ΜΗΔΕΝΙΚΗΣ ΓΝΩΣΗΣ ΕΠΑΛΗΘΕΥΣΙΜΟΤΗΤΑ ΚΑΤΑΝΕΜΗΜΕΝΗΣ ΑΠΟΘΗΚΕΥΣΗΣ

Νικόλαος Σταματελόπουλος

Επιβλέπων:
Νεκτάριος Κοζύρης

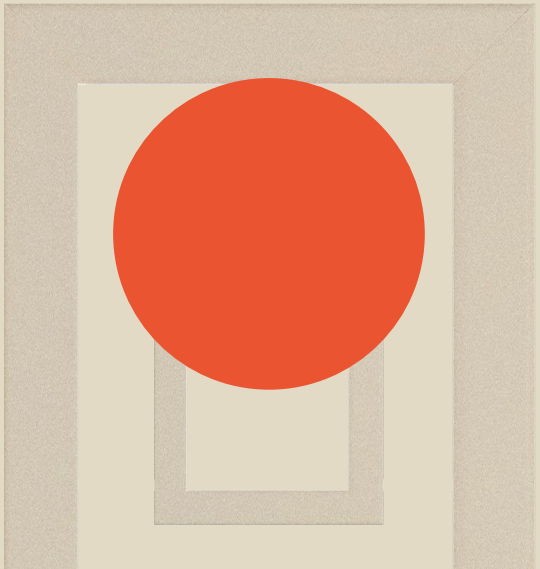
Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

01

Εισαγωγή

02

Μεθοδολογία



03

Πειραματική Αξιολόγηση

04

Συμπεράσματα

Εισαγωγή

Θεματικό πεδίο

HDFS - Κατανεμημένο Σύστημα Αρχείων

- Αποθήκευση μεγάλου όγκων δεδομένων.
- Απομακρυσμένη πρόσβαση και διαφάνεια τοποθεσίας.
- Χρήση σε κατανεμημένη επεξεργασία, Big Data

Ερώτημα

Εγγυάται την ακεραιότητα των δεδομένων;

Απάντηση

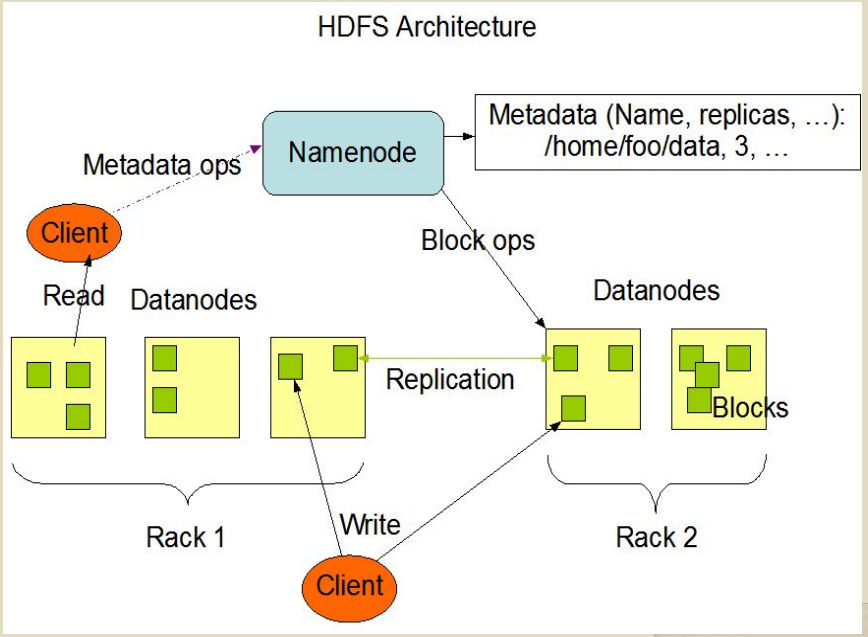
Ένας κακόβουλος κόμβος μπορεί να αποκρύψει φθορές στα δεδομένα που διαθέτει.

Σκοπός

Ενίσχυση του HDFS ώστε να ανιχνεύει φθορές στα δεδομένα, χωρίς ανάγκη εμπιστοσύνης.

HDFS - Σύντομη Επισκόπηση

- Master-Slave αρχιτεκτονική (Namenode-Datanode)
- Write-once-read-many
- Αρχεία χωρίζονται σε μπλοκ σταθερού μεγέθους.
- Διατήρηση πολλαπλών αντιγράφων (replicas), συνοδευόμενα από αθροίσματα ελέγχου.
- Περιοδική αποστολή Heartbeat και Blockreport από Datanode σε Namenode.



HDFS - Ακεραιότητα Δεδομένων

Υπάρχοντες μηχανισμοί:

- Blockreport
- BlockScanner ελέγχει κάθε αντίγραφο μπλοκ και αναφέρει τυχόν φθορές.
- Έλεγχος αρχείου από τον χρήστη κατά την ανάκτηση.

Ενδεχόμενο μη έμπιστου συμμετέχοντος κόμβου:

- Ψευδής αναφοράς κατοχής αντιγράφων κατά το blockreport.
- Η διεργασία BlockScanner μπορεί να απενεργοποιηθεί.
- Αλλαγή ή και διαγραφή των αθροισμάτων ελέγχου παράλληλα με τα δεδομένα.
- Συνεργασία κακόβουλων Namenode και Datanode για παρουσίαση αλλοιωμένων αντιγράφων στον χρήστη ως υγιή.

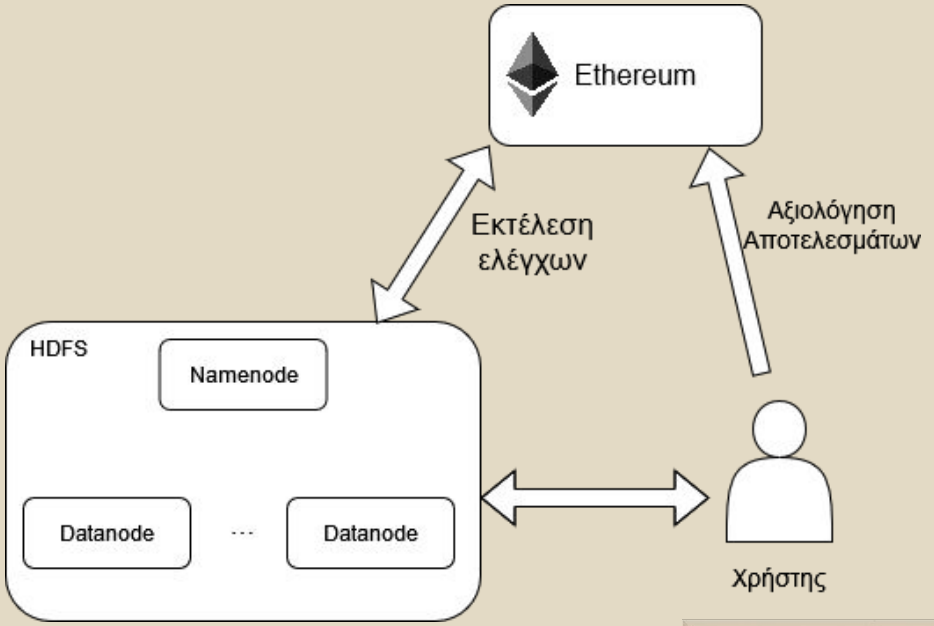
Μεθοδολογία

Εκτέλεση ελέγχων μέσω τρίτης οντότητας.

- Υποβολή “αποδείξεων” από τους Datanodes.
- Αποτελέσματα διαθέσιμα προς αξιολόγηση από χρήστες.

Γιατί Ethereum Blockchain;

- Έξυπνα συμβόλαια (smart contracts).
- Δεν απαιτεί εμπιστοσύνη (trustless).
- Επαληθευσιμότητα και μονιμότητα συναλλαγών.
- Συναλλαγές και αποτελέσματα δημοσίως διαθέσιμα.



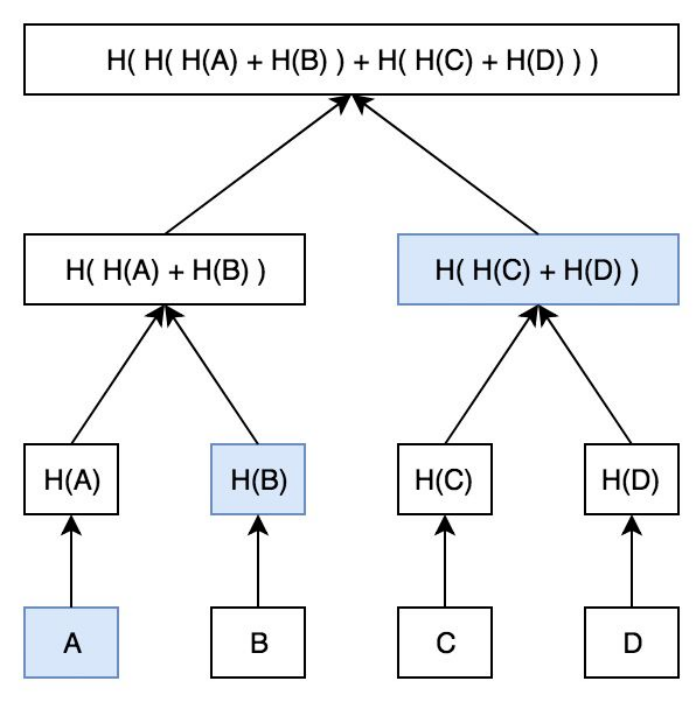
Merkle Proofs

Βασικά χαρακτηριστικά:

- Απόδειξη ακεραιότητας τμήματος δεδομένων και συμμετοχής σε ευρύτερο σύνολο.
- Δεν απαιτείται αποκάλυψη ολόκληρου του συνόλου δεδομένων.
- Δημοσιοποίηση Merkle root - ταυτότητα του συνόλου δεδομένων.

Εφαρμογή των Merkle proofs στα μπλοκ του HDFS:

1. Χωρισμός του μπλοκ σε μικρότερα τμήματα σταθερού μεγέθους (chunks) και κατασκευή Merkle tree.
2. Αποκάλυψη των δεδομένων του chunk (που επιλέχθηκε) και των γειτονικών κόμβων στο μονοπάτι προς τη ρίζα.
3. Έλεγχος μέσω ανακατασκευή μονοπατιού και σύγκριση με τη δημοσιευμένη ρίζα.



zk-SNARKs

Zero-Knowledge Succinct Non-interactive ARguments of Knowledge

- Απόδειξη ύπαρξης και κατοχής γνώσης χωρίς την ανάγκη αποκάλυψης αυτής.
- Μη διαλογική διαδικασία κατασκευής και ελέγχου των αποδείξεων.
- Μικρές αποδείξεις και σύντομη επαλήθευση.

Στάδια πρωτοκόλλου:

1. G(lambda, C) = (pk, vk)
2. P(pk, x,w) = proof
3. V(vk, x, proof) = {true, false}

zk-SNARKs - ZoKrates

Toolbox για zk-SNARK:

- Εκτέλεση επιμέρους zk-SNARK σταδίων.
- Κατασκευή έξυπνου συμβολαίου για επαλήθευση zk-SNARK απόδειξης στο Ethereum.
- Ορισμός προγράμματος ελέγχου Merkle proof σε γλώσσα υψηλού επιπέδου.
- Όμως επιβάλλεται Merkle tree και chunk σταθερού μεγέθους.

merkle_proof(index, merkleRoot, nonce, private chunk, private siblings[])

Πρόβλημα: Δυνατότητα επαναχρησιμοποίησης απόδειξης για επίλυση του ίδιου προβλήματος.

Λύση: **Χρήση nullifier** ως δημόσια είσοδος.

Random Number Generator

RNG(seed, k, max) => $\{a_1, \ldots, a_k\}$, $a_i \in [0,max-1]$

- Επιλογή των προς απόδειξη τμημάτων δεδομένων μέσω γεννήτριας ψευδοτυχαίων αριθμών.
- Γεννήτρια διαθέσιμη σε Datanodes και έξυπνο συμβόλαιο.
- Ξεχωριστός σπόρος για κάθε Datanode, που κατασκευάζεται στο Ethereum.

Σπόρος = randomness || timestamp || datanodeAddress [|| blockId]

Έξυπνο Συμβόλαιο

- Στοιχεία που αποθηκεύονται στο storage:
  - **blockId => merkleRoot**
  - **datanodeAddress => seed** (randomness || timestamp)

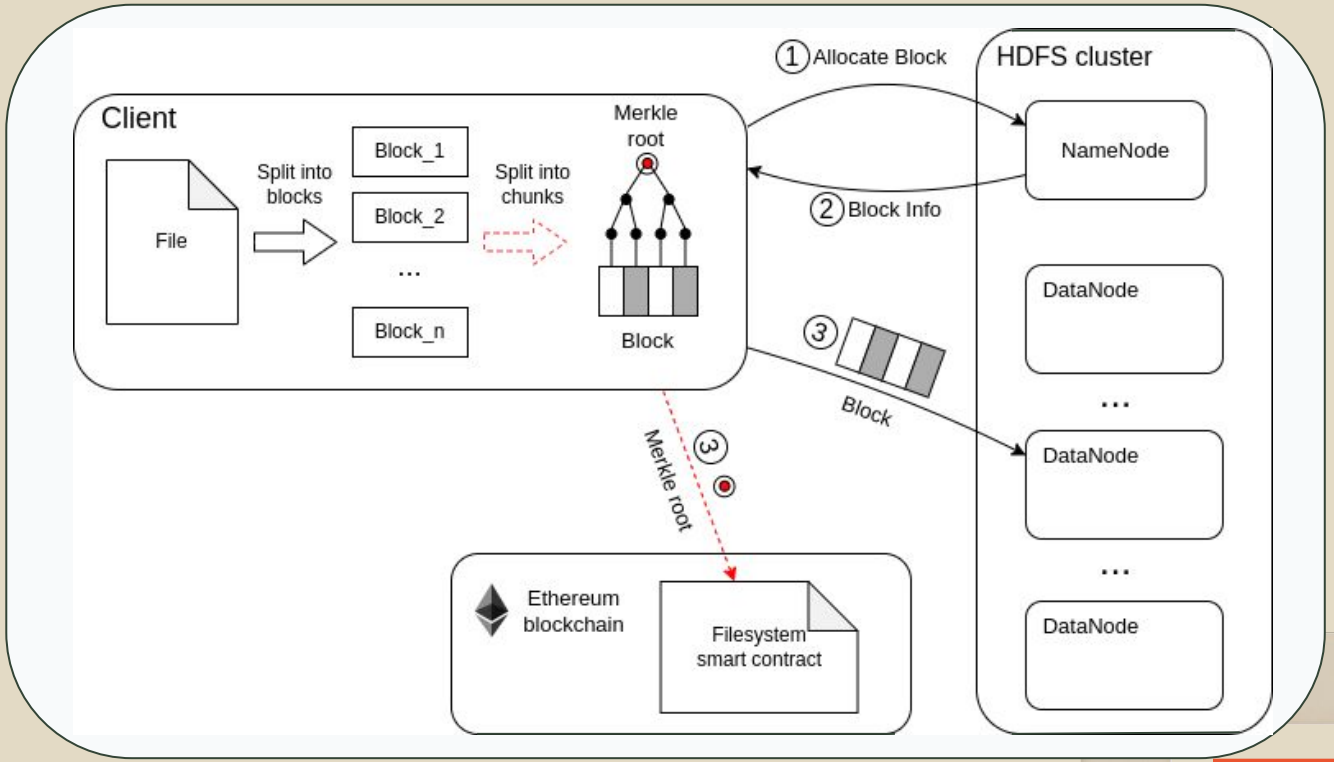
- Διεπαφές συναρτήσεων:
  - **add_root(blockId, merkleRoot)**
  - **create_seed(randomness) και peek_seed()**
  - **verify(blockId, proofs[])**

- **event(datanodeAddress, timestamp, blockId, healthStatus)**

Κάθε Datanode διαθέτει ξεχωριστό Ethereum Account.
Ως διεύθυνση του Datanode λαμβάνεται το δημόσιο κλειδί του Ethereum Account του.

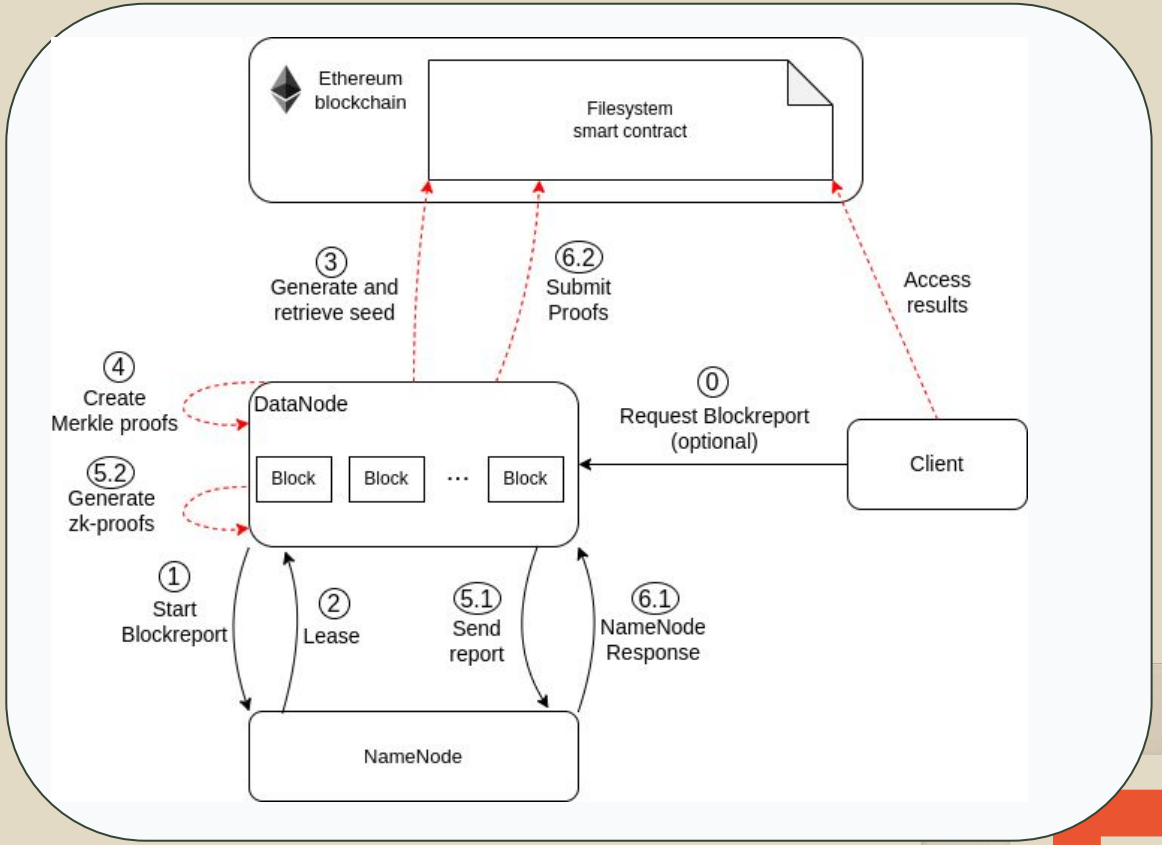
Εγγραφή αρχείου

- Αίτηση προσθήκης νέου μπλοκ και λήψη τοποθεσίας.
- Ανάγνωση και αποστολή περιεχομένου.
- Κατασκευή Merkle tree.
- Συναλλαγή για αποθήκευση Merkle root στο έξυπνο συμβόλαιο.



Blockreport

- Αίτηση εκκίνησης Blockreport από Datanode.
- Δημιουργία και ανάκτηση σπόρου.
- Κατασκευή Merkle proofs και προσωρινή αποθήκευση.
- Αποστολή blockreport στον Namenode.
- Κατασκευή zk-SNARK αποδείξεων και υποβολή για επαλήθευση
- Αποτελέσματα διαθέσιμα προς τους χρήστες.



Γιατί δουλεύει ;

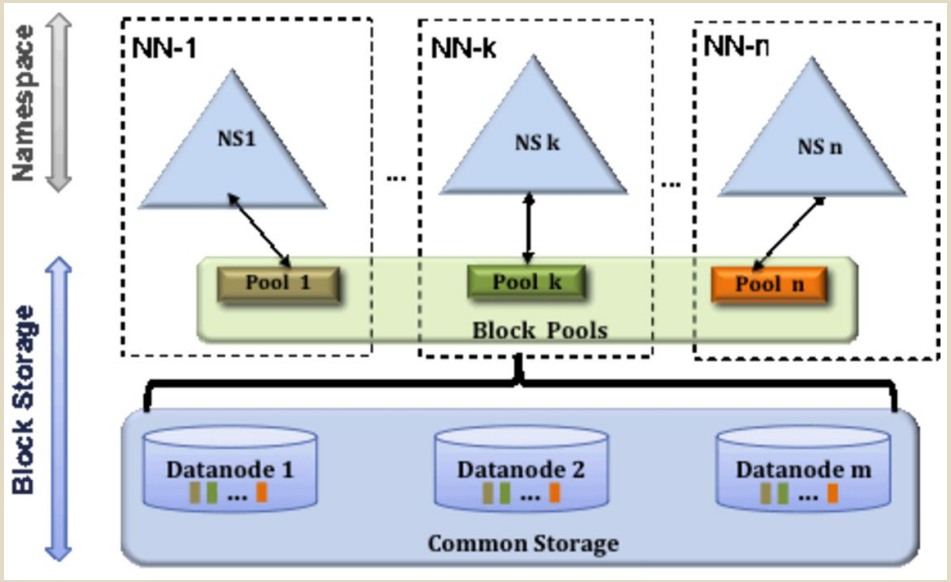
- Υπολογισμός και αποθήκευση Merkle Root από χρήστη:
  - προτού τα δεδομένα φύγουν από την κατοχή του.
  - τα δεδομένα δεν τροποποιούνται άρα ούτε και το Merkle root.
- Για κάθε αντίγραφο μπλοκ στο HDFS υποβάλλονται k πλήθους αποδείξεων :
  - Ελέγχεται ποσοστό δεδομένων: **(k * chunk_size * 100) / block_size %**
  - Οι προκλήσεις επιλέγονται ψευδοτυχαία.
  - Υποβολή αποδείξεων για διαφορετικές προκλήσεις αποτυγχάνει.
- Μοναδική ελευθερία του Datanode:
  - Αποφυγή υποβολής αποδείξεων για μπλοκ.

Federation cluster

- Επέκταση πλήθους χώρων ονομάτων μέσω προσθήκης Namenode
- Χώροι ονομάτων μεταξύ τους ανεξάρτητοι ενώ μοιράζονται εξίσου όλους τους Datanodes
- Διάκριση μεταξύ χώρων ονομάτων μέσω αναγνωριστικού BlockpoolId

Γενίκευση συστήματος:

- Χρήση διαφορετικού σπόρου για κάθε χώρο ονομάτων από τους Datanode.
- Αντιστοίχιση ζεύγους BlockpoolId-BlockId σε Merkle root.



Αξιολόγηση Συστήματος

- Ποσοτική αξιολόγηση (χρόνος εκτέλεσης)
- Απλή συστάδα HDFS
- Σταθερό μέγεθος μπλοκ (64MBytes)
- Μέγεθος chunk και πλήθος αποδείξεων παραμετροποιήσιμα
- Ιδιωτικό Ethereum Blockchain με geth (≈24 sec για εισαγωγή νέου μπλοκ)

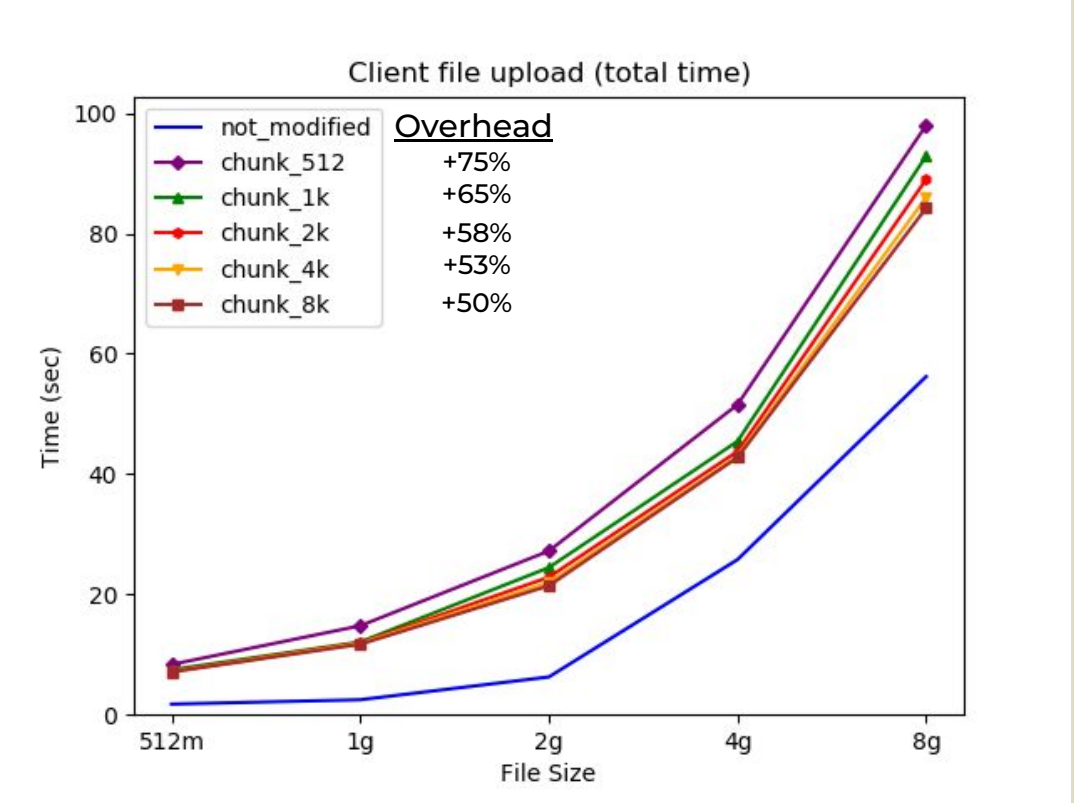
Εγγραφή Αρχείου

Στην πλευρά του χρήστη:

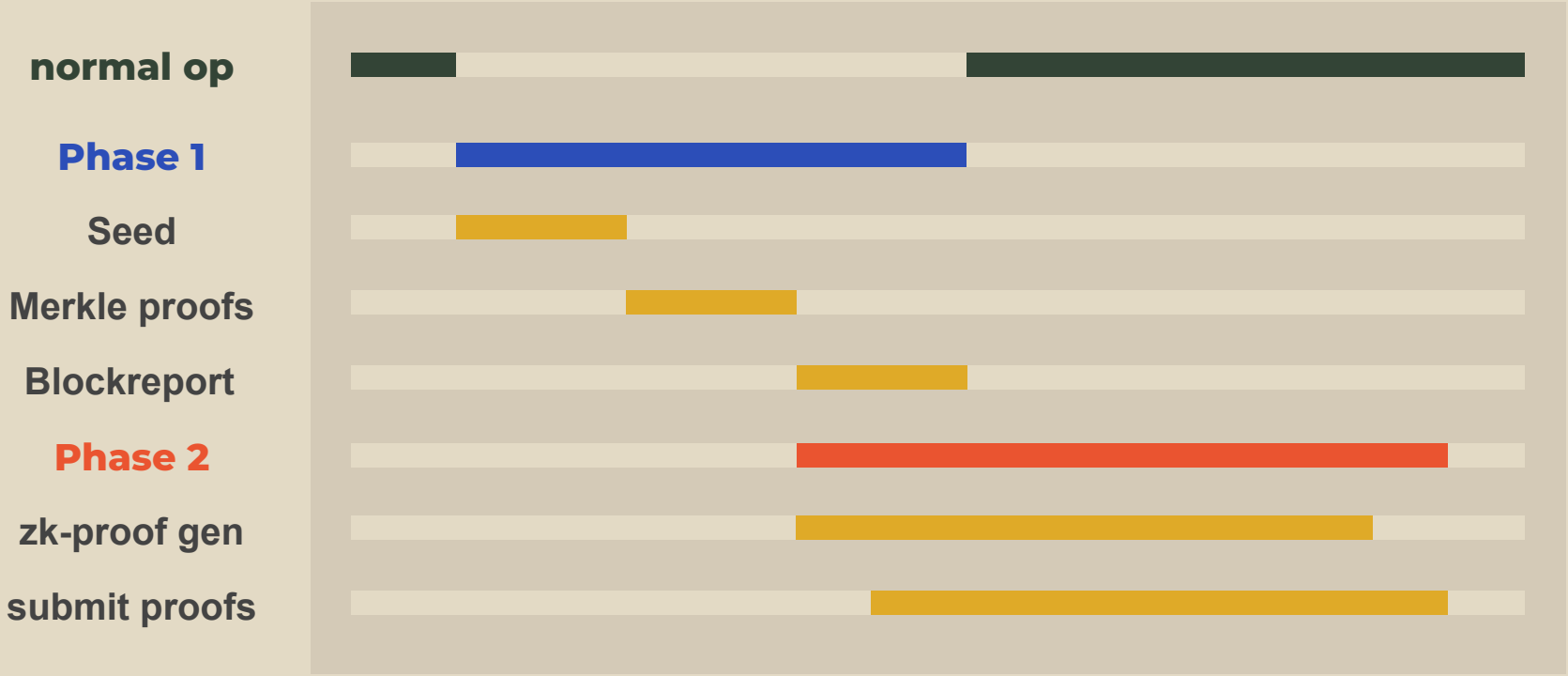
- Μεταβολή μεγέθους chunk
- Αύξηση πλήθους δεδομένων

Παρατηρήσεις:

- ➢ Καθυστέρηση λόγω κατασκευής Merkle Trees
- ➢ Μικρότερο chunk => μεγαλύτερο Merkle tree
- ➢ Αμελητέα επιβάρυνση από το blockchain



Blockreport (χρονοδιάγραμμα)

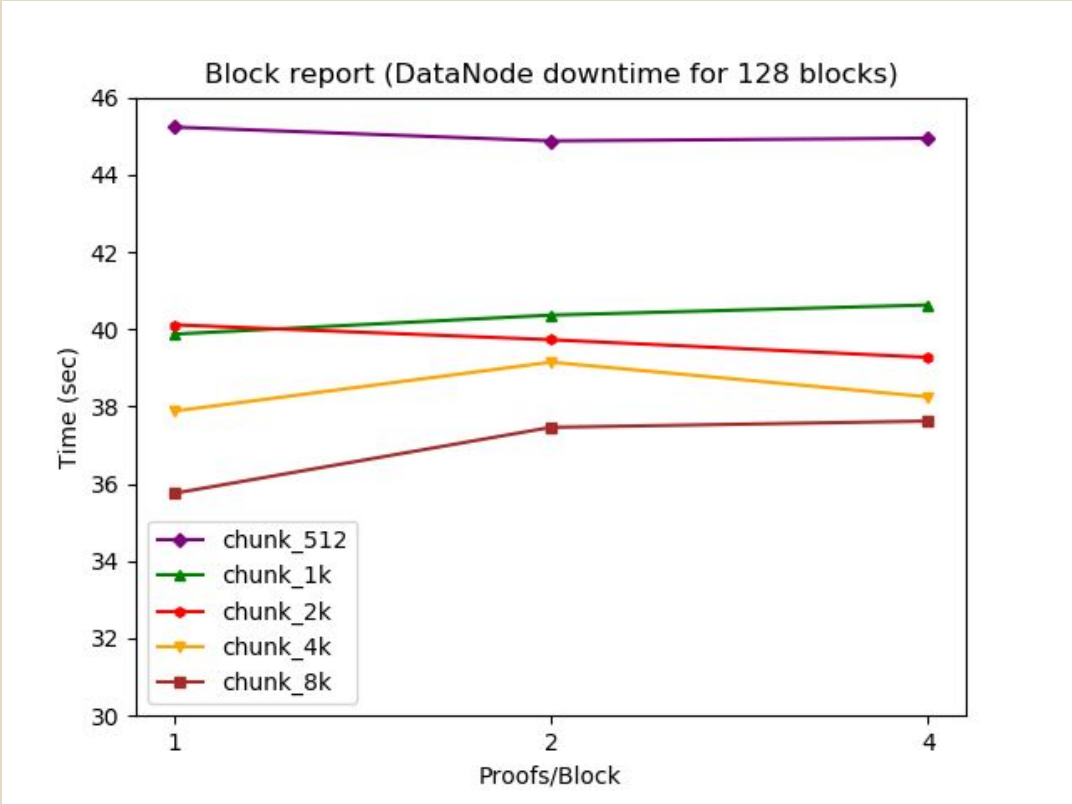


Blockreport (Namenode)

- Datanode διαθέτει 128 μπλοκς (8GBytes)
- Εκτέλεση με 4 νήματα
- Χωρίς τροποποιήσεις < 1 sec

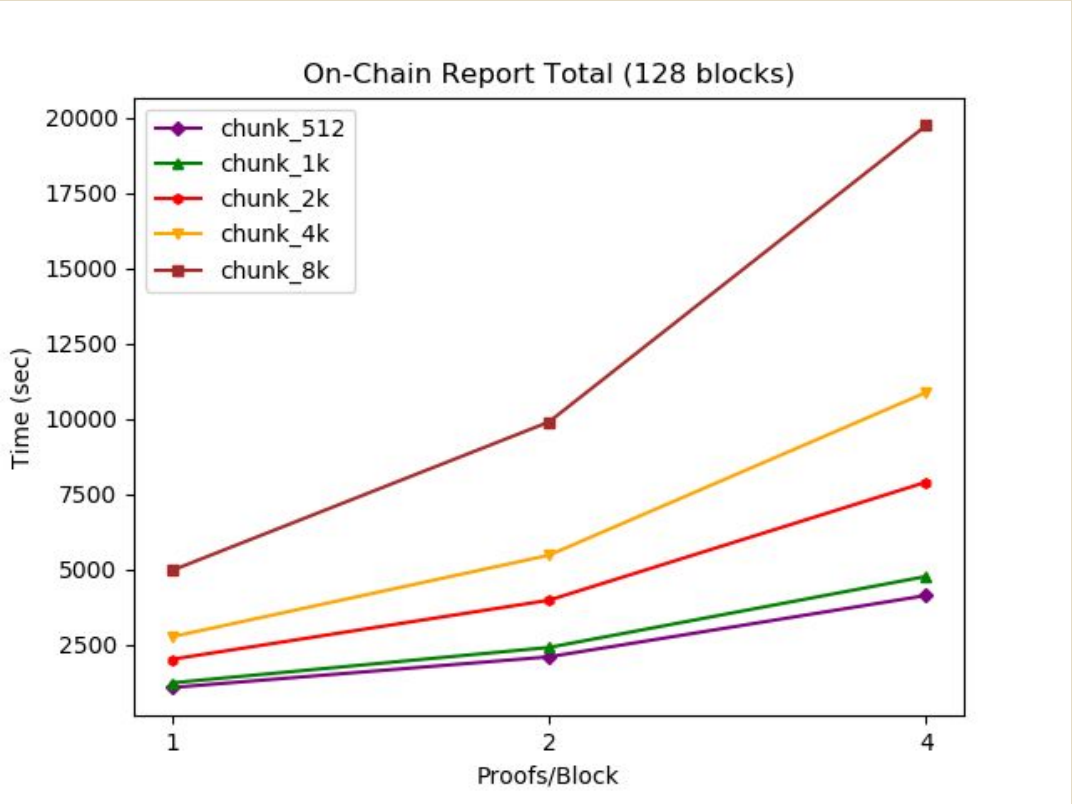
Παρατηρήσεις:

- ➢ 15 sec για τον σπόρο
- ➢ Μικρότερο chunk => μεγαλύτερη καθυστέρηση
- ➢ Σχεδόν αμελητέα καθυστέρηση με αύξηση των αποδείξεων



Blockreport (Total)

- Κυριαρχεί ο χρόνος κατασκευής των zk-SNARK αποδείξεων.
- Χρόνος εκτέλεσης γραμμικά ανάλογος με πλήθος αποδείξεων.
- Μικρότερο chunk πιο γρήγορο
- 8KBytes chunk with 4 proofs per block ≈ 5.5 ώρες



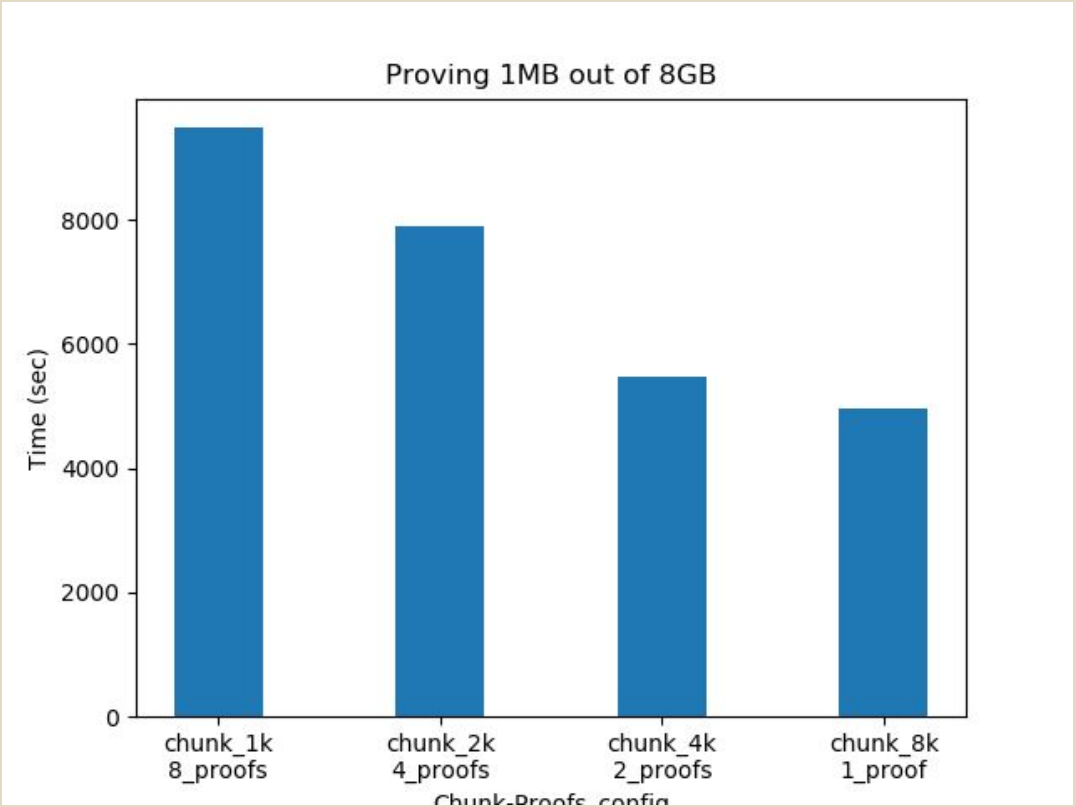
Blockreport (Coverage)

- Διατήρηση ποσοστού προς επαλήθευσης
- Αντίστοιχη μεταβολή chunk και πλήθους αποδείξεων

Παρατηρήσεις:

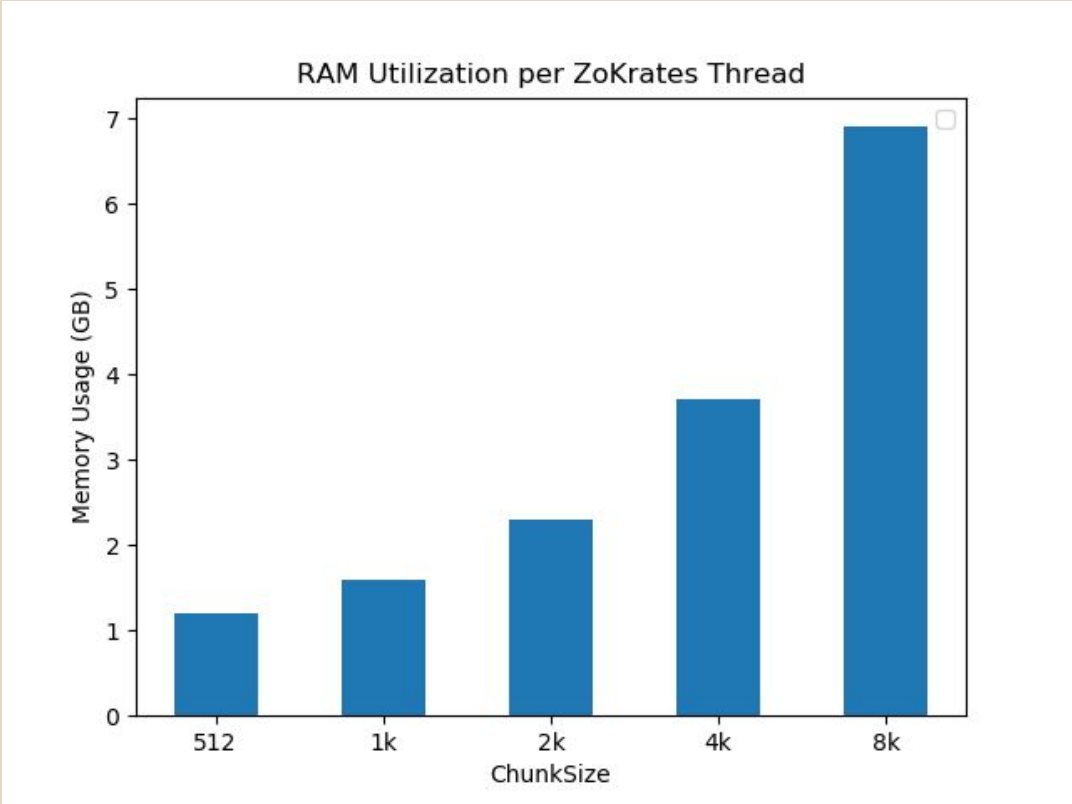
➢ Μεγαλύτερο chunk πιο αποδοτικό

➢ Μείωση chunk => μεγαλύτερο μονοπάτι στο Merkle tree



Blockreport (RAM)

- Μεγαλύτερο chunk απαιτεί περισσότερη μνήμη
- Μείωση chunk δε συνεπάγεται γραμμική μείωση μνήμης.
- Για 4 νήματα και 8KBytes chunk ≈ 28GBytes



Συμπεράσματα

- Ενίσχυση του HDFS μέσω δημοσίου κατάστιχου που εκθέτει φθορές στα δεδομένα.
- Ορθότητα ελέγχων:
  - Συναρτήσεις κατακερματισμού (RNG, Merkle proofs)
  - zk-SNARK
  - Ethereum Blockchain
- Πληρότητα συστήματος?
  - Υιοθέτηση zero-tolerance πολιτικής (ό,τι δεν αποδεικνύεται αμφισβητείται)
- Εμπιστευτικότητα δεδομένων:
  - Τα δεδομένα δε φεύγουν ποτέ από την κατοχή των Datanodes.
- Όμως μεγάλες υπολογιστικές απαιτήσεις για zk-SNARK

Βιβλιογραφία

- HDFS Architecture Guide. https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html
- What is a Block Scanner in HDFS? https://data-flair.training/forums/topic/what-is-a-block-scanner-in-hdfs/
- Merkle proofs Explained. https://medium.com/crypto-0-nite/merkle-proofs-explained-6dd429623dc5
- How does Ethereum work, anyway? https://preethikasireddy.medium.com/how-does-ethereum-work-anyway-22d1df506369
- What are zk-SNARKs? https://z.cash/technology/zksnarks/

Ευχαριστώ

Q&A

CREDITS: This presentation template was created by Slidesgo, including icons by Flaticon, infographics & images by Freepik and illustrations by Storyset

Please keep this slide for attribution