Τεχνολογίες Blockchain και Αποκεντρωμένες Εφαρμογές

Ι. Μαυρίδης, Π. Φουληράς
MSN Lab (http://msnlab.uom.gr)

Διάλεξη #09

Πώς λειτουργεί το Ethereum

Περιεχόμενα

- Από το BTC στο Ethereum
- Επισκόπηση του Blockchain στο Ethereum
  - Λογαριασμοί, Κλειδιά & Διευθύνσεις
  - Συναλλαγές
  - Μπλοκ Συναλλαγών, Tries & Σχετικές Δομές, EVM
  - Gas
  - Κόμβοι και Πελάτες
  - Δίκτυα
  - Μηχανισμοί Συναίνεσης
- Έξυπνα Συμβόλαια στο Ethereum
- Το Οικοσύστημα του Ethereum

Εισαγωγή

- Μπλοκ είναι “φουρνιές” (batches) συναλλαγών περιλαμβάνοντας ένα Hash του προηγούμενου μπλοκ, δημιουργώντας κρυπτογραφικά μία αλυσίδα από μπλοκ (Blockchain)
- Ο λόγος οργάνωσης συναλλαγών σε μπλοκ είναι ότι πρέπει να συμφωνούν όλοι οι συμμετέχοντες στο Blockchain σε απόλυτα συγκεκριμένη ιστορία συναλλαγών με αποδοτικό τρόπο (και σε χώρο και σε χρόνο)
- Ο λόγος οργάνωσης σε αλυσίδα είναι ότι οποιαδήποτε αλλαγή σε ένα μπλοκ, οδηγεί σε ακύρωση εγκυρότητος όλων των επομένων
  - Για παράδειγμα Blockchain, κάντε κλικ εδώ

Άλλοι Λόγοι Χρήσης Μπλοκ

- Δεκάδες ή εκατοντάδες συναλλαγών πρέπει να συλλεγούν, προταθούν, συμφωνηθούν και συγχρονισθούν ταυτόχρονα από όλους τους κόμβους
- Με την οργάνωσή τους σε μπλοκ δίδεται ικανός χρόνος σε όλους τους κόμβους να το κάνουν με συναίνεση
  - Αν και δεκάδες αιτήματα συναλλαγών λαμβάνουν χώρα ανά δευτερόλεπτο, τα διάφορα μπλοκ τοποθετούνται στο Ethereum, 1 φορά κάθε 12 δευτερόλεπτα

Βασική Διαδικασία

- Κάθε μπλοκ δημιουργείται από κάποιον κόμβο-Επικυρωτή (Validator)
- Αυτό διαδίδεται σε όλο το δίκτυο, με όλους τους κόμβους να το τοποθετούν στο τέλος του Blockchain (τοπικό αντίγραφο)
- Εκλέγεται νέος κόμβος-Validator για να δημιουργήσει νέο μπλοκ, σύμφωνα με το πρωτόκολλο PoS

Πρωτόκολλο PoS – (1)

- PoS (Proof-of-Stake) σημαίνει τα εξής:
  - Οι κόμβοι-Validators πρέπει να ποντάρουν 32 ETH σε ένα καταθετικό συμβόλαιο ως εγγύηση για κακή συμπεριφορά. Έτσι διευκολύνεται η προστασία του δικτύου επειδή αποδεδειγμένα ανέντιμη δραστηριότητα οδηγεί μέρος ή ολόκληρη την εγγύηση σε καταστροφή
  - Σε κάθε θυρίδα (κάθε 12 δευτερολεπτα) επιλέγεται τυχαία ένας Validator ως ο προτείνων (νέο) μπλοκ. Αυτοί επιλέγουν συναλλαγές, τις εκτελούν και προσδιορίζουν μία νέα “κατάσταση” (state). Συγκεντρώνουν όλες αυτές τις πληροφορίες σε ένα μπλοκ και το προωθούν σε άλλους Validators

Πρωτόκολλο PoS – (2)

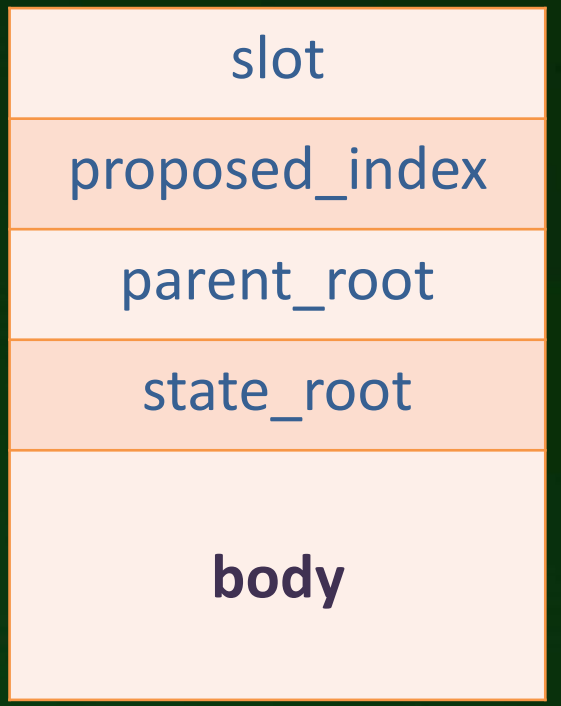
…συνέχεια:

- Άλλοι Validators που λαμβάνουν το μπλοκ επανεκτελούν τις συναλλαγές για επιβεβαίωση ότι συμφωνούν με την προτεινόμενη μεταβολή της καθολικής κατάστασης και εφόσον έγκυρο το προσθέτουν στην δική τους βάση δεδομένων

- Εάν ένας Validator μάθει ότι υπάρχουν δύο αντικρουόμενα μπλοκ, υποψήφια για την ίδια θυρίδα, χρησιμοποιεί τον αλγόριθμο “fork-choice” (περισσότερα αργότερα), για να επιλέξει το μπλοκ που υποστηρίζεται από τα περισσότερα ETH που έχουν ποντάρισθεί

Πεδία Μπλοκ

- Στο υψηλότερο επίπεδο υπάρχουν τα παρακάτω πεδία:
  - slot: η θυρίδα στην οποίαν ανήκει το μπλοκ
  - proposed_index: ID του Validator που προτείνει το μπλοκ
  - parent_root: το Hash του προηγούμενου μπλοκ
  - state_root: το root Hash του αντικειμένου κατάστασης (state object)
  - **body**: ένα αντικείμενο με πολλά πεδία, όπως ορίζονται στην συνέχεια



Πεδία Αντικειμένου “body” – (1)

- Τα πεδία αυτά είναι τα εξής:
  - randao_reveal: μία τιμή για να επιλεγεί ο προτείνων το επόμενο μπλοκ
  - eth1_data: πληροφορίες σχετικά με το καταθετικό συμβόλαιο
  - graffiti: αυθαίρετα δεδομένα για σήμανση μπλοκ
  - proposer_slashings: λίστα προτεινομένων Validators για τιμωρία
  - attester_slashings: λίστα Validators για τιμωρία (“to be slashed”)

Συνεχίζεται… 

Πεδία Αντικειμένου “body” – (2)

- …συνέχεια:
  - attestations: λίστα επικυρώσεων υπέρ του τρέχοντος μπλοκ
  - deposits: λίστα νέων καταθέσεων στο καταθετικό συμβόλαιο
  - voluntary_exits: λίστα Validators που βγαίνουν από το δίκτυο
  - sync_aggregate: υποσύνολο Validators που χρησιμοποιούνταιυ για να εξυπηρετούν light clients (κόμβους)
  - execution_payload: συναλλαγές που περάσθηκαν από τον πελάτη (κόμβο) εκτέλεσης

Πεδίο ‘attestations’

- Το πεδίο attestations περιέχει μία λίστα με όλες τις συνηγορίες (attestations) μέσα στο μπλοκ

- Κάθε συνηγορία έχει τον δικό της τύπο δεδομένων που περιέχει διάφορα δεδομένα. Τα πεδία της είναι:
  - aggregation_bits: λίστα Validators που μετείχαν σε αυτήν την συνηγορία
  - data: container με πολλαπλά υπο-πεδία
  - signature: σωρευτική υπογραφή όλων των Validators που συνηγόρησαν

Πεδίο ‘data’ του κάθε Attestation

- Το πεδίο data περιέχει τα παρακάτω (υπο-πεδία):
  - slot: η θυρίδα με την οποία σχετίζεται η συνηγορία
  - index: δείκτες για Validators που επικυρώνουν
  - beacon_block_root: το root Hash του μπλοκ Beacon που περιέχει το αντικείμενο αυτό
  - source: το τελευταίο σημείο ελέγχου με συνηγορίες
  - target: το πιο πρόσφατης εποχής συνοριακό (boundary) μπλοκ

Block μετά από Εκτέλεση Συναλλαγών του

- Εκτελώντας τις συναλλαγές στο πεδίο execution_payload ενημερώνει την καθολική κατάσταση (global state)
- Όλοι οι πελάτες (κόμβοι) επανεκτελούν τις παραπάνω συναλλαγές ώστε να διασφαλίσουν ότι η νέα κατάσταση ταιριάζει με εκείνην που υπάρχει στο πεδίο state_root του νέου μπλοκ
  - Άρα ότι το μπλοκ είναι έγκυρο και μπορούν να το προσθέσουν στο τοπικό αντίγραφο του Blockchain
- Το ίδιο το execution payload είναι αντικείμενο αρκετών πεδίων. Επίσης υπάρχει και ένα execution_payload_header που περιέχει σημαντικές περιληπτικές πληροφορίες για τα δεδομένα εκτέλεσης

‘execution_payload_header’ – (1)

- Περιέχει τα εξής πεδία:
  - parent_hash: Hash του γονικού μπλοκ
  - fee_recipient: δνση λογαριασμού για πληρωμή αμοιβών συναλλαγής
  - state_root: root Hash για την καθολική κατάσταση αφού πραγματοποιηθούν οι μεταβολές στο μπλοκ αυτό
  - receipts_root: Hash του receipts trie των συναλλαγών
  - logs_bloom: δομή δεδομένων που περιέχει event logs

Συνεχίζεται… ☞

‘execution_payload_header’ – (2)

…συνέχεια:

- prev_randao: τιμή που χρησιμοποιείται για τυχαία επιλογή Validator
- block_number: Ο αριθμός του τρέχοντος μπλοκ
- gas_limit: μέγιστη επιτρεπόμενη ποσότητα gas σε αυτό το μπλοκ
- gas_used: πραγματικά χρησιμοποιούμενη ποσότητα gas σε αυτό το μπλοκ
- timestamp: χρονοσφραγίδα του μπλοκ

Συνεχίζεται… ☞

‘execution_payload_header’ – (3)

…συνέχεια:

- extra_data: αυθαίρετα πρόσθετα δεδομένα ως ωμά byte
- base_fee_per_gas: τιμή της αμοιβής βάσης (base fee)
- block_hash: Hash του εκτελούμενου μπλοκ
- transactions_root: Hash του root συναλλαγών στο σώμα (payload)

‘execution_payload’ – (1)

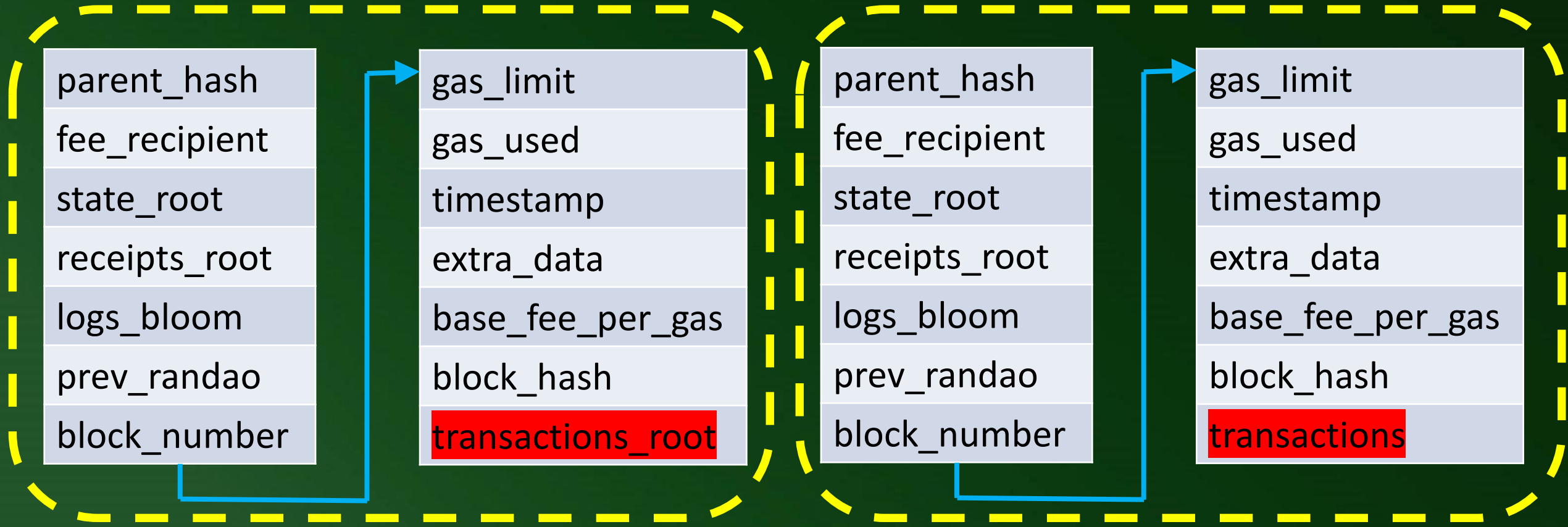
- Περιέχει τα εξής πεδία:
  - parent_hash: Hash του γονικού μπλοκ
  - fee_recipient: δνση λογαριασμού για πληρωμή αμοιβής συναλλαγής
  - state_root: root Hash για την καθολική κατάσταση μετά την εφαρμογή των μεταβολών σε αυτό το μπλοκ
  - receipts_root: Hash του trie αποδείξεων συναλλαγών
  - logs_bloom: δομή δεδομένων που περιέχει event logs
  - prev_randao: τιμή που χρησιμοποιείται για επιλογή τυχαίου Validator
  - block_number: ο αριθμός του τρέχοντος μπλοκ

Συνεχίζεται… ☞

‘execution_payload’ – (2)

- …συνέχεια:
  - gas_limit: μέγιστο επιτρεπόμενο gas στο μπλοκ αυτό
  - gas_used: το πραγματικά χρησιμοποιηθέν gas στο μπλοκ αυτό
  - timestamp: χρονοσφραγίδα του μπλοκ
  - extra_data: οιοδήποτε τύπου πρόσθετα δεδομένα ως ωμά byte
  - base_fee_per_gas: η τιμή αμοιβής βάσης (base fee)
  - block_hash: Hash του μπλοκ εκτέλεσης
  - transactions: λίστα συναλλαγών που πρόκειται να εκτελεσθούν

‘execution_payload_header’ .vs. ‘execution_payload’



Χρόνος Μπλοκ (Block Time)

- Χρόνος μεταξύ διαδοχικών μπλοκ
- Στο Ethereum ο χρόνος διαιρείται σε διαστήματα 12’’ που ονομάζονται “θυρίδες” (“slots”)
- Σε κάθε θυρίδα επιλέγεται ένας Validator για να προτείνει ένα μπλοκ
  - Εάν ο συγκεκριμένος Validator δεν είναι διαθέσιμος (π.χ. offline), δεν προτείνεται μπλοκ κατά την συγκεκριμένη θυρίδα

Μέγεθος Μπλοκ

- Κάθε μπλοκ έχει περιορισμένο μέγεθος:
  - Κάθε μπλοκ έχει μέγεθος-στόχο 15 εκατομμυρίων gas, που αυξομειώνεται σύμφωνα με τις απαιτήσεις του δικτύου, έως το όριο των 30 εκατομμυρίων gas (2x μέγεθος-στόχος)
  - Το συνολικό gas που ξοδεύεται από όλες τις συναλλαγές στο μπλοκ πρέπει να είναι < από το όριο gas του μπλοκ, διασφαλίζοντας ότι το μέγεθος οιουδήποτε μπλοκ δεν μπορεί να είναι αυθαίρετα μεγάλο
    - Σημαντικό, γιατί οι πλέον ασθενείς κόμβοι δεν θα ανταποκρίνονταν στις απαιτήσεις του δικτύου, από πλευράς χώρου και κυρίως ταχύτητας για έγκαιρη εκτέλεση των απαιτούμενων ενεργειών

Μπλοκ και base fee

- Η base fee υπολογίζεται από έναν τύπο που συγκρίνει το μέγεθος του προηγούμενου μπλοκ με το μέγεθος-στόχο
- Εάν μέγεθος τρέχοντος μπλοκ > μέγεθος-στόχο, τότε base fee αυξάνεται το πολύ κατά 12,5%/μπλοκ, άρα οικονομικά μη βιώσιμο να παραμείνει συνέχεια σε τέτοια μεγέθη (για λεπτομέρειες δείτε τον Πίνακα)

Tries: Δένδρα στο Ethereum

- Οι αναγκαίες πληροφορίες πρέπει να αποθηκεύονται έτσι ώστε να μπορούν να γίνουν γρήγορα – O(log(n)) – οι αναγκαίες αναζητήσεις, με κρυπτογραφικά αυθεντικοποιήσιμο τρόπο
- Για αυτόν τον σκοπό χρησιμοποιούνται τα λεγόμενα Merkle Trees (1 για την ακρίβεια) στο BTC
- Για το Ethereum χρησιμοποιείται μία τροποποιημένη μορφή τους (4 τέτοια για την ακρίβεια, εκ των οποίων 3 στην κεφαλίδα κάθε μπλοκ συναλλαγών)

Trie – (digital tree, prefix tree, radix tree)

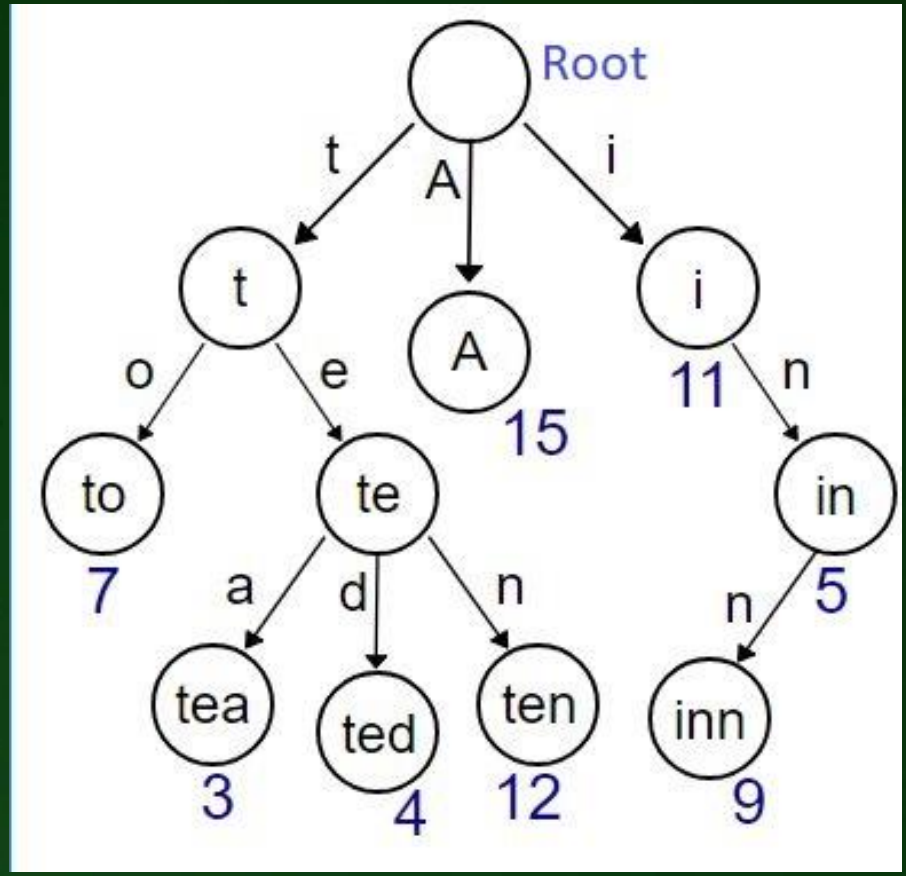
- Είδος δένδρου αναζητήσεων, όπου αντί να αποθηκεύει σε κάθε κόμβο το κλειδί που σχετίζεται με αυτόν, είναι η θέση του κόμβου μέσα στο όλο δένδρο που ορίζει το κλειδί με το οποίο αυτός σχετίζεται

- Όλοι οι απόγονοι κάθε κόμβου έχουν κοινό πρόθεμα (prefix) ως προς την συμβολοσειρά με την οποία αυτός σχετίζεται
  - Η ρίζα σχετίζεται πάντοτε με την κενή συμβολοσειρά

- Χρησιμοποιείται για την αποθήκευση Συσχετιστικού Πίνακα (Associative Array) όπου τα κλειδιά είναι συνήθως συμβολοσειρές

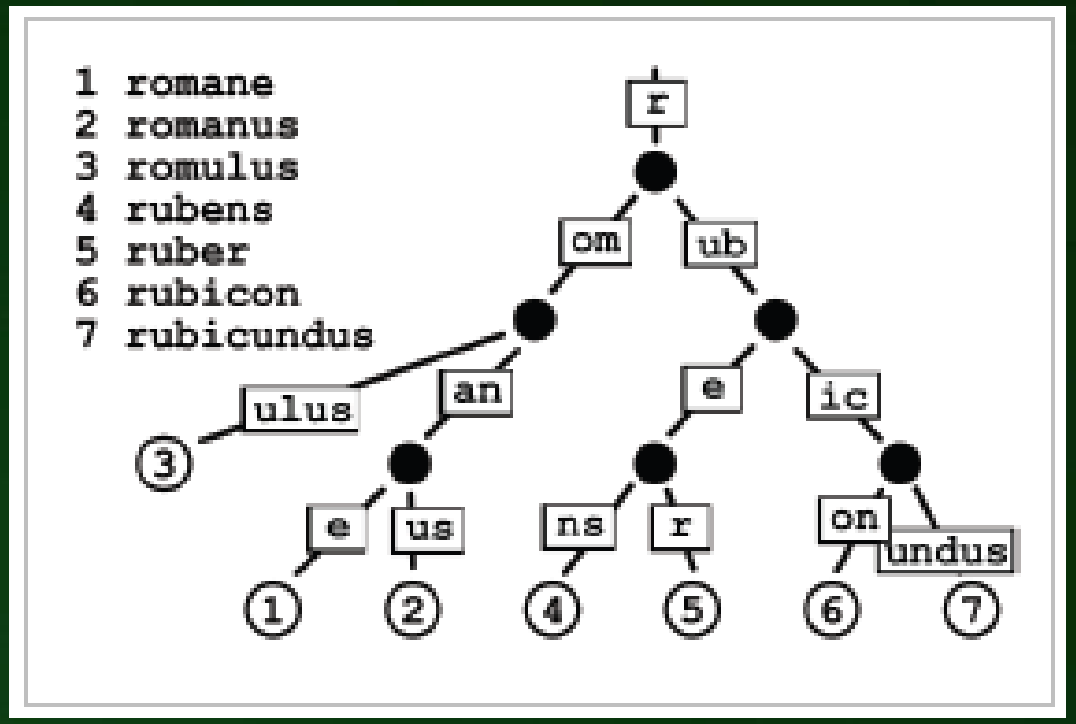
Παράδειγμα Trie

- Τα κλειδιά φαίνονται μέσα στους κόμβους
- Κάθε λέξη έχει μία αυθαίρετη ακέραιη τιμή με την οποίαν σχετίζεται



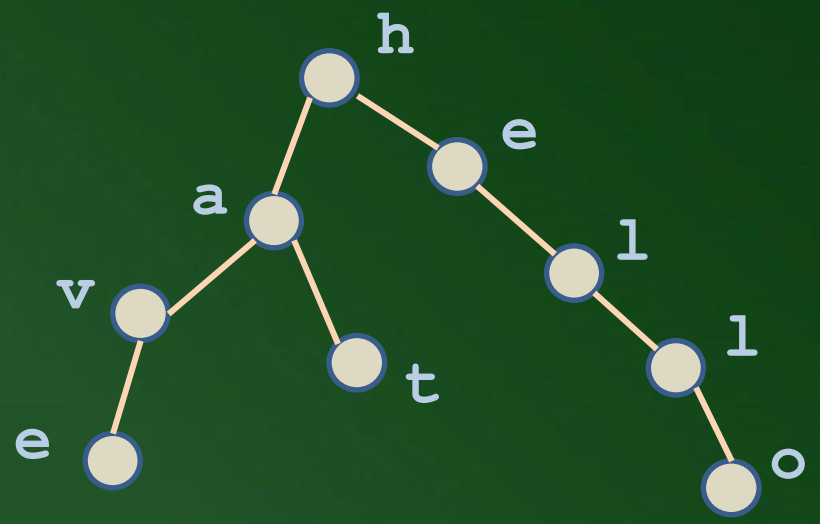
Radix Tree

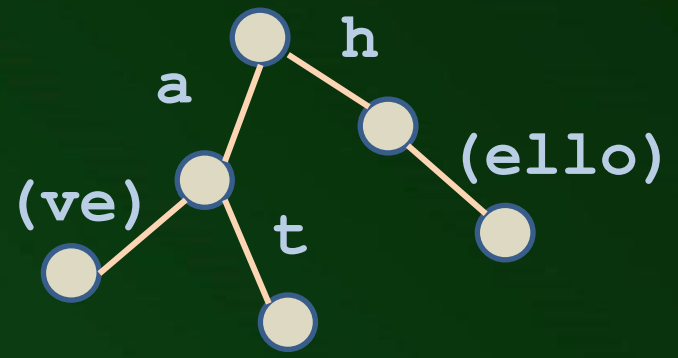
- …ή Radix Trie ή Compact Prefix Trie
- Δομή δεδομένων Trie συμπιεσμένη (βελτιστοποιημένη ως προς χώρο)
- Κάθε κόμβος που είναι το μοναδικό παιδί, συγχωνεύεται με τον γονέα του
- Αποτέλεσμα: αριθμός παιδιών κάθε *έσω κόμβου* ≤ *r*, όπου *r* το radix του radix tree, και *r* δύναμη του 2, με *r* ≥ 2



Παράδειγμα: Radix Trie καλύτερο από Απλό Trie

- Έστω οι λέξεις: `Hello, hat, have`
- Με απλό Trie θα χρειαζόμασταν 9 κόμβους, με Radix Trie μόνον 5





Radix Tries – Λεπτομέρειες

- Κάθε κόμβος έχει την μορφή:

  $[i_0, i_1, \ldots, i_n, \text{value}]$

  όπου i είναι κωδικοί αλφαριθμητικών (π.χ. ASCII) και value η τιμή που έχει αποθηκευθεί στον κόμβο αυτόν

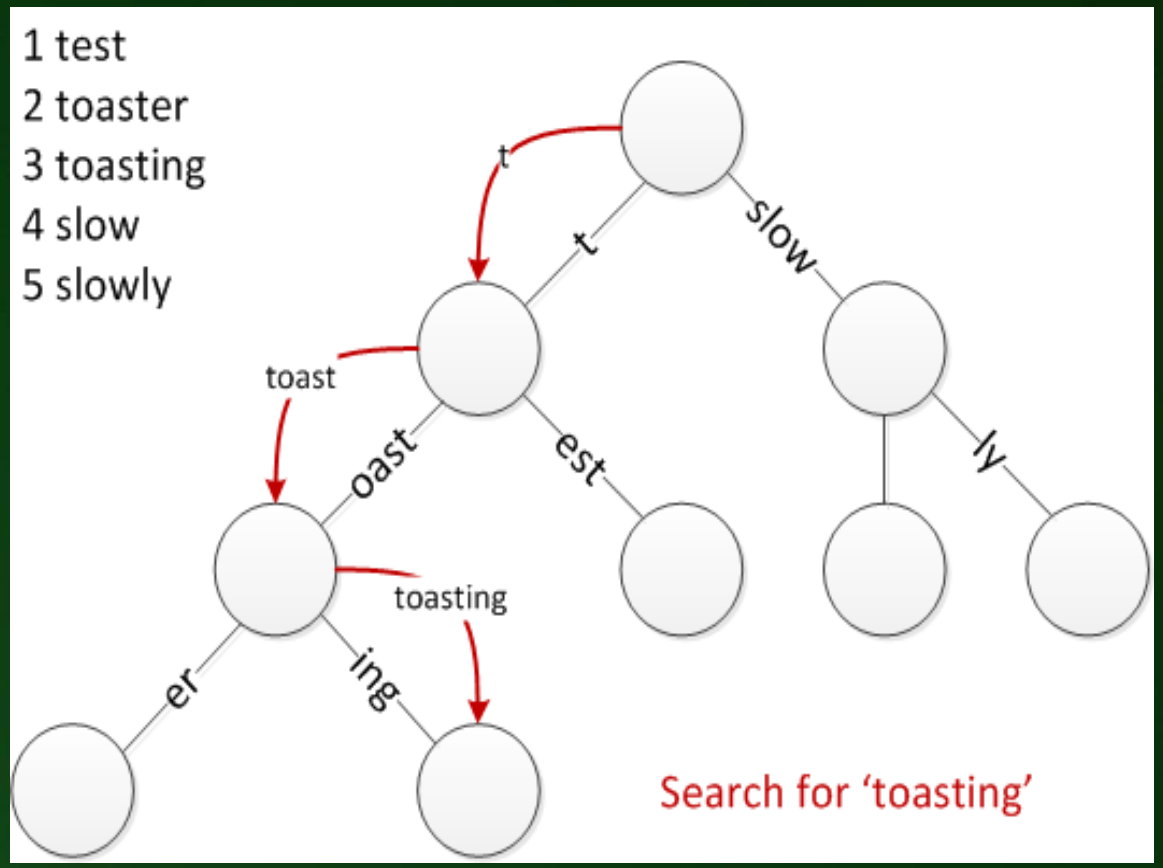
- Έτσι έχουμε ένα ζεύγος <key:value>
  - Παράδειγμα: Έστω ότι ψάχνουμε την λέξη dog, (κωδικοί ASCII 64, 6F, 57)
    - Ξεκινάμε από το Hash της ρίζας του Trie για να βρούμε τον κόμβο-ρίζα
    - Ο τελευταίους ουσιαστικά είναι πίνακας από κλειδιά προς άλλους κόμβους
    - Χρησιμοποιούμε ως κλειδί το '6' (πρώτο nibble) βρίσκοντας τον κόμβο ένα επίπεδο παρακάτω, κλπ. (root → 6 -> 4 → 6 → F → 5 → 7)

Δένδρα Patricia

- Έχουν την μορφή δένδρων Radix
- PATRICIA = “Practical Algorithm To Retrieve Information Coded In Alphanumeric”
- Το μεγάλο πλεονέκτημα:
  - Επιτρέπουν αποδοτική εισαγωγή/διαγραφή & αναζήτηση τύπου `<key:value>`
- Ας δούμε και ένα παράδειγμα… 

Αλγόριθμος αναζήτησης Patricia

- *Αναζήτηση πληροφοριών που κωδικοποιούνται με αλφαριθμητική μορφή*
- Ένα Patricia trie είναι binary radix trie (radix $r = 2$), βλ. σχήμα δεξιά
- Στο Ethereum όμως χρησιμοποιούνται 16-δικά ψηφία (nibble) $\Rightarrow$ κόμβοι trie έχουν έως 16 κόμβους-παιδιά ($r = 16$)



Merkle Patricia Trie (1)

- Στο Ethereum έχει γίνει προσπάθεια να χρησιμοποιηθούν τα πλεονεκτήματα και των δύο τύπων trie
- Αποτέλεσμα το Ethereum modified Merkle Patricia Trie (MPT)
- Υπάρχουν συνολικά τέσσερα (4) MPT στο Ethereum

Merkle Patricia Trie (2)

- Παρέχουν μια κρυπτογραφικά αυθεντικοποιημένη δομή αποθήκευσης όλων των ζευγών <key:value> στο Ethereum
- Είναι πλήρως αιτιοκρατικά (deterministic)
  - Ένα Patricia Trie με τα ίδια ζεύγη <key:value> θα είναι πάντα το ίδιο μέχρι το τέλος
- Αποδοτικές λειτουργίες Insert, Lookup, Delete
  - Πολυπλοκότητα $O(\log(n))$

Merkle Patricia Trie (3)

- Το Hash ενός κόμβου χρησιμοποιείται ως δείκτης προς τον κόμβο και το MPT δομείται ανάλογα:
  - Key = SHA3(RLP(value))
- Η πλευρά Merkle παρέχει ένα ανθεκτικό σε παραβιάσεις (tamperproof) και αιτιοκρατικό (deterministic) δένδρο, ενώ η πλευρά Patricia παρέχει μια αποτελεσματική δυνατότητα ανάκτησης πληροφοριών
- Η βασική ιδέα ενός MPT στο Ethereum είναι ότι για μία λειτουργία, θα τροποποιηθεί μόνον ένας ελάχιστος αριθμός κόμβων για τον επαναϋπολογισμό της ρίζας

Radix Tries/Merkle Tries – Λεπτομέρειες (1)

- Η διαδικασία ανευρέσεως του συνολικού κλειδιού (κόμβος που αντιστοιχεί σε αυτό) χρειάζεται τόσα βήματα όσοι οι κόμβοι στην διαδρομή έως το ζητούμενο, αλλά επειδή τα πάντα είναι αποθηκευμένα σε βάση δεδομένων, συχνά ακόμα και 1 βήμα είναι αρκετό

- Στα Merkle Tries η διαφορά έγκειται ότι χρησιμοποιείται ένα deterministic cryptographic Hash κόμβου ως δείκτης προς τον ζητούμενο κόμβο
  - Στην βάση δεδομένων έχουμε αναζήτηση: key == sha3(RLP(value))

Radix Tries/Merkle Tries – Λεπτομέρειες (2)

- Στα Merkle Tries, εάν root Hash γνωστό σε όλους, οποιοσδήποτε μπορεί να αποδείξει ότι το Trie έχει συγκεκριμένη τιμή σε συγκεκριμένη διαδρομή, εφόσον παρέχονται οι αντίστοιχοι (συνδυαζόμενοι) κόμβοι του Trie σε κάθε βήμα

- Αφού η κωδικοποίηση είναι στο 16-δικό σύστημα, η παραπάνω διαδρομή υλοποιείται κατά 1 nibble σε κάθε βήμα (4 bit/βήμα)
  - Κάθε κόμβος (ονόματι 'branch node') περιέχει πίνακα το πολύ 17 στοιχείων (16 για τις τιμές του nibble, και 1 για την τιμή-στόχο, εάν έχει ολοκληρωθεί η διαδρομή)

Merkle Patricia Trie – Δομή (1)

- Κάθε κόμβος MPT έχει μία από τις ακόλουθες τρεις μορφές (NULL η κενή συμβολοσειρά):
  -  branch: Κόμβος 17 αντικειμένων `[i0, i1, …, i15, vt]`
  -  leaf: Κόμβος 2 αντικειμένων `[encodedPath, value]`
  -  extension: Κόμβος 2 αντικειμένων `[encodedPath, key]`
- Στις 2 πρώτες μορφές μπορούμε να έχουμε τιμή που να αντιστοιχεί σε κλειδί, ενώ στο 3ο (extension) όχι τιμή, αλλά κλειδί (συνέχεια στο μονοπάτι)

Merkle Patricia Trie – Δομή (2)

- Κάθε μονοπάτι περιγράφεται αρχικά με κόμβους τύπου-branch
- Μετά από τα πρώτα λίγα επίπεδα φθάνουμε συνήθως σε κόμβο με ένα μόνον μονοπάτι έως το τελευταίο μέρος της διαδρομής. Αντί να έχουμε τιμές «κενό» σε κάθε έναν από τους 15 υπολειπόμενους δείκτες (σύνολο 16), τις περικόπτουμε χρησιμοποιώντας κόμβο τύπου-extension (1 δείκτης): encodedPath έχει το μερικό μονοπάτι έως αυτόν, key τιμή-κλειδί (Hash) για τον επόμενο κόμβο μέσα στην Β.Δ.
- Κόμβος τύπου-leaf (1 δείκτης) χρησιμοποιείται όπου έχουμε τελικό κόμβο διαδρομής για key, αλλά με value τον αναζητούμενο στόχο

Merkle Patricia Trie – Δομή (3)

- Όταν διασχίζουμε το Trie σε μία διαδρομή (βάσει κλειδιού) αυτό γίνεται κατά nibble, αλλά κάθε δεδομένο αποθηκεύεται κατά byte. Πώς διακρίνουμε το ένα nibble 1 από τα δύο nibble 01 αφού και τα δύο αποθηκεύονται ως το byte '01';
- Χρησιμοποιείται πρόθεμα nibble για τους δύο «μικρούς» τύπους κόμβων **και** για το παραπάνω πρόβλημα **και** για το εάν το εμπεριεχόμενο μέρος του path στον κόμβο αυτόν έχει μονό ή ζυγό πλήθος nibbles

16-δικό ψηφίο	bits	Τύπος κόμβου	'Μήκος' εμπεριεχόμενου path
0	0000	extension	ζυγό
1	0001	extension	μονό
2	0010	terminating (leaf)	ζυγό
3	0011	terminating (leaf)	μονό

MPT: Παράδειγμα Κωδικοποίησης – (1)

- Προαιρετικά μπορεί να υπάρχει και ένα end tag με τιμή 0x10 που υπάρχει μόνον στο τέλος του path, υποδεικνύοντας ότι αυτός είναι κόμβος τύπου-leaf. Εάν path με μονό πλήθος nibbles, συνδυάζεται με το prefix nibble σε **πλήρες byte**, αλλιώς ακολουθεί ένα ακόμα "padding nibble" (0) για να είναι πλήρες byte. Παράδειγμα:

• > [0, 1, 2, 3, 4, 5, ...] '00 01 23 45'	Χωρίς end tag, ζυγό path (6 nibbles). Περίπτωση γραμμής 1, πίνακα *** 0x00 στην αρχή ***
• > [1, 2, 3, 4, 5, ...] '11 23 45'	Χωρίς end tag, μονό path (5 nibbles). Περίπτωση γραμμής 2, πίνακα *** 0x1 στην αρχή ***
• > [0, f, 1, c, b, 8, 10] '20 0f 1c b8'	Ζυγό path με το end tag (0x10). Περίπτωση γραμμής 3, πίνακα *** 0x20 στην αρχή ***
• > [f, 1, c, b, 8, 10] '3f 1c b8'	Μονό path με το end tag (0x10). Περίπτωση γραμμής 4, πίνακα *** 0x3 στην αρχή ***

MPT: Παράδειγμα Κωδικοποίησης – (2)

- Ας δούμε και ένα πιο πλήρες παράδειγμα:
  - Έστω τα ζεύγη <key:value> ως εξής:
    <‘do’:‘verb’>, <‘dog’:‘puppy’>, <‘doge’:‘coin’>, <‘horse’:‘stallion’>
  - Πρώτα τα μετατρέπουμε σε bytes (τα value τα έχουμε γράψει ως συμβολοσειρές για ευκολία)

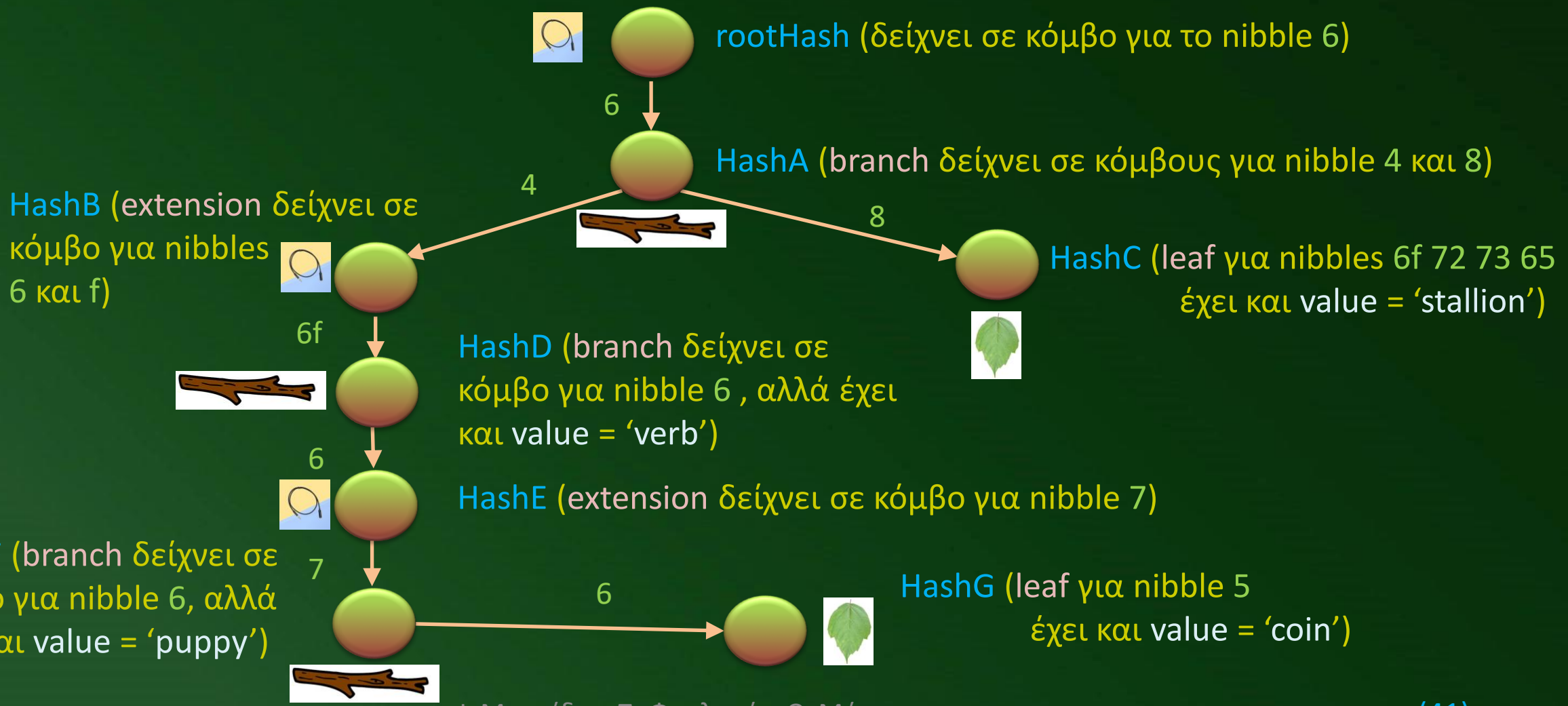
    <64 6f> : 'verb'

    <64 6f 67> : 'puppy'

    <64 6f 67 65> : 'coin'

    <68 6f 72 73 65> : 'stallion'

MPT: Παράδειγμα Κωδικοποίησης – (3)



MPT: Παράδειγμα Κωδικοποίησης – (4)

- (συνέχεια...)
  - Κατόπιν κατασκευάζουμε ένα Trie με τα παρακάτω ζεύγη <key:value> στην υποκείμενη Β.Δ. (levelDB στο Geth). H(x) = sha3(x), if len(x)>=32; else x

- rootHash: [<16>, hashA]
- hashA: [<>, <>, <>, <>, hashB→4, <>, <>, <>, hashC→8, <>, <>, <>, <>, <>, <>, <>, <>]
- hashC: [<20 6f 72 73 65>, 'stallion']
- hashB: [<00 6f>, hashD]
- hashD: [<>, <>, <>, <>, <>, <>, hashE→6, <>, <>, <>, <>, <>, <>, <>, <>, <>, 'verb']
- hashE: [<17>, hashF]
- hashF: [<>, <>, <>, <>, <>, <>, hashG →6, <>, <>, <>, <>, <>, <>, <>, <>, <>, 'puppy']
- hashG: [<35>, 'coin']

*** Περίπτωση 2 ***

*** Περίπτωση 3 ***

*** Περίπτωση 2 ***

*** Περίπτωση 4 ***

Ethereum MPTs

Η κεφαλίδα κάθε μπλοκ συναλλαγών περιέχει 3 ρίζες από 3 MPTs:

1. **'stateRoot'** → **State Trie** με

   path (key): ethereumAddress, value: rlp(ethereumAccount)

   (1b) Σε κάθε ethereumAccount αντιστοιχεί και ένα **'storageRoot'** → **Storage Trie** με

   όλα τα δεδομένα του συμβολαίου, εάν είναι CA (Contract Account)

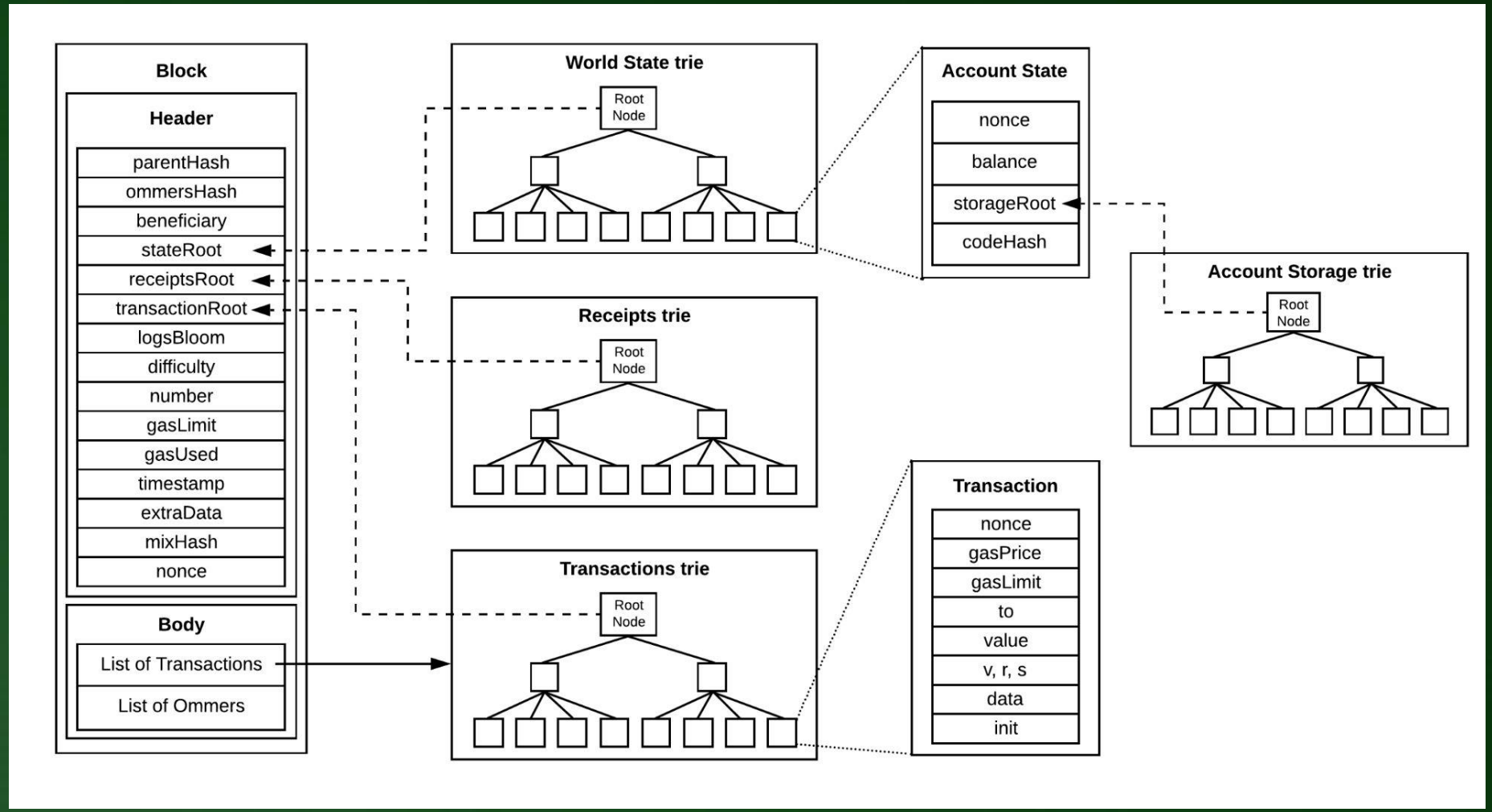
2. **'transactionsRoot'** → **Transactions Trie** με

   path (key): rlp(transactionIndex), value: συνήθως rlp(tx), βλ. EIP 2718

3. **'receiptsRoot'** → **Receipts Trie** με

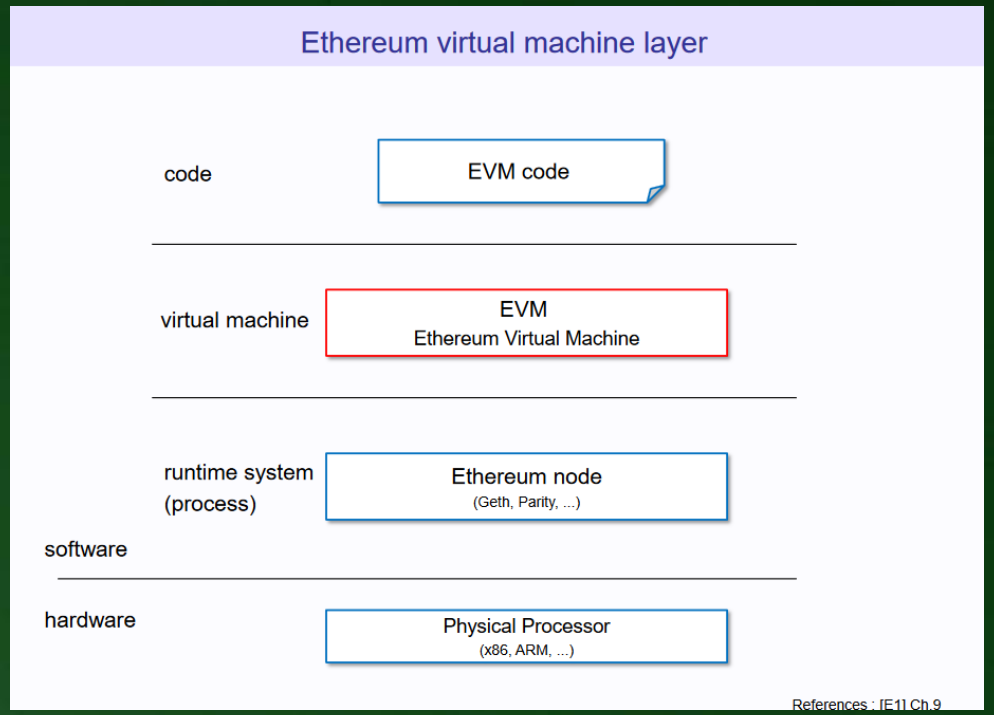
   path (key): δείκτης συναλλαγής κάποιου μπλοκ, value: receipt, βλ. EIP 2718

Η Συνολική Εικόνα Μπλοκ & MPTs



EVM

- Ethereum Virtual Machine
  - Από https://takenobu-hs.github.io/downloads/ethereum_evm_illustrated.pdf



EVM – Μηχανή Stack

- Έχει βάθος 1.024 στοιχείων, με κάθε στοιχείο 256 bit (32 byte)
  - Το τελευταίο επελέγη επειδή βολεύει για κρυπτογραφία 256-bit (π.χ. Keccak-256 Hash ή secp256k1 signatures)
  - Θυμηθείτε το Reverse Polish Notation (παραδείγματα εδώ και εδώ)
- Έχει μικρό σχετικά πλήθος εντολών (λεπτομέρειες εδώ)
- Υπάρχουν διάφορες υλοποιήσεις με κυριότερη στον πελάτη Geth

Περιεχόμενα

- Από το BTC στο Ethereum
- Επισκόπηση του Blockchain στο Ethereum
  - Λογαριασμοί, Κλειδιά & Διευθύνσεις
  - Συναλλαγές
  - Μπλοκ Συναλλαγών, Tries & Σχετικές Δομές, EVM
  - Gas
  - Κόμβοι και Πελάτες
  - Δίκτυα
  - Μηχανισμοί Συναίνεσης
- Έξυπνα Συμβόλαια στο Ethereum
- Το Οικοσύστημα του Ethereum

Κάποιες Λεπτομέρειες για Gas – (1)

- Gas: Μονάδα μέτρησης για το κόστος κάθε εκτελούμενης ενέργειας στα πλαίσια συναλλαγής
- **Πριν** από London upgrade (Αύγουστος 2021)
  - Μεταφορά 1 ETH, gaslimit = 21.000 units, gas price/unit = 200 gwei:
    - Συνολική αμοιβή: 21.000 x 200 = 4.200.000 gwei ή 0,0042 ETH

Κάποιες Λεπτομέρειες για Gas – (2)

- **Μετά** από London upgrade:
  - Μεταφορά 1 ETH, gaslimit = 21.000 μονάδες, base fee = 10 gwei:
    - Συνολική αμοιβή: 21.000 x (10 + priority fee = 2 gwei) = 0,000252 ETH
    - Επίσης τώρα δυνατόν να θέσουμε μέγιστη αμοιβή (maxFeePerGas) για την συναλλαγή
    - Εάν υπάρχει διαφορά με την πραγματική αμοιβή, αυτή η διαφορά επιστρέφεται

Κάποιες Λεπτομέρειες για Gas – (3)

- **Πριν** από London upgrade (Αύγουστος 2021)
  - Οι miners λάμβαναν την **συνολική αμοιβή** gas από οιαδήποτε συναλλαγή του μπλοκ

- **Μετά** από London upgrade:
  - Το base fee "καίγεται" πλέον και το μόνο που απομένει ως οικονομικό κίνητρο είναι το priority fee (φιλοδώρημα), για συμπερίληψη συναλλαγής σε μπλοκ
  - Το maxFeePerGas υπάρχει προαιρετικά, ώστε χρήστης να ορίζει ένα άνω όριο για το πόσο θέλει να πληρώσει για την εκτέλεση της συναλλαγής του.
    - Πρέπει όμως > (base fee + tip). Η όποια διαφορά επιστρέφεται

Κάποιες Λεπτομέρειες για Gas – (4)

- Εάν gaslimit μικρότερο από όσο αναγκαίο για κάποια συναλλαγή (π.χ. 20.000, ενώ απαιτούνται 50.000 μονάδες gas), η EVM θα προσπαθήσει να εκτελέσει την συναλλαγή
- Αφού δεν φθάνει όμως το gas, θα επιστρέψει στην προηγούμενη κατάσταση, αλλά θα κρατήσει το gas που ορίσατε στο gaslimit γιατί απασχολήθηκε

Στρατηγικές για Μείωση Gas

- Προφανώς μεγαλύτερο φιλοδώρημα/gas οδηγεί σε προτίμηση συναλλαγής για συμπερίληψη στο επόμενο μπλοκ

- Ποιες είναι όμως οι τρέχουσες τιμές gas; Κάποιες πηγές εκτίμησης:
  - Etherscan
  - Blocknative ETH Gas Estimator
  - ETH Gas Station
  - Cryptoneur Gas Fees Calculator

Ερωτήσεις