ScorpioFS
Κατανεμημένο ομότιμο σύστημα αρχείων

Αντώνης Κουζούπης

Πανεπιστήμιο Πειραιώς
Τμήμα Πληροφορικής

19 Οκτωβρίου 2012

Τι είναι το ScorpioFS

- Σύστημα αποθήκευσης αντιγράφων ασφαλείας

  Παρέχει στο χρήστη ένα τοπικά προσαρτημένο σύστημα αρχείων το οποίο αποθηκεύει τα περιεχόμενά του στο δίκτυο.
- Δίκτυο ομότιμα συνδεδεμένων υπολογιστών

  Αποτελεί ένα δίκτυο υπολογιστών που παρέχουν στην υπηρεσία μία τοπική αποθήκη καθώς και μία λειτουργία αντιγραφής των αρχείων μεταξύ των κόμβων για την εξασφάλιση της διαθεσιμότητας των δεδομένων. Όλοι οι κόμβοι στο δίκτυο είναι ομότιμα συνδεδεμένοι (peer–to–peer).
- Κατανεμημένο σύστημα

  Το σύστημα αποθήκευσης αρχείων είναι πλήρως αποκεντρωμένο. Όλοι οι κόμβοι στο δίκτυο έχουν ισότιμα δικαιώματα. Κληρονομεί τα πλεονεκτήματα και τα μειονεκτήματα των κατανεμημένων συστημάτων.

Τι είναι το ScorpioFS

- Σύστημα αποθήκευσης αντιγράφων ασφαλείας

  Παρέχει στο χρήστη ένα τοπικά προσαρτημένο σύστημα αρχείων το οποίο αποθηκεύει τα περιεχόμενά του στο δίκτυο.
- Δίκτυο ομότιμα συνδεδεμένων υπολογιστών

  Αποτελεί ένα δίκτυο υπολογιστών που παρέχουν στην υπηρεσία μία τοπική αποθήκη καθώς και μία λειτουργία αντιγραφής των αρχείων μεταξύ των κόμβων για την εξασφάλιση της διαθεσιμότητας των δεδομένων. Όλοι οι κόμβοι στο δίκτυο είναι ομότιμα συνδεδεμένοι (peer–to–peer).
- Κατανεμημένο σύστημα

  Το σύστημα αποθήκευσης αρχείων είναι πλήρως αποκεντρωμένο. Όλοι οι κόμβοι στο δίκτυο έχουν ισότιμα δικαιώματα. Κληρονομεί τα πλεονεκτήματα και τα μειονεκτήματα των κατανεμημένων συστημάτων.

Τι είναι το ScorpioFS

- Σύστημα αποθήκευσης αντιγράφων ασφαλείας

  Παρέχει στο χρήστη ένα τοπικά προσαρτημένο σύστημα αρχείων το οποίο αποθηκεύει τα περιεχόμενά του στο δίκτυο.
- Δίκτυο ομότιμα συνδεδεμένων υπολογιστών

  Αποτελεί ένα δίκτυο υπολογιστών που παρέχουν στην υπηρεσία μία τοπική αποθήκη καθώς και μία λειτουργία αντιγραφής των αρχείων μεταξύ των κόμβων για την εξασφάλιση της διαθεσιμότητας των δεδομένων. Όλοι οι κόμβοι στο δίκτυο είναι ομότιμα συνδεδεμένοι (peer–to–peer).
- Κατανεμημένο σύστημα

  Το σύστημα αποθήκευσης αρχείων είναι πλήρως αποκεντρωμένο. Όλοι οι κόμβοι στο δίκτυο έχουν ισότιμα δικαιώματα. Κληρονομεί τα πλεονεκτήματα και τα μειονεκτήματα των κατανεμημένων συστημάτων.

Τα μέρη του ScorpioFS (Chord)(Σκατά Τίτλος!!!)

Το μέρος του *ScorpioFS* που υλοποιεί το Chord πρωτόκολλο. Είναι υπεύθυνο για την εύρεση των κόμβων που είναι αποθηκευμένα τα δεδομένα, την εισαγωγή και τη διαγραφή ενός κόμβου από το δίκτυο και για την αντιγραφή των αρχείων. Γενικά είναι υπεύθυνο για το δικτυακό κομμάτι.

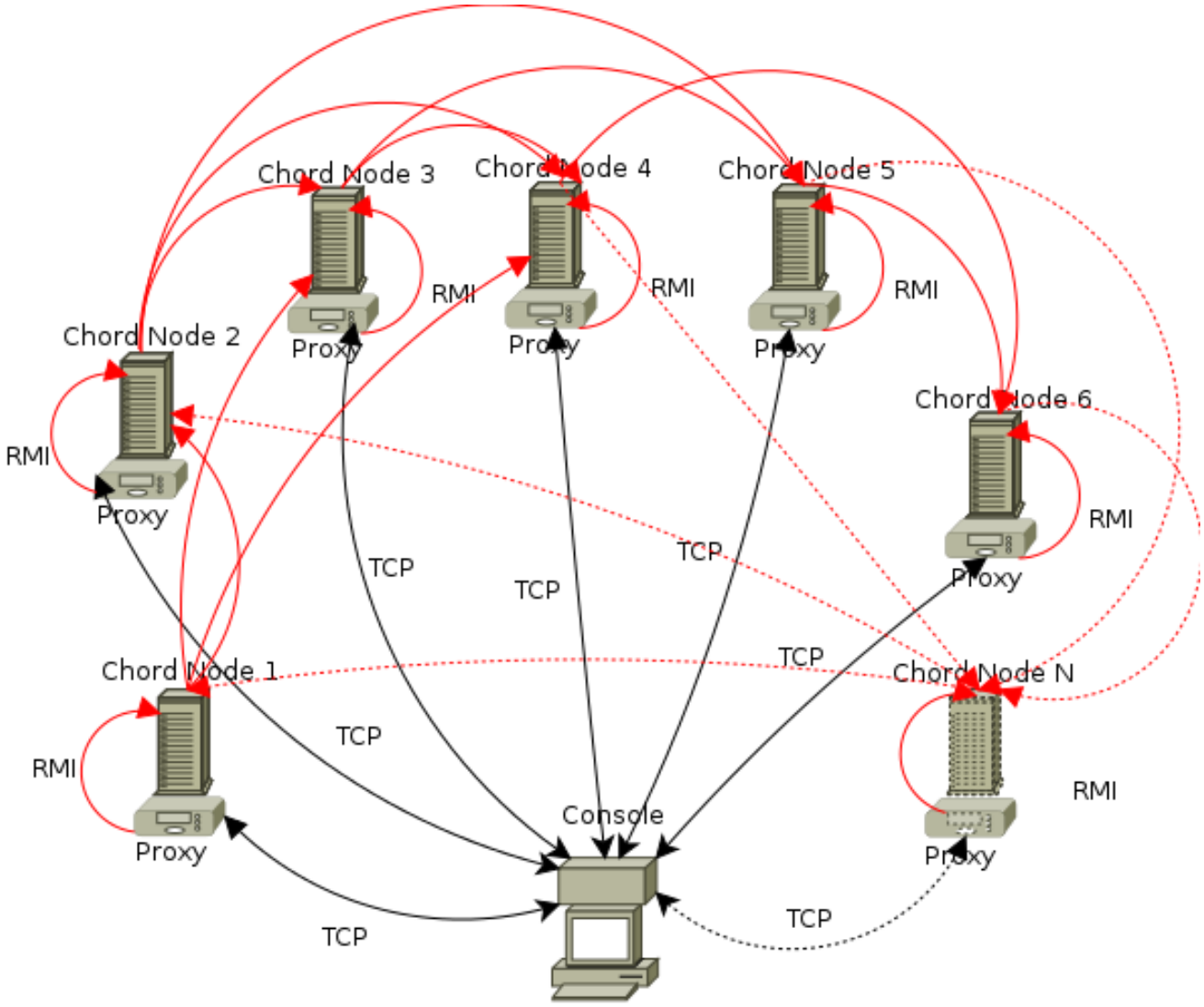
Τα μέρη του ScorpioFS (Fuse)(Σκατά Τίτλος!!!)

Υλοποιεί το τοπικό σύστημα αρχείων που αντιλαμβάνεται ο χρήστης. Υλοποιεί τις περισσότερες λειτουργίες ενός συστήματος αρχείων όπως δημιουργία, διαγραφή, επεξεργασία, αντιγραφή κτλ. Χωρίζει μεγάλα αρχεία σε μικρότερα του 1MB και επικοινωνεί με το **Chord** κομμάτι για την αποστολή και αποδοχή δεδομένων.

Τα μέρη του ScorpioFS (Console)(Σκατά Τίτλος!!!)

Κονσόλα διαχείρισης των κόμβων του δικτύου. Εκτελεί διάφορες λειτουργίες μαζικά στους κόμβους όπως δημιουργία ή καταστροφή, περισυλλογή των στατιστικών. Λειτουργεί ανεξάρτητα από το **Chord** και **Fuse** κομμάτι και επιτελεί επικουρικό ρόλο στο σύστημα.

Σχεδιάγραμμα Δικτύου



Το πρωτόκολλο Chord

- University of California, Berkeley & MIT Laboratory for Computer Science – SIGCOMM'01
- Επεκτάσιμο πρωτόκολλο για αναζήτηση σε ένα δυναμικό peer–to–peer σύστημα με συχνές αφίξεις και αναχωρήσεις κόμβων.
- Αποθηκεύει ζευγάρια key/data στον κατάλληλο κόμβο.
- Δοθέντος ενός κλειδιού το αντιστοιχίζει σε ένα κόμβο.
- *Consistent hashing* για εξισορρόπηση του φόρτου εργασίας, κάθε κόμβος είναι υπεύθυνος για περίπου τον ίδιο αριθμό κλειδιών, ελάχιστες μετακινήσεις κλειδιών όταν ένας κόμβος μπαίνει ή βγαίνει από το σύστημα.

Το πρωτόκολλο Chord

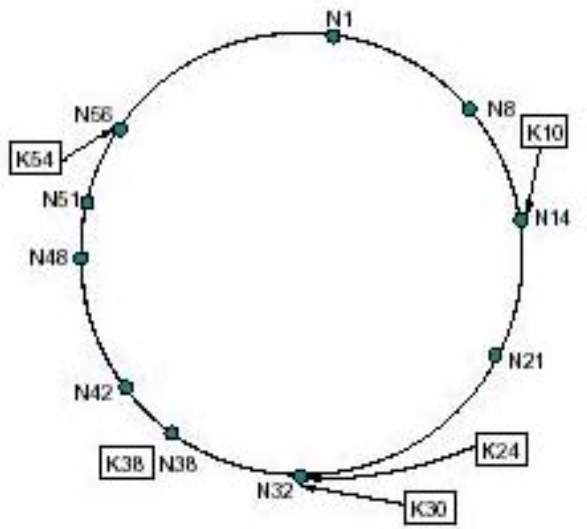
- Σε ένα σύστημα με N κόμβους, κάθε κόμβος κρατάει πληροφορία για μόνο $\bigcirc(\log N)$ άλλους κόμβους.
- Επιλύει όλες τις αναζητήσεις μέσω $\bigcirc(\log N)$ μηνυμάτων προς άλλους κόμβους.
- Το πρωτόκολλο παρέχει μία *lookup*(*key*) συνάρτηση που βρίσκει την IP διεύθυνση του κόμβου που είναι υπεύθυνος για το κλειδί.
- Το Chord ενημερώνει τους κόμβους για τις αλλαγές των κλειδιών που είναι υπεύθυνοι.
- Όταν ο N-οστός κόμβος έρθει ή φύγει από το σύστημα μόνο $\bigcirc(\frac{1}{N})$ κλειδιά μετακινούνται.

Χαρακτηριστικά του Chord

- **Load balance** – Το Chord λειτουργεί σαν κατανεμημένη συνάρτηση κατακερματισμού διαμοιράζοντας τα κλειδιά σε όλους τους κόμβους.
- **Decentralization** – Κανένας κόμβος δεν είναι πιο σημαντικός από τους άλλους. Κατάλληλο για χαλαρά συνδεδεμένες peer–to–peer εφαρμογές.
- **Scalability** – Το κόστος μιας αναζήτησης αυξάνεται λογαριθμικά σε σχέση με το πλήθος των κόμβων.
- **Availability** – Ρυθμίζει αυτόματα το δίκτυο ώστε να "κρύψει" από την εφαρμογή τις αποχωρήσεις και τις αφίξεις νέων κόμβων.
- **Flexible naming** – Δεν θέτει κάποιο περιορισμό στη μορφή των κλειδιών.

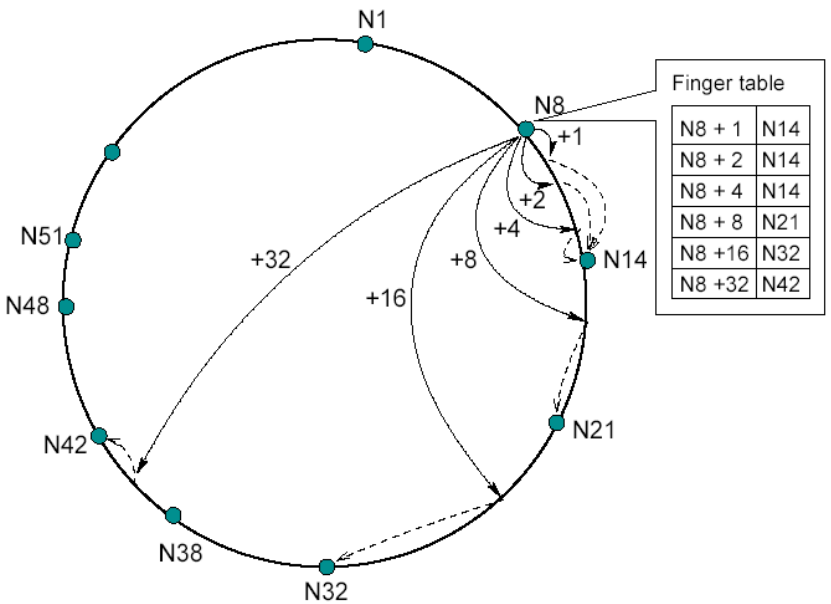
Consistent Hashing

Ένα κλειδί κ ανατίθεται στο πρώτο κόμβο που το αναγνωριστικό του ισούται ή ακολουθεί το κ. Ο κόμβος αυτός ονομάζεται *successor* κόμβος του κλειδιού κ.



Finger Table

Κάθε κόμβος n κρατάει ένα πίνακα δρομολόγησης με $m(\bigcirc(logN))$ εγγραφές που ονομάζεται *finger table*. Το i-οστό στοιχείο του πίνακα περιέχει το αναγνωριστικό του πρώτου κόμβου που επιτυγχάνει τον n κατά λιγότερο 2^{i-1}.



Finger Table

Το πρώτο στοιχείο στον πίνακα του κόμβου 8 είναι ο κόμβος 14 αφού είναι ο πρώτος κόμβος που έπεται το $(8 + 2^0) \mod 2^6 = 9$. Αντίστοιχα, το τελευταίο στοιχείο του πίνακα είναι ο κόμβος 42, καθώς είναι ο πρώτος κόμβος που έπεται του $(8 + 2^5) \mod 2^6 = 40$.

Finger Table

- Κάθε κόμβος κρατάει πληροφορία για ένα μικρό αριθμό κόμβων. Επίσης αυτοί οι κόμβοι είναι οι πιο κοντινοί του.
- Γενικά το finger table ενός κόμβου δεν περιέχει αρκετή πληροφορία ώστε να καθορίσει τον successor ενός τυχαίου κλειδιού k.
- Η διαδικασία για την εύρεση ενός successor μπορεί να γίνει αναδρομικά και στους κόμβους του finger table ενός κόμβου n, εάν το κλειδί προς αναζήτηση είναι πιο μακρυά από τον τελευταίο κόμβο στο finger table του κόμβου n.
- Καθώς κάθε κόμβος έχει εγγραφές σε διαστήματα της δύναμης του δύο, μπορεί να προωθήσει μία ερώτηση τουλάχιστον στο μισό δρόμο.

Finger Table

Εάν υποθέσουμε ότι ο κόμβος 8 θέλει να βρει τον successor για το κλειδί 54. Αφού η μεγαλύτερη εγγραφή στο finger table του κόμβου 8 που προηγείται του 54 είναι ο κόμβος 42, ο κόμβος 8 θα ρωτήσει τον κόμβο 42 να εξυπηρετήσει την ερώτηση.
Ο κόμβος 42 βρίσκει ότι η μεγαλύτερη εγγραφή που προηγείται του 54 είναι ο κόμβος 51. Ο κόμβος 51 θα βρει ότι ο successor του 54 είναι ο κόμβος 56. Τελικά ο κόμβος 51 θα επιστρέψει στον κόμβο 8 ότι ο successor του κλειδιού 54 είναι ο κόμβος 56.

Εισαγωγή Κόμβου

Ένας νέος κόμβος n εισέρχεται στο σύστημα:

1. Ο κόμβος n καλεί όποιο κόμβο γνωρίζει και του ζητάει να του βρει τον successor του.
2. Ο κόμβος n αντιγράφει όσα στοιχεία από το finger table του successor του είναι μικρότερα από n.
3. Ανά τακτά διαστήματα γίνεται "stabilize" στο δίκτυο. Ο τρέχοντας κόμβος ρωτάει τον successor του, για τον predecessor του successor του. Με αυτό τον τρόπο ένας νέος κόμβος γίνεται γνωστός στο δίκτυο.
4. Κάθε κόμβος ελέγχει περιοδικά τον predecessor του για ενημερώσει τυχών λανθασμένες εγγραφές.

Εισαγωγή Κόμβου

Ο κόμβος 26 εισέρχεται στο σύστημα μεταξύ των κόμβων 21 και 32. Ο κόμβος 26 βρίσκει τον successor του (κόμβος 32). Ο κόμβος 26 αντιγράφει όλα τα κλειδιά του successor του που είναι μικρότερα από 26. Με τη διαδικασία του "stabilize" ενημερώνεται ο κόμβος 21, ότι ο successor του είναι ο κόμβος 26

Να scan-aro και να βάλω το παράδειγμα από το paper.

Αποχώρηση Κόμβου

Για να αυξηθεί η διαθεσιμότητα του συστήματος, κάθε κόμβος κρατάει μία λίστα από successors ($\Omega(\log N)$) και όχι μόνο έναν. Εάν μία κλήση προς ένα successor αποτύχει, τότε γίνεται κλήση στον αμέσως επόμενο. Θα πρέπει να αποτύχουν όλοι οι κόμβοι για να υπάρχει δυσλειτουργία.

Ένας κακόβουλος χρήστης θα μπορούσε κάνει ορισμένους κόμβους να αποτύχουν αλλά όχι συγκεκριμένους κατ' επιλογή κόμβους.

Αποχώρηση Κόμβου

Η εφαρμογή που χρησιμοποιεί το Chord πρωτόκολλο, ScorpioFS, αντιγράφει τα δεδομένα και σε άλλους κόμβους. Έτσι για τα ίδια δεδομένα είναι υπεύθυνοι παραπάνω από ένας κόμβοι αλλά με άλλο κλειδί.

Μία οικειοθελής αποχώρηση μπορεί να χειριστεί σαν μία αποτυχία του κόμβου.

Virtual File System

- Αφαιρετικό επίπεδο πάνω από ένα πιο συμπαγές σύστημα αρχείων.
- Επιτρέπει σε εφαρμογές χρήστη (user space) να έχουν πρόσβαση σε ένα συμπαγές σύστημα αρχείων.
- Διαφανής χρήση αποθηκευτικών μέσων χωρίς ο χρήστης να καταλαβαίνει τη διαφορά.
- Το VFS προσθέτει μία διεπαφή μεταξύ του πυρήνα και του συμπαγούς συστήματος αρχείων. Συνεπώς είναι εύκολη η δημιουργία νέων συστημάτων αρχείων.

Filesystem in USErspace

- Μέρος του προγράμματος A Virtual Filesystem (AVFS), αλλά τώρα είναι ανεξάρτητο.
- Ανοιχτό λογισμικό με άδεια χρήσης GNU GPL και GNU Library GPL.
- Διαθέσιμο για Unix-like λειτουργικά συστήματα, Linux, FreeBSD, NetBSD, Mac OS X, OpenSolaris, GNU/Hurd.
- Επίσημα στον πυρήνα του Linux από την έκδοση 2.6.14

Filesystem in USErspace

- Unix kernel module που επιτρέπει σε μη εξουσιοδοτημένους χρήστες να δημιουργήσουν το δικό τους σύστημα αρχείων.
- Είναι μία "γέφυρα" μεταξύ του πυρήνα και της εφαρμογής του χρήστη.
- Χρησιμοποιεί inode cache και data buffers για βελτίωση στην ταχύτητα.
- Απλό API με bindings σε πολλές γλώσσες προγραμματισμού.
- Απλή εγκατάσταση χωρίς την μεταγλώττιση του πυρήνα.
- Ανάπτυξη με γνώμονα την ασφάλεια.
- Σταθερό!

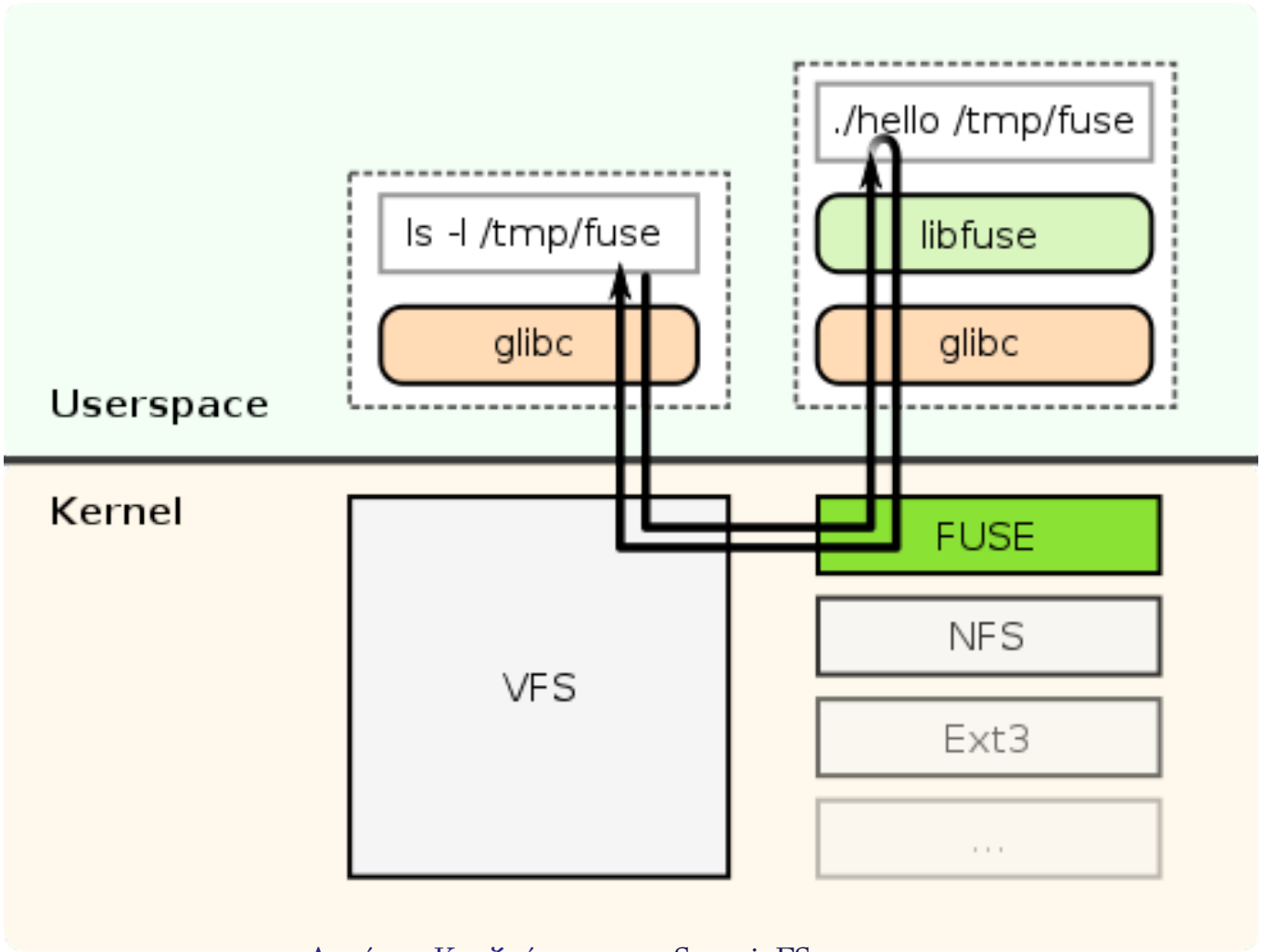
Filesystem in USErspace

Εφαρμογές που χρησιμοποιούν το FUSE:

- Wuala
- SSHFS
- NTFS-3G
- TrueCrypt
- vmware-mount
- ...

Filesystem in USErspace

Η δομή του FUSE:



FUSE-J

- Java API που παρέχει στο χρήστη bindings για το FUSE.
- Χρησιμοποιεί το framework Java Native Interface (JNI), που επιτρέπει σε κλάσεις Java να καλέσουν ή να καλεστούν (από) native προγράμματα γραμμένα σε γλώσσες όπως C, C++, assembly.
- Ανοιχτό λογισμικό με άδεια GNU Library GPL.

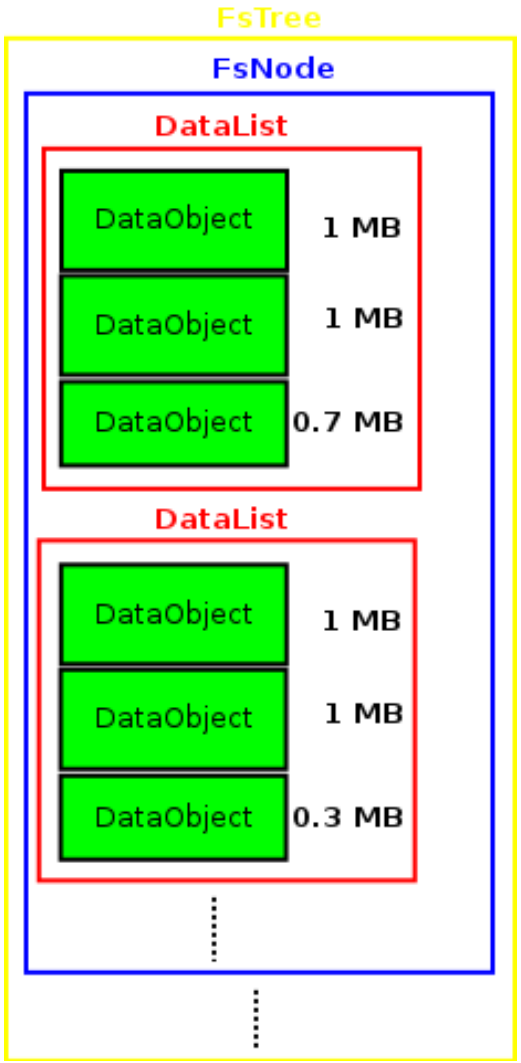
Αναπαράσταση Αρχείων

- Τα αρχεία χωρίζονται σε chunks μεγέθους 1MB.
- Τα δεδομένα είναι σε μορφή πίνακα από bytes.
- Κάθε chunk υλοποιείται από ένα instance της κλάσης *DataObject*.
- Όλα τα chunks – DataObject που αποτελούν ένα αρχείο κρατούνται στην κλάση *DataList*. Κάθε αρχείο έχει ένα instance αυτής της κλάσης.

Αναπαράσταση Αρχείων

- Κάθε "κόμβος" στο ScorpioFS αποτελεί ένα instance της κλάσης *FsNode*.
- Κρατάει διάφορες πληροφορίες για έναν "κόμβο" όπως όνομα, μέγεθος, τον πατέρα του κόμβου, *DataList*, κτλ
- Όλοι οι κόμβοι – *FsNode* s κρατούνται στην κλάση *FsTree*.
- Το *FsTree* είναι μία συλλογή από *FsNode* s. Επομένως η κλάση *FsTree* υλοποιεί το σύστημα αρχείων.
- Η κλάση *FsTree* είναι serializable, αποθηκεύεται και στέλνεται στο δίκτυο σαν ένα κανονικό αρχείο.

Αναπαράσταση Αρχείων



Έναρξη ScorpioFS

- Υπάρχει τοπικά ένα αρχείο το οποίο περιέχει τα keys από τα chunks από τα οποία αποτελείται το *FsTree*.
- Το αρχείο αυτό είναι κρυπτογραφημένο με AES-128 αλγόριθμο κρυπτογράφησης και PBKDF2 (Password-Based Key Derivation Function 2) συνάρτηση παραγωγής κλειδιού.
- Αρχικά η εφαρμογή αποκρυπτογραφεί το αρχείο και ανακτά από το δίκτυο τα chunks του *FsTree*.
- Κάνει deserialize το αρχείο, προσαρτάται ένας κατάλογος και "χτίζει" εκεί το σύστημα αρχείων.

Λήξη ScorpioFS

- Γίνεται serialize το *FsTree*.
- Εάν είναι μεγαλύτερο από 1 MB, σπάει σε μικρότερα chunks και στέλνονται στο δίκτυο όπως τα κανονικά αρχεία.
- Αποθηκεύονται τα keys των παραπάνω chunks σε ένα αρχείο.
- Κρυπτογραφείται το αρχείο και αποθηκεύεται τοπικά.
- Αποπροσαρτάται ο τοπικός κατάλογος και σταματάει το πρόγραμμα.

Εγγραφή ενός αρχείου

- Κλήση της κατάλληλης μεθόδου που επικοινωνεί με το FUSE για την εγγραφή του αρχείου στο σύστημα αρχείων.
- Δημιουργία ενός νέου *FsNode* και αρχικοποίησή του.
- Δημιουργία *DataObject* s με μέγεθος 1 MB.
- Δημιουργία *DataList* με τα παραπάνω *DataObject* s.
- Προσάρτηση του τρέχοντα κόμβου – *FsNode* στο κατάλληλο σημείο του *FsTree*.
- Αφού κλείσει ο file descriptor για το συγκεκριμένο αρχείο, τότε ξεκινάει η διαδικασία για την αποστολή του στο δίκτυο.

Εγγραφή ενός αρχείου

- Δημιουργείται το SHA-1 hash του *DataObject*.
- Ο κόμβος συμβουλεύεται το Finger Table του και σύμφωνα με το παραπάνω hash αποφασίζει ποιος κόμβος είναι υπεύθυνος για αυτό.
- Επικοινωνεί με τον κόμβο αυτό με RMI κλήση και του στέλνει το chunk.

Ανάγνωση ενός αρχείου

- Καλείται η *read* μέθοδος από το FUSE.
- Αν το συγκεκριμένο *DataObject* υπάρχει στην τοπική αποθήκη του κόμβου τότε διαβάζεται από εκεί.
- Διαφορετικά μέσω της λειτουργίας που ορίζει το Chord πρωτόκολλο, ο κόμβος βρίσκει ποιος κόμβος είναι υπεύθυνος για το chunk αυτό.
- Μέσω RMI κλήσης φέρνει το chunk και το ανοίγει.
- Υπάρχει η δυνατότητα για την ανάκτηση συγκεκριμένων chunks και όχι όλου του αρχείου.

Αντιγραφή Αρχείων

- Ανά τακτά χρονικά διαστήματα – 5 δευτερόλεπτα – ξεκινάει η διαδικασία αντιγραφής αρχείων μεταξύ των κόμβων στο δίκτυο.
- Για κάθε ένα αρχείο που έχει στην αποθήκη του ο κόμβος, υπολογίζεται πάλι το SHA-1 hash του.
- Αναζητάται ο υπεύθυνος κόμβος για το νέο πλέον key και μέσω RMI κλήσης αντιγράφεται σε αυτόν.
- Αν ο νέος υπεύθυνος κόμβος δεν είναι γνωστός για τον τρέχοντα τότε ψάχνει αναδρομικά.
- Η παραπάνω διαδικασία περιορίζεται από ένα Replication Factor.

Κονσόλα Διαχείρισης

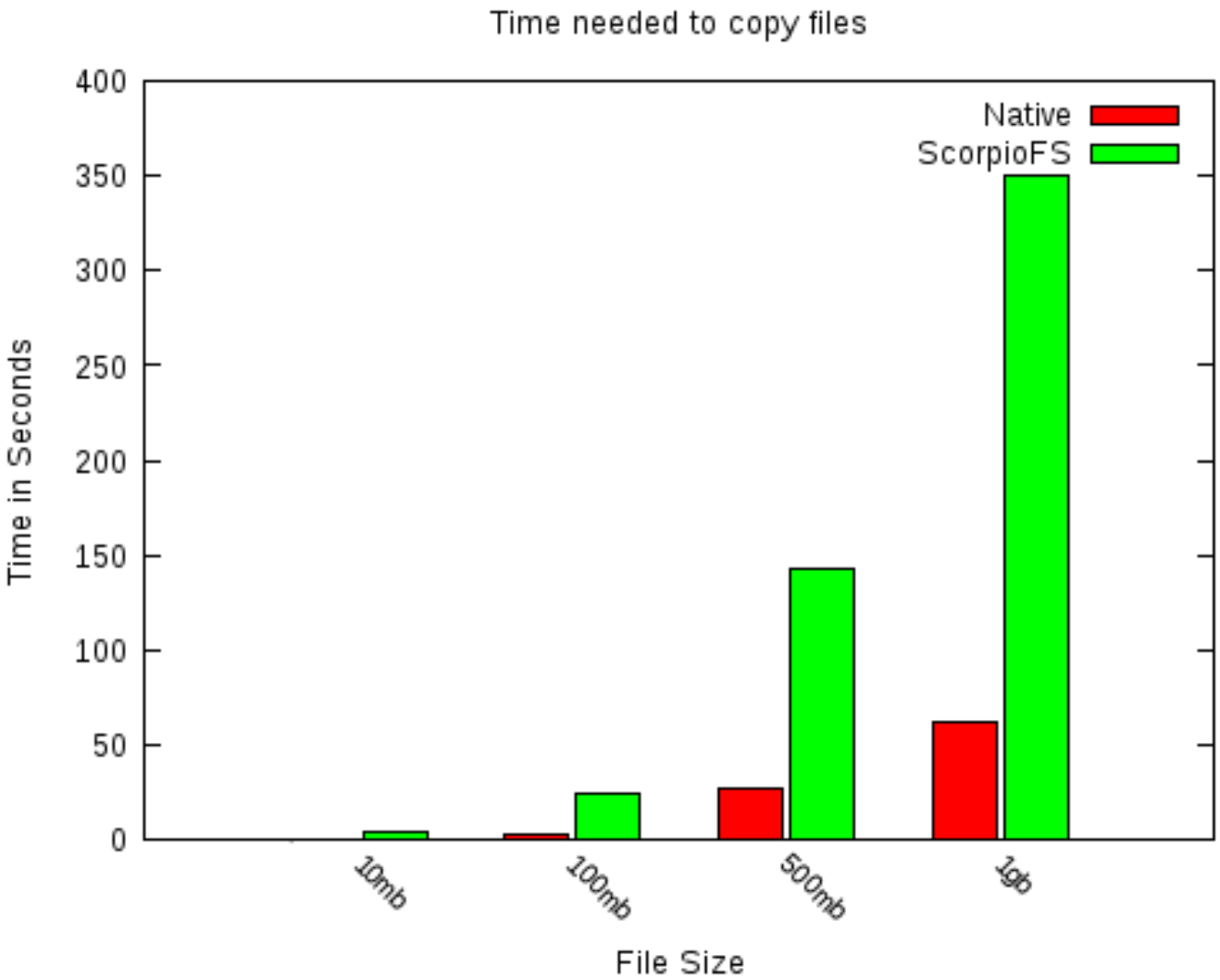
- Έχει επικουρικό ρόλο στο σύστημα. Βοηθάει για τη μαζική εκτέλεση ενεργειών στους κόμβους.
- Αρχιτεκτονική πελάτη–εξυπηρετητή χρησιμοποιώντας το TCP πρωτόκολλο επικοινωνίας.
- Η κονσόλα διαχείρισης στέλνει μηνύματα στους κόμβους.
- Σε κάθε Chord κόμβο υπάρχει ένας proxy εξυπηρετητής που λαμβάνει τα μηνύματα και τα προωθεί.
- Στην κονσόλα διαχείρισης υπάρχει ένας πολυνηματικός εξυπηρετητής που λαμβάνει τα μηνύματα των κόμβων.

Κονσόλα Διαχείρισης

Οι λειτουργίες που προσφέρει η κονσόλα διαχείρισης είναι:

- Δημιουργία κόμβων μεμονωμένα ή μαζικά.
- Σταμάτημα κόμβων μεμονωμένα ή μαζικά.
- Απαρίθμηση των ενεργών κόμβων στο δίκτυο.
- Συγκέντρωση και εξαγωγή σε αρχεία διάφορων στατιστικών από τους κόμβους.

Αντιγραφή Αρχείων



Διαγραφή Αρχείων