RaiBlocks: Masrafsız Dağıtılmış Kriptopara Ağı

Colin LeMahieu
clemahieu@gmail.com

***Özet*—Son zamanlarda, yüksek talep ve sınırlı ölçeklenebilirlik piyasada bulunan popüler kriptoparaların transfer süresini ve işlem ücretlerini arttırdı, bu da memnuniyetsiz bir deneyim kazanılmasına neden oldu. Sizleri RaiBlocks ile tanıştırıyoruz, herkesin hesabın kendine ait blockchain'i olduğu Block-lattice teknolojisine sahip yeni bir kriptora para ,neredeyse anında transfer hızı ve sınırsız ölçeklenebilirlik ile birlikte. Her kullanıcının kendine has blockchaini olması, kullanıcıları eş zamanlı olmaksızın ağın geri kalanı ile senkronize etmesini sağlamakta, bu da hızlı transfer zamanları ve minimum maliyeti sağlamaktadır. İşlemler para transfer miktarları yerine hesap bakiyelerini takip eder, bu da güvenlikten ödün vermeden yoğun veritabananı rahatlatır. Bugüne kadar, RaiBlocks ağı sadece 1.7GBlık veri ile 4.2 milyon transfer yaptı. RaiBlock'ın işlem ücretsiz, saliselik transferleri tüketiciler için onu birinci sınıf bir kriptopara yapıyor.**

***Anahtar Kelimeler*—kriptopara, blockchain, raiblocks, dağıtılmış veritabanı, dijital, transferler**

I. Giriş

BITCOIN'IN 2019 yılında ortaya çıkmasından beri, geleneksel ve devlet destekli finansal sistemlerden kriptografi üzerine kurulu modern,güvenilir ve kişilere bağımlı olmaksızın depolamaya ve transfer etmeye uygun olan alana doğru hızla kaydı. [1]. Etkili bir şekilde çalışması için,bir kriptoparanın kolayca transfer edilebilmesi,tek yönlü çalışan,düşük veya hiçbir masrafı olmamalıdır. Transfer sürelerinin artışı,yüksek işlem ücreti ve tartışılır ağ ölçeklenebilirliği Bitcoin'in günlük yaşamda kullanılabilir bir para birimi olup olmadığını tartışılır hale getirdi.

Bu sayfada,size RaiBlocks'u sunuyoruz,sınırsız ölçeklenebilirlik sağlayan ve işlem ücreti olmayan block-lattice veri yapısında kurulmuş düşük gecikmeye sahip kriptopara. RaiBlocks dizayn olarak yüksek performanslı ve günlük kullanıma uygun olacak bir kriptopara olmak üzere kurulmuştur.RaiBlocks protokolü düşük güçteki donanımlarda çalışabilir olması bunu pratik hale getirip günlük kullanım için uygun hale getirir.

Kriptopara istatislikleri yayınlandığı gün itibariyle bu sayfanın mantıklı olduğunu bildirdi.

II. Arka plan

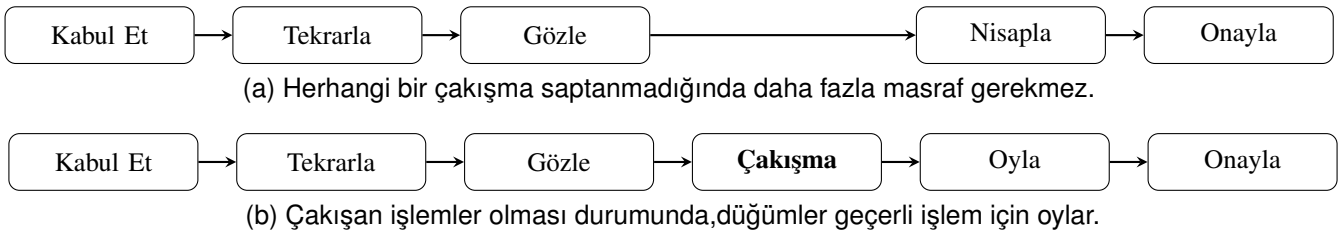
2008 yılında, Satoshi Nakamato takma adında isimsiz bir kişi dünyanın ilk merkezsiz kriptoparasını yayınladı, Bitcoin [1]. Önemli bir yenilik getiren Bitcoin para biriminin işlemlerinin defteri olarak kullanılan kamuya açık, değiştirilemez ve merkezi olmayan bir veri yapısı olan blok zinciri olarak ortaya çıktı. Ne yazık ki, Bitcoin olgunlaştıkça, Protokolde yer alan bazı konular Bitcoin'in birçok uygulama için yetersiz kalmasını sağladı:

1) Düşük ölçeklenebilirlik: Blockchain içerisindeki her blok belirli bir miktarda veri taşıyabilir, bu da sistemin saniyede defalarca işlem yapması demektir bu ise blok üzerinde yoğunluğa neden olur. Şu anki medyan işlem ücreti $10.38'dir. [2].
2) Yüksek gecikme süresi: Ortalama onaylama süresi 164 dakikadır. [3].
3) Güç İsrafi: Bitcoin ağı yıllık yaklaşık olarak 27.28TWh elektrik harcar, bu da ortalama her işlem için 260KWh demektir. [4].

Bitcoin, ve diğer kriptoparalar kötü niyetli aktörlere karşı direnirken işlemlerini doğrulamak için küresel defterlerine karşı fikirbirliği elde ederek işlev görürler., bu fikirbirliğine ekonomik bir ölçüt olan Proof of Work (PoW) denir. PoW sistemlerinde kullanıcılar adeta bir sayı karmaşasını çözmek üzere yarışırlar, buna *nonce* denir, böylece tüm blok karması bir hedef aralığındadır.Bu geçerli aralık, geçerli bir nonce bulmak için ve tutarlı bir ortalama zaman alımı sağlamak için, tüm Bitcoin ağının kümülatif hesaplama gücüne ters orantılıdır Geçerli bir notun bulucusuna daha sonra bloğu ekleme izni verilir.Bu nedenle, nonce hesaplamak için daha fazla hesaplama kaynağı tükenenler Blockchain içerisinde daha büyük bir rol oynamaktadır. PoW,Sybil saldıralarına karşı koruma sağlar, ek güç kazanmak için merkezi olmayan sistem içerisinde birden çok öğe gibi davranır ve ayrıca küresel bir veri yapısına erişirken doğal olarak mevcut olan yarış koşullarını büyük ölçüde azaltır.

Bir alternatif uzlaşma protokolu, Proof of Stake (PoS), ilk defa Peercoin tarafından 2012 yılında ortaya çıkarıldı. [5].PoS sistemlerinde katılımcılar belli bir kriptokraside sahip oldukları servet miktarına eşdeğer güç verilir.Bu düzenlemeyle, daha büyük bir finansal yatırıma sahip olanlara daha fazla güç verilir ve sistemin dürüstlüğünü korumak ya da yatırımlarını kaybetme riskini azaltmak amacıyla teşvik edilir.PoS sistemi bunun yanında güç israfındandan kaçınır durumdadır ve sadece hafif bir uygulama ile düşük donanımda çalışır.

İlk orijinal Raiblocks raporu ve beta uygulaması Aralık 2014 tarihinde yayınlandı bu da onu ilk Directed Acyclic Graph (DAG) kriptoparalardan yapar. [6].Çok az süre sonra, diğer DAG kriptoparaları geliştirilmeye başladı,bunlardan en ünlü olanları DagCoin/Byteball ve IOTA'dır. [7], [8].Bu DAG tabanlı kriptoparalar alışılagelmiş blockchain kalıplarını yıkarak sistem performansını ve güvenliğini arttırdı. Byteball, dürüst, saygın ve kullanıcı tarafından güvenilir "şahitler" den oluşan bir "ana-zincir" e güvenerek fikir birliğine vararken, IOTA, istiflenmiş işlemlerin birikmiş PoW'i vasıtasıyla yenilik sağlıyor.RaiBlocks, çakışan işlemler üzerinde denge ağırlıklı bir oy birliği ile çözüm sağlıyor.Bu çözüm sistemi daha güçlü, merkezi olmayan bir sistemi korurken daha hızlı, daha deterministik işlemler gerçekleştirir. Raiblocks gelişimine devam ederken ve şimdiden kendini en hızlı kriptoparalardan biri olarak kendini konumlandırdı.



Skl. 1. RaiBlocks tipik işlemler için ek yük gerektirmez. Çakışan işlemler durumunda, düğümler tutunabilmek işlemi için oy vermeleri gerekir

III. RaiBlocks Bileşenleri

RaiBlocks mimarisini size anlatmadan önce,sistemi oluşturan bileşenleri tanıyalım.

A. Hesap

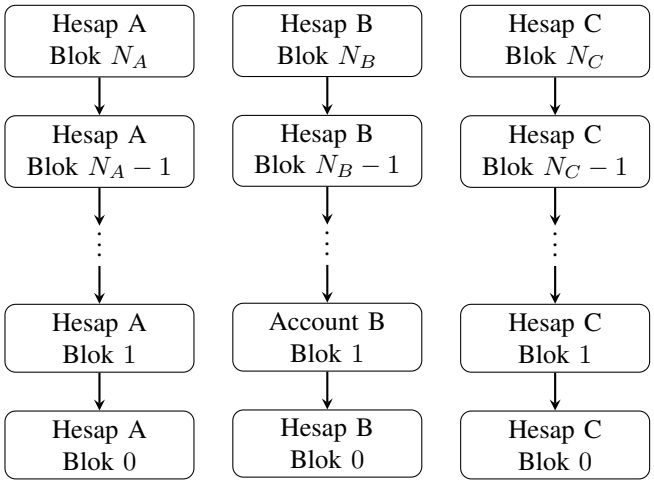
Her hesap, bir dijital imza anahtar-çiftinin ortak anahtar kısmıdır.Açık anahtar, adres olarak da bilinir,özel anahtar gizli tutulurken diğer ağ katılımcılarıyla paylaşılır.Dijital olarak imzalanmış bir veri paketi içeriğin, özel anahtar sahibi tarafından onaylandığını söyler. Bir kullanıcı birçok hesabı kontrol edebilir, ancak her hesap için yalnızca bir genel adres olabilir.

B. Blok/İşlem

"Blok" ve "işlem" terimleri, bir bloğun tek bir işlemi içerdiği yerde genellikle birbirinin yerine kullanılır.İşlem, özellikle, bloğun işlemin dijital kodlamasına atıf yaparken blok eylemi ifade eder. İşlemler, işlemin yapıldığı hesaba ait özel anahtar tarafından imzalanır.

C. Hesap Defteri

Hesap Defteri, her hesabın kendi işlem zincirinin bulunduğu global hesap grubudur. (Şekil 2). Bu, tasarım zamanı anlaşmasıyla bir çalışma anı sözleşmesinin değiştirilmesi kategorisine giren kilit bir tasarım bileşenidir; herkes, sadece bir hesap sahibinin kendi zincirini değiştirebileceğini kontrol ederek dijital imza yoluyla kabul eder. Bu, görünüşte paylaşılan bir veri yapısını, dağıtılmış bir hesap defterini, paylaşılmayan bir veri kümesine dönüştürür.



Skl. 2. Her hesapta, hesabın bakiye geçmişi içeren kendi blok zinciri bulunur. Blok 0 açık işlem olmalıdır. (Bölüm IV-B)

D. Düğüm

Bir *düğüm* bir bilgisayarda çalışan ve RaiBlocks protokolüne uyan aynı zamanda RaiBlocks ağına katılan bir yazılım parçasıdır.Yazılım eğer varsa hesap defterini ve düğümün kontrol edebileceği hesapları yönetir. Bir düğüm, tüm muhasebeyi veya her bir hesabın blok zincirinin yalnızca son birkaç bloğunu içeren budama geçmişini saklar.Yeni bir düğüm oluştururken, geçmişin tamamını doğrulamanız ve yerel olarak azaltmanız önerilir.

IV. Sistem Görünümü

Diğer şifreleme para birimlerinde kullanılan blok zincirlerin aksine, RaiBlocks *block-lattice* yapısını kullanır. Her hesabın, hesabın işlem / bakiye geçmişine eşdeğer kendi blok zinciri (hesap zinciri) vardır.(Şekil 2). Her bir hesap zinciri yalnızca hesap sahibince güncellenebilir; bu, her bir hesap zincirinin blok-kafesin geri kalanına hemen ve eşzamansız olarak güncellenmesini ve böylece hızlı işlemlerin yapılmasını sağlar. RaiBlocks protokolü son derece hafif; her işlem, internet üzerinden iletilmek üzere gerekli minimum UDP paket boyutuna uymaktadır. Düğümlerin donanım gereksinimleri de çok az, çünkü düğümlerin çoğu işlem için blokları kaydetmek ve yeniden yayınlamak zorundalardır.(Şekil 1).

Sistem bir *oluşum bakiyesi*'ne sahip *oluşum hesabı* tarafından başlatılır. Oluşum dengesi sabit bir miktardır ve asla artırılamaz.Oluşum bakiyesi bölünerek, oluşum hesabı zincirinde kayıtlı olan gönderme işlemleri vasıtasıyla diğer hesaplara gönderilir. Tüm hesapların bakiyelerinin toplamı, sisteme miktar üst sınırı ve arttırma kabiliyeti vermeyen ilk genez dengesi asla aşmayacaktır.

Bu bölüm, farklı işlem türlerinin ağ boyunca nasıl oluşturulduğunu ve yayılımını anlatacaktır.

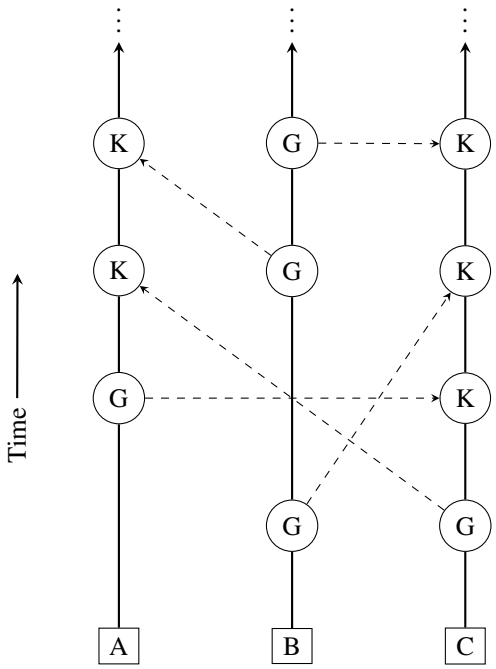
A. İşlemler

Bir hesaptan diğerine para aktarma işlemi iki işlem gerektirir: bir *gönderme* tutarı gönderenin bakiyesinden düşer ve bir *almak* tutarı alıcı hesap bakiyesine ekleme (Şekil 3).

Gönderenlerin ve alıcının hesaplarında tutarlar ayrı işlemler olarak aktarılması, birkaç önemli amaca hizmet eder:

1) Asenkron olmayan gelen aktarmaları sıralamak.
2) İşlemleri UDP paketlerine uyacak şekilde küçük tutma.
3) Veri alanını en aza indirgeyerek defteri budama işlemini kolaylaştırmak.
4) Yerleşik işlemleri, kararsız işlemlerden izole etmek.

Aynı hedef hesaba aktarılmış birden fazla hesap, eşzamansız bir işlemdir; şebeke gecikmesi ve gönderen hesaplar mutlaka



Skl. 3. Block-lattice teknolojisinin görseli. Her bir para transferi, hesap zincirinin sahibi (A, B, C) tarafından imzalanmış bir gönderme bloğu (G) ve bir alma bloğu (K) gerektirir.

birbirleriyle iletişim halinde olmadıkları anlamına gelir; önce hangi işlemin gerçekleştiğini bilmenin hiçbir yolu yoktur. Toplama, birleştirici olduğu için, girdilerin diziliş sırası önemli değil ve bu nedenle sadece bir küresel anlaşmaya ihtiyacımız var. Bu, çalışma anı sözleşmesini tasarım zamanı sözleşmesine dönüştüren önemli bir tasarım bileşenidir. Alıcı hesap, hangi aktarımın önce geldiğine karar vermede kontrol sahibidir ve gelen blokların imzalı sırası ile ifade edilir.

Bir hesap birçok küçük aktarım grubu olarak alınan büyük bir aktarım yapmak istiyorsa, biz bunu bir UDP paketi içine uyacak şekilde temsil etmek istiyoruz. Bir alıcı hesap giriş aktarmalarını sıraladığında, hesap bakiyesinin toplamının bir miktarını tutar; böylece herhangi bir zamanda sabit bir boyutta işlemle herhangi bir miktarı aktarma yeteneği kazanır. Bu, Bitcoin ve diğer kripto para birimleri tarafından kullanılan girdi / çıktı işlem modelinden farklıdır.

Bazı düğümlerin, bir hesabın tam işlem geçmişi kaydetmek için kaynakları harcamaları ilgisizdir; onlar sadece her hesabın şu andaki bakiyesiyle ilgilenirler. Bir hesap bir işlem yaparken, biriken bakiyesini kodlar ve bu düğümlerin yalnızca doğruluğunu korurken geçmiş verilerini atmasına olanak tanıyan en yeni bloğu takip etmeleri yeterlidir.

Tasarım-zaman sözleşmelerine odaklanmış olsa dahi, ağdaki kötü aktörleri tanımlama ve işleme tabi tutma nedeniyle işlemleri onaylarken bir gecikme penceresi vardır. RaiBlocks'daki sözleşmelere milisaniye ile saniye arasında hızlı bir şekilde ulaşıldığından, kullanıcıya iki farklı kategoride gelen işlemler sunuyoruz: Yerleşik ve kararsız. Yerleşik işlemler, bir hesabın alma blokları oluşturduğu işlemlerdir. Tespit edilmemiş işlemler henüz alıcının kümülatif dengesine dahil edilmemiştir. Bu, diğer kripto para birimlerindeki daha karmaşık ve yabancı teyit metriğinin yerine geçer.

B. Hesap Oluşturmak

Hesap oluşturabilmek için, *açık* işlem oluşturmalısınız.(Şekil 4). Açık bir işlem, her hesap zincirinin daima ilk işlemidir ve fonların ilk alındığında oluşturulabilir. *Hesap* alanı imzalama için kullanılan özel anahtardan türetilen ortak anahtarı (adres) depolar. *Kaynak* alanı ise , fonları gönderen işlemin karmasını içerir.Hesap yaratmada, sizin adınıza oy verecek bir temsilci seçilmelidir; bu daha sonra istenirse eğiştirilebilir. (Bölüm IV-F). Hesap kendisini kendi temsilcisi olarak ilan edebilir.

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

Skl. 4. Açık bir işlemin anatomisi

C. Hesap Bakiyesi

Hesap bakiyesi hesap defterinin kendisi içinde kaydedilir. Bir işlem tutarını kaydetmek yerine, doğrulama yaparak (Bölüm IV-I) gönderme bloğundaki denge ve önceki bloğun dengesi arasındaki farkı kontrol eder. Alıcı hesap, daha sonra, yeni alınan blokta verilen son bakiye olarak ölçülen önceki bakiyeyi artırabilir. Bu, yüksek miktarda blok indirirken işlem hızını artırmak için yapılır. Hesap geçmişi talep ederken tutarlar verilir.

D. Bir hesaptan başka hesaba gönderme

Bir adresten başka bir adrese bakiye yollayabilmeniz için karşı hesabın açık bir blok olması,yani dengede olması gerekmektedir. (Şekil 5). *öncekiö*nceki alan, hesap zincirindeki önceki bloğun karmasını içerir. The *hedef* alanı, gönderilecek fonların hesabını içerir. Gönderim bloğu onaylandıktan sonra değiştirilemez. Bir kez şebekeye gönderildiğinde, para gönderenin hesabının bakiyesinden derhal kesilir ve alıcı taraf bu fonları kabul etmek için bir blok imzalayana kadar *askıda kalır* bu alıcı taraf bu fonları kabul etmek için bir blok imzalayana kadar sürer . Bekleyen fonlar, gönderenin hesabından düştüğü ve gönderen işlemi iptal edemediğinden onay bekliyor gibi düşünülmemelidir.

E. İşlem Almak

Bir işlemi tamamlamak için, gönderilen fonların alıcısı, kendi hesap zincirinde bir alma bloğu oluşturmalıdır (Şekil 6).Kaynak alanı, ilişkili gönderme işleminin karmasını referans alır. Bu blok oluşturulup yayınlandıktan sonra, hesabın bakiyesi güncellenir ve fonlar resmen hesaplarına geçer.

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

Skl. 5. Gönderim işleminin anatomisi

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

Skl. 6. Alma işleminin anatomisi

F. Temsilci Atama

Hesap sahipleri, kendi adına oy kullanacak bir temsilci seçme imkânı bulurlar, çünkü Proof of Work ya da Proof of Stake protokollerinde güçlü bir benzerliği olmayan bir yerinden yönetim aracıdır. Geleneksel PoS sistemlerinde, hesap sahibinin düğümü oylamaya katılmak için çalışıyor olmalıdır. Bir düğümü sürekli olarak çalıştırmak birçok kullanıcı için pratik değildir; bir temsilci olarak bir hesaba oy verme yetkisi bu şartı rahatlatır. Hesap sahipleri, fikir birliğini herhangi bir hesaba yeniden atama hakkına sahiptir. Bir *değiştirme* işlemi, eski temsilciden oy ağırlığının çıkarılması ve ağırlığın yeni temsilciye eklenmesiyle bir hesabın temsilcisini değiştirir (Şekil 7). Bu işlemde para yatırılmaz ve temsilcinin hesap fonlarının harcaması olmaz.

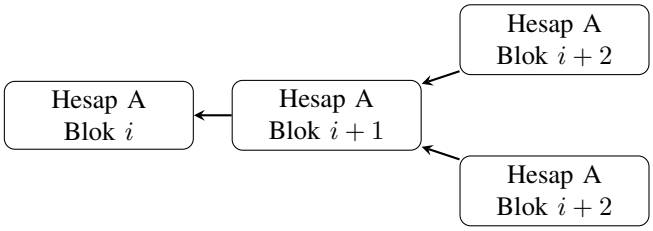
```
change {
   previous: DC04354B1...AE8FA2661B2,
   representative: xrb_1anrz...posrs,
   work: 0000000000000000,
   type: change,
   signature: 83B0...006433265C7B204
}
```

Skl. 7. Bir değişim işleminin anatomisi

G. Çatallar ve Oylama

Bir çatallanma oluşması için j ile $b_1, b_2, \ldots, b_j$ imzalanması gerekir (Şekil 8). Bu bloklar, bir hesabın durumuyla çelişkili bir görüş oluşturur ve çözülmesi gerekir. Yalnızca hesap sahibinin blokları hesap zincirine ekleme yeteneği vardır. Dolayısıyla, bir çatal, hesap sahibinin kötü programlamanın veya kötü niyetin (iki kez harcama) sonucunda olması gerekir.

Tespit edildikten sonra, bir temsilci bloğa referans eden bir oyu hesap defterine yaratacaktır $\hat{b}_i$ ve bunu ağa yayınlayacaktır. Bir düğümün oy ağırlığı, w_i, onun temsilcisi olarak adlandırdığı tüm hesapların bakiyelerinin toplamıdır. Düğüm, diğer M çevrimiçi temsilcilerinden gelen oyları gözlemleyecek ve toplam 4 dakika boyunca toplam 1 dakika kümülatif bir hesap tutacak ve kazanan bloğu onaylayacaktır. (Denklem 1).



Skl. 8. Bir çatallaşma, iki (veya daha fazla) imzalanmış bloklar aynı önceki bloğa referans verdiğinde oluşur. Eski bloklar solda; Yeni bloklar sağda

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

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

En popüler blok b^* oyların çoğuna sahip olacak ve düğümün hesap defterine tutulacaktır (Denklem 2). Oy kaybeden blok(lar) atılır. Bir temsilci,hesap defterindeki bir bloğun yerini alırsa, daha yüksek bir sıra numarası olan yeni bir oy yaratacak ve yeni oyu ağa yayınlayacaktır. Bu **sadece** temsilcilerin oy verdiği bir senaryodur

Bazı durumlarda, kısa ağ bağlantı sorunları yayınlanan bir bloğun tüm chain tarafından kabul edilmemesine neden olabilir. Bu hesaptaki sonraki tüm bloklar, ilk yayını görmeyen emsalleri tarafından geçersiz sayılır. Bu bloğun yeniden yayınlanması kalan emsalleri tarafından kabul edilecek ve sonraki bloklar otomatik olarak alınacaktır. Çatallanma veya eksik blok oluştuğunda bile, yalnızca işlemde referanslanan hesaplar etkilenir; ağın geri kalanı, diğer tüm hesaplar için işlemleri işlemeye ile devam eder.

H. Proof of Work

Dört işlem türünün tümünün doğru doldurulması gereken bir çalışma alanı vardır. Çalışma alanı, işlem yaratıcısı, alıcı / gönderme / değiştirme işlemlerinde önceki alana zincirin eklenmesinin veya açık bir işlemdeki hesap alanının belirli bir eşik değerinin altında kalacağı şekilde bir nonce hesaplamasına olanak tanır. Bitcoin'den farklı olarak, RaiBlocks'daki PoW, Hashcash'e benzer bir anti-spam aracı olarak kullanılır ve saniye cinsinden hesaplanabilir [9]. Bir işlem gönderildiğinde, sonraki blok için PoW önceden hesaplanır çünkü önceki blok alanı bilinir; işlemler arasındaki zaman PoW'yi hesaplamak için gereken zamandan daha uzun olduğu sürece, işlemler son kullanıcıya anında görünür hale gelecektir.

I. İşlem Doğrulaması

Bir bloğun geçerli sayılması için aşağıdaki niteliklere sahip olması gerekir:

1) Blok, hesap defterinin üzerinde olmamalıdır (yinelenen işlem).

2) Hesabın sahibi tarafından imzalanmış olmalıdır.
3) Önceki blok hesap zincirinin baş bloğudur. Var ise ancak baş değilse, çatal.
4) Hesabın açık bir blok olması gerekir.
5) Hesaplanan karma PoW eşiği gereksinimini karşılamaktadır.

Bir alıcı bloğuysa, kaynak bloğu karmasının beklemede olup olmadığını kontrol edin, yani itfa edilmemiş demektir. Bir gönderme bloğu ise, bakiye önceki bakiyeden daha az olmalıdır.

V. Saldırı Vektörleri

RaiBlocks, tüm merkezi olmayan kripto para birimleri gibi, kötü niyetli kişiler taraflar tarafından mali kazanç veya sistem kaybı girişimi için saldırıya uğradı. Bu bölümde, olası olası saldırı senaryolarını, böyle bir saldırının sonuçlarını ve RaiBlock protokolünün önleyici tedbirleri nasıl aldığını özetledik.

A. Boşluk-Engel Senkronizasyonu

Bu bölümde, bir bloğun düzgün şekilde yayınlanamadığı, ağın sonraki blokları yoksaymasına neden olan senaryoyu tartıştık. Bir düğüm, başvurulan önceki bloğun bulunmadığı bir bloğu izlerseniz, iki seçeneğe sahiptir:

1) Kötü amaçlı bir çöp bloğu olabileceği için bloğu yok sayın.
2) Başka bir düğümle yeniden senkronizasyon isteğinde bulunun.

Bir yeniden senkronizasyon durumunda, yeniden senkronun gerektirdiği artan trafik miktarını kolaylaştırmak için bir TCP bağlantının bir önyükleme düğümü ile oluşturulması gerekir. Bununla birlikte, eğer blok aslında kötü bir bloksa, o zaman yeniden senkronizasyon gereksizdi ve gereksiz yere ağdaki trafiği arttırdı. Bu, Bir Ağ Yükseltme Saldırısıdır ve hizmet reddine neden olur.

Gereksiz tekrar başlatılmasını önlemek için, düğümler, potansiyel olarak kötü amaçlı bir blok için belirli bir eşik eşiğinin gözlemlenmesini bekleyecek ve senkronize etmek için bir önyükleme düğümüyle bağlantı kurmaya başlamadan önce bekleyecektir. Bir blok yeterince oy almıyorsa, önemsiz veri olduğu varsayılabilir.

B. Gereksiz İşlemler

Kötü amaçlı bir varlık, ağın doyurulması amacıyla kendi kontrolü altındaki hesaplar arasında gereksiz fakat geçerli işlemler gönderebilir. Hiçbir işlem ücreti olmadan bu saldırıyı süresiz sürdürebilirler. Bununla birlikte, her işlem için gereken PoW, kötü niyetli varlığın hesaplama kaynaklarına önemli miktarda yatırım yapmaksızın verebileceği işlem oranını sınırlar. Defteri şişirmek için böyle bir saldırıda bile tam tarihsel düğüm olmayan düğümler eski işlemlerini kendi zincirlerinden temizleyebilir; bu, hemen hemen tüm kullanıcılar için depolama türü kullanımını bu tip bir saldırıdan etkisiz hale getirir.

C. Sybil Saldırısı

Bir varlık tek bir makinede yüzlerce RaiBlocks düğümü oluşturabilir; Ancak, oylama sistemi hesap bakiyesine dayanılarak ağırlıklandırıldığından, ağa fazladan düğüm eklenmesi bir saldırganın ekstra oy kazanmasına neden olmaz. Dolayısıyla bir Sybil saldırısı ile kazanılmanın bir avantajı yoktur.

D. Kuruş-Harcama Saldırısı

Bir kuruş harcama saldırısı, bir saldırganın düğümlerin depolama kaynaklarını boşa harcamak için çok az sayıdaki hesaba sonsuz küçük miktarda harcamasıdır. Blok yayıncılığı PoW tarafından hızla sınırlandırılmıştır, bu nedenle hesapların ve işlemlerin oluşturulmasını belirli bir ölçüde sınırlandırır. Tam tarihsel düğümler olmayan düğümler, hesapların büyük olasılıkla geçerli bir hesap olmadığı istatistiksel bir metrikin altına budama yapabilir. Son olarak, RaiBlocks, minimum kalıcı depolama alanını kullanacak şekilde ayarlanmıştır; bu nedenle, bir ek hesap depolamak için gereken alan open block + indexing $= 96\text{B} + 32\text{B} = 128\text{B}$'dır. Bu, 8 milyon kuruş harcama hesabı saklayabilecek 1 GB'a eşittir. Düğümler daha agresif olarak budamak istediklerinde, erişim sıklığına dayalı bir dağılım hesaplayabilir ve nadiren kullanılan hesapları daha yavaş depolara devredebilirler.

E. Önceden hesaplanmış PoW Saldırısı

Hesap sahibinin hesap zincirine blok ekleyen tek varlık olması nedeniyle, sıralı bloklar, PoW ile birlikte ağa yayınlanmadan önce hesaplanabilir. Burada saldırgan, her biri en az değeri olan sayısız ardışık bloğu uzun süre üretir. Belirli bir noktada, saldırgan, ağın mümkün olduğunca çabuk yankılanan ve diğer düğümlerin işleyeceği çok sayıda geçerli işlemle sellenerek bir Hizmet Reddi (DoS) gerçekleştirir. Bu işlem selesinin gelişmiş bir versiyonudur Bölüm V-B.Böyle bir saldırı sadece kısaca çalışır, ancak etkinliği artırmak için >50% Attack (Bölüm V-F) gibi diğer saldırılarla birlikte kullanılabilir. İşlem oranını sınırlayan ve diğer teknikler şu anda saldırıları azaltmak için araştırılmaktadır.

F. >50% Saldırısı

RaiBlocks için oylaşmanın bir metriği denge ağırlıklı oylama sistemidir. Bir saldırgan oylama gücünün 50%'sinden fazlasına sahip olabilirse, ağın oybirliğine varıp sistemin bozulmasına neden olabilir. Bir saldırgan, iyi bir düğümün bir ağ DoS aracılığıyla oy vermesini engelleyerek, kaybedilen denge miktarını düşürebilir. RaiBlocks, böyle bir saldırıyı önlemek için aşağıdaki önlemleri alır:

1) Bu türden saldırılara karşı birincil savunma, sisteme yapılan yatırımla bağlantılı oylama ağırlıklıdır. Bir hesap sahibi, sistemin yatırımlarını korumak için dürüstlüğünü korumak için özünde teşvik edilir. Muhasebeyi çevirmeye çalışmak, sistemi tamamen yok ederek yatırımlarını yok edecektir.
2) Bu saldırının maliyeti RaiBlock'ların piyasa değeri ile orantılıdır. PoW sistemlerinde, parasal yatırımla karşılaştırıldığında orantısız denetim sağlayan teknoloji

icat edilebilir ve eğer saldırı başarılı olursa, saldırı tamamlandıktan sonra bu teknoloji yeniden hazırlanabilir. RaiBlocks ile sisteme saldırmanın maliyeti sistemin kendisiyle birlikte ölçeklenir ve bir saldırı başarılı olursa, saldırıya yapılan yatırım geri alınamaz.

3) Azami seçmen sayısını korumak için bir sonraki savunma hattı temsili oylama niteliğindedir. Bağlantı nedenleriyle oylamaya güvenilir şekilde katılamayan hesap sahipleri, bakiyelerinin ağırlığıyla oy kullanabilecek bir temsilci seçebilirler. Temsilcilerin sayısını ve çeşitliliğini en üst düzeye çıkarmak, ağ esnekliğini artırır.
4) RaiBlocks'taki çatallar kazara olmaz, bu nedenle düğümler çatallı bloklarla nasıl etkileşim kuracağına dair politika kararları verebilir. Saldırgan olmayan hesapların, yalnızca bir saldırgan hesaptan bir bakiye alması durumunda, çatal uçurumlara karşı savunmasız olduğu vakittir. Blok çatallardan güvenli olmak isteyen hesaplar, çatal üreten bir hesaptan almadan önce biraz bekleyebilir veya hiçbir zaman hiç almayacağını seçebilir. Alıcılar, diğer hesapları izole etmek için şüpheli hesaplardan para alırken kullanılacak ayrı hesaplar da üretebilirler.
5) Henüz uygulanmayan son bir savunma hattı*blok çimentolama*dır. RaiBlocks, blok çatalılarını oylamayla hızla halletmek için büyük çaba harcar. Düğümler, belirli bir süre sonra geri çevrilmesini önleyecek şekilde çimento bloklarına göre yapılandırılabilir. Ağ, belirsiz çatalları önlemek için hızlı yerleşim süresine odaklanarak yeterince güvenli bir durumda olur.

$> 50\%$ saldırısının daha sofistike bir hali burdadır Şekil 9. Çevrimdışı adında olan, ancak çevrimiçi oy kullanmayan temsilcilerin yüzdesi:"Pay" saldırganın oylayacağı yatırım miktarı."Aktif", çevrimiçi olan ve protokole göre oy kullanan temsilciler. Bir saldırgan, diğer bir seçmenleri bir ağ DoS saldırısı yoluyla offline duruma getirerek tahakkuk ettirilmesi gereken miktarı telafi edebiliyor.Bu saldırı devam ederse, saldırıya uğramış temsilcilerin senkronize edilmiyor ve bu "Eşzamansız" ile gösteriliyor.Son olarak, bir saldırgan, eski saldırı defterini yeniden senkronize ederken Hizmet Reddini saldırılarını yeni bir temsilci grubuna geçirerek göreceli oylama gücünde kısa sürede bir kazanç elde edebilir; Bu, "Attack" tarafından gösterilir.

Offline	Unsync	**Attack**	Active	Stake

Skl. 9. 51% saldırı gereksinimlerini düşürebilecek olası bir oylama düzenlemesi.

Bir saldırgan, bu şartların bir kombinasyonu ile Stake >Etksin'e neden olabiliyorsa, kazancının bedeli karşılığında,hesap defterinin üstünde oyları başarıyla bulabilirler. Diğer sistemlerin piyasa değerini inceleyerek bu tür saldırıların maliyetinin ne olacağını tahmin edebiliriz. Eğer 33% temsilcileri DoS aracılığıyla çevrimdışı veya saldırıya uğramış ise,bir saldırgan saldırabilmek için market kapasitesinin 33%'üne sahip olmalıdır

G. Önyükleme Zehirlenmesi

Bir saldırgan, eski bir özel anahtarı ne kadar uzun süre dengede tutabilirse, o zaman var olan dengelerin, hesap bakiyeleri veya temsilcileri yeni hesaplara geçtiği için katılan temsilcileri olmayabilir. Bu, bir düğüm, saldırganın o sırada o noktada temsilcilere kıyasla oylama payı olan eski bir ağ temsilciliğine önyüklenebilirse o düğüme oylama kararları verebilir. Bu yeni kullanıcı, saldıran düğümün yanı sıra herkesle etkileşim kurmak isterse, işlemlerinin tamamı farklı kafa bloklarına sahip olduklarından reddedilecektir. Net sonuç, düğümlerin ağdaki yeni düğümlerin zamanını kötü bilgiler besleyerek harcayabilmesidir. Bunu önlemek için düğümler ilk hesap veritabanı ve bilinen iyi blok başları ile eşleştirilebilir; bu, veritabanını genesis bloğuna geri yüklemek için kullanılan bir yedektir. Karşıdan yüklemenin mevcut olabilmesi ne kadar yakınsa, bu saldırıya karşı doğru şekilde savunma olasılığı da o kadar yüksektir. Sonunda, bu saldırı, çağdaş bir veritabanına sahip olan herhangi biriyle işlem yapamayacakları için önemsiz verileri önyükleme sırasında düğümü beslemekten daha kötüdür.

VI. Uygulama

Şu anda referans uygulaması C ++'da uygulanmaktadır ve 2014'ten bu yana Github'da bülten üretilmektedir. [10].

A. Dizayn Özellikleri

RaiBlocks uygulaması, bu yazıda özetlenen mimari standardına uymaktadır. Ek spesifikasyonlar burada açıklanmaktadır.

1) İmzalama Algoritması: RaiBlocks, tüm dijital imzalar için Blake2b karma ile değiştirilmiş bir ED25519 eliptik eğri algoritması kullanır [11]. Hızlı imzalama, hızlı doğrulama ve yüksek güvenlik için ED25519 seçildi.

2) Hashing Algoritması: Karma algoritması yalnızca ağ spamini önlemek için kullanıldığından algoritma seçimi madenciğe dayalı kripto para birimleri ile karşılaştırıldığında daha az önemlidir. Uygulamamız blok içeriğine karşı Blake2b'yi bir özet algoritması olarak kullanmaktadır. [12].

3) Anahtar Tespit Fonksiyonu: Referans cüzdanında, anahtarlar bir parola ile şifrelenir ve ASIC kırma girişimlerine karşı koruma sağlamak için parola bir anahtar türetme işlevi üzerinden beslenir. Şu an Argon2 [13]esnek bir anahtar türetme fonksiyonu yaratmayı amaçlayan tek halk rekabeti birincisidir.

4) Blok Aralığı: Her hesabın kendi blok zinciri olduğundan, güncellemeler ağ durumuna eşzamansız yapılabilir. Bu nedenle blok aralıkları yoktur ve işlemler anında yayınlanabilir.

5) UDP İleti Protokolü: Sistemimiz mümkün olan en düşük miktarda bilgi işlem kaynağı kullanarak sınırsız olarak çalışacak şekilde tasarlanmıştır. Sistemdeki tüm iletiler stateless olarak tasarlanmış ve tek bir UDP paketine sığdırılmıştır. Bu ayrıca, kesintili bağlantıya sahip lite eşlerinin, kısa vadeli TCP bağlantılarını yeniden kurmadan ağa katılmasını kolaylaştırır. TCP, blok zincirlerini toplu haliyle önyüklemek istediklerinde yalnızca yeni eşler için kullanılır.

Düğümler, işlemlerinin diğer düğümlerden gelen işlem yayın trafiğini gözlemleyerek ağ tarafından alındığından emin olabilir, çünkü birkaç kopya kendini tekrar yansıtır.

B. IPv6 ve Multicast

Bağlantısız UDP'nin üstünde kurulmak gelecekteki uygulamaların geleneksel çoklu trafik akışı ve oy yayınında yerini alması için IPv6 çok noktaya yayın'ı kullanmasına olanak tanır. Bu, ağ bant genişliği tüketimini azaltacak ve düğümlere daha fazla politika esnekliği sağlayacaktır.

C. Performans

Bu yazının yazıldığı tarihte, RaiBlocks ağı tarafından 4.2 milyon işlem gerçekleştirildi ve 1.7GB'lık bir blokaj boyutu elde edildi. İşlem süreleri saniye cinsinden ölçülür. Örnek SSD'lerinde çalışan mevcut bir referans uygulaması, esas olarak IO'ya bağlı olan saniyede 10.000'den fazla işlemi işleyebilir.

VII. Kaynak Kullanimi

Bu, bir RaiBlocks düğümünün kullandığı kaynakların bir özetidir. Ayrıca, özel kullanım örnekleri için kaynak kullanımını azaltmak için fikirler üzerinde duruyoruz. İndirgenmiş düğümlere genellikle hafif, budama veya basitleştirilmiş ödeme doğrulama (SPV) düğümleri denir.

A. Ağ

Ağ etkinliği miktarı, ağın bir ağın sağlığına ne kadar katkıda bulunduğu üzerine bağlıdır

1) Temsilci: Temsilci bir düğüm, diğer temsilcilerin oy pusulasını gözlemleyip kendi oylarını yayınladığı için maksimum ağ kaynağı gerektirir.

2) Aracısız: Aracısız düğüm temsili bir düğüme benzer ancak yalnızca bir gözlemci, temsili bir hesabın özel anahtarını içermez ve kendi oylarını yayınlamaz.

3) Güvenilir: Bir düğüm, fikir birliği sağlanması için güvendiği bir temsilcinin oy trafiğini gözlemlemektedir. Bu, bu düğüme giden temsilcilerin gelen oy trafiği miktarını azaltır.

4) Hafif: Hafif bir düğüm, en az ağ kullanımına izin verdiği, yalnızca ilgilendiği hesapların trafiğini gözlemleyen güvenen bir düğümdür.

5) Önyükleme: Bir önyükleme düğümü, kendilerini çevrimiçi duruma getiren düğümler için defterin tamamına veya tamamına hizmet eder. Gelişmiş akış denetimini gerektiren büyük miktarda veri içerdiğinden, UDP yerine TCP bağlantısı üzerinden yapılır.

B. Disk Kapasitesi

Kullanıcı taleplerine bağlı olarak, farklı düğüm yapılandırmaları farklı depolama gereksinimlerini gerektirir.

1) Tarihsel: Tüm işlemlerin tam bir geçmiş kaydını tutmak isteyen bir düğüm, maksimum miktarda depolama alanı gerektirir.

2) Şimdiki: Bloklarla birikmiş bakiyelerin tutulması tasarımı nedeniyle, düğümlerin fikir birliğine katılabilmesi için her bir hesap için en yeni veya en başta gelen blokları tutması yeterlidir. Eğer bir düğüm tam geçmişi korumaktan ilgisiz ise, yalnızca baş engellerini korumayı tercih edebilir.

3) Hafif: Hafif bir düğüm, yerel bir defter veri saklamaz ve ilgilenen hesaplarda etkinliği gözlemlemek veya isteğe bağlı olarak tuttuğu özel anahtarlarla yeni işlemler oluşturmak için yalnızca ağa katılır.

C. CPU

1) İşlem Yaratma: Yeni işlemler oluşturmak isteyen bir düğüm, RaiBlock'un kısma mekanizmasını geçirebilmek için Proof of Work üretmelidir. Çeşitli donanımların hesaplanması Ek'te kıyaslanmıştır. A.

2) Temsilci: Bir temsilci blokların imzalarını doğrulamak, oy kullanmak ve fikir birliğine katılmak için kendi imzalarını üretmek zorundadır. Temsili bir düğüm için CPU kaynaklarının miktarı, işlem üretmekten çok daha azdır ve çağdaş bir bilgisayardaki herhangi bir tek CPU ile çalışmalıdır.

3) Gözlemci: Bir gözlemci düğümü kendi oylarını kendi başına üretemez. İmza oluşturma yükü minimum olduğundan, işlemci gereksinimleri temsil eden bir düğüm çalıştırmakla hemen hemen aynıdır.

VIII. Sonuç

Bu yazıda, yeni bir block-lattice yapısını ve aracısız, sınırsız, düşük gecikmeli bir kriptoparayı temel hatları ile sizlere sunduk. Ağ, minimum kaynak gerektirir, yüksek güç gerektiren madencilik donanımı gerektirmez ve yüksek işlem çıktılarını işleyebilir. Tüm bunlar, her hesap için ayrı blok zincirlere sahip olmak suretiyle sağlanır ve erişim sorunlarını , küresel veri yapısının verimsizliklerini ortadan kaldırır. Sistem üzerinde olası saldırı vektörleri tespit ettik ve RaiBlocks'ın bu saldırı türlerine karşı nasıl direneceğini gösteren argümanları sizlere sunduk.

Ek A
PoW Donanim Testleri

Daha önce belirtildiği gibi, RaiBlocks'daki PoW, ağ spamini azaltmaktır. Düğüm uygulamanız OpenCL uyumlu GPU'lardan yararlanabilecek ivmesağlar. Tablo I da bazı donanımların karşılıklı testleri verilmiştir.Şu anda PoW eşiği sabit ancak adaptif eşik ortalama bilgi işlem gücü olarak uygulanalınabilir.

TABLO I
Donanim PoW Performansi

Device	Saniyedeki İşlem Sayısı
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

Teşekkürler

Brian Pugh'a bu yazıyı düzenlediği için teşekkür ederiz.

REFERANSLAR

[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