουργήσουν ως μία λογικά ενιαία cache. Ο μηχανισμός χειρίζεται το σύνολο των απαντήσεων που γίνονται cached συλλογικά μεταξύ της ομάδα (array) των proxies ως μία μεγάλη cache. Μία συνάρτηση κατακερματισμού του κλειδιού (hash function) χρησιμοποιείται για να διαιρεθεί το σύνολο των URL μεταξύ των caches. Ενας client που προσπαθεί να εντοπίσει ένα cached πόρο μπορεί να κατευθύνει την αίτηση στην κατάλληλη cache εφαρμόζοντας την hash function. Η hash function χρησιμοποιεί το αιτούμενο URL καθώς και την ταυτότητα των μελών του proxy για να διαμορφώσει ένα resolution path (μονοπάτι επίλυσης). Εάν συγκριθεί με το ICP, το CARP έχει ντετερμινιστικό μονοπάτι επίλυσης της αίτησης, εξαλείφοντας έτσι την ανάγκη για μηνύματα ερωταποκρίσεων (queries). Επίσης, υπάρχουν λιγότερα διπλότυπα αποθηκευμένων πόρων στο CARP από το

ICP. Το CARP χρησιμοποιεί το HTTP καθώς και Remote Procedure Calls για την επικοινωνία μεταξύ των proxies. Ο κάθε proxy συσχετίζεται με ένα παράγοντα φορτίου (load factor) που λαμβάνεται υπόψη πριν μία αίτηση οδηγηθεί σε ένα συγκεκριμένο proxy. Το CARP διατίθεται ως προϊόν από την Microsoft.

Cache Digest Protocol (CDP)

Το Cache Digest Protocol αποτελεί μία επέκταση του ICP. Η βασική ιδέα στο Cache Digest είναι η δυνατότητα ανταλλαγής μίας περίληψης (digest) των περιεχομένων της cache. Η περίληψη αποτελεί μία ένδειξη της συλλογής των αντικειμένων σε μία cache. Οταν μία cache έχει στην διάθεση της μία περίληψη από όλους τους ομότιμους κόμβους μπορεί, πολύ εύκολα να ανατρέξει στην περίληψη και να εξετάσει εάν το αιτούμενο αντικείμενο είναι διαθέσιμο σε μία από τις caches. Εάν η διερεύνηση αυτή επιτύχει, ο συγκεκριμένος ομότιμος κόμβος είναι υποψήφιος να δεχθεί μία αίτηση ανάκτησης του πόρου. Εάν ο έλεγχος στις περιλήψεις αποτύχει, οι αντίστοιχες caches δεν ερωτούνται με προφανές αποτέλεσμα την δραστική μείωση των ερωτημάτων που απευθύνονται στο σύνολο των ομότιμων κόμβων.

Ενα πρόβλημα του μηχανισμού Cache Digest είναι η εωλότης των περιλήψεων και τα λανθασμένα ερωτήματα τα οποία οφείλονται σε αυτήν. Ενα αντικείμενο μπορεί να αφαιρεθεί από μία cache μετά την διαμόρφωση της σχετικής περίληψης. Ενα ακόμη πρόβλημα είναι το μέγεθος των περιλήψεων και η ανταλλαγή τους μεταξύ των ομοτίμων κόμβων. Οι περιλήψεις ανταλλάσονται μέσα σε HTTP μηνύματα, πάνω από το TCP, για λόγους αξιοπιστίας. Μία περίληψη μπορεί να θεωρηθεί σαν ένας κανονικός πόρος και οι τεχνικές ελέγχου της ορθότητας του HTTP (resource revalidation) μπορούν να χρησιμοποιηθούν για την διερεύνηση του επίκαιρου της περίληψης.

Web Cache Coordination Protocol (WCCP)

Το WCCP είναι ένας μηχανισμός συντονισμού, στενά δεμένος με το επίπεδο δικτύου. Ο σκοπός του WCCP είναι η παρεμπόδιση (intercept) της HTTP αίτησης και η ανακατεύθυνση της στην μηχανή cache. Επειδή η αίτηση θα αποτύχει εάν η cache δεν είναι, για κάποιο λόγο, διαθέσιμη, ένας μηχανισμός συντονισμού απαιτείται. Το αντικείμενο του μηχανισμού συντονισμού είναι να εξισορροπεί τον φόρτο μεταξύ διαφορετικών caches, έχοντας πλήρη γνώση της διαθεσιμότητας τους. Ελέγχοντας περιοδικά την διαθεσιμότητα μίας cache, ο μηχανισμός εξασφαλίζει ότι δεν πρόκειται να προωθηθούν πακέτα σε μία cache που δεν ανταποκρίνεται σε έλεγχο διαθεσιμότητας (hearbeat check). Ενας τέτοιος μηχανισμός

αποτελεί την βάση του πρωτοκόλλου WCCP που υλοποιείται σαν τμήμα της Cisco Cache Engine. Η μηχανή cache ρυθμίζεται ώστε να δέχεται WWW αιτήσεις που ανακατευθύνονται σε αυτήν από ένα δρομολογητή. Ο δρομολογητής, που έχει ενεργοποιημένο το WCCP, μπορεί να επεξεργάζεται όλες τις IP επικεφαλίδες. Ενα TCP πακέτο που στοχεύει στην θύρα 80 ανακατευθύνεται στο cache engine. Επιπλέον, οι δρομολογητές που διαθέτουν WCCP επικοινωνούν περιοδικά με τις μηχανές caching για να εξασφαλίσουν την διαθεσιμότητα τους.

Διανομή Περιεχομένου (Content Distribution)

Η τεχνική του content distribution αναφέρεται στην εφαρμογή επιλεκτικού κατοπτρισμού (selective mirroring). Η βασική ιδέα στην διανομή περιεχομένου είναι να μειωθεί ο φόρτος στον origin server. Αυτό μπορεί να επιτευχθεί παρέχοντας τμήμα ή όλο το περιεχόμενο από ένα σύνολο αντιγράφων (replicas). Διάφορες τεχνικές χρησιμοποιούνται για την ανακατεύθυνση των αιτήσεων στα αντίγραφα (π.χ., τεχνικές που βασίζονται στο DNS). Ενας τρόπος για την διαίρεση (partitioning) ενός πόρου είναι στα συστατικά βάσης (base) και εμφωλευμένων στοιχείων (embedded). Ενα έγγραφο βάσης (base document) αποτελεί το container έγγραφο και τα εμφωλευμένα συστατικά είναι οι εικόνες ή τα scripts τα οποία αποτελούν τμήμα της WWW σελίδας. Οι servers που χρησιμοποιούνται για να εξυπηρετήσουν το non-container τμήμα του πόρου καλούνται content distribution servers. Μπορεί να εντοπίζονται κοντά στον origin server ή οπουδήποτε στο διαδίκτυο. Τα εμφωλευμένα στοιχεία υπάρχουν ως αντίγραφα (replicated) σε αυτούς τους servers. Κατά την αίτηση, η υπηρεσία διανομής περιεχομένου (content distribution service) προσπαθεί να εντοπίσει τον content distribution server που βρίσκεται «πλησιέστερα» στον χρήστη για να του επιστρέψει τα εμφωλευμένα στοιχεία. Η εγγύτητα ενός content distribution server μπορεί να αναφέρεται σε γεωγραφική απόσταση, σε δικτυακή απόσταση και μετρικές υστέρισης (latency metrics). Αυτή η προσέγγιση ελαττώνει τον φόρτο στον base server και βελτιώνει τον χρόνο απόκρισης για τον τελικό χρήστη.

Ο στόχος του content distribution δεν είναι διαφορετικός από αυτόν του caching. Και οι δύο προσεγγίσεις μετακινούν το περιεχόμενο κοντά στον τελικό χρήστη φιλοδοξώντας να μειώσουν την υστέρηση που αντιλαμβάνεται ο χρήστης καθώς και τον φόρτο στον origin server. Με το caching, τα proxies πρέπει να διατηρούν την συνέπεια (consistency) και να επαληθεύουν την ορθότητα των αποθηκευμένων πόρων. Με το μηχανισμό content

distribution, οι content servers έχουν πλήρη έλεγχο επί του περιεχομένου και μπορούν να προβούν σε ρυθμίσεις με τους servers που διαθέτουν περιεχόμενο για λογαριασμό τους.

Στο μηχανισμό cache mirroring, μεγάλα τμήματα ενός δικτυακού τόπου (site) κατοπτρίζονται (mirrored) σε διάφορους κόμβους του διαδικτύου. Στην προσέγγιση content distribution, οι origin servers αποφασίζουν ποιοι πόροι μπορούν να αντιγραφούν, και μεταφέρουν το έργο του κατοπτρισμού σε άλλο οργανισμό. Οι origin servers είναι υποχρεωμένοι να ειδοποιούν την εταιρίας που αναλαμβάνει το content distribution όταν οι πόροι τους υφίστανται μεταβολές.

Ενα παράδειγμα εταιρίας που αναλαμβάνει content distribution είναι η Akamai. Ενας δικτυακός τόπος που θέλει να διανεμηθούν τα περιεχόμενα του μέσω του Akamai, μετονομάζει αυτά τα URLs με συγκεκριμένο πρόθεμα. Το πρόθεμα περιέχει το όνομα ενός κόμβου (hostname string). Η DNS επίλυση του ονόματος κόμβου επιστρέφει την IP διεύθυνση ενός Akamai mirror server που είναι πολύ πιθανό να περιέχει αντίγραφο του πόρου. Η απόφαση για την επιστροφή μίας IP διευθύνσεως λαμβάνεται από τον DNS server του Akamai δικτύου. Σχεδιαστικά, ο προσδιοριζόμενος Akamai server είναι πλησιέστερα στον τοπικό DNS server του client που αιτήθηκε τον πόρο. Η προσδοκία είναι ότι ο client είναι αρκετά κοντά στον DNS server (δικτυακή απόφαση) και ο πόρος θα πρέπει να μεταφερθεί για μία μικρή σχετικά απόσταση. Επειδή πρέπει να απεικονιστεί το όνομα κόμβου που περιέχεται στο URL, είναι δυνατό για το Akamai να χρησιμοποιήσει τον DNS server για να προσδιορίσει τον κατάλληλο Akamai server που διαθέτει τον ζητούμενο πόρο.

Η τεχνική του Akamai πρέπει να εξασφαλίσει ότι το DNS lookup επιστρέφει το πλησιέστερο mirror site. Οι DNS τιμές TTL πρέπει να ρυθμιστούν κατάλληλα ώστε να αποφευχθεί να παραμένουν cached οι DNS απαντήσεις για μεγάλο διάστημα. Διαφορετικά, ένας client που αναζητάει κάποιον κόμβο μπορεί να χρησιμοποιήσει μία παλιά IP διεύθυνση ενός Akamai server που δεν αποτελεί, πλέον, την καταλληλότερη επιλογή για τον ζητούμενο πόρο. Υπάρχει ένα trade-off μεταξύ του εντοπισμού της καλύτερης επιλογής και του κόστους των DNS lookups.

Υπάρχουν όμως ορισμένα προβλήματα με την προσέγγιση διανομής περιεχομένου. Ο origin server οφελείται από τον περιορισμένο φόρτο ενώ οι τελικοί χρήστες οφελούνται από την άντληση πόρων από «κοντινούς» χρήστες. Η θέση των content distribution servers μπορεί να

είναι ένα πρόβλημα για τους clients. Είναι δυνατόν ορισμένοι clients να έχουν καλύτερη δικτυακή συνδεσιμότητα (χαμηλότερα RTT) προς τον origin server από τους content distribution servers. Τεχνικά, οι content distribution servers εργάζονται για λογαριασμό των base servers. Οι clients εγκαθιστούν απευθείας συνδέσεις με τους content distribution servers και αναμένουν να είναι αυτοί συμβατοί με το HTTP πρωτόκολλο. Τα content distribution sites χρησιμοποιούν διαφορετικά πρωτόκολλα στην επικοινωνία τους με τους base servers και μπορούν να χρησιμοποιούν και άλλους μηχανισμούς για να διασφαλίσουν ότι διαθέτουν επικαιροποιημένο περιεχόμενο.

Αναφορές

[Krishnamurty, 2001] B. Krishnamurty, and J. Rexford, «Web Protocols and Practice», Addison Wesley, 2001.