RaiBlocks: Jaringan Kriptocurrency Didistribusikan Biaya Rendah

Colin LeMahieu
clemahieu@gmail.com

***Abstrak*—Baru saja, Tingginya permintaan dan skalabilitas terbatas telah meningkatkan rata-rata waktu dan biaya transaksi dalam kripto yang populer, menghasilkan pengalaman yang tidak memuaskan. Di sini kami memperkenalkan RaiBlocks, sebuah kripto dengan arsitektur blok-lattice baru dimana masing-masing akun memiliki blockchain tersendiri, memberikan kecepatan transaksi seketika dan skalabilitas tak terbatas. Setiap pengguna memiliki blockchain sendiri, memungkinkan mereka memperbaruinya secara asinkron ke jaringan lainnya, sehingga menghasilkan transaksi cepat dengan overhead minimal. Transaksi melacak saldo akun daripada jumlah transaksi, sehingga pemangkasan database agresif tanpa mengorbankan keamanan. Sampai saat ini, jaringan RaiBlock telah memproses 4,2 juta transaksi dengan ukuran buku besar yang tidak dipasangkan hanya 1,7GB. Raiblocks biaya rendah, transaksi dalam sekejap membuatnya menjadi kripto utama bagi transaksi konsumen.**

***Ketentuan Indeks*—kriptokokus, blockchain, raiblocks, buku besar terdistribusi, digital, transaksi**

I. Kata Pengantar

IMPLEMENTASI Bitcoin di tahun 2009, telah terjadi pergeseran yang berlanjut dari sistem mata uang dan sistem keuangan tradisional yang didukung pemerintah ke sistem pembayaran modern berdasarkan kriptografi, yang menawarkan kemampuan untuk menyimpan dan mentransfer dana secara aman dan aman [1]. Agar berfungsi efektif, mata uang harus mudah dipindahtangankan, tidak dapat dibalikkan, dan memiliki biaya terbatas atau tidak ada. Waktu transaksi yang meningkat, biaya yang besar, dan skalabilitas jaringan yang dipertanyakan telah menimbulkan pertanyaan tentang kepraktisan Bitcoin sebagai mata uang sehari-hari.

Dalam tulisan ini, kami memperkenalkan RaiBlocks, sebuah kriptocurrency latensi rendah yang dibangun di atas struktur data blok-kisi yang inovatif yang menawarkan skalabilitas tak terbatas dan tidak ada biaya transaksi. RaiBlocks dengan design adalah protokol sederhana dengan tujuan tunggal menjadi kriptourrency berkinerja tinggi. Protokol RaiBlocks dapat berjalan pada perangkat keras berdaya rendah, memungkinkannya menjadi kripto yang praktis dan terdesentralisasi untuk penggunaan sehari-hari.

Statistik kriptocurrency yang dilaporkan dalam makalah ini akurat pada tanggal publikasi.

II. Latar Belakang

Pada tahun 2008, seorang individu anonim dengan nama samaran Satoshi Nakamoto menerbitkan sebuah white paper yang menguraikan Kriptocurrency terdesentralisasi pertama di dunia, Bitcoin [1]. Inovasi utama yang dibawa oleh Bitcoin adalah struktur data blockchain, public, immutable dan desentralisasi yang digunakan sebagai buku besar untuk transaksi mata uang. Sayangnya, saat Bitcoin matang, beberapa masalah dalam protokol membuat Bitcoin menjadi penghalang bagi banyak aplikasi:

1) Skalabilitas yang buruk: Setiap blok di blockchain dapat menyimpan data dalam jumlah terbatas, yang berarti sistem hanya dapat memproses begitu banyak transaksi per detik, membuat titik di blok menjadi komoditas. Saat ini biaya transaksi median adalah $10.38 [2].
2) Latency tinggi: Waktu konfirmasi rata-rata adalah 164 menit [3].
3) Daya tidak efisien: Jaringan Bitcoin mengkonsumsi 27.28TWh sepanjang tahun, dengan menggunakan rata-rata 260KWh per transaksi [4].

Bitcoin, dan cryptocurrencies lainnya, berfungsi dengan mencapai konsensus pada buku besar global mereka untuk memverifikasi transaksi yang sah sambil menolak pelaku kejahatan. Bitcoin mencapai konsensus melalui ukuran ekonomi yang disebut Proof of Work (PoW). Dalam sebuah sistem PoW, peserta bersaing untuk menghitung angka yang disebut nonce, sehingga hash dari keseluruhan blok berada pada kisaran target. Kisaran yang valid ini berbanding terbalik dengan kekuatan komputasi kumulatif seluruh jaringan Bitcoin untuk mempertahankan waktu rata-rata yang konsisten yang diambil untuk menemukan nonce yang valid. Pencari sebuah nonce yang valid kemudian diizinkan untuk menambahkan blok pada blockchain; Oleh karena itu, mereka yang menghabiskan lebih banyak sumber daya komputasi untuk menghitung nonce memainkan peran lebih besar dalam keadaan blockchain. PoW memberikan perlawanan terhadap serangan Sybil, di mana entitas berperilaku sebagai entitas ganda untuk mendapatkan kekuatan tambahan dalam sistem yang terdesentralisasi, dan juga sangat mengurangi kondisi balapan yang secara inheren ada saat mengakses struktur data global.

Protokol konsensus alternatif, Proof of Stake (PoS), pertama kali diperkenalkan oleh Peercoin pada tahun 2012 [5]. Dalam sistem PoS, peserta memilih dengan bobot yang setara dengan jumlah kekayaan yang mereka miliki dalam kripto yang diberikan. Dengan pengaturan ini, mereka yang memiliki investasi finansial lebih besar diberi lebih banyak kekuatan dan secara inheren diberi insentif untuk menjaga kejujuran sistem atau berisiko kehilangan investasinya. PoS tidak melakukan kompetisi daya komputasi yang boros, hanya memerlukan perangkat lunak ringan yang berjalan pada perangkat keras berdaya rendah.

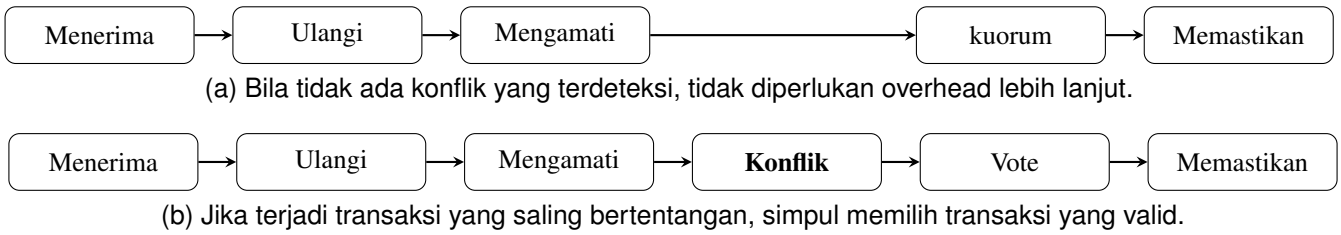

Fig. 1. RaiBlock tidak memerlukan biaya tambahan untuk transaksi biasa. Jika terjadi transaksi yang saling bertentangan, simpul harus memilih agar transaksi tetap terjaga

Kertas RaiBlocks asli dan implementasi beta pertama diterbitkan pada bulan Desember 2014, menjadikannya salah satu kriptocurrency berbasis Directed Acyclic Graph (DAG) terdistribusi pertama [6]. Segera setelah itu, kripto Gangguan DAG lainnya mulai berkembang, terutama DagCoin / Byteball dan IOTA [7], [8]. Cryptocurrencies berbasis DAG ini memecahkan cetakan blockchain, meningkatkan kinerja dan keamanan sistem. Byteball mencapai konsensus dengan mengandalkan "rantai utama" yang terdiri dari saksi "jujur, terpercaya dan dipercaya pengguna", sementara IOTA mencapai konsensus melalui PoW kumulatif dari transaksi bertumpuk. RaiBlocks mencapai konsensus melalui pemungutan suara tertimbang mengenai transaksi yang bertentangan. Sistem konsensus ini menyediakan transaksi yang lebih cepat dan lebih deterministik sambil tetap mempertahankan sistem yang kuat dan terdesentralisasi. RaiBlocks melanjutkan perkembangan ini dan telah memposisikan dirinya sebagai salah satu kripto yang memiliki kinerja tertinggi.

III. RaiBlocks Komponen

Sebelum menggambarkan arsitektur RaiBlock secara keseluruhan, kita mendefinisikan komponen individual yang membentuk system.

A. Akun

Akun adalah bagian kunci publik dari pasangan kunci tanda tangan digital. Kunci publik, yang juga disebut sebagai alamat, dibagi dengan peserta jaringan lainnya sementara kunci privat dirahasiakan. Paket data yang ditandatangani secara digital memastikan bahwa isinya disetujui oleh pemegang kunci privat. Satu pengguna dapat mengendalikan banyak akun, namun hanya satu alamat publik yang mungkin ada per akun.

B. Blok/Transaksi

Istilah "blok" dan "transaksi" sering digunakan secara bergantian, di mana satu blok berisi satu transaksi. Transaksi secara khusus mengacu pada tindakan sementara blok mengacu pada pengkodean digital transaksi. Transaksi ditandatangani oleh kunci privat milik akun tempat transaksi dilakukan.

C. Ledger

Ledger adalah kumpulan akun global dimana masing-masing akun memiliki rantai transaksi sendiri (Gambar 2). Ini adalah komponen desain utama yang termasuk dalam kategori mengganti perjanjian run-time dengan perjanjian disain; Semua orang setuju melalui pengecekan tanda tangan bahwa hanya pemilik akun yang bisa memodifikasi rantai mereka sendiri. Ini mengubah struktur data yang tampaknya ada bersama, sebuah buku besar terdistribusi, ke kumpulan yang tidak dibagi.

D. Node

Node adalah perangkat lunak yang berjalan pada komputer yang sesuai dengan protokol RaiBlocks dan berpartisipasi dalam jaringan RaiBlocks. Perangkat lunak ini mengelola buku besar dan akun yang dapat dikendalikan oleh node, jika ada. Simpul dapat menyimpan keseluruhan buku besar atau riwayat pemangkasan yang hanya berisi beberapa blok terakhir dari setiap blokir akun. Saat membuat simpul baru, disarankan untuk memverifikasi keseluruhan riwayat dan memangkas secara lokal.

IV. Gambaran Sistem

Tidak seperti blockchains yang banyak digunakan dalam cryptocurrencies lainnya, RaiBlock menggunakan struktur *blok-kisi*. Setiap akun memiliki blockchain sendiri (akun-chain) yang setara dengan riwayat transaksi/saldo akun (Gambar 2). Setiap rantai akun hanya dapat diperbarui oleh pemilik akun; Hal ini memungkinkan setiap rantai akun diperbarui segera dan secara asynchronous ke sisa blok-kisi, sehingga terjadi transaksi cepat. Protokol RaiBlocks sangat ringan;

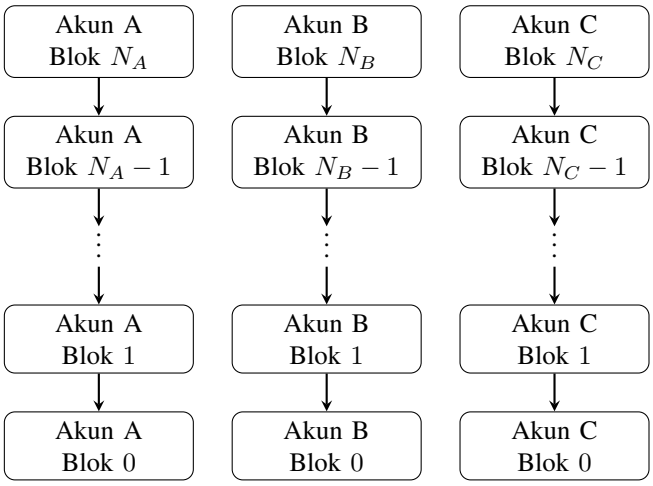

Fig. 2. . Setiap akun memiliki blockchain sendiri yang berisi riwayat saldo akun. Blok 0 harus berupa transaksi terbuka (Bagian IV-B)

setiap transakasi sesuai dengan ukuran paket UDP minimum yang diperlukan untuk dikirim melalui internet. Kebutuhan perangkat keras untuk node juga minimal, karena node hanya perlu merekam dan mencairkan blok sebagian besar transaksi (Gambar 1).

Sistem ini dimulai dengan *akun genesis* yang mengandung *keseimbangan genesis*. Keseimbangan genesis adalah kuantitas tetap dan tidak dapat ditingkatkan. Keseimbangan genesis dibagi dan dikirim ke akun lain melalui transaksi kirim yang terdaftar pada genesis rantai akun. Jumlah saldo semua akun tidak akan pernah melebih saldo awal genesis yang memberi System batas atas pada kuantitas dan tidak ada kemampuan untuk meningkatkannya.

Bagian ini akan berjalan melalui bagaimana berbagai jenis transaksi dibangun dan disebarluaskan ke seluruh jaringan.

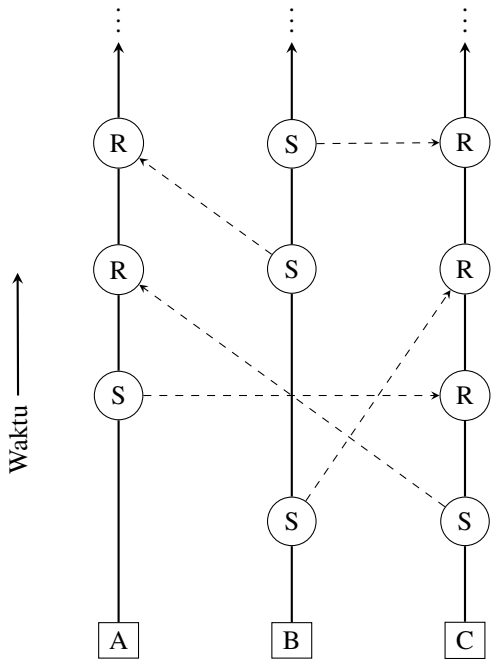

Fig. 3. Visualisasi blok-kisi. Setiap transfer dana memerlukan blok kirim (S) dan blok penerima (R), masing-masing ditandatangani oleh pemilik rantai akun mereka (A, B, C)

A. Transaksi

Mentransfer dana dari satu akun ke akun lainnya memerlukan dua transaksi: sebuah pengiriman dikurangi jumlah dari saldo pengirim dan jumlah penambahan yang ditambahkan ke saldo rekening penerima (Gambar 3).

Mentransfer jumlah sebagai transaksi terpisah di akun pengirim dan penerima melayani beberapa tujuan penting:

1) Pengurutan transaksi masuk yang secara inheren asynchronous.
2) Menjaga transaksi kecil agar sesuai dengan paket UDP.
3) Memfasilitasi pemangkasan buku besar dengan meminimalkan data jejak kaki.
4) Mengisolasi transaksi yang diselesaikan dari yang tidak beres.

Lebih dari satu akun mentransfer ke tujuan yang sama Akun adalah operasi asinkron; latency jaringan dan akun pengirim yang tidak selalu saling berkomunikasi berarti tidak ada cara yang menyenangkan untuk mengetahui transaksi mana yang terjadi lebih dulu. Karena penambahan bersifat asosiatif, urutan input diurutkan tidak masalah, dan karenanya kita memerlukan kesepakatan global. Ini adalah komponen desain utama yang mengubah perjanjian run-time menjadi perjanjian waktu desain. Rekening penerima memiliki kendali atas penentuan transfer mana yang datang lebih dulu dan dinyatakan dengan perintah masuk blok masuk.

Jika sebuah akun ingin melakukan transfer besar yang diterima sebagai sekumpulan banyak transfer kecil, kami ingin mewakili Ini dengan cara yang sesuai dengan paket UDP. Ketika transfer masukan urutan akun penerima, jumlah total saldo akunnya terus berjalan sehingga setiap saat ia memiliki kemampuan untuk mentransfer jumlah apapun dengan transaksi ukuran tetap. Ini berbeda dengan model transaksi input / output yang digunakan oleh Bitcoin dan cryptocurrencies lainnya.

Beberapa node tidak tertarik untuk mengeluarkan sumber daya untuk menyimpan riwayat transaksi penuh akun; mereka hanya tertarik pada saldo masing-masing akun. Ketika sebuah akun melakukan transaksi, ia mengkodekan saldo terakumulasinya dan simpul ini hanya perlu melacak blok terakhir, yang memungkinkan mereka membuang data historis sambil mempertahankan kebenaran.

Bahkan dengan fokus pada perjanjian desain, ada jendela penundaan saat memvalidasi transaksi karena mengidentifikasi dan menangani aktor buruk di jaringan. Karena kesepakatan di RaiBlock tercapai dengan cepat, sesuai urutan milidetik hingga detik, kami dapat menyajikan pengguna dengan dua kategori transaksi masuk yang diketahui: diselesaikan dan tidak diselesaikan. Transaksi yang diselesaikan adalah transaksi dimana akun telah menghasilkan menerima blok. Transaksi yang belum diselesaikan belum dimasukkan ke dalam saldo kumulatif penerima. Ini adalah pengganti untuk metrik konfirmasi yang lebih kompleks dan tidak dikenal di kripto yang lain.

B. Membuat Akun

Untuk membuat akun, Anda perlu mengeluarkan transaksi terbuka (Gambar 4). Transaksi terbuka selalu merupakan transaksi pertama dari setiap rantai akun dan dapat dibuat pada saat penerimaan dana pertama. Bidang akun menyimpan kunci publik (alamat) yang berasal dari kunci privat yang digunakan untuk masuk. Bidang sumber berisi hash dari transaksi yang mengirim dana. Pada pembuatan akun, perwakilan harus dipilih untuk memberikan suara atas nama Anda; ini bisa diubah nanti (Bagian IV-F). Akun tersebut dapat menyatakan dirinya sebagai perwakilannya sendiri.

C. Saldo Akun

Saldo akun dicatat dalam buku besar itu sendiri. Daripada mencatat jumlah transaksi, verifikasi (Bagian IV-I) mengharuskan pengecekan selisih antara saldo di blok kirim dan saldo blok sebelumnya. Account penerima kemudian dapat menambahkan saldo sebelumnya yang diukur ke dalam saldo akhir yang diberikan di blok penerima yang baru. Hal ini

```
open {
   account: DC04354B1...AE8FA2661B2,
   source: DC1E2B3F7C...182A0E26B4A,
   representative: xrb_1anr...posrs,
   work: 0000000000000000,
   type: open,
   signature: 83B0...006433265C7B204
}
```

Fig. 4. Anatomi transaksi terbuka

dilakukan untuk meningkatkan kecepatan proses saat mendownload volume blok tinggi. Saat meminta riwayat transaksi, jumlah sudah diberikan.

D. Mengirim dari Akun

Untuk mengirim dari sebuah alamat, alamat tersebut harus sudah memiliki blok terbuka yang ada, dan oleh karena itu keseimbangan (Gambar 5). Bidang sebelumnya berisi hash dari blok sebelumnya dalam rantai akun. Bidang tujuan berisi akun dana yang akan dikirim ke. Sebuah blok kirim tidak dapat diubah begitu dikonfirmasi. Begitu disiarkan ke jaringan, dana langsung dikurangkan dari saldo rekening pengirim dan menunggu sampai tertunda sampai pihak penerima menandatangani satu blok untuk menerima dana tersebut. Dana yang tertunda tidak boleh dianggap menunggu konfirmasi, karena sama baiknya dengan pengeluaran dari rekening pengirim dan si pengirim tidak dapat mencabut transaksi tersebut.

```
send {
   previous: 1967EA355...F2F3E5BF801,
   balance: 010a8044a0...1d49289d88c,
   destination: xrb_3w...m37goeuufdp,
   work: 0000000000000000,
   type: send,
   signature: 83B0...006433265C7B204
}
```

Fig. 5. Anatomi transaksi kirim

E. Menerima Transaksi

Untuk menyelesaikan transaksi, penerima dana yang dikirim harus membuat blok penerimaan pada rantai akun mereka sendiri (Gambar 6). Bidang sumber merujuk pada hash dari transaksi kirim yang terkait. Begitu blok ini dibuat dan disiarkan, saldo akun akan diperbarui dan dana telah masuk ke akun mereka secara resmi.

F. Menugaskan Perwakilan

Pemegang akun yang memiliki kemampuan untuk memilih perwakilan untuk memilih atas nama mereka adalah alat desentralisasi yang kuat yang tidak memiliki analog yang kuat dalam Bukti Karya atau Bukti Protokol Pasokan. Dalam sistem PoS konvensional, simpul pemilik akun harus berjalan untuk

```
receive {
   previous: DC04354B1...AE8FA2661B2,
   source: DC1E2B3F7C6...182A0E26B4A,
   work: 0000000000000000,
   type: receive,
   signature: 83B0...006433265C7B204
}
```

Fig. 6. Anatomi transaksi menerima

berpartisipasi dalam pemungutan suara. Terus menjalankan simpul tidak praktis bagi banyak pengguna; memberikan representative kekuatan untuk memberikan suara atas nama sebuah akun melemaskan persyaratan ini. Pemegang rekening memiliki kemampuan untuk menugaskan kembali konsensus ke akun kapan saja. Perubahan transaksi mengubah perwakilan akun dengan mengurangi bobot suara dari perwakilan lama dan menambahkan bobot Ke perwakilan baru (Gambar 7). Tidak ada dana yang dipindahkan dalam transaksi ini, dan perwakilannya tidak memiliki daya beli dari dana akun.

```
change {
   previous: DC04354B1...AE8FA2661B2,
   representative: xrb_1anrz...posrs,
   work: 0000000000000000,
   type: change,
   signature: 83B0...006433265C7B204
}
```

Fig. 7. Anatomi transaksi perubahan

G. Forks and Voting

Fork terjadi ketika blok yang ditandatangani $b_1, b_2, \ldots, b_j$ mengklaim blok yang sama dengan pendahulunya (Gambar 8). Blok ini menyebabkan pandangan yang bentrok tentang status akun dan harus diselesaikan. Hanya pemilik akun yang memiliki kemampuan untuk menandatangani blok ke dalam rantai akun mereka, jadi garpu harus merupakan hasil dari pemrograman yang buruk atau niat jahat (pembelanjaan ganda) oleh pemilik akun.

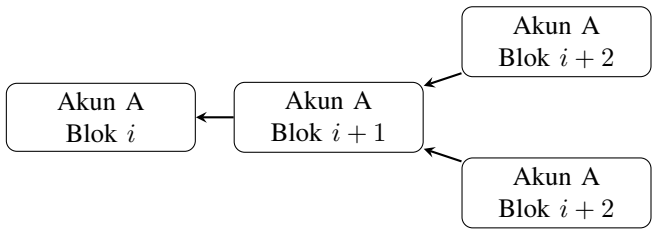

Fig. 8. Fork terjadi ketika dua (atau lebih) blok yang ditandatangani merujuk pada blok sebelumnya yang sama. Blok yang lebih tua ada di sebelah kiri; blok yang baru ada di sebelah kanan

Setelah deteksi, seorang perwakilan akan membuat sebuah referensi pemungutan suara yang membaur di buku, $\hat{b}_i$, besar itu dan menyiarkannya ke jaringan. Bobot suara simpul, w_i,

adalah jumlah saldo semua akun yang menamainya sebagai perwakilannya. Simpul tersebut akan mengamati suara yang masuk dari perwakilan online M lainnya dan menyimpan penghitungan kumulatif selama 4 periode suara, total 1 menit, dan mengkonfirmasi blok kemenangan (Persamaan 1).

$$v(b_j) = \sum_{i=1}^{M} w_i \mathbb{1}_{\hat{b}_i = b_j} \tag{1}$$

$$b^* = \arg\max_{b_j} v(b_j) \tag{2}$$

Blok paling populer b^* akan memiliki mayoritas suara dan akan disimpan di buku besar node (Persamaan 2). Blokir yang kehilangan hak suara dibuang. Jika seorang perwakilan mengganti satu blok di buku besarnya, akan membuat pemungutan suara baru dengan nomor urut yang lebih tinggi dan menyiarkan suara baru ke jaringan. Inilah satu-satunya skenario dimana perwakilan memilih.

Dalam beberapa keadaan, masalah konektivitas jaringan singkat dapat menyebabkan blok yang disiarkan agar tidak diterima oleh semua rekan sejawat. Setiap blok berikutnya pada akun ini akan diabaikan karena tidak valid oleh rekan-rekan yang tidak melihat siaran awal. Sebuah siaran ulang blok ini akan diterima oleh rekan-rekan yang tersisa dan blok berikutnya akan diambil secara otomatis. Bahkan ketika sebuah garpu atau blok yang hilang terjadi, hanya akun yang dirujuk dalam transaksi yang terpengaruh; sisa jaringan melanjutkan transaksi pemrosesan untuk semua akun lainnya.

H. Proof of Work

Keempat jenis transaksi memiliki bidang kerja yang harus diisi dengan benar. Bidang kerja memungkinkan pencipta transaksi untuk menghitung nonce sehingga hash dari nonce digabungkan dengan bidang sebelumnya dalam transaksi menerima / mengirim / mengubah atau bidang akun dalam transaksi terbuka berada di bawah nilai ambang batas tertentu. Tidak seperti Bitcoin, PoW di RaiBlocks hanya digunakan sebagai alat anti-spam, mirip dengan Hashcash, dan dapat dihitung berdasarkan urutan detik [9]. Setelah transaksi dikirim, PoW untuk blok berikutnya dapat didekomposisi karena bidang blok sebelumnya diketahui; ini akan membuat transaksi muncul seketika ke pengguna akhir asalkan waktu antar transaksi lebih besar dari waktu yang dibutuhkan untuk menghitung PoW.

I. Verifikasi Transaksi

Agar blok dianggap valid, harus ada atribut berikut:

1) Blok tidak boleh ada di buku besar (duplikat transaksi).
2) Harus ditandatangani oleh pemilik akun.
3) Blok sebelumnya adalah blok utama rantai akun. Jika ada tapi bukan kepala, itu adalah fork.
4) Akun harus memiliki blok terbuka.
5) Hash dihitung memenuhi persyaratan ambang PoW.

Jika itu adalah blok penerima, periksa apakah hash blok sumber tertunda, artinya belum ditukarkan. Jika itu adalah blok kirim, saldo harus kurang dari saldo sebelumnya.

V. Attack Vectors

RaiBlocks, seperti semua kripto yang terdesentralisasi, dapat diserang oleh pihak-pihak jahat karena usaha memperoleh keuntungan finansial atau sistem gugur. Pada bagian ini kami menguraikan beberapa kemungkinan skenario serangan, konsekuensi dari serangan semacam itu, dan bagaimana protokol RaiBlocks mengambil tindakan pencegahan.

A. Blok Gap Sinkronisasi

Pada Bagian IV-G, kita membahas skenario di mana sebuah blok mungkin tidak disiarkan dengan benar, menyebabkan jaringan mengabaikan blok berikutnya. Jika sebuah simpul mengamati sebuah blok yang tidak memiliki blok sebelumnya yang direferensikan, ia memiliki dua pilihan:

1) Abaikan blok karena bisa menjadi blok sampah berbahaya.
2) Meminta resync dengan node lain.

Dalam kasus resync, koneksi TCP harus dibentuk dengan simpul bootstrap untuk memfasilitasi peningkatan jumlah lalu lintas yang dibutuhkan oleh resync. Namun, jika blok itu sebenarnya blok yang buruk, maka resync tidak perlu dan tidak perlu meningkatkan lalu lintas di jaringan. Ini adalah Network Amplification Attack dan menghasilkan penolakan layanan.

Untuk menghindari resyncing yang tidak perlu, node akan menunggu sampai ambang batas suara tertentu telah diamati untuk blok yang berpotensi berbahaya sebelum memulai koneksi ke node bootstrap untuk melakukan sinkronisasi. Jika sebuah blok tidak menerima cukup suara maka bisa diasumsikan sebagai data sampah.

B. Transaksi Flooding

Entitas malicious dapat mengirim banyak transaksi yang tidak perlu namun valid antar akun di bawah kendalinya untuk menjinakkan jaringan. Tanpa biaya transaksi mereka bisa melanjutkan serangan ini tanpa batas waktu. Namun, PoW yang dibutuhkan untuk setiap transaksi membatasi tingkat transaksi yang dapat dihasilkan entitas berbahaya tanpa berinvestasi secara signifikan dalam sumber daya komputasi. Bahkan di bawah serangan seperti itu dalam upaya untuk mengembang buku besar, simpul yang tidak memiliki simpul sejarah penuh dapat memangkas transaksi lama dari rantai mereka; ini klem penggunaan penyimpanan dari jenis serangan untuk hampir semua pengguna.

C. Sybil Attack

Entitas dapat membuat ratusan node RaiBlock pada satu mesin tunggal; Namun, karena sistem voting berbobot berdasarkan saldo akun, menambahkan simpul ekstra ke jaringan tidak akan mendapatkan suara tambahan penyerang. Oleh karena itu tidak ada keuntungan yang bisa didapat melalui serangan Sybil.

D. Penny-Spend Attack

A penny-spend attack adalah tempat penyerang menghabiskan jumlah kecil-jumlah imal ke sejumlah besar akun untuk menyia-nyiakan sumber penyimpanan simpul. Pemblokiran adalah tingkat yang dibatasi oleh PoW, jadi ini membatasi penciptaan akun dan transaksi sampai batas tertentu. Simpul yang bukan simpul sejarah penuh dapat memangkas akun di bawah metrik statistik tempat akun tersebut kemungkinan besar bukan akun yang valid. Akhirnya, RaiBlock disetel untuk menggunakan ruang penyimpanan minimal permanen, sehingga ruang yang dibutuhkan untuk menyimpan satu akun tambahan sebanding dengan ukuran blok terbuka + pengindeksan $= 96\text{B} + 32\text{B} = 128\text{B}$. Ini setara dengan 1GB yang mampu menyimpan 8 juta akun penny-spend. Jika node ingin memangkas lebih agresif, mereka dapat menghitung distribusi berdasarkan frekuensi akses dan mendelegasikan akun yang jarang digunakan ke penyimpanan yang lebih lambat.

E. Precomputed PoW Attack

Karena pemilik akun akan menjadi satu-satunya entitas yang menambahkan blok ke rantai akun, blok berurutan dapat dihitung, bersama dengan PoW mereka, sebelum disiarkan ke jaringan. Di sini penyerang menghasilkan segudang blok berurutan, masing-masing bernilai minimal, dalam jangka waktu yang lama. Pada titik tertentu, penyerang melakukan Denial of Service (DoS) dengan membanjiri jaringan dengan banyak transaksi yang valid, dimana node lain akan memproses dan menggema secepat mungkin. Ini adalah versi lanjutan dari banjir transaksi yang dijelaskan di Bagian V-B. Serangan seperti itu hanya akan bekerja sebentar, tapi bisa digunakan bersamaan dengan serangan lainnya, seperti serangan¿ 50% (Bagian V-F) untuk meningkatkan efektivitas. Transaction rate-limiting dan teknik lainnya saat ini sedang diselidiki untuk mengurangi serangan.

F. >50% Attack

Metrik konsensus untuk RaiBlock adalah sistem voting berimbang. Jika penyerang dapat memperoleh lebih dari 50% kekuatan pemungutan suara, mereka dapat menyebabkan jaringan berosilasi konsensus membuat sistem rusak. Seorang penyerang dapat menurunkan jumlah saldo yang harus mereka hilangkan dengan mencegah kelalaian dari pemungutan suara melalui jaringan DoS.

RaiBlock mengambil tindakan berikut untuk mencegah serangan semacam itu:

1) Pertahanan utama melawan jenis serangan ini adalah bobot suara yang terkait dengan investasi di sistem. Pemegang rekening secara inheren diberi insentif untuk menjaga kejujuran sistem agar melindungi investasinya. Mencoba membalik buku besar akan merusak sistem secara keseluruhan yang akan menghancurkan investasi mereka.
2) Biaya serangan ini sebanding dengan kapitalisasi pasar RaiBlocks. Dalam sistem PoW, technol-ogy dapat ditemukan yang memberikan kontrol yang tidak proporsional dibandingkan dengan investasi moneter dan jika serangannya berhasil, teknologi ini dapat dilakukan kembali setelah serangan selesai. Dengan RaiBlock biaya serangan sistem dengan sistem itu sendiri dan jika sebuah serangan berhasil maka investasi dalam serangan tidak dapat dipulihkan.
3) Untuk mempertahankan kuorum maksimum pemilih, barisan pertahanan berikutnya adalah pemilihan perwakilan. Pemegang rekening yang tidak dapat berpartisipasi secara andal dalam memilih alasan konektivitas dapat memberi nama perwakilan yang dapat memilih dengan berat saldo mereka. Memaksimalkan jumlah dan keragaman perwakilan meningkatkan ketahanan jaringan.
4) Fork di Raiblocks tidak pernah kebetulan, jadi node dapat membuat keputusan kebijakan tentang bagaimana berinteraksi dengan blok bercabang. Satu-satunya akun non-penyerang yang bisa memblok fork adalah jika mereka menerima saldo dari akun yang menyerang. Akun yang ingin aman dari fork blok bisa menunggu sedikit atau lebih lama sebelum menerima dari akun yang menghasilkan fork atau memilih untuk tidak pernah menerima sama sekali. Penerima juga dapat membuat akun terpisah untuk digunakan saat menerima dana dari akun yang meragukan untuk melindungi akun lainnya.
5) Garis pertahanan terakhir yang belum dilaksanakan adalah *penyemenan blok*. RaiBlocks berusaha keras untuk menyelesaikan garpu blok dengan cepat melalui voting. Simpul dapat dikonfigurasikan ke blok semen, yang akan mencegahnya digulung kembali setelah jangka waktu tertentu. Jaringan cukup aman melalui fokus pada waktu penyelesaian cepat untuk mencegah garpu ambigu.

Versi yang lebih canggih dari serangan $> 50\%$ dirinci pada Gambar 9. “Offline” adalah persentase perwakilan yang diberi nama tapi tidak online untuk memilih. “Pasak” adalah jumlah investasi yang diserang oleh penyerang. “Aktif” adalah perwakilan yang sedang online dan memberikan suara sesuai dengan protokol. Seorang penyerang dapat mengimbangi jumlah saham yang harus mereka keliru dengan mengetuk pemilih lain secara offline melalui serangan jaringan DoS. Jika serangan ini dapat dipertahankan, perwakilan yang diserang akan menjadi tidak sinkron dan ini ditunjukkan oleh “Unsync.” Akhirnya, penyerang dapat memperoleh ledakan singkat dengan kekuatan pemungutan suara relatif dengan mengalihkan serangan Denial of Service mereka ke satu set perwakilan baru sementara set lama menyinkronkan buku besar mereka, ini ditunjukkan oleh “Attack.”

Offline	Unsync	**Attack**	Active	Stake

Fig. 9. Pengaturan voting potensial yang bisa menurunkan 51% persyaratan serangan.

Jika penyerang dapat menyebabkan Stake $>$ Active dengan kombinasi keadaan ini, mereka dapat berhasil membalik suara pada buku besar dengan mengorbankan saham mereka. Kita dapat memperkirakan berapa banyak jenis serangan ini dapat dikenai biaya dengan memeriksa pangsa pasar sistem lain.

Jika kita memperkirakan 33% perwakilan offline atau diserang melalui DoS, penyerang perlu membeli 33% dari harga pasar untuk menyerang sistem tersebut melalui pemungutan suara.

G. Bootstrap Poisoning

Semakin lama penyerang mampu menahan kunci pribadi lama dengan keseimbangan, semakin tinggi probabilitas bahwa saldo yang ada pada saat itu tidak akan memiliki perwakilan yang berpartisipasi karena saldo atau perwakilan mereka telah ditransfer ke akun yang lebih baru. Ini berarti jika sebuah simpul di-bootstrap ke representasi lama dari jaringan di mana penyerang memiliki kuorum hak suara dibandingkan dengan perwakilan pada saat itu, mereka akan dapat mengosongkan keputusan pemungutan suara ke simpul tersebut. Jika pengguna baru ini ingin berinteraksi dengan siapapun selain node penyerang, semua transaksi mereka akan ditolak karena mereka memiliki blok kepala yang berbeda. Hasil bersihnya adalah simpul yang bisa menyia-nyiakan waktu simpul baru di jaringan dengan memberi mereka informasi yang buruk. Untuk mencegah hal ini, node dapat dipasangkan dengan database awal akun dan kepala blok yang dikenal; Ini adalah pengganti untuk mendownload database sepanjang jalan kembali ke blok genesis. Semakin dekat unduhan untuk menjadi saat ini, semakin tinggi probabilitas untuk bertahan secara akurat terhadap serangan ini. Pada akhirnya, serangan ini mungkin tidak lebih buruk daripada menyuntikkan data sampah ke node saat bootstrap, karena mereka tidak dapat bertransaksi dengan siapa saja yang memiliki database kontemporer.

VI. Implementasi

Saat ini implementasi referensi diimplementasikan di C++ dan telah menghasilkan rilis sejak tahun 2014 di Github [10].

A. Fitur Desain

Implementasi RaiBlocks sesuai dengan standar arsitektur yang digariskan dalam makalah ini. Spesifikasi tambahan dijelaskan di sini.

1) Algoritma Penandatanganan: RaiBlock menggunakan algoritma kurva elips ED25519 yang dimodifikasi dengan hashing Blake2b untuk semua tanda tangan digital [11]. ED25519 dipilih untuk penandatanganan cepat, verifikasi cepat, dan keamanan yang tinggi.

2) Algoritma Hashing: Karena algoritma hashing hanya digunakan untuk mencegah spam jaringan, pilihan algoritma kurang penting bila dibandingkan dengan kriptocurrency berbasis pertambangan. Implementasi kami menggunakan algoritma Blake2b sebagai algoritma pencerna terhadap isi blok [12].

3) Fungsi Derivatif Kunci: Di dompet referensi, kunci dienkripsi dengan kata kunci dan kata kunci diumpankan melalui fungsi derivasi kunci untuk melindungi dari upaya cracking ASIC. Saat ini Argon2 [13] adalah pemenang satu-satunya kompetisi publik yang bertujuan menciptakan fungsi derivasi kunci yang tangguh.

4) Blok Interval: Karena setiap akun memiliki blockchain tersendiri, pembaruan dapat dilakukan tanpa asinkron ke keadaan jaringan. Karena itu tidak ada selisih waktu dan transaksi bisa dipublikasikan seketika.

5) UDP Message Protocol: Sistem kami dirancang untuk beroperasi tanpa batas waktu dengan menggunakan sumber daya komputasi minimum semaksimal mungkin. Semua pesan dalam sistem telah ditandatangani untuk tidak berkewarganegaraan dan sesuai dengan paket UDP tunggal. Ini juga mempermudah rekan-rekan lite dengan konektivitas berselang untuk berpartisipasi dalam jaringan tanpa membangun kembali koneksi TCP jangka pendek. TCP digunakan hanya untuk rekan-rekan baru ketika mereka ingin bootstrap rantai blok secara massal.

Node dapat dipastikan transaksi mereka diterima oleh jaringan dengan mengamati lalu lintas transaksi yang ditransmisikan dari node lain karena harus melihat beberapa salinan bergema kembali ke dirinya sendiri.

B. IPv6 and Multicast

Membangun di atas koneksi-kurang UDP memungkinkan implementasi masa depan untuk menggunakan multicast IPv6 sebagai pengganti siaran tradisional dan siaran suara. Ini akan mengurangi konsumsi bandwidth jaringan dan memberikan fleksibilitas kebijakan yang lebih banyak ke node ke depan.

C. Performance

Pada saat penulisan ini, 4,2 juta transaksi telah diproses oleh jaringan RaiBlocks, menghasilkan ukuran blockchain sebesar 1,7GB. Waktu transaksi diukur berdasarkan urutan detik. Implementasi referensi saat ini yang beroperasi pada SSD komoditas dapat memproses lebih dari 10.000 transtindakan per detik yang terutama terikat IO.

VII. Penggunaan Sumber daya

Ini adalah ikhtisar sumber daya yang digunakan oleh node RaiBlocks. Selain itu, kami membahas gagasan untuk mengurangi penggunaan sumber daya untuk kasus penggunaan tertentu. Simpul yang dikurangi biasanya disebut node verifikasi pembayaran yang ringan, dipangkas, atau disederhanakan (SPV).

A. Jaringan

Jumlah aktivitas jaringan bergantung pada seberapa banyak kontribusi jaringan terhadap kesehatan jaringan.

1) Representatif: Simpul perwakilan membutuhkan sumber daya jaringan maxi-mum karena mengamati lalu lintas suara dari perwakilan lain dan menerbitkan suaranya sendiri.

2) Trustless: A trustless node mirip dengan simpul perwakilan tapi hanya pengamat, tidak berisi kunci pribadi akun perwakilan dan tidak mempublikasikan suaranya sendiri.

3) Trusting: A trusting node Mengamati lalu lintas suara dari satu perwakilan yang dipercayainya untuk melakukan konsensus dengan benar. Ini mengurangi jumlah lalu lintas suara masuk dari perwakilan yang menuju ke simpul ini.

4) Light: A light node juga merupakan simpul kepercayaan yang hanya mengamati lalu lintas untuk akun yang diminati sehingga memungkinkan penggunaan jaringan minimal.

5) Bootstrap: A bootstrap menyajikan sebagian atau semua buku besar untuk node yang membawa dirinya online. Ini dilakukan melalui koneksi TCP daripada UDP karena melibatkan sejumlah besar data yang memerlukan kontrol aliran lanjutan.

B. Kapasitas Disk

Bergantung pada tuntutan pengguna, konfigurasi node yang berbeda memerlukan persyaratan penyimpanan yang berbeda.

1) Historical: Simpul yang tertarik untuk menyimpan catatan sejarah lengkap dari semua transaksi akan memerlukan jumlah penyimpanan maksimum.

2) Current: Karena perancangan menyimpan saldo terakumulasi dengan blok, node hanya perlu menyimpan blok terbaru atau kepala untuk setiap akun agar dapat berpartisipasi dalam konsensus. Jika simpul tidak tertarik untuk menyimpan sejarah lengkap, ia dapat memilih untuk tetap mempertahankan blok kepala.

3) Light: Simpul cahaya tidak menyimpan data buku besar lokal dan hanya berpartisipasi dalam jaringan untuk mengamati aktivitas pada akun yang diminati atau secara opsional membuat transaksi baru dengan kunci privat yang dimilikinya.

C. CPU

1) Menghasilkan Transaksi: Simpul yang tertarik untuk membuat transaksi baru harus menghasilkan Bukti Kerja untuk meloloskan mekanisme pelepasan RaiBlock. Perhitungan berbagai perangkat keras mengacu pada Lampiran A.

2) Perwakilan: Perwakilan harus memverifikasi tanda tangan untuk blok, suara, dan juga menghasilkan tanda tangan sendiri untuk berpartisipasi dalam konsensus. Jumlah sumber daya CPU untuk node perwakilan secara signifikan kurang dari transaksi yang dilakukan dan harus bekerja dengan CPU tunggal manapun di komputer kontemporer.

3) Observer: Sebuah node pengamat tidak menghasilkan suara sendiri. Karena overhead pembuatan tanda tangan minimal, persyaratan CPU hampir identik dengan menjalankan simpul represen-tatif.

VIII. Kesimpulan

Dalam makalah ini, kami mempresentasikan kerangka untuk kripto yang terpercaya, biaya rendah, rendah-latency yang menggunakan struktur blok-lattice baru dan mendelegasikan Bukti Pemungutan Suara. Jaringan membutuhkan sumber daya minimal, tidak ada perangkat keras pertambangan berdaya tinggi, dan dapat memproses throughput transaksi yang tinggi. Semua ini dicapai dengan memiliki masing-masing blockchains untuk masing-masing akun, menghilangkan masalah akses dan inefisiensi dari struktur data global Kami mengidentifikasi kemungkinan serangan vektor pada sistem dan mempresentasikan argumen tentang bagaimana RaiBlocks tahan terhadap bentuk serangan ini.

Appendix A
PoW Hardware benchmarks

Seperti disebutkan sebelumnya, PoW di RaiBlocks adalah untuk mengurangi spam jaringan Implementasi node kami memberikan akselerasi yang bisa memanfaatkan GPU OpenCL yang kompatibel. Tabel I memberikan perbandingan benchmark kehidupan nyata dari berbagai hard- ware. Saat ini ambang PoW tetap, tapi adaptif Ambang batas dapat diimplementasikan sebagai daya komputasi rata-rata berlangsung.

TABLE I
Hardware PoW Performance

Device	Transactions Per Second
Nvidia Tesla V100 (AWS)	6.4
Nvidia Tesla P100 (Google,Cloud)	4.9
Nvidia Tesla K80 (Google,Cloud)	1.64
AMD RX 470 OC	1.59
Nvidia GTX 1060 3GB	1.25
Intel Core i7 4790K AVX2	0.33
Intel Core i7 4790K,WebAssembly (Firefox)	0.14
Google Cloud 4 vCores	0.14-0.16
ARM64 server 4 cores (Scaleway)	0.05-0.07

Acknowledgment

Kami mengucapkan terima kasih kepada Brian Pugh atas kompilasi dan memformat tulisan ini.

Diedit dan diterjemahkan oleh gotowerdown untuk Raiblocks Whitepaper (Indonesia)

Referensi

[1] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008. [Online]. Available: http://bitcoin.org/bitcoin.pdf
[2] “Bitcoin median transaction fee historical chart.” [Online]. Available: https://bitinfocharts.com/comparison/bitcoin-median_transaction_fee.html
[3] “Bitcoin average confirmation time.” [Online]. Available: https://blockchain.info/charts/avg-confirmation-time
[4] “Bitcoin energy consumption index.” [Online]. Available: https://digiconomist.net/bitcoin-energy-consumption
[5] S. King and S. Nadal, “Ppcoin: Peer-to-peer crypto-currency with proof-of-stake,” 2012. [Online]. Available: https://peercoin.net/assets/paper/peercoin-paper.pdf
[6] C. LeMahieu, “Raiblocks distributed ledger network,” 2014.
[7] Y. Ribero and D. Raissar, “Dagcoin whitepaper,” 2015.
[8] S. Popov, “The tangle,” 2016.
[9] A. Back, “Hashcash - a denial of service counter-measure,” 2002. [Online]. Available: http://www.hashcash.org/papers/hashcash.pdf
[10] C. LeMahieu, “Raiblocks,” 2014. [Online]. Available: https://github.com/clemahieu/raiblocks
[11] D. J. Bernstein, N. Duif, T. Lange, P. Shwabe, and B.-Y. Yang, “High-speed high-security signatures,” 2011. [Online]. Available: http://ed25519.cr.yp.to/ed25519-20110926.pdf
[12] J.-P. Aumasson, S. Neves, Z. Wilcox-O’Hearn, and C. Winnerlein, “Blake2: Simpler, smaller, fast as md5,” 2012. [Online]. Available: https://blake2.net/blake2.pdf
[13] A. Biryukov, D. Dinu, and D. Khovratovich, “Argon2: The memory-hard function for password hashing and other applications,” 2015. [Online]. Available: https://password-hashing.net/argon2-specs.pdf