Adamnite: Ölçeklenebilir ve erişilebilir bir blockchain geliştirme platformu

**Archie Chaudhury**

**archchaudhury02@gmail.com**

### Giriş

Bitcoin'in 2009 yılında ortaya çıkmasından bu yana, kripto para birimleri ve diğer dijital varlıklar finansı yeniden şekillendirdi. Bitcoin, temel varlık için geçerliliği aşılamak için hiçbir merkezi otoriteye veya güce dayanmayan, eşler arası ağ tarafından desteklenen merkezi olmayan bir para biriminin ilk başarılı uygulamasıydı. Son zamanlarda, dijital varlıklar, sadece para birimi transferlerinin ötesinde kullanım durumları nedeniyle büyük ilgi gördü, Non-Fungible-Tokens (NFT'ler) ve diğer kullanım durumları kendi başlarına güçlü pazarlar haline geldi. Yine de, dijital varlıkların önemine rağmen, Bitcoin'in icadı, veri depolama, doğrulama ve fikir birliği için bir araç olarak blok zinciri teknolojisinin yaratılmasına da yol açtı. Yakın muadili Dağıtılmış Defter Teknolojisi (DLT) ile birlikte Blockchain teknolojisi, İnternet'i yeniden şekillendirdi; Merkezi Olmayan Otonom Kuruluşlar (DAO'lar), Gezegenler Arası Dosya Depolama (IPFS) ve Merkezi Olmayan Uygulamalar (DApp'ler), yaygın olarak “Web3” olarak adlandırılan yeni İnternet'in parçalarıdır. Buna göre Adamnite, dünya çapındaki geliştiricilerin güvenle Web3 uygulamaları oluşturmasına izin verecek tamamen izinsiz bir blok zinciri platformu üretmeyi hedefliyor. Güvenli bir DPOS tabanlı fikir birliği mekanizması kullanan, zincir üstü programlar için verimli bir yürütme modelinden yararlanan ve kullanımı kolay yerel bir üst düzey programlama diliyle gelen Adamnite, mevcut blok zinciri çözümlerinin çoğundan daha ölçeklenebilir ve erişilebilir olacaktır.

### Arka plan

Blockchain'i kapsayan daha büyük bir yazılım grubu olan Dağıtılmış Defter Teknolojisi (DLT), kökleri Roma İmparatorluğu'nun başlarında, vatandaşlarının bir kayıt tutma sistemi kullanarak imparatorluğun tamamında takas yapmasına izin verdi. Birden çok düğüm arasında paylaşılan ve merkezi bir yöneticiye bağlı olmayan herhangi bir varlık veritabanı DLT olarak tanımlanabilir. DLT'ler, potansiyellerine rağmen, sistemi ele geçiren kötü niyetli aktörlerin korkusu nedeniyle modern yazılımlarda hiçbir zaman toplu olarak benimsenmedi. Bu problem Leslie Lamport, Robert Shostak ve Marshall Pease tarafından 1982 yılında yayınladıkları aynı isimli gazetede “Bizans Generalleri Problemi” olarak özetlenmiştir. Basitçe söylemek gerekirse, Bizans Generalleri Sorunu, Bizans Ordusu'ndaki birden fazla generalin, merkezi olmayan iletişim platformlarında hainlerin varlığı nedeniyle bir plan üzerinde fikir birliğine varamaması veya daha kötüsü, kötü bir plan seçmesi örneğini kullanır. Tüm dürüst generaller, sistemde kötü niyetli aktörlerin varlığına bakılmaksızın doğru plan üzerinde anlaşmalıdır. Bizans Generalleri Sorununa bir çözüm bulmak veya Bizans Hata Toleransına ulaşmak, DLT'lerin hem geliştirilmesinde hem de benimsenmesinde önemli bir engeldi. Satoshi Nakamoto, dağıtılmış bir defter üzerinde fikir birliğine varmak için kriptografik problemleri çözen katılımcılara dayanan yeni bir algoritma kullanan Bitcoin aracılığıyla Bizans Generalleri Problemini çözdü.

Bitcoin'in ilk başarısı, Proof-of-Work Consensus'un, en uzun geçerli defterin en meşru defter olduğunu varsayma felsefesine atıfta bulunan "En Uzun Zincir Kuralı" ile benzersiz entegrasyonunun doğrudan bir sonucuydu. Bu uygulama Nakamoto Konsensüsü olarak bilinir hale geldi. Bitcoin'de Nakamoto Consensus, madenciler olarak da bilinen blok doğrulayıcıları en uzun ve en doğru zinciri doğrulamaya teşvik ederek Bizans Generalleri Sorununu özel olarak çözdü. Bu, tüm katılımcıların merkezi bir otoriteye bakmak zorunda kalmadan mevcut işlem zincirinin geçerliliğine güven duymalarını sağladı. Bitcoin ayrıca, her işlem için bir zaman damgası kaydı sunarak, Digicash gibi önceki dijital para birimlerini rahatsız eden çifte harcama sorununu da çözdü; bu, tek bir Bitcoin'in aynı hesap tarafından birden çok kez "harcanmamasını" sağladı. Bu işlemleri, daha sonra Nakamoto Konsensüsü aracılığıyla doğrulanan ve halka açık bir defterde saklanan bloklar halinde gruplandırma fikri devrim niteliğindeydi ve şu anda blok zinciri teknolojisi olarak bilinen şeyin ilk gerçek kullanımını temsil ediyordu. Konsensüs mekanizması aynı zamanda ağı kötü niyetli aktörlere karşı da korur: bir saldırganın blok zincirinin kontrolünü ele geçirmek için ağdaki bilgi işlem gücünün %50'sinden fazlasını kontrol etmesi gerekir (genellikle %51 saldırısı olarak bilinir). Bitcoin'in başarısı, genel olarak blockchain teknolojisi ve DLT'nin yolunu açtı. Bitcoin'in piyasaya sürülmesini takip eden yıllarda, her yeni zincir Bitcoin'in orijinal modelini geliştirmek için farklı bir yönteme odaklanan çoklu blok zincirleri ve bunlara karşılık gelen dijital varlıklar piyasaya sürüldü. Örneğin, Ethereum, hem akıllı sözleşmelerin hem de DApp'lerin geliştirilmesini ilerletmek için yerleşik bir Turing Complete programlama diline sahip bir blok zinciri önerirken, Algorand ve Cardano gibi platformlar, kurumsal kullanım için daha ölçeklenebilir blok zincirleri oluşturmak için farklı fikir birliği mekanizmalarından yararlandı.

### Merkezi Olmayan Finansın Ötesindeki Uygulamalar

Bitcoin'in yükselişi, yeni yazılım alt kümelerinin yaratılmasını ve hatta tamamen benzersiz sektörlerin gelişimini teşvik etti; örneğin, Merkezi Olmayan Finans veya DeFi, tamamen Bitcoin'in merkezi olmayan bir para birimini başarılı bir şekilde uygulamasından doğdu. En önemlisi, merkezi bir otorite veya güvenilir bir üçüncü taraf olmadan parasal işlemler yapabilme yeteneği, finans sektörü için devrim niteliğindeydi ve hâlâ da öyle. Bununla birlikte, blok zinciri teknolojisi, yalnızca finansal işlemler ve varlık yönetiminin ötesinde uygulamalara sahiptir. Örneğin, oylama için merkezi olmayan bir yönetim uygulaması, oyların doğrudan halka açık deftere kaydedildiği blok zinciri teknolojisi kullanılarak kolayca oluşturulabilir. Ek olarak, blok zinciri üzerine bir depolama hizmeti de kurulabilir, bu da dosyaların ve diğer veri biçimlerinin halka açık defterde kodlanmasını sağlar. Daha yeni bir yenilik, Non-Fungible-Tokens veya NFT'lerdir. NFT'ler, sahiplik kanıtını temsil eden tekil belirteçlerdir ve dijital sanat, emlak ve içerik yönetiminde çok sayıda kullanım örneği görmüştür. Çoğu geliştirici için blok zinciri teknolojisini kullanarak yeni yazılım oluşturmanın iki ana yolu vardır. Ya tamamen yeni bir blok zinciri oluşturabilirler ya da kendi özel ihtiyaçlarına uyan mevcut bir blok zincirinden yararlanabilirler. Çoğu DApp ve token sistemi, mevcut zincirlerin üzerine inşa edilmiştir, bu nedenle mevcut blok zinciri çözümlerinin mümkün olduğunca ölçeklenebilir olması ihtiyacını belirtir.

Ne yazık ki, çoğu alternatif kullanım durumu şu anda en azından Bitcoin'de sınırlıdır. Bu, büyük ölçüde, eşler arası bir ağ tarafından desteklenen merkezi olmayan bir para birimi için mükemmel olan, ancak daha yüksek seviyeli uygulamalar için kullanıldığında başarısız olan Bitcoin'in fikir birliği modelinden kaynaklanmaktadır. Özellikle, Bitcoin'in POW kullanımı, bireysel işlemlerin maliyetli, verimsiz (ortalama olarak, bir işlem yaklaşık 10 dakika sürer) ve çevre dostu olmadığı anlamına gelir. Bu, Bitcoin'i yalnızca kurumsal kullanım için uygunsuz kılmakla kalmaz, aynı zamanda temel blok zinciri teknolojisinin kullanım durumlarına da kısıtlamalar getirir. İşlem bloklarını işlemek için gereken hesaplama gücü miktarı Bitcoin'i biraz kısıtlar; yönetişim ve oyun uygulamalarının dakikada 1000'lerce işlemi ucuz ve verimli bir şekilde işlemesi gerekebilir; bu, Bitcoin'in şu anda yerel olarak halledemeyeceği bir şeydir.

### Bitcoin Scripti

Bitcoin üzerinde geliştirme öncelikle iki yolla yapılır: Bitcoin Core yazılımına doğrudan eklemeler ve mevcut Bitcoin Protokolünün Bitcoin Script programlama dili ile uygulanması. Burada, Bitcoin blok zincirini merkezi olmayan bir uygulamaya veya akıllı sözleşmeye entegre etmenin bir yöntemi olduğu için Bitcoin Komut Dosyası programlama diline odaklanacağız. Komut dosyası, bir işlemin, alıcının harcanacak veya başka bir yere aktarılacak madeni paraların kilidini nasıl açabileceğine ilişkin doğrudan özelliklere sahip olmasına izin veren yığın tabanlı bir programlama dilidir. Bu, para biriminin belirli bir bölümünün ne zaman ve nasıl kullanıma sunulacağını esasen yöneten Harcanmamış İşlem Çıktıları (UTXO'lar) üzerindeki işlemler aracılığıyla uygulanır. Bu, belirli gereksinimlerin karşılanmasına dayalı olarak ödemeler sunan uygulamalar oluşturmak için kullanılabilecek temel akıllı sözleşmelerin oluşturulmasına olanak tanır. Bu gereksinimler, program tarafından doğrulanabilen görevlerin tamamlanmasından, karşılık gelen miktarda Bitcoin almak için başka bir varlığın “x” miktarını göndermeniz gerektiğini belirten basit bir takas ödemesine kadar değişebilir. Aşağıda, bir ortak anahtar adresine basit bir işlem için tasarlanmış bir Bitcoin Script programının bir kod parçası tanımlanmıştır.

*Kaynak:* [*Bitcoin Wiki*](https://en.bitcoin.it/wiki/Script)

OP\_DUP, OP\_EQUALVERIFY ve OP\_CHECKSIG işlem kodları, sırasıyla en üst yığın öğesinin kopyalanması, tüm girişlerin eşit olup olmadığının kontrol edilmesi ve yığının en üst öğesinin geçerli olup olmadığının kontrol edilmesi için işlevleri tanımlar. Komut dosyası programlama dili, tercih edilen kullanım durumuna bağlı olarak benzersiz sözleşmeler oluşturmak için kullanılabilecek birden çok işlem koduyla birlikte gelir.

Script, doğrudan Bitcoin blok zincirine bağlanan çok sayıda uygulamayı destekleyebilirken, blok zinciri geliştirmede uzun vadeli ölçeklenebilir bir çözüm olmasını engelleyen birkaç temel özellikten yoksundur. Komut dosyası yalnızca UTXO'lara odaklanır, böylece uygulamaları yalnızca blok zincirindeki işlemlerle sınırlar. Bu, geliştiricilerin zincir üzerindeki verileri hesaba katabilecek veya değiştirebilecek akıllı sözleşmeler oluşturmasını engeller. Bu nedenle komut dosyası, uygulamaların blok zincirinin genel durumu ile etkileşime girmesine izin vermez. Senaryo da kesinlikle karmaşıktır; sözdizimi ve yapısı, daha zor kullanım durumları için uygulamayı zorlaştırır. Örneğin, bir MultiSig İşlemi için ortalama bir komut dosyası, işlem işlevinin amaçlandığı gibi olması için tüm kullanıcıların özel komut dosyaları göndermesini gerektirir. Bu, daha karmaşık işlemleri ve işlemleri işleyen komut dosyaları oluştururken verimsizliğe yol açar.

### Ethereum Programlama

İlk teknik incelemesinde “Yeni Nesil Akıllı Sözleşme ve Merkeziuyetsiz Uygulama Platformu” olarak tanımlanan Ethereum Blockchain, ölçeklenebilirliği açısından Bitcoin üzerinde doğrudan bir gelişme olması gerekiyordu. Yerleşik bir Turing-Complete programlama dili ile gelen Ethereum Blockchain, kendi blok zinciri temel çözümlerini güçlendirmek için platformu kullanan çok sayıda geliştirici ve kuruluşla DApp Geliştirme için ana platform haline geldi. Ethereum'da yazılan programlar, hem Turing Eksiksizlikleri hem de durumu işleme kapasiteleri nedeniyle Bitcoin muadillerinden önemli ölçüde daha ölçeklenebilirdir; döngüler ve zincir üstü veriler, hem verimli hem de yanıt veren programlar oluşturmak için sıklıkla kullanılır.

Ethereum'un en popüler programlama dili, ağırlıklı olarak Javascript'e dayanan üst düzey bir programlama dili olan Solidity'dir. Solidity, Turing-Complete'dir ve Solidity'de yazılan herhangi bir kodun daha düşük seviyeli bayt koduna geçirilmesi ve Ethereum Sanal Makinesinde (EVM) çalıştırılması amaçlanmıştır. Tipik bir Solidity programı, esasen daha büyük bir program içinde çeşitli yapıları tanımlayan sınıflar olan sözleşmeler etrafında inşa edilmiştir. Yeni bir varlık, yönetim mekanizması veya kimlik doğrulama aracı için işlemler oluşturmak ve tanımlamak için bir sözleşme kullanılabilir. Solidity's Docs'tan alınan yeni bir varlık oluşturmak için tasarlanmış basit bir akıllı sözleşme aşağıda gösterilmiştir:

|  |
| --- |
| // SPDX-License-Identifier: GPL-3.0  pragma solidity >=0.7.0 <0.9.0;  contract Test\_Coin{   address public minter;  mapping (address => uint) public balances;   event Sent(address from, address to, uint amount);    constructor() {  minter = msg.sender;  }    function mint(address receiver, uint amount) public {  require(msg.sender == minter);  balances[receiver] += amount;  }   error InsufficientBalance(uint requested, uint available);    function send(address receiver, uint amount) public {  if (amount > balances[msg.sender])  revert InsufficientBalance({  requested: amount,  available: balances[msg.sender]  });   balances[msg.sender] -= amount;  balances[receiver] += amount;  emit Sent(msg.sender, receiver, amount);  } } |

Sözleşmeler, işlevlerin, olayların, hataların vb. kullanımını destekler. Bu, geliştiricilerin doğrudan girdiye yanıt veren karmaşık akıllı sözleşmeler oluşturmasına olanak tanır. Örneğin yukarıdaki sözleşme, varlığın bir taraftan diğerine transferini ifade eden ‘Sent’ olayını tanımlar.

Ethereum'un avantajlarına rağmen, yaygın kurumsal kabulün önünde bir engel olarak hareket eden birkaç soruna sahiptir. Yani, yaygın olarak Gaz olarak adlandırılan Ethereum'daki yüksek işlem ücretleri, dağıtım maliyetlerini azaltmak için her akıllı sözleşmenin dikkatli bir şekilde optimize edilmesi gerektiği anlamına gelir. Mevcut ağ tıkanıklığına bağlı olarak, yukarıda açıklanan gibi basit bir akıllı sözleşmenin devreye alınması 400 USD'ye mal olabilir. Ek olarak, Ethereum programları, Bitcoin muadillerinden önemli ölçüde daha yüksek seviyede olsa da, kendi nüansları ile birlikte gelir. En önemlisi, Ethereum kodu yüksek derecede birleştirilebilirlik ve yeniden kullanılabilirlikten yoksundur, bu nedenle aynı kodun sıklıkla tekrar tekrar konuşlandırıldığı bir ekosistem yaratır. Solidity'deki kütüphaneler geçici bir düzeltme sağlamış olsa da, hala yaygın olarak kullanılmaya devam etmektedirler. İç işlevler özyinelemeli olarak çağrılabilir; ancak, hem yığının boyutuna hem de alttaki belleğe bağlı olarak net sınırlar vardır. Sonuç olarak, Ethereum blok zincirinde kurumsal düzeyde bir ölçekte programlama, temel nedenler gaz optimizasyonu ve önemli modülerlik eksikliği ile temel karmaşıklıkları nedeniyle ortalama programcı için son derece zordur. Bu, geliştiricilerin basit hatalar yapmasına neden olarak basit hatalar sonucu ölümcül güvenlik açıklarına sahip uygulamalara yol açar. Kötü niyetli aktörler bu uygulamaları yönlendiren akıllı sözleşmelerdeki mantıksal boşluklardan yararlandığından, bu güvenlik açıkları her yıl önemli mali kayıplara neden olur.

### Alternatif yöntemler ve zincirler

Şu anda Algorand, Cardano ve Polkadot gibi alternatif blok zinciri platformları, blok zinciri teknolojisini hem kurumsal benimseme hem de DApp geliştirme için daha ölçeklenebilir hale getirmeye çalışıyor. Bu platformlar genellikle mevcut çözümler üzerinde bir iyileştirmedir: Algorand ve Cardano, işlem maliyetlerini önemli ölçüde azaltan bir hisse kanıtı uygulaması aracılığıyla zincir üzerinde fikir birliğine ulaşırken, Polkadot birlikte çalışabilirlik, zincirler arası transferler ve aşağıdakilere dayalı verimli bir yürütme modeli sunar. WebAssembly (WASM). Bununla birlikte, bu zincirlerin, merkezileşme, ölçeklenebilirlik eksikliği ve güvenlik eksikliği gibi en sık belirtilen sorunlardan birkaçı gibi kendine özgü sorunları vardır. Mevcut blok zinciri çözümleri genellikle belirli bir alan veya kullanım durumu için altyapı geliştirmeye odaklanır ve çoğu zincir DeFi'ye büyük önem verir. Bu, varlık aktarımlarını dikte etmek için çok uygun, ancak veri depolama gibi diğer kullanım durumları için daha az uygun olan platformlara yol açar.

Ayrıca, Ethereum ve Bitcoin'de olduğu gibi, çoğu alternatif zincir için öğrenme eğrisi son derece diktir. Yetenekli bir geliştirici, otomatik işlemleri nispeten kolaylıkla gerçekleştirebilmelidir, ancak zincir üzerindeki verileri manipüle eden karmaşık akıllı sözleşmeler veya uygulamalar oluşturmak çok daha zordur. Tam teşekküllü bir DApp oluşturmak için geliştiricilerin genellikle çok az belge içeren düşük seviyeli dillerle uğraşmaları, uygulamalarının güvenli olduğundan emin olmaları ve gereksiz kullanım durumları için yapısal destek bulmaları gerekir. Topluluk temelli geliştirme (Algorand'da birden fazla SDK'nın oluşturulması iyi bir örnektir) blok zinciri geliştirmeyi daha erişilebilir hale getirmiş olsa da, bunlar hala blok zincirinin çekirdek akıllı sözleşme mimarisine entegre edilmemiştir. Mevcut blok zinciri çözümleri, merkezi olmayan para birimlerini kolaylaştırmada başarılı olsa da, standart uygulama geliştirme için kullanımları zor olmaya devam ediyor. Bu, çoğu geliştirici daha verimli ve daha kolay eski çözümleri tercih edeceğinden, blok zincirinin finansın ötesinde benimsenmesini önemli ölçüde sınırlar. Ayrıca, alternatif zincirlerin çoğu, anlamlı bir benimseme oluşturmak için genellikle EVM uyumluluğuna bağımlıdır ve çoğu zaman, yüksek seviyeli dilleri olarak Solidity'ye yerleşir, bu da akıllı sözleşmeler geliştirmek için çok az veya hiç geçerli alternatifi olmayan bir ekosistemle sonuçlanır.

### Adamnite

Adamnite, hem profesyonel geliştiricilerin hem de hobicilerin zorlu bir öğrenme sürecinden geçmek zorunda kalmadan veya büyük miktarda hesaplama kaynağından ödün vermeden blok zinciri teknolojisinden yararlanmalarına izin vererek blok zinciri geliştirmenin geleceğini temsil etmeyi amaçlamaktadır. Ayrıca, kurumsal düzeyde kullanıma önemli ölçüde odaklanılacaktır: Adamnite blok zinciri, mevcut blok zinciri çözümlerinden daha hızlı, daha güvenli, daha güçlü ve uygulanması daha ucuz olmalıdır. Adamnite bunu, kullanıcı/geliştirici odaklı ve aynı zamanda büyük ölçekli uygulama geliştirme için optimize edilmiş bir blok zinciri geliştirme platformu sunarak başarır. Adamnite, daha sezgisel ve anlaşılması daha kolay bir sistem sağlayarak, daha fazla geliştiriciyi ve kuruluşu blok zinciri gelişimini benimsemeye teşvik eden bir platform oluşturmayı umuyor.

#### Temel Özellikler

**Kullanımı Kolay**: Adamnite'ın temel çekiciliği basitliğidir: programlama dili ve ilgili ortamının öğrenilmesi Python, Javascript, vb. kadar kolay olmalıdır. Soyutlama ve modülerlik anahtar olacaktır; Adamnite blok zinciri, artan depolama veya zincirde şişme pahasına gelse bile geliştirme kolaylığını vurgulamalıdır (ancak, ortak sözleşmeler nedeniyle depolamadaki marjinal düşüşün herhangi bir acil depolama maliyetini dengeleyeceğini tahmin ediyoruz). Kitaplıkların kullanımı buna bir örnek olacaktır: Adamnite kullanan geliştiriciler, basit programlar oluştururken kendilerine yardımcı olacak, topluluk tarafından denetlenen kitaplıkları indirme yeteneğine sahip olmalıdır.

**Erişilebilirlik**: Kuruluşlar ve işletmeler, güvenlik, işlem maliyetleri veya yüksek bir öğrenme eğrisi hakkında endişelenmeden Adamnite'ı mevcut çerçevelerine kolayca uygulayabilmelidir. Adamnite Blok Zinciri de erişilebilir kalmalıdır: Bir düğüm işleten herkes blok zincirinin farklı yönlerini doğrulamaya uygun olmalı veya yeni blokları önermek/doğrulamak için lider seçim sürecinde seçilmelidir.

**Ölçeklenebilirlik**: Adamnite'ın çok taraflı akıllı sözleşmeler kullanan blok zinciri üzerine kurulu uygulamalar, çok sayıda kullanıcıyı ve genel olarak yüksek bir mesaj verimini destekleyebilmelidir. Diğerlerinin yanı sıra Polkadot ve Cosmos tarafından kullanılan popüler WASM paradigmasına dayanan bir devlet yürütme modeli, diğer geliştirme platformlarıyla hem verimliliğe hem de potansiyel birlikte çalışabilirliğe izin vermelidir. Bu yürütme modeli, hem ademi merkeziyetçiliği hem de depolama verimliliğini korurken verimli olmalıdır.

### Adamnite İşlemleri

Adamnite, herhangi bir kripto para birimi gibi, blok zincirinde çalışan yerel cüzdanlara ve hesaplara sahiptir. Ethereum ve diğer 2./3. nesil blok zinciri ağları gibi, Adamnite de hesap tabanlı bir model kullanır; bu, işlemlerin ve durum geçişlerinin Bitcoin'de olduğu gibi harcanmamış işlem çıktıları yerine bireysel hesaplarda izlendiği anlamına gelir. Adamnite'ın iki hesap türü vardır: otonom (zincir üstü kodla kontrol edilir) ve manuel (harici bir tarafça kontrol edilir). Tüm hesaplar işlem gönderme ve alma özelliğine sahiptir ve aynı standart alanları içerir:

1. Nonce: Hesabın gönderdiği işlem miktarı.
2. Balance: Hesabın sahip olduğu toplam NITE miktarı
3. Code Hash: Hesabı kontrol eden kodun hashı. Hesap manuel ise bu alan boştur.
4. Storage Hash: Hesap için depolama merkle ağacının kökünü temsil eden bir hash. Hesap manuel ise bu alan boştur.

Adamnite ağındaki işlemler, herhangi bir kripto para birimi platformuyla aynı standart alanları içerir:

1. İşlem türü
2. Gönderenin Herkese Açık Adamnite Adresi
3. Gönderilen NITE miktarı
4. Alıcının Herkese Açık Adresi
5. Mesaj, ek verilerin saklanabileceği isteğe bağlı bir alan.
6. Message\_Size, iletinin bayt cinsinden ne kadar büyük olduğunu açıklayan isteğe bağlı bir tam sayı.
7. ATE\_Max, işlemin maksimum işlem ücreti
8. Gönderenin imzası

Adamnite, geliştiricilerin daha büyük uygulamaları desteklemek için isteğe bağlı boyutta mesajlar göndermesine özellikle izin verir. Adamnite'ta bu mesajlar sıkıştırılır ve doğrudan işlemlerde saklanır. Özerk hesaplar için, bu iletiler daha sonra bir amaca hizmet etmek için sıkıştırılabilir. Bu, merkezi hizmetlere dönmek yerine doğrudan blok zincirinde daha geniş bir uygulama yelpazesinin oluşturulmasına izin verir.

İşlem türü, işlemin bir ödeme işlemi mi (NITE'nin bir hesaptan diğerine transferi) yoksa bir mesaj araması mı olduğunu ifade eder.

ATE\_MAX net ücreti, Ethereum'un net gaz fiyatına benzer şekilde belirlenir: ister harici bir hesap ister akıllı sözleşme ile yapılmış olsun, her işlem, blok zincirine uygulanan toplam işlem gücü veya depolama miktarını belirlemek için analiz edilir. Ethereum'un modeli gibi, bu da kötü niyetli oyuncuların çok fazla bilgi işlem gücü tüketen tekrarlayan işlemler göndererek ağdan yararlanmasını önlemek içindir. Bununla birlikte, Adamnite'ın fikir birliği modeli nedeniyle, net işlem ücretleri eski POW zincirlerinden çok daha düşüktür, bu da zincir üstü uygulamalar için ölçeklenebilirliği ve performansı önemli ölçüde artırır.

### 

### Konsensüs Mekanizması

Adamnite'ın blok zinciri ve konsensüs protokolü, Delegated Proof of Stake'in (DPOS) bir varyasyonunu kullanır. DPOS, katılımcıların onları ağda temsil etmek için bir grup doğrulayıcı veya tanık üzerinde oy kullandığı bir fikir birliği mekanizmasıdır. Bu doğrulayıcılara daha sonra yeni bloklar oluşturma ve onaylama yeteneği verilir. Ödüller iki kez gerçekleşir: yeni bir işlem bloğunu başarıyla öneren doğrulayıcılar, o blokla ilişkili işlem ücretlerinin bir kısmını ödüllendirirken, oylama amacıyla tokenlarını düzenli olarak stake eden aktif katılımcılar, stake ettikleriyle orantılı bir miktarda ödüllendirilir. Önemli bir not, katılmak için bir düğümün yalnızca belirli bir katılım işlemi göndermesi gerektiğidir (gerçek varlıklar olmadan). Bu katılım işlemi, esasen, l'nin keyfi bir sayı olduğu sonraki l blokları için bireysel düğümün tanıklar için tercihlerini (bir düğümün birden fazla tanık seçmesine izin verilir) iletecektir. Bu ayrıca, kötü niyetli faaliyetten şüphelenilirse tanıkların her I blokta pozisyonlarını kaybedebilecekleri anlamına gelir.

Adamnite'ın fikir birliği mekanizmasının yarı resmi bir tanımı aşağıdaki gibidir. Her turun başında (bir tur I bloklarından oluşur), tüm düğümler tanıklar için tercihlerini iletme fırsatına sahiptir. En çok oyu alan tanıklar daha sonra, I + 1 turunun başlangıcına kadar tutarlı kalacak olan m büyüklüğündeki bir tanık havuzuna (havuz A) ayrılır. Her k. blok (k, I'in keyfi bir faktörüdür), bir dizi n/m tanık (n, m'nin keyfi bir faktörüdür), sonraki k blok için doğrulayıcı olarak hareket etmek üzere rastgele seçilir, toplam gibi faktörlerle tartılır. aldıkları oy sayısı (her NITE bir oydur), tanık olarak geçmişleri/itibarları ve protokoldeki bireysel payları. Bu yeni havuzu B havuzu olarak tanımlıyoruz. Son olarak, her blok için B havuzundan rastgele bir blok öneren seçilir. Bir blok önerildiğinde, doğrulanması ve eklenmesi için B havuzunun en az ⅔ tarafından onaylanması gerekir. blok zinciri. Bu süreç, ağ tarafından seçilen tanıklara göre değişen A ve B havuzlarıyla süresiz olarak tekrarlanır. Bir tanığın, örneğin, seçtikleri blok içinde geçersiz işlemleri kodlayarak hatalı davrandığı tespit edilirse, o zaman A havuzundan başka bir tanıkla değiştirilir. katılımcılar sabit bir miktarda kesilir. Bu, onlara oy veren düğümlerin nihayetinde jetonlarını kaybetmelerine ve böylece gelecekte söz konusu tanık için oy kullanma olasılıklarının azalmasına neden olur.

Blok Zinciri ayrıca, blok zincirinde protokol ve yapısal değişiklikler yapmak için Merkezi Olmayan Otonom Organizasyonun (DAO)kine benzer bir yönetişim kullanmayı planlıyor. Bu, Algorand tarafından oluşturulan, kullanıcıların çeşitli tekliflere oy verdiği ve karşılığında ödül kazanma fırsatına sahip olduğu yönetişim mekanizmasına benzer. Adamnite'ta, hem delegeleri (ağın yönetimini denetleyen bireyler) seçmek hem de yetki verenlerden gelen meşru tekliflere oy vermek için yönetişim kullanılacaktır. Bu, hem tanıklara ve paydaşlarına verilecek ödüllerin miktarı gibi protokol düzeyindeki kararları hem de iyileştirme önerileri için ilk tarama gibi geliştirme kararlarını oylamak için kullanılabilir.

DPOS, hızı ve ademi merkeziyetçiliği nedeniyle özellikle bir fikir birliği mekanizması olarak seçilmiştir; blok teklifi ve doğrulamasını az sayıda kişiye devrederek blok zincirinin işlemleri hızlı bir şekilde işlemesine izin verirken, ağdaki her kullanıcının aktif bir söz sahibi olmasına izin verir. Örneğin, Adamnite Ekosisteminde i katılımcı varsa, o zaman n'nin bir sabit olduğu doğrulayıcı havuzu için yalnızca ilk n oy alan seçilecektir. Doğrulayıcılar, hisseleri veya geçmişleri ne olursa olsun, ağ tarafından bir sonraki turda her zaman oylanabilir. Adamnite, söz konusu düğümleri dürüstçe hareket etmeye teşvik ederken yeni bloklar önermek ve doğrulamak için gereken düğüm miktarını kısıtlar, böylece birinci nesil İş Kanıtı veya Kanıt Kanıtı zincirlerinden daha hızlı ve daha verimli olmasına izin verir.

### Blockchain

Adamnite Blok Zinciri, diğer hisse ispatı blok zincirlerine (POS) benzer: hem blok teklifi hem de blok doğrulama için gereken hesaplama kaynakları minimumdur, bloklar bir hesaplama problemini çözmek yerine paylaşılan fikir birliği ile doğrulanır ve tüm bloklar blok zincirine kaydedilir. Ancak, önemli farklılıklar vardır. Adamnite blokları, her bir işlem için tüm bireysel depolama mesajlarının bir listesini ve bayt cinsinden net depolama boyutunu içerir. Ayrıca, bir doğrulama listesi (hem blok teklif edenin hem de doğrulayıcıların genel adreslerinin bir kopyası ve bunların hepsinin gerçekten o belirli blok için seçilen doğrulayıcılar kümesinde olduğuna dair kriptografik bir kanıt) ve blok numarası da kaydedilir. Bir bloğu doğrulamak için, bir tanık W'nin önce önceki bloğun geçerli olduğundan, mevcut bloğun makul bir zaman damgasına sahip olduğundan emin olması gerekir *t*n (tn-1 < tn < tn-1 + 10, burada *t* is dakika olarak kaydedilir) işlem parametrelerine dayalı olarak her işlemin başarılı olduğundan emin olun, blok numarasının geçerli olduğundan emin olun ve son olarak blok teklif sahibinin seçili tanıklar havuzunun bir parçası olduğundan emin olun (önceden B havuzu olarak tanımlanıyordu)Hem seçilen tanıkların kamuya açık kaydına hem de her bir blokta doğrulayıcı listesinin dahil edilmesine dayalı olarak belirli bir blok için tanıkları özel olarak hedef alan kötü niyetli bir taraf hakkında endişeler olsa da, Adamnite'nin benzersiz DPOS planı bunu önlemelidir. Spesifik olarak, Algorand ve Witnet tarafından kullanılanlara benzer bir kriptografik sıralama şeması, bireyi oylama ile belirlenen en üst n adresten seçmek için kullanılır; burada n, doğası gereği ekosistemdeki toplam katılımcı sayısına bağlıdır. Bu işlem 5 bloktan oluşan bir turda her blok için tekrarlanır. Kriptografik sıralama şeması, doğrulanabilir bir rastgele işlevin (VRF) her blok için doğrulayıcıları seçmede kilit bir unsur olmasını sağlar ve böylece saldırganların belirli doğrulayıcıları hedeflemesini son derece zorlaştırır.

Yukarıdaki denklemde h, Doğrulanabilir Rastgele İşlevi (VRF) temsil eder, imza işlevi zaman içinde alır, rastgele bir değer ve M anahtarı ve K, belirli bir adresin önceki "seçim" sırasında aldığı toplam oy sayısını temsil eder. Bu denklem, ek ağırlıklar alacak şekilde değiştirilebilir: örneğin, Adamnite'ın nihai uygulaması için, diğer değişkenlerin yanı sıra hesabın sahip olduğu toplam payı da hesaba katacaktır. Her blok için, onu doğrulamak için n içindeki bir adresin seçilme olasılığı, kendisine tahsis edilen oyların miktarına doğal olarak bağlıdır. Kriptografik sıralama şeması, blok teklif edenin kimliğinin, hatta zincir içindeki çeşitli blokları doğrulamak için hareket eden tanık grubunun bile makul bir şekilde korunmasını sağlamalıdır.

### Adamnite Programlama

Adamnite üzerindeki tüm işlemler ve özerk sözleşmeler, ADM kodu olarak tanımlanan yerel bayt kodunu yürüten Adamnite Sanal Makinesi (ADVM) tarafından yürütülür. ADVM, tüm Adamnite düğümleri arasında paylaşılan ve yüksek seviyeli bir kaynak dilde yazılmış programların blok zincirinde yürütülmesine izin veren bir hesaplama motorudur. ADM kodu Turing-Complete'tir: geliştiricilerin uygulamalar oluştururken mümkün olduğunca fazla esnekliğe sahip olmalarını sağlamak içindir. Geliştiriciler tarafından üst düzey bir dilde yazılan kod, daha sonra ADVM'de çalıştırılan opcode'lara derlenir. Bu işlem kodları, basit işlemleri ve ödemeleri tanımlayabilir veya ağın genel durumunu değiştirmek için harici verileri dikkate alabilir. Adamnite'ın hem güvenlik hem de modülerlik hedefine ulaşmak için ADVM (ve buna bağlı olarak ADM kodu), popüler Web Montajı (WASM) paradigmasının uygulamalarıdır. WASM, farklı ortamlarda ikili talimatları yürütmek için genel bir formattır. En popüler kullanım durumu, internet tabanlı web uygulamaları için ortak bir standart olarak hizmet etmektir. WASM, standardizasyonu ve modülerliği nedeniyle Adamnite için seçildi: geliştiricilerin, WASM çerçevesinde işlemleri yürüten bir aile olması daha olasıdır (örneğin, standart aritmetik, Ethereum Sanal Makinesinde olduğu gibi 256 bit yerine 64 bit kullanır) ve çekirdek uygulamayı desteklemek için gerektiği gibi belirli WASM ikili modülleri. WASM'nin standardizasyonu aynı zamanda zincir üzerinde kod okunabilirliğine de yardımcı olur, böylece kullanıcıların aslında zincir üzerinde dağıtılan düşük seviyeli bayt kodunu doğrulamasını sağlar.

ADVM, sözleşmeler ve zincir üstü uygulamalar yazmak için tasarlanmış üst düzey bir pitonik programlama dili olan A1 Programlama Dili'nin birincil hedefi olacaktır. A1'in diğer popüler alternatif akıllı sözleşme programlama dillerine göre ana avantajı, kullanım kolaylığı, modülerliği ve iddiaya dayalı güvenliğinden gelir: geliştiriciler, dilin temel sözdizimini/semantiğini kolayca öğrenebilmeli, karmaşık oluşturmak için incelenmiş ve önceden tanımlanmış kitaplıkları kullanabilmelidir. programlar ve derleme sırasında kontrol etmek için çeşitli özellikler bildirir. Modülerlik ve kullanım kolaylığı, geliştiricilerin karmaşık uygulamaları hızla oluşturmasını sağlarken, mülk tabanlı testler, bu uygulamaları oluşturan bireysel sözleşmelerin ve programların güvenli olmasını sağlar. A1, 1997'de Mark Miller, Dan Bornstein ve diğerleri tarafından dağıtılmış bilgi işlem için özel olarak oluşturulmuş genel amaçlı bir programlama dili olan E programlama dilinden büyük ölçüde ilham almıştır. E aynı zamanda akıllı sözleşmeler kavramını tanıtan ilk programlama dillerinden biriydi. geliştiricilerin birden fazla taraf arasında anlaşmaları yürütebilecek programları tanımlamasına izin vererek. A1, E'nin temel ilkelerini, nihayetinde bir blok zincirinde yürütülen üst düzey talimatları tanımlayabilen bir programlama dili oluşturmak için genişletir. A1 ayrıca, WASM modüllerine doğal olarak uyan modüler yapısı nedeniyle ADVM ile doğal olarak uyum sağlar.

A1 ekosisteminin ve bir bütün olarak Adamnite'ın gelişimi yinelemeli bir süreç olacaktır; A1 Programlama Dili için daha fazla uygulama, kitaplık ve paket oluşturuldukça, yeni geliştiricilerin başlaması daha kolay olacaktır. Adamnite'ın açık kaynaklı doğası aynı zamanda standart kitaplığının zaman içinde tamamen topluluk tarafından işletileceği ve sürekli bakım için merkezi bir otoriteye veya partiye bağlı olmayan kendi kendini idame ettiren bir ekosisteme yol açacağı anlamına gelir. Bu anlamda, Adamnite'ın topluluğu Python veya C++'dan farklı olmayacak şekilde gelişecektir; amacı, paylaşılan bir geliştirici topluluğunun aşağıdaki gibi işlevsel programlar oluşturmasını sağlamaktır:

|  |
| --- |
| # Ballot example # # Implements voting process  class Proposal:  name: string # proposal's name  count: number # number of accumulated votes  class Voter:  weight: number # weight is accumulated by delegation  hasVoted: boolean # true if person has already voted, false otherwise  vote: number # index of the voted proposal  contract Ballot:  voters: map[address, Voter]  proposals: list   def vote(proposalIdx: number):  voter = voters[msg.sender]  voter.hasVoted = True  voter.vote = proposalIdx   proposals[proposalIdx].count += voter.weight |

### Depolama ve Yürütme

Adamnite, daha önce açıklandığı gibi, özerk hesaplar için bilgi depolamak için benzersiz bir depolama mekanizması kullanır. Çekirdek blok zinciri ikili bir merkle trie şeması (esas olarak Buterin ve Ballet tarafından EIP-3102'de hesap ve durum bilgilerini depolamak için açıklananla aynı) kullanırken, zincir dışı bir veritabanı hem gerçek kodu hem de özerk hesaplar için verileri depolar. Bir hash (muhtemelen güvenli tek yönlü hash fonksiyonu kullanılarak hesaplanır) gerçek blok zincirinde depolanır. Bu şema Adamnite'a iki şekilde yardımcı olur: zincir üzerinde kod yürütmenin verimliliğini artırır (kodla etkileşimlerden kaynaklanan durum değişiklikleri gruplar halinde gerçek blok zincirine yüklenir) ve olması gereken defterin boyutunu azaltarak Adamnite'ın daha vatansız olmasına yardımcı olur. müşteriler ve düğümler tarafından saklanabilir, böylece bireylerin zincirin bir kopyasını indirmesi önündeki engeli azaltır ve böylece blok zincirinin mevcut durumunu bağımsız olarak doğrular. Benzer bir iki katmanlı mimari Algorand'da önerilmiş ve Polkadot'ta uygulanmıştır.

Gerçek dağıtılmış defteri depolamak için kullanılan ikili merkle ağacı, anahtar değerler ve bunların karmaları arasında bir eşlemedir ve çoğunlukla standart yapıyı takip eder. Sözleşme kodunu depolamak için kullanılan zincir dışı veritabanı, daha geleneksel bir anahtar yapıyı takip eder, hesap adreslerini, kod içeren dosyaları ve depolamayı eşler. Bu, ADVM için WASM yapısıyla sorunsuz bir şekilde çalışır, çünkü kod ve ilişkili meta veriler, karmaşık bir dağıtım sürecine gerek kalmadan veritabanına kolayca yüklenebilir. Zincir dışı veritabanının, gerçek blok zincirinin durum bilgilerine ve çeşitli kod yürütmeleri ve işlemleri arasında veritabanının içeriğini doğrulamak için istemcilerin/düğümlerin varlığına kolayca erişebileceğini varsayıyoruz. Ayrıca, veritabanının kendisini depolamak ve içeriğini doğrulamak için tasarlanmış farklı türde istemcilerin varlığını da varsayıyoruz.

Zincir içi kodun yürütülmesi veya mesaj çağrıları, zincir dışı veritabanı aracılığıyla gerçekleştirilir. Çekirdek blok zinciri için tanıkları seçmek için kullanılan aynı DPOS seçimiyle seçilen ayrı bir doğrulayıcı grubu tanımlıyoruz. Bu doğrulayıcılar, yalnızca veritabanında yapılan değişiklikleri doğrulamakla kalmaz, aynı zamanda temel kodu yürüterek özerk hesaplara yapılan çağrıları da yürütür ve özerk hesapların oluşturulmasını yönetir. Çekirdek blok zincirinden farklı olarak, kodun yürütülmesi güvenlik veya canlılık yerine hız için optimize edilmiştir. Tur için tüm kod yürütmelerini yürütmek için doğrulayıcılar grubu arasından tek bir lider seçilir. Bu lider, belirli bir süre içinde (örneğin 10 bloğun yürütülmesine eşdeğer) bir toplu durum geçişi üretir ve verim için optimize eder. Kod yürütmenin bir sonucu olarak gerçek blok zincirindeki depolama ve bakiyelerdeki değişikliklere ilişkin fikir birliği, Solana'nın uygulamasına benzer bir Geçmiş Kanıtı (POH) şeması kullanılarak işlenir. Lider hem yeni kodun yüklenmesini hem de mevcut koda çağrıları yürütür, hem veritabanına hem de zincir üzerindeki duruma bir dizi işlem/değişiklik önerir, zamanın geçtiğini doğrulayan her değişiklik için bir kriptografik kanıt üretir ve seti iletir. daha sonra kontrol eden diğer doğrulayıcılara. POH kullanımı nedeniyle, doğrulayıcılar daha büyük kümedeki bireysel değişiklikleri kontrol edebilir, böylece tüm komitenin bir dizi durum değişikliği üzerinde çok daha hızlı fikir birliğine varmasına izin verir. Onaylayıcıların çoğunluğu (⅔) değişiklikleri onayladıktan sonra, çekirdek blok zinciri için mevcut tanık grubuna ya hep ya hiç toplu olarak gönderilir ve daha sonra durumu güncellemeden önce bağımsız olarak kontrol edilir. Çekirdek blok zincirinin tanıklarının sadece durumla ilgili kontrolleri doğrulaması gerekeceğini belirtmekte fayda var (eğer herhangi bir normal işlem, mesaj yürütmelerinden birinin geçersiz kılınmasına neden olduysa). Yerel bir zincir üstü uygulamanın canlılığını, oluşturulmasından kullanıcı tarafından kullanımına kadar açıklayan bir örnek sunuyoruz.

1. Geliştirici, uygulama mantığını bir A1 akıllı sözleşme biçiminde yazar ve onu ADM koduna (ilişkili meta verilerle birlikte bir WASM dosyası) derler.Developer uploads ADM code to the off-chain database, thus creating a new autonomous account.
2. Zincir dışı doğrulayıcı, yeni bir özerk hesabın oluşturulmasını onaylar, meta verilere dayalı bir depolama anahtarı ve koda dayalı bir kod anahtarı ile veritabanında yeni bir hesap oluşturur. Otonom hesap, onaylandıktan sonra çekirdek blok zincirinde oluşturulur.A user interacts with the autonomous account, thus submitting a transaction or message call.
3. Zincir dışı doğrulayıcılar, kullanıcının hem kullanıcının zincir üzerindeki bakiyesine hem de veritabanındaki özerk hesabın depolama içeriğine çapraz referans vererek yürütmeyi tamamlayabildiğini onaylar.
4. Zincir dışı doğrulayıcılar, çekirdek blok zincirinin tanıkları için kullanıcının gerekli bakiyesini içeren bir durum kontrolü oluşturur ve bunu ya hep ya hiç partisinin bir parçası olarak sunar.

Sonuç olarak, Adamnite'ta kod yürütme hız ve verimlilik için optimize edilirken, temel düzeyde ödemeler ve fikir birliği, canlılık ve güvenlik için optimize edilmiştir. Bu, temel sistemin bütünlüğünü oluşturan temel düzeydeki işlemlerin güvenliğinden ödün vermeden hem oyunlar hem de akış uygulamaları gibi yüksek verimli uygulamaların bulunabileceği bir ekosistem oluşturulmasına olanak tanır.

### Gelecekteki Sorunlar ve Araştırmalar

Akıllı sözleşmeler için sürekli güvenlik, Adamnite'ın önerilen ekosisteminin belirli bir bölümünün devreye girdiği yerdir: Adamnite'ın programlama dillerinin her biri, mantıksal geçersizliği ve geliştirici tarafından tanımlanan belirli iddiaları kontrol eden bir doğrulama motoruna sahip bir derleyiciye sahip olacaktır. Bunu A1 programlama ile entegre etmenin birincil hedeflerinden biri, derleyici içinde bir doğrulama motorunu entegre etmek, böylece WASM'a derlemeden önce mantıksal geçersizlik ve iddialar üzerinde kontrollere izin vermektir. Bir örnek aşağıdadır. Bir geliştirici, akıllı bir sözleşmenin, katılımcı Alice'in bir döngü aracılığıyla keyfi olarak para çekmesine izin vermeyeceğinden emin olmak ister. Bunu, özellikle derleme için kullanılan ayrı bir komut dosyasındaki bir iddia yoluyla bildirmeleri gerekecek. Bu iddia basitçe tek bir değişkenin kontrolü veya bir fonksiyonun çıktısının ara değerini kontrol etmek gibi daha karmaşık bir şey olabilir. Durum ne olursa olsun, derleyici, iddiayı tanımlandığı kod gövdesine göre kontrol edebilecek ve geçerli olup olmadığını görebilecek. Derleyici ayrıca, bir döngünün her iterasyonunun başlangıcında doğru olan koşullar olan döngü değişmezlerini de kontrol edecektir. Derleyicinin güvenliği çoğunlukla mülk tabanlı testler etrafında toplanacak ve böylece geliştiricilerin, dağıtımdan önce amaçlandığı gibi çalıştığından emin olmak için kodlarının tek tek bölümlerini kontrol etmelerine olanak tanıyacak. Bu özellikler, sözleşmeyle etkileşime giren bir tarafın bildiği bilgileri belirleyen karmaşık bir bilgi iddiasıyla iki değişkeni karşılaştıran ikili bir ifade kadar basit olabilir. Bu, geliştiricilerin aynı anda birden fazla platform için akıllı sözleşmeler yazmasına olanak tanıyan bir blok zinciri geliştirme platformu olan Reach tarafından sağlanan doğrulama motoruna benzer. Ancak, A1 için önerilen doğrulama motoru derleyicilerin kendisinde yerleşik olacak ve böylece geliştiricilerin aşina olmadıkları yerel olmayan bir dil öğrenmelerini gerektirmeyecek. Ayrıca, derleyici doğrudan testlerin kullanımını da destekler; geliştiriciler, kendi programlarını test etmek için istismar komut dosyaları yazabilmelidir. QuickCheck'in Haskell'de yazılan programların çok daha güvenli olmasını sağlaması gibi, A1'in doğrulama yazılımı da geliştiricilerin Adamnite ağında güvenli akıllı sözleşmeleri kolayca yazmasına olanak tanır. Doğrulamayı her programın önemli bir parçası haline getirerek, resmi bir doğrulama sürecinin geliştirilmesi, suistimallere daha az eğilimli DApp'lerin oluşturulmasına umarız yardımcı olacak düzenli ve güvenli bir geliştirme sürecine olanak tanır.

### Adamnite'ı daha vatansız hale getirmek veya Adamnite'ı doğrulamaya yönelik istemcilerin/düğümlerin vatansız bir şekilde hareket etmesini sağlamak da önemli bir araştırma hedefidir. Spesifik olarak, bir doğrulayıcının blok zincirinin tüm geçmişini doğrulamak için en son bloğu doğrulamasını sağlayan kısa bir blok zinciri oluşturmak için sıfır bilgi kanıtlarının (ZKP'ler) kullanılması, bireylerin ağı bağımsız olarak doğrulaması için gereken engelleri daha da azaltacaktır. Ayrıca blok zincirinin genel boyutunu büyük ölçüde azaltacak ve böylece onu hesaplama açısından daha verimli hale getirecektir. ZKP'leri kullanan benzer bir çözüm, tüm boyutunun sadece 22 kb olduğu bildirilen MINA blok zincirinde uygulandı. Durumu azaltmanın daha acil bir yolu, ağa gönderilen işlemler için bir son kullanma tarihine sahip olmak ve bakiyesi veya depolaması olmayan boş hesapları budamaktır. Benzer bir çözüm, işlemler için bir son kullanma tarihi getirmenin ve kullanılmayan hesaplardan kurtulmanın bir kripto para birimi ağındaki verimliliği büyük ölçüde artırabileceğini teorileştiren Vault'un yazarları tarafından önerildi. Zincir üstü durumun boyutunu azaltmak, sonuçta hem geliştiricilerin hem de kullanıcıların sansüre dayanıklı olmasına yardımcı olur: bağımsız olarak doğrulayarak ve blok zincirine bağlanarak, ağdaki katılımcılar, kendi başlarına hareket edebilen büyük ölçekli düğüm operatörlerine daha az bağımlı hale gelir. faiz.

### Potansiyel Kullanım Durumları

Varlık Oluşturma

Mevcut blok zincirlerinin üzerine inşa edilen varlıkların popülaritesi hızla arttı ve şimdi kripto para piyasasının önemli bir bölümünü oluşturuyor. Bu varlıklar, fiziksel bir şirketteki öz sermaye payını temsil edebilir, bir tür benzersiz faydaya sahip olabilir veya bir DAO'da yönetişim için kullanılabilir. Alt Jetonlar, genellikle çağrıldıkları gibi, genellikle farklı bir kullanım durumunu desteklemeye veya simgeleştirmeye hizmet eder. Adamnite'ta, doğrudan blok zinciri üzerine yeni bir varlık kolayca oluşturulabilir. Bir Adamnite-Alt Simgesinin oluşturulması, basitçe, yaratıcının adı, miktarı ve adresi gibi standart belirteç parametrelerinin kodlanmasını içerir. İşlem işlevleri, Adamnite'taki her üst düzey dilde yerleşiktir ve geliştiricilerin önceden tanımlanmış parametrelerle varlıklar oluşturmasına olanak tanır. Adamnite'ın varlık yaratma ile başarmayı umduğu şeyin bir örneği, aşağıda yüksek seviyeli Pythonic tabanlı bir dilde verilmiştir:

|  |
| --- |
| def test\_token(minter):  Self.amount = 100000  Self.name = "Test Token"  Self.decimal\_values = 0  Self.symbol = "TT"  Self.creator = minter  Self.Inital\_sender = True |
|  |

Varlığın daha fazla basılmasını, belirli ödülleri, işlemleri veya ücret hesaplamalarını tanımlayan ek işlevler geliştirici tarafından eklenebilir. Adamnite'taki tüm üst düzey diller, geliştiricilerin hem varlıkları kolayca oluşturmalarına hem de akıllı sözleşmelerine belirli bir düzeyde benzersizlik getirmelerine olanak tanıyan çok sayıda yerleşik işlevi destekleyecektir. Adamnite Programlama Yığınının esnekliği, geliştiricilerin, yerleşik sözleşmeleri daha derin bir düzeyde değiştirmek için ya uygulamalı bir yaklaşım benimsemelerini ya da kaynağa daha derinlemesine bakmalarını sağlayacaktır.

Merkeziyetsiz Finans

Adamnite'ın hesapları, otonom işlemlerin doğrudan zincir üzerinde işlenmesine izin veren akıllı sözleşmeler oluşturmak için kolayca kullanılabilir. Bu akıllı sözleşmeler, yalnızca belirli gereksinimlerin karşılanmasına dayalı olarak işlem gönderen DeFi uygulamaları oluşturmak için kullanılabilir. Örneğin, bu, zincir üzerindeki bir varlığın bağımsız bir hesaba transferinin, o hesabın ayrı bir varlıkta eşdeğer değeri ödemesiyle sonuçlandığı bir borsa oluşturmak için kullanılabilir. Adamnite'ın depolama yetenekleri sayesinde, akıllı sözleşmelerde analiz için çok çeşitli veriler kullanılabilir ve böylece daha çeşitli DeFi Uygulamalarının oluşturulmasına izin verilir. Bir örnek, varlıkları teminat olarak ödünç veren ve geçmiş işlemlerine dayalı olarak belirli bir hesap için bir kredi geçmişi oluşturan bir bankacılık uygulaması olacaktır.

Merkezi Olmayan Akış ve Kodlama

Merkezi Olmayan Akış, doğrudan blok zincirinde iletişim merkezleri kurmaktan, merkezi bir tarafa bağımlı olmadan eşler arası iletişimi sağlamaya kadar çeşitli uygulamalara sahiptir. Son yıllarda başarıya ulaşan LivePeer gibi merkezi olmayan akış uygulamaları olmuştur. Adamnite'ın temel programlama dilinin modülerliği sayesinde, Adamnite'ta merkezi olmayan akış uygulamaları kolayca oluşturulabilir. Geliştiriciler, veri depolama ve kodlamayı işlemek için önceden derlenmiş sözleşmelerden yararlanabilecektir. Adamnite'ın hızı da bu uygulamalara yardımcı olacak; Adamnite'tan yararlanan akışlı DApp'ler, Web2 benzerleriyle aynı verimliliğe sahip olmalıdır. Geliştiriciler ayrıca, veri madencilerinin (zincir içi doğrulayıcılarla karıştırılmaması gereken) videoları kod dönüştürme ve amaç için "depolama kiralama" için ödüllendirildiği bir sistem oluşturabilir. Blok zincirinde gönderilen işlemlerde büyük miktarda veriyi doğrudan kodlama yeteneği nedeniyle, kullanıcılar video verilerini doğrudan blok zincirine koyabilecekler ve işlem ücretleri tek endişe kaynağı olacak.

Merkezi Olmayan Özerk Kuruluşlar (DAO)

DAO, en basit haliyle, bir organizasyonun kuralları veya yapısı ile ilgili kararlar almak için bir araya gelen bir grup insandır. Kararlar genellikle, bir bilgisayar programının sonuca göre otomatik olarak değişiklik yaptığı standart bir yönetişim süreci aracılığıyla oylama yoluyla alınır. Kurallar ve kararlar kodun ötesine geçebilir; birçok DAO, dahili sermayenin, yatırımların ve daha fazlasının tahsisi konusunda kararlar alır. Temel bir DAO yapısını uygulamanın anahtarı, bir kuruluşun üyeleri tarafından varılan fikir birliğine ve söz konusu fikir birliğini belirlemek için temel bir yönetişim modeline dayanan değişken koda sahip olmaktır. Yönetim modelleri, merkezi olmayan kuruluşlar ve daha geniş topluluğun belirli bir teklif üzerinde fikir birliğine varmasını sağlamak için yönetişimi kullanan bazı blok zinciri ağları ile blok zinciri topluluğunda da popülerlik kazanmıştır. Basit bir DAO veya yönetim modeli Adamnite üzerinde aşağıdaki şekilde uygulanabilir:

* Yalnızca belirli mantıksal parametrelere göre etkinleştirilen ayrı kod blokları oluşturun. Örneğin, bir varlığın X tutarını bir hesaptan diğerine aktarmayı amaçlayan ve yalnızca yeterli sayıda seçmen işlemi onaylamayı seçtiğinde yürütülen bir kod bloğu olabilir.
* Yeni bir teklif tanımlayan ve seçmenlerin bunun uygulanıp uygulanmaması konusunda oy kullanmasına izin veren akıllı sözleşmeler oluşturun. Akıllı sözleşmenin fiili olarak oluşturulması esnektir ve çeşitli şekillerde yapılabilir. Basit bir yöntem, her kararı temsil eden otonom Adamnit Adresleri oluşturmak ve bu adreslerin ya bir varlığın transferi ya da diğer bazı zincir üstü verilerin manipülasyonu yoluyla oyları kaydetmesini sağlamaktır.
* Bileşenler tarafından verilen karara göre doğru kod bloğunu uygulayın.

Adamnite ayrıca herhangi bir topluluk tarafından yönetişim yöntemleri için kullanılabilir, böylece herhangi bir grup insanın doğrudan blok zinciri üzerinde karar vermesine izin verir.

### Miscellaneous

Neden DPOS?

Bir konsensüs mekanizması olarak DPOS'un iki ortak kusuru olduğu algılanıyor: güvenlik ve merkezileşme. Bu konular el ele; DPOS eleştirmenleri, kötü niyetli bir saldırganın tanıkları tam olarak belirleyebileceği için aynı hesapların tekrar tekrar tanık olarak seçilmesini önlemek için oylama sürecine aktif katılımın olması gerektiğine işaret ediyor, bu nedenle merkezileşme ve güvenlik endişelerine yol açıyor. Adamnite bu sorunu iki şekilde çözer. İlk olarak, Adamnite oylama için teşvikler sunarak, katılımcıların oylama sürecine katılma olasılığının daha yüksek olmasını sağlar. İkinci olarak, Adamnite, kriptografik sorgulamadan yararlanarak, belirli bir blok için tanık havuzunun yarı rastgele olmasını ve en yüksek oy alanların ilk havuzunun tek halka açık bilgi olmasını sağlar.

DPOS, hızı, güvenliği ve güvenilirliği nedeniyle özellikle bir fikir birliği mekanizması olarak seçilmiştir. DPOS'un hem POW'dan hem de POS'tan daha hızlı ve verimli olduğu gösterilmiştir, çünkü çok sayıda adres arasında bir hesaplama gücü kanıtı veya fikir birliği gerektirmek yerine yeni blokları doğrulamak için az sayıda tanık kullanır. Ayrıca, her token sahibi oylamaya katılma fırsatına sahip olduğundan, DPOS merkezi değildir. Son olarak, tanıklar, her an hem konumlarını (ve doğrulama ödülleri kazanma potansiyellerini) kaybedebilecekleri için, dürüst davranma konusunda büyük bir teşvike sahiptirler. Sonuç olarak, Adamnite'ın DPOS konsensüs mekanizması onu daha ölçeklenebilir hale getirmeye yardımcı olacaktır: hem blok önerisini hem de blok doğrulamayı demokratik olarak seçilmiş birkaç doğrulayıcının elinde toplayarak, Adamnite diğer zincirlerden daha hızlı olurken, ademi merkeziyetçiliğin belirli bir yönünü muhafaza edecektir.

Yönetim

Adamnite, yönetişimi, katılımcıların hem delegeler hem de teklifler için oy kullanmaları için bir yol olarak kullanmayı planlıyor. Başlangıçta, Adamnite'taki yönetişim, Cardano ve Algorand tarafından kullanılan yönetişim mekanizmalarına benzer şekilde geleneksel bir madeni parayla hak kazanma planını izleyecektir. Katılımcılardan bir teklife veya delegelere oy vermek için belirli bir NITE miktarını kilitlemeleri istenecektir. Daha sonra belirli bir seçeneğe oy verebilecekler. Kendisine tahsis edilen en yüksek NITE miktarına sahip seçenek, uygulanan seçenek olacaktır. Delegeler söz konusu olduğunda, toplam katılımcı sayısına bağlı olarak yine m olacak şekilde en iyi m adres seçilecektir. Bu sonuçta bir jetonla oylama mekanizmasıdır, çünkü bireysel bir katılımcı daha fazla söz sahibi olmak için daha büyük miktarda NITE'yi kilitleyebilecektir. Madeni parayla oylama belirli bir derecede merkezileşmeye neden olsa da, şu anda en çok düşünülmüş ve yaygın olarak uygulanan çözümdür. Gelecekte Adamnite, hem ademi merkeziyetçiliği hem de güvenliği sağlamak için katılım kanıtı gibi farklı bir yönetişim mekanizmasına geçiş yapacaktır.

Yönetim teklifleri, ödül yapısındaki basit bir ayarlamadan zincir üstü geliştirme için finansmana kadar değişebilir. Delegeler, yeni teklifleri onaylamak ve seçmekten nihai olarak sorumlu olacaklar, ancak herkes inceleme için yeni bir teklif oluşturabilir. Teklifler, delegelerin hem teklifi oluşturan kişiyle hem de daha geniş toplulukla aktif olarak etkileşime girmesiyle Bitcoin İyileştirme Tekliflerine (BIP'ler) benzer şekilde onaylanacak. İçerik oluşturucu, onay için incelenmeden önce teklifini gözden geçirme fırsatına sahip olur. Bu, Adamnite ağını daha merkezi olmayan hale getirmeye yardımcı olan açık bir süreç yaratır.

POH

Zincir dışı veritabanının durumundaki değişiklikler üzerinde hızlı bir şekilde fikir birliğine varmak için kullanılan POH şeması, güvenli, tek yönlü ve çarpışmaya dayanıklı bir hash fonksiyonunun tekrar tekrar hesaplanmasına dayanmaktadır. Bu durumda, kodu yürütmek için seçilen lider, Solana'nın protokolündeki POH oluşturucuya benzer. Bir hesap için bakiye veya depolamada bir değişiklikle sonuçlanan her durum geçişinin yürütülmesinin ardından lider, verilerin bir karmasını oluşturur ve özel anahtarıyla imzalar. Bir kriptografik karma işlevinin kullanılması, diğer doğrulayıcıların, durumdaki iki değişiklik arasında belirli bir süre geçtiğini kanıtlamasına olanak tanır; bir hash fonksiyonu kullandığımız için, değişiklik, onu oluşturmak için gereken süreden çok daha kısa sürede doğrulanabilir. Hash'ler nihai olarak birbirine eklenir; Bu, gelecekteki bir yürütmenin geçerliliği nihai olarak önceki bir yürütmeden türetilen duruma bağlı olduğundan, durum istikrarını sağlar. Bu planın önerilen verimliliği, nihai olarak yürütmenin tek bir etkin liderle sınırlandırılmasına bağlıdır ve diğer doğrulayıcılar bir onay damgası görevi görür. Aşağıdaki tablo, her değişiklik için karşılık gelen POH'nin oluşturulmasıyla birlikte kod ve mesaj çağrılarının yürütülmesini açıklar.

|  |  |
| --- | --- |
| Execution | Hash |
| A Kullanıcısı bir NFT satın alır (depolama ve bakiye değişiklikleri) | HASH (NFT, State) |
| Kullanıcı B, özerk hesap oluşturur (bakiye değişti, yeni depolama ve kod anahtarları oluşturuldu) | HASH(ACCOUNT,HASH(NFT)) |
| A Kullanıcısı, B kullanıcısı tarafından oluşturulan bağımsız hesapla etkileşime girer (bakiye ve depolama değişti) | HASH(INTERACT, HASH(ACCOUNT)) |

Diğer doğrulayıcılar, her bir hash'i hesaplamak için gereken süre nedeniyle farklı durumlarla sonuçlanan hesaplamalar arasında belirli bir süre geçtiğini kanıtlayabilir. Önceki hash'in kullanılması, bir yürütmenin sonucunun değiştirilmesinin, kendisinden önceki hesaplamaların sonuçlarını değiştirmeyi gerektireceği ve ana akım Proof of Work zincirlerinde görülene benzer bir kriptografik yaratacağı bir sistem yaratır.

Para Birimi ve Tokenomik

Adamnite blok zinciri, nite adı verilen kendi yerleşik belirtecine sahip olacaktır. Bu, zincir içi işlem ücretlerini, oylama ödüllerini vb. işlemek için kullanılacaktır. Ayrıca, USD Para Birimi Sistemindeki sentlere ve Bitcoin'deki Satoshilere benzer şekilde daha küçük nite değerleri olacaktır. Ethereum'un ilk teknik incelemesinde olduğu gibi, bu mezhepler, kriptografi ve blok zincirindeki en önde gelen katkıda bulunanlardan bazılarının adını almıştır. Aşağıdaki gibi tanımlanırlar:

* 1: micali
* 1010: sunny
* 1012: vitalik
* 1014: nite

Nite, büyüme oranını kontrol eden matematiksel bir fonksiyonla, kalıcı olarak büyüyen bir arza sahip olacak. Bitcoin gibi sabit arzın aksine, tutarlı bir büyüme oranına sahip olmanın ana argümanı, basitçe ademi merkeziyetçiliktir. Sabit kaynaklara sahip varlıklar genellikle erken benimseyenlerin elinde yoğunlaşır, bu da yeni bireylerin ekosisteme katılmasını engeller. Büyüyen bir arz, ağın her zaman yeni kullanıcıları desteklemeye hazır olmasını sağlar. Token ihracı, yalnızca Adamnite ağını büyütmek amacıyla bir grup seçilmiş delege veya resmi bir kuruluş olabilen, yetki verilmiş bir parti tarafından kontrol edilen bir hesaptan yapılacaktır.

Büyüme fonksiyonu aynı zamanda enflasyonu kontrol etmek içindir. Şu anda, başlangıç bloğunun basımında önerilen enflasyon oranı %10'dur ve ardından %3'lük uzun vadeli enflasyon oranına bir azalma gelir. Bu parametreler, yıllık azalma ile birlikte, yönetişim tekliflerinde daha geniş topluluk tarafından değiştirilebilir. Ayrıca, enflasyon oranı bir ortalama veya hedeften ziyade bir maksimumdur: büyüme işlevi, yanma nedeniyle kaybedilen jetonları, yanlış yerleştirilmiş özel anahtarları vb. hesaba katmaz. Etherem ve Solana tarafından ilgili blok zincirleri için önerilen ihraç modelleri gibi, büyüme oranı sonunda yukarıda önerilen sabite ulaşır:

### Inflation (%) versus Time

A picture containing shoji, tiled

Description automatically generated

### Son

Adamnite, blockchain gelişiminin geleceğini temsil ediyor. Geliştiricilere hızlı, verimli ve güvenli uygulamalar oluşturma araçları sağlayan Adamnite, blok zinciri teknolojisinin daha yaygın olarak kullanıldığı ve eski Web2 çözümlerine tercih edildiği bir dünya için zemin hazırlıyor. Adamnite ayrıca eşler arası hesaplamayı ileriye taşımaya da hizmet eder; Adamnite'ın depolama yetenekleri ve fikir birliği sistemi, bireysel düğümlerin/tanıkların, platformu kullanan diğer geliştiriciler tarafından kodlanmış zincir üstü verileri barındırmak için esasen ödüllendirildiği anlamına gelir. Ancak Adamnite'ın en benzersiz özelliği, geliştirme ekosistemi olacak. Geliştiriciler, DLT veya blok zinciri teknolojileriyle ilgili önceki deneyimlerinden bağımsız olarak, sezgiselliği ve güvenliği nedeniyle Adamnite üzerinde güvenle uygulamalar oluşturabilecekler. Bu, günlük gelişimde blok zincir teknolojisinden yararlanmaya başlamak için gereken engelleri önemli ölçüde azaltacaktır.

Yeni nesil bir blok zinciri platformu fikri tamamen yeni değil: Ethereum gibi protokoller, blok zincirinin benimsenmesine ve yeniliğine şimdiden önemli katkılarda bulundu. Adamnite, mevcut çözümlerden daha verimli, daha güvenli ve kullanımı daha kolay bir platform sağlayarak bu trendi hızlandırmayı hedefliyor. Adamnite, blok zinciri teknolojisinin tam potansiyeliyle kullanıldığı ve herkesin, her yerde ademi merkeziyetçilik ve dağıtılmış bilgi işlemin gücüyle güçlü uygulamalar oluşturmasına izin veren bir dünyanın ön saflarında yer alacak.