**Capítulo 17  
Escalabilidade**

Mesmo que múltiplos casos de uso e sistemas de Prova de Conceito (PoC) tenham sido desenvolvidos usando blockchains, e que a tecnologia funcione bem para muitos cenários, ainda há a necessidade de abordar algumas limitações fundamentais presentes nas blockchains para tornar a tecnologia mais adotável.

No topo da lista desses problemas está a escalabilidade, que é uma limitação significativa. A falta de escalabilidade é uma preocupação geral, onde as blockchains não atendem aos níveis de desempenho adequados esperados pelos usuários quando a cadeia é usada em larga escala.

Existem várias técnicas para resolver o problema de escalabilidade em blockchains, que serão discutidas em detalhes nas seções seguintes. Ao longo do caminho, abordaremos os seguintes tópicos:

* O que é escalabilidade?
* Trilema da escalabilidade em blockchain
* Métodos para melhorar a escalabilidade da blockchain
* Camadas 0, 1, 2 e além

**O que é escalabilidade?**  
Esse é o problema mais importante nas blockchains, o qual pode significar a diferença entre uma ampla adaptabilidade das blockchains ou uso limitado apenas por consórcios privados.

Podemos definir escalabilidade como a capacidade de um sistema de preservar ou reajustar seus atributos para se adaptar à demanda aumentada por seus recursos à medida que sua utilização cresce.

Como resultado de pesquisas substanciais nesta área, muitas soluções foram propostas, as quais são discutidas na seção seguinte.

De uma perspectiva teórica, a abordagem geral para lidar com o problema de escalabilidade geralmente gira em torno de aprimoramentos no nível do protocolo. Por exemplo, uma solução comumente mencionada para a escalabilidade do Bitcoin é aumentar o tamanho do bloco. Isso significaria que um número maior de transações pode ser agrupado em um bloco, resultando em maior escalabilidade.

Outras propostas incluem soluções que transferem certos processamentos para redes fora da cadeia; por exemplo, redes de estado off-chain.

Com base nas soluções mencionadas acima, geralmente, as propostas podem ser divididas em duas categorias: soluções on-chain, que se baseiam na ideia de alterar os protocolos fundamentais nos quais a blockchain opera, e soluções off-chain, que utilizam recursos de rede fora da cadeia para melhorar a blockchain.

Embora existam muitas soluções para o problema de escalabilidade atualmente, e a situação não seja tão ruim quanto na primeira década após a introdução do Bitcoin, observe que a escalabilidade não é fácil de alcançar. Alguns compromissos precisam ser feitos para que uma solução de escalabilidade funcione. Foi conjecturado que os três objetivos das blockchains – descentralização, escalabilidade e segurança – não podem ser alcançados simultaneamente – o chamado trilema da blockchain.

**Trilema da blockchain**  
Com base no problema apresentado por Vitalik Buterin, um dos cofundadores do Ethereum, entende-se que apenas duas das três principais propriedades centrais de uma blockchain podem ser utilizadas ao mesmo tempo. Essas três propriedades centrais são as seguintes:

* **Descentralização** – Isso significa que o sistema opera para participantes com acesso apenas a recursos normais, por exemplo, um computador básico com hardware comum. Em outras palavras, nenhum hardware especializado é necessário para participar da rede, um requisito que poderia favorecer aqueles com mais recursos.
* **Escalabilidade** – Isso significa que o sistema geral é capaz de processar um número maior de transações do que o número de transações que um participante individual pode processar. Em termos mais simples, o sistema é capaz de realizar um alto número de transações.
* **Segurança (ou consistência)** – Isso significa que o sistema é seguro contra adversários mesmo com altos níveis de recursos (embora não ilimitados).

O trilema significa que os três objetivos principais de uma blockchain escalável – segurança, escalabilidade e descentralização – não podem ser plenamente alcançados; é preciso sacrificar pelo menos um deles para atingir os outros dois. Em outras palavras, o trilema da blockchain afirma que uma arquitetura simples de blockchain só pode alcançar duas das três propriedades, por exemplo, se você precisa de uma blockchain segura e descentralizada, será necessário sacrificar a escalabilidade. Da mesma forma, se você deseja alcançar escalabilidade e segurança, então a descentralização deve ser sacrificada.

Isso é conhecido como o trilema da blockchain e é visto como um problema fundamental que precisa ser resolvido antes que a adoção global do Ethereum e de outras blockchains possa ser alcançada.

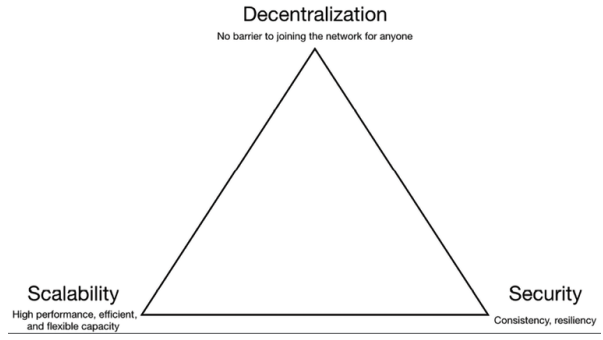


Figura 17.1: Trilema da blockchain

(continua...)

Blockchains monolíticas tradicionais como o Bitcoin realizam todas as operações da blockchain, incluindo disponibilidade de dados, consenso, liquidação e execução, on-chain. A disponibilidade de dados garante que os dados do cabeçalho do bloco estejam publicamente disponíveis, de modo que qualquer pessoa possa recriar o estado para fins de verificação. O consenso, como vimos no Capítulo 5, Algoritmos de Consenso, garante a consistência da cadeia ao obter um acordo entre os participantes sobre a inclusão e ordenação das transações. Liquidação significa a finalização das transações on-chain. Por fim, temos a execução, que significa a execução (cálculo) de uma transação para transicionar do estado atual para o novo estado.

Há três gargalos que afetam negativamente o desempenho da blockchain: execução, verificação e comunicação. Por design, em blockchains monolíticas, todo participante precisa aceitar, processar, encaminhar e executar cada transação, o que torna a rede fundamentalmente lenta.

Pesquisas intensivas estão em andamento e avanços significativos foram feitos nesse sentido na comunidade de pesquisa em blockchain. Com soluções como camada 2, redes multichain e melhorias na camada 1, o problema de escalabilidade foi abordado até certo ponto, embora ainda haja alguns compromissos a serem feitos. Mesmo com todos os avanços mencionados, o escalonamento da blockchain não é simples. Não há uma solução única que sirva para todos os casos devido às diferentes exigências e casos de uso. No futuro, é provável que várias soluções de escalabilidade coexistam. Soluções múltiplas também levam a arquiteturas heterogêneas distintas, o que torna a adoção por desenvolvedores um pouco desafiadora. Além disso, embora um bom progresso esteja sendo feito, o ecossistema da camada 2 passa por evolução constante, o que torna difícil adotar uma única solução de longo prazo.

Uma coisa importante a entender é que, devido ao trilema da blockchain, quando alto desempenho e eficiência são alcançados, ou a segurança ou a descentralização deve ser parcialmente sacrificada. Todas as três propriedades não podem ser alcançadas ao mesmo tempo.

No entanto, isso é um tanto debatível, já que os proponentes da escalabilidade na camada 1, como a Solana, acreditam que todas as três propriedades podem ser alcançadas, e a Solana em particular afirma ter demonstrado isso com a blockchain Solana. Contudo, há alguma preocupação de que essa solução não seja tão descentralizada quanto redes de Prova de Trabalho, pois o número de validadores na primeira não é suficiente para permitir uma descentralização total. Em termos gerais, desde que mais de 50% da rede de validadores esteja sob o controle de uma maioria honesta, espera-se que a rede continue segura.

Observe que, para alcançar desempenho e velocidade, pode ser aceitável abrir mão de algum nível de descentralização como um compromisso, mas isso não deve ser tanto a ponto de comprometer a segurança e, consequentemente, resultar em censura de transações.

Há muitas pesquisas nessa área, e o objetivo é abordar esse trilema e encontrar o equilíbrio correto entre todas as três propriedades em vez de fazer concessões e escolher apenas duas das três. Nesse sentido, o Algorand propôs uma solução chave que afirma ter resolvido o trilema da blockchain sem sacrificar nenhum dos três objetivos. O truque é permitir a seleção aleatória de validadores para escolher o próximo conjunto de nós para adicionar blocos. O algoritmo pode escolher qualquer um aleatoriamente para se tornar o próximo validador que adicionará o bloco, sem abrir mão da descentralização. O algoritmo alcança escalabilidade selecionando aleatoriamente um pequeno conjunto de representantes (comitê) que executam o protocolo em vez de usar todos os nós. Esse pequeno conjunto de proponentes e verificadores de blocos permite que os usuários recebam apenas um número pequeno e fixo de mensagens para alcançar consenso sobre o próximo bloco, alcançando assim velocidade e escalabilidade. Isso é chamado de algoritmo de prova de participação pura.

Com o advento de protocolos de camada 2, aprimoramentos inovadores na camada 1, mecanismos de consenso mais rápidos e outras técnicas como sharding e paralelização, uma combinação razoavelmente equilibrada de todas as três propriedades de descentralização, escalabilidade e segurança (consistência) pode ser alcançada.

Agora, discutiremos alguns métodos para melhorar a escalabilidade.

**Métodos para melhorar a escalabilidade**  
Como essa é uma área muito ativa de pesquisa, ao longo dos anos muitas técnicas e propostas foram feitas para abordar o problema de escalabilidade das blockchains. Nesta seção, introduziremos muitas dessas técnicas.

Discutimos o coeficiente de Nakamoto no Capítulo 2, Descentralização, sobre descentralização.  
Acontece que cadeias como a Solana têm um bom coeficiente de Nakamoto (cerca de 31), mas talvez não o suficiente para impedir ataques de formação de cartel. Outras cadeias como a Polygon têm um coeficiente de Nakamoto ainda menor (cerca de 4), o que não é adequado. Aqui, abre-se mão de muita descentralização em favor da eficiência (velocidade), o que não é o ideal. A Solana parece ser uma escolha melhor, com um coeficiente de Nakamoto de 31. Essas estatísticas são acompanhadas online em <https://nakaflow.io>.

O Algorand é um desenvolvimento muito importante e, de fato, refutou o trilema da blockchain.  
Com o Algorand, é possível alcançar todas as três propriedades, ou seja, escalabilidade, descentralização e segurança. Iremos introduzir o Algorand no Capítulo 23, Blockchains Alternativos.

**Podemos dividir abordagens para resolver o problema de escalabilidade em quatro categorias principais com base na camada na pilha de blockchain em que operam.**

Descrevemos essas categorias aqui:

* **Métodos de Camada 0** (também chamados de métodos multichain, blockchains modulares ou arquitetura polilítica de blockchain) envolvem um ecossistema multichain, permitindo a interoperabilidade entre cadeias e permitindo que desenvolvedores criem cadeias personalizadas.
* **Métodos de Camada 1** também são chamados de métodos on-chain, onde a blockchain e o próprio protocolo de rede são aprimorados para melhorar a escalabilidade. Esta camada representa a blockchain e a rede.
* **Métodos de Camada 2**, ou métodos off-chain, são aqueles em que mecanismos que não fazem parte da blockchain e existem fora da blockchain principal são usados para melhorar a escalabilidade da blockchain.
* **Métodos de Camada 3** (também chamados de escalonamento multilayer ou hiperescalonamento) são baseados na observação fundamental de que se a camada 2, usando rollups, pode comprimir dados até n vezes, então poderia ser possível adicionar outra camada por cima e alcançar compressão de dados múltiplas (até n × n) vezes. Note que há algumas outras visões sobre o que vem além da camada 2, como propostas feitas pela StarkWare – o chamado escalonamento fractal, que introduz camadas 3 específicas para aplicações que são construídas recursivamente sobre a camada 2.

A escalabilidade tem duas facetas. Uma é o aumento da velocidade de processamento das transações para alcançar melhor número de transações por segundo (TPS), e a outra é o aumento do número de nós na rede. Ambas são desejáveis em muitas situações; no entanto, a velocidade de transação é mais procurada em redes públicas, já que a escalabilidade de nós não é um problema em redes públicas de blockchain. Isso é evidente em redes de blockchain como Ethereum e Bitcoin, onde milhares de nós operam na rede, mas a taxa de transferência de transações é aproximadamente de 7 e 15 TPS, respectivamente.

**Camada 0 – soluções multichain**  
Descreveremos algumas das soluções de camada 0, ou camada de rede, nas seções a seguir.

As soluções de camada 0 surgiram por causa das arquiteturas multichain (às vezes chamadas de “blockchain de blockchains”) onde uma cadeia central atua como um relé entre múltiplas sidechains. Alguns exemplos principais de cadeias de camada 0 são Polkadot, Avalanche e Cosmos.

Uma breve introdução ao Polkadot é dada abaixo.

**Polkadot**  
Polkadot é um protocolo moderno de blockchain que conecta uma rede de blockchains criadas para propósitos específicos e permite que elas operem juntas. É um ecossistema multichain heterogêneo com consenso compartilhado e estado compartilhado.

O Polkadot tem uma cadeia principal central chamada de **Relay Chain** (Cadeia de Revezamento). Esta Relay Chain gerencia as parachains – os fragmentos heterogêneos conectados à Relay Chain. A Relay Chain mantém os estados de todas as parachains. Todas essas parachains podem se comunicar e compartilhar segurança, levando a um ecossistema melhor e mais robusto.

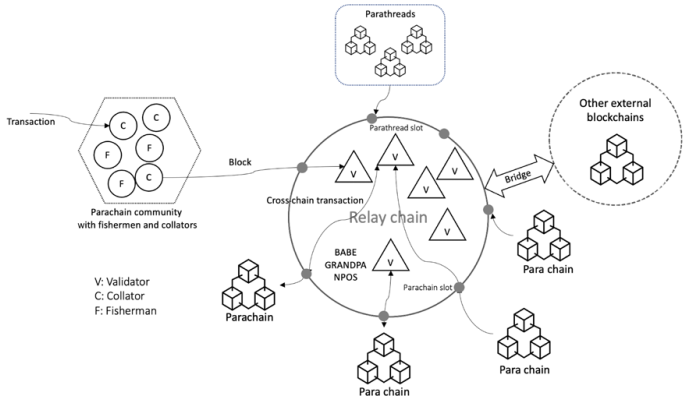


Figura 17.2: Uma representação da rede Polkadot

Como as parachains são heterogêneas, elas podem servir a diferentes propósitos. Por exemplo, uma cadeia pode ser especificamente para contratos inteligentes, outra para jogos, outra para fornecer algum serviço público, e assim por diante. A Relay Chain é protegida por prova de participação nomeada (nominated proof of stake).

Essa arquitetura baseada em sharding com Relay Chain e parachains, com muitas blockchains rodando em paralelo, é como o Polkadot alcança escalabilidade. O sharding permite que muitos cálculos sejam executados em paralelo. O Polkadot também pode se conectar com cadeias privadas, outras cadeias públicas, cadeias de consórcio e oráculos, o que permite interoperabilidade e resulta em maior escalabilidade da rede, melhora do “efeito de rede” e rendimento geral.

Os validadores da Relay Chain produzem blocos, comunicam-se com as parachains e finalizam blocos. A governança on-chain por meio de esquemas de votação baseados em participação decide qual deve ser o número ideal de validadores.

Existem vários componentes no Polkadot. A Relay Chain é a cadeia principal responsável por gerenciar as parachains, interoperabilidade entre cadeias, mensagens intercadeias, consenso e segurança.

Como mostrado na Figura 17.2, a rede consiste em nós e funções. Os nós podem ser clientes leves, nós completos, nós de arquivamento ou nós sentinela. Clientes leves consistem apenas do tempo de execução e do estado. Nós completos são aparados em intervalos configuráveis. Nós de arquivamento mantêm todo o histórico dos blocos, e nós sentinela protegem os validadores e impedem ataques DDoS para fornecer segurança à Relay Chain.

Existem várias funções que os nós podem desempenhar: validador, nomeador, colator e pescador.  
Os **validadores** são o nível mais alto responsável pelo sistema. Eles são os produtores de blocos e, para se tornarem produtores de blocos, precisam fornecer um depósito de caução suficiente. Eles produzem e finalizam blocos e se comunicam com as parachains.  
Os **nomeadores** são participantes com tokens e contribuem com o depósito de segurança dos validadores. Eles confiam que os validadores serão “bons” e produzirão blocos.  
Os **colators** são responsáveis pela execução de transações. Eles criam blocos válidos, mas ainda não selados, para os validadores.  
Os **pescadores** são nós utilizados para detectar comportamentos maliciosos. Pescadores são recompensados por fornecer prova do mau comportamento dos participantes.

As **parachains** são blockchains heterogêneas conectadas à Relay Chain. Estas são fundamentalmente o núcleo de execução do Polkadot. Parachains podem existir com seus próprios tempos de execução, chamados de blockchains específicas de aplicação. Outro componente chamado de **parathread** é uma blockchain que funciona dentro do host do Polkadot e se conecta à Relay Chain. Elas podem ser vistas como cadeias pague-conforme-o-uso. Uma parathread pode se tornar uma parachain por meio de um mecanismo de leilão. **Pontes** são usadas para conectar parachains com redes externas de blockchain como o Bitcoin e o Ethereum.

**Em seguida, passemos a algumas soluções centrais on-chain, ou soluções de camada 1.**

**Camada 1 – soluções de escalabilidade on-chain**  
Nesta seção, descreveremos soluções de camada 1 (nível de rede; on-chain), que visam elementos centrais da blockchain como blocos, transações e outras estruturas de dados on-chain para abordar o problema de escalabilidade.

**Kadcast**  
Este é um novo protocolo que permite a propagação rápida, eficiente e segura de blocos para a rede da blockchain do Bitcoin.

**bloXroute**  
Outra solução de camada de rede é o bloXroute, que visa resolver problemas de escalabilidade ao criar uma rede de distribuição blockchain sem necessidade de confiança.

Outra opção que foi proposta para melhorar a escalabilidade é a **paralelização de transações**, que introduziremos a seguir.

Mais informações estão disponíveis no artigo:  
Rohrer, E. e Tschorsch, F., 2019, outubro. Kadcast: Uma abordagem estruturada para broadcast em redes blockchain. Nos Anais da 1ª Conferência ACM sobre Avanços em Tecnologias Financeiras (pp. 199-213).  
<https://dl.acm.org/doi/pdf/10.1145/3318041.3355469>

Mais informações sobre o bloXroute estão disponíveis em <https://bloxroute.com> e no seguinte whitepaper:  
<https://bloxroute.com/wp-content/uploads/2019/11/bloXrouteWhitepaper.pdf>

**Paralelização de transações**  
Normalmente, nos projetos de blockchain, as transações são executadas de forma sequencial. Por exemplo, no Ethereum, toda a execução de transações é sequencial, o que permite que seja segura e consistente. Isso também levanta a questão: se, de alguma forma, as execuções de transações puderem ser feitas em paralelo sem comprometer a consistência e a segurança da blockchain, isso resultaria em um desempenho muito melhor.

Algumas sugestões já foram feitas para o Ethereum, por exemplo:

* **Paralelização fácil**, que introduz um novo tipo de transação:  
  <https://github.com/ethereum/EIPs/issues/648>
* **Paralelização de transações no Ethereum**, que propõe um mecanismo para permitir execução paralela de transações no Ethereum. O artigo está disponível aqui:  
  <https://arxiv.org/pdf/1901.09942.pdf>

Transações paralelas são suportadas em várias plataformas de blockchain, incluindo o Hyperledger Sawtooth e o Solana. O Solana permite execução paralela de contratos inteligentes usando um runtime chamado **Sealevel**.

**Aumento no tamanho do bloco**  
Essa é a proposta mais debatida para aumentar o desempenho da blockchain (capacidade de processamento de transações). Atualmente, o Bitcoin pode processar apenas cerca de três a sete transações por segundo, o que é um fator limitante importante na adaptação do Bitcoin para processar microtransações. O tamanho do bloco no Bitcoin é codificado rigidamente como 1 MB, mas se o tamanho do bloco for aumentado, ele poderá conter mais transações e resultar em tempos de confirmação mais rápidos. Existem várias Propostas de Melhoria do Bitcoin (BIPs) feitas em favor do aumento do tamanho de bloco. Essas incluem: BIP 100, BIP 101, BIP 102, BIP 103 e BIP 109.

No Ethereum, o tamanho do bloco não é limitado por codificação rígida; em vez disso, é controlado por um limite de gás (gas limit). Em teoria, não há limite no tamanho de um bloco no Ethereum porque depende da quantidade de gás, que pode aumentar ao longo do tempo. Isso é possível porque os mineradores podem aumentar o limite de gás para os blocos subsequentes se o limite tiver sido atingido no bloco anterior. O **Segregated Witness do Bitcoin (SegWit)** abordou essa questão ao separar os dados de assinatura dos dados da transação, o que resultou em mais espaço para transações. Outras propostas para o Bitcoin incluem o **Bitcoin Unlimited**, **Bitcoin XT** e **Bitcoin Cash**. Você pode consultar o Capítulo 6, Arquitetura do Bitcoin, para mais detalhes.

Um relato interessante com referências históricas e discussão está disponível em:  
<https://en.bitcoin.it/wiki/Block_size_limit_controversy>

Para mais informações sobre propostas do Bitcoin, consulte os seguintes endereços:

* <https://www.bitcoinunlimited.info>
* <https://www.bitcoincash.org>

**Redução do intervalo de blocos**  
Esta é outra proposta que trata da redução do tempo entre cada geração de bloco. O tempo entre os blocos pode ser diminuído para alcançar finalização mais rápida dos blocos, mas pode resultar em menor segurança devido ao aumento no número de forks. O Ethereum alcançou um tempo de bloco de aproximadamente 14 segundos.

Isso é uma melhora significativa em relação à blockchain do Bitcoin, que leva 10 minutos para gerar um novo bloco. No Ethereum, o problema de muitos blocos órfãos resultantes de tempos menores entre os blocos é mitigado pelo uso do protocolo **GHOST (Greedy Heaviest Observed Subtree)**, pelo qual blocos órfãos ou obsoletos (também chamados de “tios” na cadeia Ethereum) também são incluídos na determinação da cadeia válida. Uma vez que o Ethereum migre para Proof of Stake (PoS), isso se tornará irrelevante, pois não será necessário mineração e será possível obter quase imediata finalização das transações.

**Tabelas de Pesquisa Bloom Invertíveis (IBLTs)**  
Essa é outra abordagem que foi proposta para reduzir a quantidade de dados que precisa ser transferida entre os nós do Bitcoin. As **IBLTs** foram originalmente propostas por Gavin Andresen, e o principal atrativo dessa abordagem é que ela não resulta em um hard fork do Bitcoin se implementada. A ideia principal baseia-se no fato de que não há necessidade de transferir todas as transações entre os nós; em vez disso, apenas aquelas que ainda não estão disponíveis no pool de transações do nó em sincronização são transferidas. Isso permite uma sincronização mais rápida do pool de transações entre os nós, aumentando assim a escalabilidade e a velocidade geral da rede Bitcoin.

**Sharding**  
Sharding não é uma técnica nova e tem sido usada há muito tempo em bancos de dados distribuídos como MongoDB e MySQL para escalabilidade. A ideia principal por trás do sharding é dividir as tarefas em vários pedaços que são então processados por múltiplos nós. Isso resulta em maior rendimento e menor necessidade de armazenamento. Em blockchains, um esquema semelhante é empregado, pelo qual o estado da rede é particionado em vários shards. O estado geralmente inclui saldos, código, nonce e armazenamento. Shards são partições fracamente acopladas de uma blockchain que operam na mesma rede. Há alguns desafios relacionados à comunicação entre shards e ao consenso sobre o histórico de cada shard. Esta é uma área de pesquisa ativa e tem sido extensivamente estudada no contexto de escalonamento do Ethereum.

**Blockchains privadas**  
A maioria das blockchains privadas é inerentemente mais rápida porque não é necessária descentralização real e os participantes da rede não precisam minerar usando PoW; em vez disso, eles podem apenas validar transações. Isso pode ser considerado como uma solução alternativa para o problema de escalabilidade em blockchains públicas; no entanto, não é a solução para o problema de escalabilidade. Também deve-se observar que blockchains privadas são adequadas apenas em áreas e configurações específicas, como ambientes corporativos, onde todos os participantes são conhecidos.

**Propagação de blocos**  
Além da proposta anterior, também foi sugerido o **pipeline** da propagação de blocos, que é baseado na ideia de antecipar a disponibilidade de um bloco. Nesse esquema, a disponibilidade de um bloco já é anunciada sem aguardar sua real disponibilidade, reduzindo assim o tempo de ida e volta entre os nós.

Finalmente, o problema das longas distâncias entre o originador da transação e os nós também contribui para a lentidão da propagação de blocos. Foi demonstrado em uma pesquisa conduzida por Christian Decker et al. que aumentos na conectividade podem reduzir o atraso na propagação de blocos e transações. Isso é possível porque, se em qualquer momento um nó do Bitcoin estiver conectado a muitos outros nós, ele pode acelerar a propagação de informações na rede.

Uma solução elegante para os problemas de escalabilidade será muito provavelmente uma combinação de algumas ou todas as abordagens gerais mencionadas anteriormente. Algumas iniciativas realizadas para enfrentar os problemas de escalabilidade e segurança em blockchains já estão quase prontas para implementação ou já foram implementadas. Por exemplo, o **SegWit do Bitcoin** é uma proposta que pode ajudar imensamente com escalabilidade e requer apenas um soft fork para ser implementado. A ideia chave por trás do chamado SegWit é separar os dados de assinatura das transações, o que resolve o problema de maleabilidade das transações e permite o aumento do tamanho do bloco, resultando assim em maior taxa de transferência.

**Bitcoin-NG**  
Outra proposta, o **Bitcoin-NG**, baseada na ideia de microblocos e eleição de líder, tem ganhado alguma atenção recentemente. A ideia central é dividir os blocos em dois tipos, a saber: blocos de líder (também chamados de blocos-chave) e microblocos:

* **Blocos de líder**: são responsáveis pela Prova de Trabalho, enquanto os microblocos contêm as transações reais.
* **Microblocos**: não exigem PoW e são gerados pelo líder eleito a cada ciclo de geração de blocos. Esse ciclo é iniciado por um bloco de líder. A única exigência é assinar os microblocos com a chave privada do líder eleito. Os microblocos podem ser gerados em altíssima velocidade pelo líder eleito (minerador), resultando assim em desempenho e velocidade de transação aumentados.

Por outro lado, o **Ethereum 2.0 Mauve Paper**, escrito por Vitalik Buterin e apresentado na Devcon2 do Ethereum em Xangai, descreve uma visão diferente de uma blockchain escalável.

Essa proposta é baseada em uma combinação de sharding e uma implementação do algoritmo de consenso de Prova de Participação (PoS). O artigo definiu certos objetivos, como obter eficiência via PoS, minimizar o tempo de bloco e garantir finalização econômica, escalabilidade, comunicação entre shards e resistência à censura. Parte da visão apresentada no Mauve Paper foi implementada no Ethereum como o upgrade conhecido como “The Merge”. Discutimos o The Merge e o futuro do Ethereum no Capítulo 13, *The Merge and Beyond*.

**Cadeias baseadas em DAG**  
**DAG** significa *Directed Acyclic Graph* (Grafo Acíclico Dirigido). É visto como uma alternativa à tecnologia de blockchain baseada em cadeia linear. Uma blockchain é fundamentalmente uma lista encadeada, enquanto um DAG é um grafo acíclico onde os links entre os nós têm apenas uma direção. Em outras palavras, blockchains baseadas em DAG não se parecem com uma cadeia linear, mas sim com um grafo que se assemelha a uma árvore. Um DAG consiste em vértices e arestas.

*Dirigido* significa que o grafo se move apenas em uma direção e *acíclico* significa que não há possibilidade de retornar a um nó anterior a partir do nó atual.

A estrutura do DAG permite a criação e confirmação paralela de transações e blocos, alcançando assim alta taxa de transferência de transações. A ideia principal por trás das cadeias baseadas em DAG é que, como sabemos, em blockchains normais, há uma sequência linear de blocos um após o outro e, no caso de forks, apenas um fork sobrevive devido à regra de seleção de forks (por exemplo, a cadeia mais longa), enquanto os demais forks são ignorados (descartados). Isso significa que os blocos/transações podem ser gerados de forma limitada, um após o outro. Agora, se de alguma forma, formos capazes de manter esses forks sem comprometer a segurança, então isso significa que podemos produzir mais blocos e muito mais rapidamente, até mesmo quase em paralelo. Então, em vez de ter apenas um filho e um pai numa sequência linear de blocos, nos DAGs há blocos com múltiplos filhos e múltiplos pais, melhorando assim a velocidade de produção de blocos/transações.

Existem dois tipos de livros-razão distribuídos baseados em DAG:

* **DAGs sem blocos** (*Block-less DAGs*): aqui, os vértices são transações, e não existem blocos. Exemplos desse tipo de cadeia são **IOTA** (<https://www.iota.org>) e **Obyte** (<https://obyte.org>).
* **DAGs baseados em blocos** (*Block-based DAGs*): aqui, os vértices são blocos e os blocos podem se referir a diversos blocos predecessores. Exemplos desse tipo de cadeia são os protocolos **FANTOM**, **Ghost** e **Spectre**. Veja mais em: <https://eprint.iacr.org/2018/104.pdf>

**Mecanismos de consenso mais rápidos**  
Tradicionalmente, blockchains são baseadas em algoritmos de PoW ou PoS, que são inerentemente lentos em termos de desempenho. O PoW, especialmente, pode processar apenas algumas transações por segundo. Se um mecanismo de consenso diferente como **Raft** ou **PBFT** for usado, então a velocidade de processamento pode aumentar significativamente. Examinamos esses conceitos em maior detalhe no Capítulo 5, *Algoritmos de Consenso*.

Blockchains baseadas em algoritmo de PoS são fundamentalmente mais rápidas porque algoritmos PoS não exigem a conclusão de cálculos demorados e que consomem energia, como ocorre com o PoW.

Até agora, discutimos soluções de camada 0 (multichain) e camada 1 (on-chain) para o problema de escalabilidade.  
Na próxima seção, introduziremos uma abordagem mais popular que visa resolver o problema de escalabilidade usando componentes **off-chain** ou **de camada 2**.

**Camada 2 – soluções off-chain**

As soluções de camada 2 são baseadas na ideia de que, em vez de modificar a cadeia principal para alcançar escalabilidade, o objetivo deve ser transferir parte do processamento para mecanismos mais rápidos fora da cadeia subjacente principal, realizar o processamento ali e depois escrever o resultado de volta na cadeia principal como uma garantia de integridade. Diversas técnicas desse tipo são descritas aqui.

**Camada 2**  
Camada 2 é o nome dado às tecnologias que ajudam a escalar o Ethereum usando técnicas off-chain.  
Blockchains como Bitcoin e Ethereum atualmente podem realizar até 7 e 15 transações por segundo, respectivamente. Isso é muito mais lento do que métodos centralizados tradicionais de pagamento, como a Visa, que pode processar 100.000 transações por segundo.

A menos que a taxa de transferência de transações da blockchain possa ser significativamente melhorada, ela não pode ser usada para negócios cotidianos. Muitas pesquisas foram feitas nesse sentido. Existem duas maneiras de resolver esse problema: melhorando a camada base (ou seja, camada 1) ou transferindo parte do trabalho para outra camada (um sistema off-chain).

Observe que existem algumas cadeias complexas de camada 1 que estão tentando alcançar todas as três propriedades ao mesmo tempo com resultados aparentemente bons, por exemplo, a Solana. No entanto, existem alguns obstáculos no caminho, e só o tempo dirá se ela terá sucesso – até agora, parece promissora.  
Então, se não podemos escalar a camada 1, há alguma outra opção? Sim – transferimos o trabalho para outro sistema off-chain, chamado camada 2.

As soluções de camada 2 são ferramentas, mecanismos e protocolos off-chain que permitem que a camada 1 escale sem exigir nenhuma mudança na própria camada 1. Algumas mudanças podem ser feitas na camada 1 para torná-la mais amigável à camada 2, no entanto, nenhuma mudança é necessária. De certa forma, a camada 2 complementa a camada 1 e resulta em um sistema de blockchain mais escalável.  
O estado da camada 2 pode ser verificado pela camada 1 por meio de **provas de validade** ou **provas de fraude**. Esse mecanismo é o mais importante, pois garante que os validadores da L2 não possam trapacear e incluir transações inválidas em um bloco da L2, por exemplo, criar moedas do nada ou roubar suas moedas. O segundo uso da L1 é como uma **camada de disponibilidade de dados** (validação de estado) para transações da L2, de modo que, em caso de disputa, os usuários possam recriar de forma independente o estado da L2 e garantir a validade dos dados e a operação contínua do sistema, ou sair de forma segura (sem confiança) para a L1.  
As principais vantagens da camada 2 incluem taxas mais baixas, aproveitamento da segurança da L1 enquanto permite a execução mais rápida de transações e a possibilidade de mais casos de uso do que na L1. As soluções de camada 2 buscam alcançar garantias de segurança e descentralização semelhantes às da cadeia de camada 1 herdando essas garantias de segurança da camada 1.

**Canais de estado**  
Também chamados de canais de pagamento, os canais de estado são outra proposta para acelerar as transações em uma rede de blockchain. A ideia básica é usar canais laterais para atualização de estado e processamento de transações fora da cadeia principal; uma vez que o estado esteja finalizado, ele é escrito de volta na cadeia principal, aliviando assim a carga das operações demoradas da blockchain principal.

Os canais de estado funcionam executando as seguintes três etapas:

1. Primeiro, uma parte do estado da blockchain é bloqueada sob um contrato inteligente, garantindo o acordo e a lógica de negócios entre os participantes.
2. Agora, começa o processamento de transações off-chain e a interação entre os participantes que atualizam o estado (apenas entre si, por enquanto). Nesta etapa, praticamente qualquer número de transações pode ser realizado sem exigir a blockchain. É isso que torna o processo tão rápido e, argumentavelmente, o melhor candidato para resolver problemas de escalabilidade da blockchain. Pode-se argumentar que isso não é uma solução real on-blockchain como, por exemplo, o sharding, mas o resultado final é uma rede mais rápida, leve e robusta, que pode ser muito útil em redes de micropagamentos, redes de IoT e muitas outras aplicações.
3. Quando o estado final é alcançado, o canal de estado é encerrado e o estado final é escrito de volta na blockchain principal. Neste estágio, a parte bloqueada da blockchain também é desbloqueada.

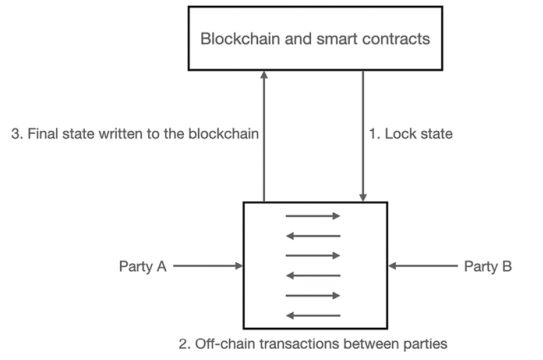


Figura 17.3: Canais de estado

Essa técnica foi usada na **Lightning Network do Bitcoin** e na **Raiden do Ethereum**.  
A principal diferença entre Lightning e Raiden é que Lightning funciona apenas para transações de Bitcoin, enquanto Raiden suporta todos os tokens compatíveis com ERC20, tornando a rede Raiden uma opção mais flexível.

**Sidechains**

As sidechains podem melhorar a escalabilidade indiretamente permitindo que muitas sidechains operem juntamente com a blockchain principal, enquanto permitem o uso de sidechains comparativamente menos seguras e mais rápidas para realizar transações, mas que ainda estão conectadas à blockchain principal. A ideia central das sidechains é chamada de **dois-vias peg** (*two-way peg*), que permite a transferência de moedas de uma cadeia principal para uma sidechain e vice-versa.

**Sub-chains**

Esta é uma técnica relativamente nova proposta recentemente por **Peter R. Rizun** e é baseada na ideia de blocos fracos, que são criados em camadas até que um bloco forte seja encontrado.

Blocos fracos podem ser definidos como aqueles blocos que não conseguiram ser minerados cumprindo o critério de dificuldade padrão da rede, mas realizaram trabalho suficiente para atender a outro alvo de dificuldade, mais fraco. Os mineradores podem construir sub-chains empilhando blocos fracos uns sobre os outros até que um bloco que satisfaça o alvo de dificuldade padrão seja encontrado.

O artigo de pesquisa sobre sub-chains de Rizun está disponível em:  
<https://ledger.pitt.edu/ojs/ledger/article/view/40/55>  
Rizun, P. R. (2016). Subchains: A Technique to Scale Bitcoin and Improve the User Experience. *Ledger*, 1, 38-52.

Neste ponto, a sub-chain é encerrada e torna-se o bloco forte. As vantagens dessa abordagem incluem o tempo de espera reduzido para a primeira verificação de uma transação. Essa técnica também reduz a chance de blocos órfãos e acelera o processamento das transações. Essa também é uma forma indireta de abordar o problema da escalabilidade. Sub-chains não exigem soft fork nem hard fork para serem implementadas, mas exigem aceitação da comunidade.

**Tree chains**

Também há outras propostas para aumentar a escalabilidade do Bitcoin, como as **tree chains**, que alteram a estrutura da blockchain de um modelo sequencial linear para uma árvore. Essa árvore é basicamente uma árvore binária que descende da cadeia principal do Bitcoin. Essa abordagem é semelhante à implementação de sidechains, eliminando a necessidade de mudanças significativas no protocolo ou aumento no tamanho do bloco. Ela permite maior taxa de transferência de transações. Nesse esquema, as próprias blockchains são fragmentadas e distribuídas pela rede para alcançar escalabilidade.

Além disso, a mineração não é necessária para validar os blocos nas tree chains; em vez disso, os usuários podem verificar independentemente o cabeçalho do bloco. No entanto, essa ideia ainda não está pronta para produção e mais pesquisas são necessárias para torná-la prática.

Além das técnicas gerais mencionadas anteriormente, algumas melhorias específicas do Bitcoin também foram propostas por **Christian Decker** em seu livro *On the Scalability and Security of Bitcoin*. Essa proposta é baseada na ideia de acelerar o tempo de propagação, já que o mecanismo atual de propagação da informação resulta em forks na blockchain. Essas técnicas incluem **minimização da verificação**, **pipeline de propagação de blocos** e **aumento de conectividade**. Essas mudanças não exigem alterações fundamentais no protocolo; em vez disso, podem ser implementadas de forma independente no software do nó do Bitcoin.

Em relação à **minimização da verificação**, observou-se que o processo de verificação de blocos contribui para o atraso de propagação. O motivo por trás disso é que um nó leva muito tempo para verificar a exclusividade do bloco e das transações dentro do bloco. Foi sugerido que um nó pode enviar a mensagem de inventário assim que as verificações iniciais de PoW e de validação do bloco forem concluídas. Dessa forma, a propagação pode ser melhorada realizando apenas a primeira verificação de dificuldade e não aguardando a finalização da verificação das transações.

**Plasma**

Outra proposta de escalabilidade é o **Plasma**, que foi proposta por **Joseph Poon** e **Vitalik Buterin**. Essa proposta descreve a ideia de executar contratos inteligentes na blockchain raiz (mainnet do Ethereum) e ter blockchains-filhas que realizam um grande número de transações para depois enviar pequenas quantidades de compromissos de volta para a cadeia-mãe. Nesse esquema, as blockchains são organizadas em uma hierarquia em árvore, com mineração sendo feita apenas na blockchain raiz (principal), que transmite as provas de segurança para as cadeias-filhas. Isso também é um sistema de camada 2, pois, como os canais de estado, o Plasma também opera off-chain.

A ideia original foi proposta no seguinte artigo de pesquisa:  
<https://eprint.iacr.org/2016/545.pdf>

**Plasma vs sidechains**

As **Plasma chains** são quase como sidechains, no entanto, elas sacrificam alguma funcionalidade para obter segurança adicional. As chains Plasma podem ser vistas como sidechains não-custodiais.

Uma sidechain é uma cadeia alternativa à cadeia principal (*parent chain*), enquanto o Plasma é uma estrutura para cadeias-filhas (*child chains*). As sidechains operam como uma blockchain separada em paralelo com uma blockchain de camada 1 como o Ethereum. Essas duas cadeias podem se comunicar entre si para que ativos possam ser movimentados entre elas. Sidechains também possuem um mecanismo de consenso. As chains Plasma possuem um mecanismo de consenso usado para produzir blocos, porém, diferentemente das sidechains, a **raiz Merkle** de cada bloco da chain Plasma é submetida ao Ethereum. As raízes Merkle dos blocos são usadas para provar a correção dos blocos.

**Escalabilidade assistida por hardware confiável**

Essa técnica é baseada na ideia de que cálculos complexos e pesados podem ser transferidos para recursos fora da cadeia (*off-chain*) de maneira verificavelmente segura. Uma vez que os cálculos são concluídos, os resultados verificados são enviados de volta à blockchain. Um exemplo de tal solução é o **Truebit**:  
<https://truebit.io/>

**Commit chains**

Commit chains são um termo mais genérico para a proposta Plasma de Vitalik Buterin. Elas também são chamadas de sidechains não-custodiais, mas sem um novo mecanismo de consenso, como é o caso das sidechains. Elas dependem do mecanismo de consenso da cadeia principal, por isso podem ser consideradas tão seguras quanto a cadeia principal.  
O operador da commit chain é responsável por facilitar a comunicação entre os participantes da transação e enviar atualizações regulares para a cadeia principal.

**Rollups**

Nas soluções de **rollup**, as transações são submetidas diretamente à camada 2 em vez da camada 1. As transações submetidas são agrupadas em lotes (*batches*) e eventualmente enviadas para a camada 1. As L2s são blockchains independentes com nós compatíveis com o Ethereum. Todos os estados e execuções são processados na camada 2, incluindo verificação de assinaturas e execução de contratos. A L1 apenas armazena os dados da transação, resultando assim no ganho de desempenho.

Em outras palavras, os rollups fornecem um ambiente de execução fora da camada 1, isto é, camada 2, o que resulta em execução mais rápida devido à ausência das limitações da L1. Quando a execução é concluída, a prova e o resumo dos dados são enviados para a camada 1, onde o consenso é alcançado. Portanto, a camada 1 é geralmente chamada de **camada de liquidação** (*settlement layer*).

O artigo de pesquisa sobre contratos Plasma está disponível em:  
<http://plasma.io>

Mais informações sobre commit chains estão disponíveis aqui:  
**Khalil, R., Zamyatin, A., Felley, G., Moreno-Sanchez, P. e Gervais, A.**, 2018. *Commit-Chains: Secure, Scalable Off-Chain Payments*. Cryptology ePrint Archive, Report 2018/642.  
<https://eprint.iacr.org/2018/642.pdf>

Essa prova é uma **prova de integridade computacional**, demonstrando que a transição de estado é válida após a execução da transação. Essas provas fornecem evidência da validade das transações no lote, de que a lógica interna da aplicação no rollup foi seguida corretamente e de que a transição de estado ocorreu de forma válida.

Os rollups são agora amplamente utilizados e constituem a principal solução de escalabilidade para blockchains.

Os rollups podem ser divididos em duas categorias com base em seu uso. Primeiro, temos os **rollups específicos de aplicação**, onde uma parte da aplicação com uso intensivo de recursos (cara) — ou seja, a execução — é empacotada em um rollup. Em segundo lugar, temos os **rollups gerais**, que ajudam a escalar redes baseadas em EVM por sua capacidade de gerar provas de validade para qualquer transição de estado na EVM.

A camada 2 pode melhorar a TPS (*transactions per second*) para dezenas de milhares de transações por segundo sem sacrificar a descentralização e a segurança porque sua segurança é herdada da cadeia de camada 1, por exemplo, o Ethereum.

**Validade dos dados**

Validade dos dados refere-se ao requisito de que o estado da camada 2 possa ser verificado pela camada 1 usando provas de validade ou provas de fraude. Esse mecanismo garante que os validadores da camada 2 não possam agir de forma maliciosa, como incluir transações inválidas nos blocos da L2, censurar transações, criar dinheiro do nada ou roubar fundos. De forma coloquial, a validade dos dados garante que "ninguém possa gastar fundos que não pertencem a eles".  
Uma blockchain de camada 2 pode publicar seu estado periodicamente na cadeia de camada 1 escrevendo o hash de sua mais recente **raiz de estado**. Essa raiz de estado é fornecida como uma prova criptográfica de validade usando conhecimento zero (*zero-knowledge*) e verificada na L1 por um contrato inteligente. Outra técnica é o uso de provas de fraude, que são baseadas no paradigma onde observadores honestos monitoram a cadeia da camada 2 e, em caso de submissão incorreta de raiz de estado à camada 1, eles podem acionar um alarme e fornecer a prova de fraude, que resultará no rollback automático da cadeia.

**Disponibilidade dos dados**

A disponibilidade dos dados refere-se ao requisito de que, em caso de disputas sobre uma transação, os usuários devem ser capazes de recriar de forma independente o estado da camada 2 e executar uma saída segura e sem confiança (*trustless exit*) para a camada 1.  
De maneira coloquial, a disponibilidade dos dados garante que "qualquer um possa gastar seus próprios fundos". A disponibilidade dos dados é útil quando os usuários precisam provar para uma cadeia de camada 1 que eles são donos dos fundos que estão tentando sacar ou gastar.

Para que isso aconteça, a camada 1 precisa ter acesso a todas as transações na camada 2 ou ao seu estado mais recente (atual). Uma técnica comum para alcançar isso é registrar as transações da L2 na L1 usando **calldata**.  
Outra técnica é armazenar os registros em uma camada de disponibilidade de dados externa separada, onde o provedor garante a disponibilidade dos dados por meio de mecanismos criptoeconômicos ou criptográficos. Note que publicar dados no Ethereum usando calldata é caro, no entanto, essa é a única técnica prática disponível para publicar dados facilmente na L1 do Ethereum. Qualquer outra técnica poderia ser muito mais cara.  
Outra técnica é armazenar os dados totalmente off-chain, onde outro provedor torna-se o custodiante dos dados.

Uma maneira simples de resolver o problema da disponibilidade dos dados é baixar os dados completos em cada entidade da rede, ou seja, cada nó mantém uma cópia completa dos dados — o que, é claro, não é sustentável devido ao volume de dados envolvido e à sobrecarga de replicação. A outra opção, mais comum e prática, são as **provas de disponibilidade de dados**, que permitem que clientes verifiquem se todos os dados de um bloco foram publicados, fazendo o download apenas de uma pequena parte do bloco.

**Computation e dados** são os dois principais gargalos da blockchain que são resolvidos pelos rollups.  
Rollups comprimem os dados para que apenas uma pequena quantidade de dados seja publicada na camada 1, reduzindo assim o volume de dados. Os rollups usam várias técnicas de compressão de dados, incluindo agregação de assinaturas (uma assinatura por lote em vez de uma por transação).

Execução e cálculo são abordados ao realizar as execuções off-chain e ao submeter provas de fraude ou provas de validade para garantir a validade dos dados.

De modo geral, podemos dizer que os **principais objetivos de segurança dos rollups** são:

* disponibilidade de dados
* integridade da transição de estado
* resistência à censura

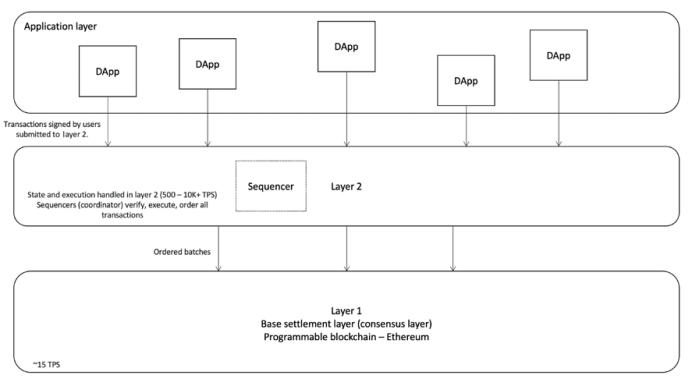
A disponibilidade de dados pode ser resumida como a pergunta: “todas as atualizações de estado feitas na base de dados da camada 2 estão publicamente disponíveis?”  
A integridade da transição de estado questiona se todas as atualizações de estado feitas são válidas.  
A resistência à censura pergunta se um usuário é capaz de garantir que, uma vez submetida uma transação, ela será eventualmente executada e não será censurada por um adversário.

**Como os rollups funcionam**

Os rollups operam utilizando um contrato inteligente (isto é, **contrato de rollup**) que existe on-chain. Esse contrato inteligente é responsável por verificar e manter a **raiz de estado** (*state root*). A raiz de estado, ou raiz de lote (*batch root*), é de fato a **raiz Merkle** de todo o estado do rollup, incluindo saldos de contas, código de contratos inteligentes, transações etc.

Uma coleção de transações em formato compactado, geralmente chamada de **lote** (*batch*), pode ser publicada por qualquer pessoa. A raiz de estado anterior e a nova são publicadas junto com o lote. O contrato de rollup on-chain verifica a raiz de estado anterior no lote e a compara com sua raiz de estado atual e, se ambas corresponderem, ele atualiza a raiz de estado para a nova.

O sistema de rollup lida com depósitos, transferências e saques. Podemos visualizar um sistema genérico de rollup na **Figura 17.4**:

  
**Figura 17.4: Um sistema de rollup**

A ideia central é tão simples quanto descrita, mas o problema-chave é como garantir que os novos estados sejam válidos e corretos. É aí que entram as **provas de fraude** e **provas de validade**.

Se alguém observar que um lote teve uma nova raiz de estado incorreta (*post-state root*), então uma prova de fraude é enviada como evidência à cadeia via contrato de rollup para refutar o lote. O contrato de rollup mantém o histórico completo de raízes de estado e o hash de cada lote de transações. O contrato de rollup verifica a prova de fraude e, se for válida, reverte o lote em questão e todos os lotes subsequentes.

**Provas de validade** são usadas pelos **ZK-rollups**, que são provas criptográficas incluídas com cada lote e que provam que a nova raiz de estado é o resultado correto da execução do lote de transações.

Um lote de transações pode ser submetido por vários tipos de usuários, dependendo da segurança e da arquitetura do sistema. A segurança central do sistema gira em torno da regra de que qualquer pessoa que deseje submeter lotes deve ter uma grande participação/depósito no sistema, o qual atua como uma garantia contra comportamentos fraudulentos ou maliciosos. Se o usuário enviar um lote fraudulento que for provado inválido por meio de uma prova de fraude, então o depósito é queimado conforme as regras do sistema.

Qualquer pessoa pode submeter um lote, mas tal abordagem aberta não é eficiente nem segura. Existem outras opções também, por exemplo:

* um **leilão de sequenciador** pode ser realizado em intervalos regulares para determinar o próximo sequenciador por um período fixo de tempo;
* o sequenciador também pode ser escolhido aleatoriamente de um conjunto de participantes com participação em um mecanismo PoS;
* um sequenciador com desempenho inadequado pode ser retirado por votação dos detentores de tokens da rede, e um novo leilão pode ser realizado.

**Tipos de rollups**

Existem dois tipos de rollups que variam dependendo do modelo de segurança no qual se baseiam:

* **Rollups otimistas** (*Optimistic Rollups*)
* **Rollups com prova de conhecimento zero** (*ZK-Rollups*)

**Rollups otimistas (Optimistic Rollups)**

Sob este modelo, assume-se que as transações são válidas por padrão, a menos que sejam contestadas por uma **prova de fraude**.  
A ideia fundamental é que o provedor de rollup publica os dados da transação na cadeia e aguarda alguns dias. Se alguém (um validador externo) apresentar uma reclamação e provar, por meio de uma prova de fraude, que a transação publicada é inválida, o provedor de rollup é penalizado (*slashed*, ou seja, perde sua garantia) e as transações são revertidas.

Os optimistic rollups utilizam uma **sidechain** que roda em paralelo com a cadeia principal do Ethereum.  
A ideia principal é mover a computação e o armazenamento para fora da camada 1. Após uma transação ser executada na L2, o rollup otimista propõe um novo estado à L1 ou autentica a transação. Os dados da transação da L2 são escritos na L1 como **calldata**.  
O sistema de rollup compacta muitas transações em um lote para reduzir a quantidade de dados postados na cadeia L1 principal e, portanto, o custo (em taxas).

O sistema utiliza **provas de fraude**, o que significa que, se um verificador (alguém na rede) detectar uma transação fraudulenta, a rede do optimistic rollup executará uma prova de fraude e rodará a computação da transação usando os dados de estado disponíveis (validade dos dados). Nesse caso, o gás consumido para executar a prova de fraude é reembolsado ao verificador.

Rollups otimistas são mais adequados para **computações EVM de uso geral**, por causa da sua compatibilidade direta com a EVM e baixo custo.

As transações são processadas off-chain, mas são postadas on-chain no contrato de rollup. Os fundos em um sistema de optimistic rollup são armazenados em um contrato inteligente na cadeia principal L1 do Ethereum. Esse contrato inteligente é responsável por lidar com depósitos e saques de fundos, registrar agregadores e submeter provas de fraude.  
Qualquer pessoa pode se registrar como agregador ao fazer um depósito de garantia no contrato inteligente.

**Como funcionam os optimistic rollups – passo a passo**

1. O **usuário** envia uma transação para a **camada 2** (rollup otimista).
2. A transação é **executada** e **agrupada** com outras transações em um **lote** pelo **sequenciador** (também chamado de agregador).
3. O sequenciador **comprime o lote** e publica os dados da transação na **camada 1**, onde são armazenados como *calldata* em um contrato inteligente.
4. O sequenciador propõe uma **nova raiz de estado** da L2 como sendo válida.
5. A comunidade (qualquer participante da rede) tem uma **janela de tempo para disputar** a transação — chamada de **período de contestação** (*challenge period*). Esse período pode durar dias.
6. Se nenhuma disputa for feita, a nova raiz de estado torna-se válida na L1 e os fundos podem ser sacados da L2 de volta para a L1.
7. Se houver uma disputa, então uma **prova de fraude** é submetida à L1. A L1 executa a transação novamente para verificar sua validade.
8. Se a transação for inválida, o sequenciador é penalizado (*slashed*) e os dados são revertidos.

**Vantagens dos optimistic rollups**

* Eles são **compatíveis com a EVM**, o que significa que os contratos inteligentes existentes podem ser portados facilmente para a camada 2.
* Os optimistic rollups são **eficientes em termos de custo**, já que não exigem cálculos criptográficos pesados como os ZK-rollups.
* Não exigem **mudanças significativas na infraestrutura** existente.

**Desvantagens dos optimistic rollups**

* O **tempo de retirada é longo**, pois é necessário esperar o período de contestação.
* A **segurança depende da presença de um observador honesto** capaz de detectar fraudes.
* Pode ser necessário **executar novamente a transação inteira** para verificar uma prova de fraude.

**ZK-rollups**

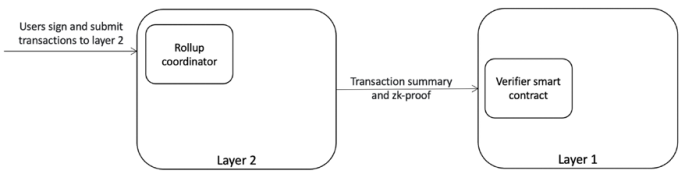
A ideia principal por trás dos ZK-rollups é que um nó na camada 2 processa e comprime transações e produz uma prova de correção que é submetida à blockchain. A verificação dessa prova é muito mais rápida do que executar e verificar todas as transações. Assim, o insight aqui é que não importa se a cadeia principal é lenta e só pode lidar com sete transações por segundo. Se cada transação contém milhares de transações dentro dela, então processar uma única transação na cadeia principal na verdade significa processar milhares de transações. É assim que a escalabilidade é alcançada.

A principal diferença entre os rollups otimistas e os ZK-rollups é que os ZK-rollups usam provas de validade de conhecimento zero, enquanto os rollups otimistas usam provas de fraude. A ideia principal é semelhante à dos rollups otimistas, no sentido de que o provedor de rollup publica dados de transação na cadeia da camada 1, mas com uma prova de conhecimento zero que prova a execução correta das transações. Essa prova de conhecimento zero garante a validade dos dados. Os ZK-rollups comprimem uma grande quantidade de computação e verificação em uma única operação. Os ZK-rollups usam essa técnica para agrupar muitas transações que são executadas na L2, em uma única transação na mainnet da L1 via um contrato inteligente localizado na L1. A partir dos dados submetidos, o contrato inteligente verifica todas as transferências que estão incluídas no rollup. A vantagem principal aqui é que verificar as transações não requer todos os dados relevantes, apenas a prova de conhecimento zero. As transações são escritas no Ethereum como *calldata* para reduzir as taxas de gás.

Os ZK-rollups consistem em dois tipos de atores. Os transatores, que criam e transmitem as transações, e os retransmissores (*relayers*, validadores), que coletam as transações e criam os rollups. Os retransmissores criam as provas ZK-ZNARK ou ZK-STARK, que são basicamente um hash que representa a diferença entre o estado antigo e o novo. Qualquer pessoa pode se tornar um retransmissor, desde que atenda aos requisitos específicos de participação do contrato de rollup.

Os ZK-rollups são geralmente melhores para pagamentos simples (transferências de tokens) e trocas simples porque eles não executam computações de uso geral de maneira eficiente. Isso porque eles não são prontamente compatíveis com a EVM padrão e, portanto, não conseguem executar computações de uso geral (Turing-completas) de forma eficiente. No entanto, grandes esforços estão sendo feitos para alcançar isso, como o desenvolvimento da ZK-EVM. Vários projetos trabalhando na construção de ZK-EVMs são zkSync, Applied ZKP, Scroll, Polygon Hermez e Polygon Miden.

A ideia principal por trás dos ZK-rollups é representada na Figura 17.5. Em vez de os usuários enviarem suas transações para a blockchain principal da camada 1, eles as enviam para uma entidade chamada provedor de rollup ou coordenador de rollup:



*Figure 17.5: The idea behind the ZK-rollup-based scalability solution*

O coordenador de rollup verifica, executa e comprime muitas (geralmente milhares de) transações em uma única prova que é publicada on-chain e verificada pelos verificadores (mineradores). Essa prova é muito mais rápida de verificar em comparação com a execução em si na cadeia. A prova é computada usando técnicas como SNARKs e STARKs.

**Escalabilidade**

Semelhante aos rollups otimistas, o sistema ZK-rollup é composto por contratos inteligentes de rollup on-chain da L1 e máquinas virtuais off-chain. As transações são finalizadas e liquidadas na camada 1 do Ethereum e só são liquidadas se o contrato de rollup da L1 aceitar a prova de validade submetida pelo coordenador (prover) da camada 2. Como essa prova prova a integridade da transação e todas as transações devem ser aprovadas pela camada 1, isso remove o risco de operadores maliciosos tentarem submeter transações fraudulentas.

**Tecnologias utilizadas para construir ZK-rollups**

Existem duas tecnologias que são usadas para construir ZK-rollups, incluindo ZK-SNARKs e ZK-STARKs.

ZK-STARKs são transparentes, ou seja, não requerem configuração confiável (*trusted setup*). Eles são escaláveis, pois o tempo de prova é quase linear e o tempo de verificação é logarítmico. A configuração é sucinta, exigindo no máximo um tamanho logarítmico. O tamanho da prova é um pouco grande. Também são seguros contra ataques quânticos. Exemplos-chave de camadas 2 baseadas em STARK são StarkNet e Polygon Miden. Os STARKs geram provas mais rapidamente e são seguros contra ataques quânticos. Como a prova é maior em tamanho, eles exigem mais espaço de *calldata* – aproximadamente 50KB. Assim, também exigem mais gás para verificação e são mais caros para os usuários. Também observe que os STARKs são baseados em segurança de 80 bits, significando uma chance de 1/1000 de falsificação de prova, o que pode ser um grande problema, dado que esses sistemas devem ser usados para dApps financeiros lidando potencialmente com milhões e bilhões de dólares. Portanto, isso precisa ser cuidadosamente considerado antes de escolher uma solução de rollup baseada em STARK.

ZK-SNARKs são não-interativos, o que significa que há apenas uma mensagem postada pelo verificador. Eles são sucintos, com tempos de verificação logarítmicos. Requerem uma configuração confiável, que pode levar tempo linear ou mais para ser feita. O tamanho da prova é menor. Não são seguros contra ataques quânticos. Alguns exemplos importantes de sistemas baseados em ZK-SNARK são Aztec, Loopring, zkSync, Hermez e Polygon Zero.

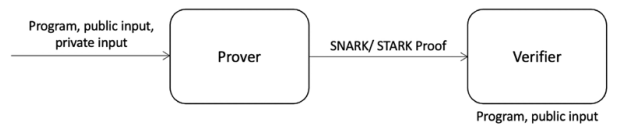
A tabela a seguir mostra uma comparação entre SNARKs e STARKs:

|  | **Tempo de Prova** | **Tempo de Provedor** | **Tempo de Verificação** |
| --- | --- | --- | --- |
| SNARKs | Pequeno | Medio | Pequeno |
| STARKs | Grande | Pequeno | Medio |

Vamos agora ver como SNARKs e STARKs são usados para criar provas.

Pense em um programa que sempre termina — vamos chamá-lo de **c**. Esse programa **c** recebe uma entrada pública chamada **x**, e uma entrada privada chamada **w**. O sistema SNARK/STARK é composto por um provador, um verificador, o programa **c** e uma prova **π**. O provador conhece o programa **c**, a entrada pública **x** e a entrada privada **w**. O verificador conhece apenas o programa **c** e a entrada pública **x**. O provador produz uma prova curta que prova que ele conhece uma testemunha (entrada privada) **w** que faz com que o programa produza um determinado valor esperado. O verificador executa um processo de verificação na prova, que o convence de que o provador de fato conhece uma testemunha (entrada privada) **w** que, quando inserida no programa **c** com a entrada pública **x** e executada, produz o valor esperado. O ponto-chave aqui, do ponto de vista da escalabilidade, é que o tempo de execução do verificador para verificar a prova é muito menor do que a execução do programa em si.

Em outras palavras, o tempo do provador é linear ao programa, enquanto o tempo do verificador é sublinear:

****

*Figure 17.6: SNARK/STARK fundamental idea*

O que faz um provedor de rollup? Fundamentalmente, os provedores de rollup mantêm árvores de Merkle, que são compostas por contas e, consequentemente, representam um estado com uma raiz de Merkle. Essa raiz de Merkle é publicada na blockchain principal da camada 1, verificada pelos verificadores. Quando o provedor de rollup executa todas as transações submetidas a ele, ele cria uma nova raiz de Merkle como um novo estado. Neste ponto, a raiz de Merkle antiga não é mais efetiva. O provedor de rollup cria uma prova SNARK e a submete à blockchain da camada 1 junto com o resumo da transação. Na camada 1, os verificadores (mineradores) verificam a prova.

A nova raiz de Merkle é o resultado da aplicação de uma sequência de transações válidas à raiz de Merkle antiga. Aqui podemos escrever o programa **c** (lembre-se do programa **c**!) que verifica isso. Aqui, a entrada pública **x** é a raiz de Merkle antiga, a nova raiz de Merkle e os resumos das transações. A entrada privada do programa **c** é o saldo das contas (estado) de todas as contas que mudaram devido à execução da transação, os saldos antigos das contas e as assinaturas digitais que assinaram essas transações. O programa **c** executará todas as verificações, incluindo validações de assinatura, verificações de saldo de conta e verificações de transação, e garantirá que as execuções estejam corretas. Basicamente, isso verifica se a nova raiz é válida e consistente com a raiz antiga após a execução da transação ter sido concluída. Em seguida, o provedor de rollup publica a nova raiz de Merkle atualizada na blockchain da camada 1 junto com a prova SNARK de que a nova raiz é consistente com a raiz antiga de acordo com a transação que o provedor de rollup recebeu e executou. Os dados postados na blockchain incluem a raiz de Merkle, a prova SNARK e um resumo dos dados da transação, excluindo assinaturas de transações. Os mineradores agora simplesmente registram a raiz de Merkle e verificam se a prova ZK-SNARK é válida. Agora os verificadores basicamente verificaram a nova raiz de estado, que foi criada como resultado da execução de muitas (geralmente milhares de) transações na camada 2, ou seja, pelo provedor de rollup. É assim que o escalonamento massivo na taxa de transferência de transações é alcançado.

A arquitetura dos ZK-rollups é amplamente baseada em dois componentes principais: contratos on-chain e máquinas virtuais off-chain. O protocolo ZK-rollup é gerenciado por contratos inteligentes executando na cadeia principal da camada 1 do Ethereum. Os contratos ZK-rollup são responsáveis por armazenamento de blocos de rollup, rastreamento de depósitos e saques, monitoramento de atualizações de estado e verificação das provas de conhecimento zero submetidas pelos produtores de blocos da camada 2. A máquina virtual off-chain na camada 2 é responsável pela execução de transações e armazenamento de estado. Aqui, as provas de validade verificadas na cadeia principal do Ethereum fornecem uma garantia criptográfica da correção das transições de estado na máquina virtual off-chain da camada 2.

Os ZK-rollups dependem da cadeia principal do Ethereum para disponibilidade de dados, finalidade da transação e resistência à censura. Observe que apenas a execução foi transferida para a camada 2 – a descentralização e a segurança ainda são baseadas na cadeia principal do Ethereum.

**Escalabilidade**

O que acontece se o provedor de rollup falhar? Como os provedores de rollup são entidades *trustless* (sem confiança), os usuários podem encontrar novos provedores de rollup. No entanto, o problema é que esses novos provedores de rollup não sabem nada sobre as informações de estado (saldos de contas, etc.) mantidas pelo provedor de rollup anterior. Então, como o novo provedor de rollup pode obter o estado do provedor antigo? Mais geralmente, o domínio do problema diz respeito a como transferimos ativos/estado/dados de conta de e para a camada 2. Se o provedor de rollup antigo (também chamado de coordenador) não estiver mais disponível, como tornamos os dados do provedor de rollup antigo disponíveis para o novo provedor de rollup? Para esse propósito, existem duas soluções: disponibilidade de dados on-chain e disponibilidade de dados off-chain. Um exemplo de solução de disponibilidade de dados on-chain é o zkSync e uma solução de dados off-chain é o zkPorter. Apresentamos essas soluções a seguir.

No zkSync, todos os dados de resumo da transação são armazenados na blockchain da camada 1. Quando o novo provedor de rollup é iniciado, ele pode ler todas as informações de transação da camada 1 que o provedor de rollup antigo processou antes de falhar e pode reconstruir todo o estado (estado mundial) a partir desses dados. Uma vez que o novo provedor de rollup tem o estado mais recente, ele pode começar a funcionar como o novo provedor de rollup, assim como o antigo. Esta é uma maneira segura de tornar os dados disponíveis novamente — uma solução segura (devido às garantias de segurança da L1) para o problema de disponibilidade de dados, porém a desvantagem aqui é que os provedores de rollup precisam escrever grandes quantidades de dados de transações na cadeia da camada 1, resultando em taxas de transação mais altas. Como é um método seguro, mas caro, é recomendado para ativos digitais criptográficos de alto valor.

No zkPorter, os dados de resumo da transação não são publicados na camada 1, mas sim em uma nova blockchain separada (off-chain). Esse mecanismo é muito mais barato que o zkSync. Essa blockchain separada existe apenas para armazenar dados de transações e é gerenciada por coordenadores sob um mecanismo de prova de participação (*proof-of-stake*) ou algum outro mecanismo de incentivo criptoeconômico. Esse mecanismo é menos seguro que o zkSync porque não possui as garantias de segurança da camada 1, mas é barato porque não requer a publicação e o armazenamento de dados na cadeia da camada 1. Isso significa que, quando precisamos dos dados de resumo da transação, eles podem não estar disponíveis. Embora as chances disso acontecer sejam mínimas, a garantia de disponibilidade de dados ainda não é tão forte quanto a da camada 1. Quando há necessidade de reconstruir os dados de estado/estado mundial/estado de contas novamente, esses coordenadores na rede fornecem os resumos das transações, que então são usados para reconstruir o estado do zero, por exemplo, em um novo coordenador, resolvendo assim efetivamente o problema de disponibilidade de dados.

Há uma escolha para o usuário aqui: ele pode usar disponibilidade de dados on-chain ou off-chain, dependendo dos requisitos de segurança e de uso.

Em resumo, os ZK-rollups são ferramentas de escolha para permitir alta taxa de transferência de transações ao executar as computações off-chain e usar provas de conhecimento zero para verificar essas execuções on-chain. Como a verificação de transações é mais rápida que a execução, tais técnicas proporcionam um aumento considerável na taxa de transferência de transações, medida em transações por segundo (TPS). O futuro do Ethereum é centrado em rollups, e várias atualizações estão em andamento para remover gargalos de desempenho e altos custos, tornando o Ethereum mais amigável a rollups. Cobrimos esse roteiro centrado em rollups no Capítulo 13, *The Merge and Beyond*.

**Prós dos ZK-rollups:**  
Sem atraso, eficientes, finalização mais rápida, criptograficamente seguros e menos vulneráveis a ataques econômicos. Devido às provas criptográficas de validade, a correção das transações off-chain pode ser garantida, o que, por sua vez, garante segurança, resistência à censura e descentralização porque os dados necessários para recuperar o estado off-chain são armazenados na camada 1. Tempo de saque mais rápido.

**Contras dos ZK-rollups:**  
ZK-rollups são limitados apenas a transferências simples, pois não são compatíveis com EVM. Isso se deve ao fato de que as provas de validade são computacionalmente pesadas e, para processá-las de forma eficiente, os ZK-rollups precisam construir sua própria linguagem de programação. A introdução de uma nova linguagem personalizada significa que os ZK-rollups não são compatíveis com Solidity. No entanto, para aliviar esse problema, já está em andamento algum trabalho na construção de compiladores (transpiladores) de Solidity para linguagens ZKP. Por exemplo, Warp, que é um transpilador de Solidity para Cairo (linguagem ZKP do StarkNet). ZK-rollups não são eficientes o suficiente para executar contratos inteligentes. É melhor usar ZK-rollups para aplicações com uma taxa de transação simples e de alto volume, como exchanges. Eles são adequados para aplicações com pouca atividade on-chain. A computação e verificação da prova de validade é custosa, o que pode resultar em taxas mais altas. A compatibilidade com EVM não é fácil de alcançar. Produzir provas de validade pode exigir hardware especial de alto desempenho, o que poderia resultar em centralização devido aos requisitos de recursos, onde apenas algumas partes seriam capazes de arcar com esse hardware caro. Operadores centralizados poderiam, possivelmente, censurar ou reordenar transações para sua vantagem. SNARKs exigem uma configuração confiável, o que pode levantar preocupações quanto à segurança.

A principal diferença entre rollups e sidechains é que as sidechains são baseadas em seus próprios pressupostos de segurança e executam seus próprios validadores, enquanto os rollups herdam sua segurança e confiança da camada 1. Sidechains podem ser vistas como outra cadeia de camada 1, sofrendo assim com o trilema da blockchain, trocando descentralização por escalabilidade. Além disso, sidechains dependem de pontes confiáveis (geralmente *multisig*) para se conectar à cadeia principal (Ethereum). Às vezes, essas pontes também são um tanto centralizadas, onde a maioria das assinaturas *multisig* é controlada por membros do mesmo grupo, por exemplo, membros fundadores. Rollups usam pontes sem confiança com melhores pressupostos de confiança.

ZK-rollups são bons para transações básicas, mas não são de uso geral. Eles são bons o suficiente para realizar transações específicas básicas dentro do sistema de rollup e postar provas para verificação na cadeia principal, mas e quanto a executar programas de uso geral como contratos inteligentes Solidity dentro do rollup? É possível, usando EVMs de Conhecimento Zero (*ZK-EVMs*). Isso significa que o provedor de rollup pode produzir prova da execução correta de um programa Solidity. Isso é feito da mesma forma que a prova padrão de atualizações de saldo do Ethereum é criada, significando que a prova de transições de estado que ocorreram devido à execução de um programa Solidity reside em uma folha da árvore de Merkle.

Provar depósitos, transferências e saques off-chain é muito útil, mas não é de uso geral. Ainda não podemos provar uma execução arbitrária como a de um contrato inteligente. Se pudermos, de alguma forma, provar a correção da execução da EVM off-chain, então você pode imaginar quantos casos de uso diferentes isso permitiria. Isso é exatamente o que os ZK-EVMs tentam fazer, o que exploraremos a seguir.

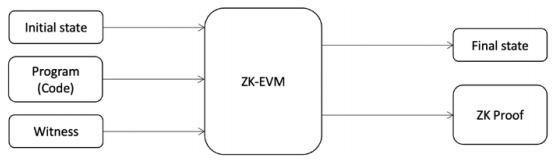
Tecnicamente, os ZK-rollups poderiam (e deveriam) ser chamados de *validity rollups* porque eles não usam ZK para privacidade, apenas para escalabilidade, e o acrônimo ZK pode passar uma impressão incorreta. Rollups que fornecem privacidade deveriam ser chamados de ZK-rollups ou *private rollups*. Lembre-se de que os ZK-rollups usam ZK para integridade dos dados, em vez de confidencialidade dos dados.

**ZK-EVM**

Uma ZK-EVM é um tipo específico de ZK-rollup. Em vez de gerar prova de alguma lógica específica de aplicação, como é o caso com ZK-rollups normais, ZK-EVMs produzem prova da correção da execução de bytecode da EVM. Isso significa que programas padrão da EVM da camada 1 do Ethereum podem ser executados em uma ZK-EVM e a ZK-EVM produzirá a prova de execução correta do código da EVM. Uma prova ZK de transição de estado válida na camada 2 é publicada na camada 1.

Podemos definir uma ZK-EVM como uma máquina virtual de camada 2 que executa contratos inteligentes como computações de prova de conhecimento zero. Ela visa replicar o sistema Ethereum como um ZK-rollup para escalar o Ethereum sem exigir nenhuma mudança no código ou perda de compatibilidade com as ferramentas de desenvolvimento existentes do Ethereum.

A arquitetura geral de alto nível de uma ZK-EVM é mostrada na Figura 17.7:

****

*Figure 17.7: ZK-EVM high-level architecture*

Lembre-se, a ideia principal aqui é provar a execução do bytecode da EVM. Os seguintes são os passos envolvidos:

1. Primeiro, a camada 2 executa as transações e, como resultado, ocorrem mudanças de estado.
2. A ZK-EVM extrai o estado anterior, as execuções e o novo estado e calcula a diferença entre o estado antigo e o novo.
3. A ZK-EVM executa e produz um *trace* (rastreamento) da execução. O rastreamento da execução geralmente contém as etapas da execução, cabeçalho de bloco, dados da transação, bytecode do contrato inteligente e provas de Merkle.
4. O provador pega o *trace* e produz uma prova. É aqui que circuitos aritméticos são construídos, como o circuito da EVM, o circuito de RAM e o circuito de armazenamento. Cada circuito produz uma prova e é então agregado em um circuito menor. Eventualmente, uma prova final é produzida.
5. Essa prova é publicada na cadeia principal.

Esse processo permite que todas as transações da camada 2 (possivelmente milhares) sejam representadas como uma única transação na cadeia principal.

Projetar uma ZK-EVM é difícil. Isso exige não apenas a refatoração da EVM, mas também a refatoração do mecanismo de transição de estado do Ethereum para torná-lo compatível com provas de conhecimento zero.

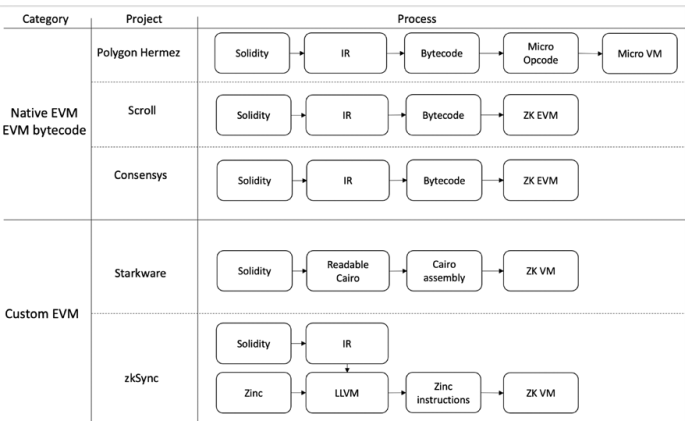
A ideia principal aqui é que ZK-EVMs escalem a taxa de transferência de transações ao multiplexar (consolidar) blocos e transações da camada 2 na camada 1 como uma transação.

Existem duas maneiras de fazer isso. A primeira é uma ZK-EVM nativa ou compatível com bytecode, que cria circuitos para todos os opcodes da EVM. A outra opção é construir uma nova Máquina Virtual (VM) personalizada e uma linguagem de programação para essa nova VM. Não é fácil construir nem a ZK-EVM nativa nem a máquina virtual personalizada, no entanto, a ZK-EVM nativa é mais difícil de construir. A ZK-EVM nativa é mais amigável ao desenvolvedor devido à compatibilidade com ferramentas do Ethereum que já existem e com as quais os desenvolvedores estão familiarizados, enquanto VMs personalizadas exigem o aprendizado de novas linguagens e ferramentas de programação, o que não é tão amigável ao desenvolvedor. Como as ZK-EVMs nativas herdam o modelo de segurança da EVM padrão bem como do Solidity, uma linguagem conhecida e relativamente madura, elas são mais seguras, enquanto as EVMs personalizadas podem não oferecer o mesmo nível de segurança.

Scroll, ConsenSys e Applied ZKP são exemplos de EVMs nativas (compatíveis com bytecode), enquanto StarkNet e zkSync são dois exemplos principais de VMs personalizadas. Em todas essas opções, o requisito chave é que as provas sejam geradas rapidamente e com custo mínimo. O principal benefício das ZK-EVMs é que podemos reutilizar o bytecode já implantado na camada 1 sem mudar nenhum código, o que torna os dApps já implantados imediatamente mais escaláveis simplesmente ao implantá-los novamente na camada 2. Além disso, podemos usar ferramentas e linguagens existentes como MetaMask, Truffle e Solidity para escrever contratos inteligentes na camada 2, da mesma forma que faríamos na camada 1.

ZK-EVMs permitem que os desenvolvedores escrevam contratos inteligentes usando as mesmas ferramentas e linguagens utilizadas na camada 1, por exemplo, Truffle e Solidity, e também permitem a reutilização do bytecode já implantado. Ambas as funcionalidades são permitidas enquanto ainda garantem integridade de estado.

O diagrama na Figura 17.8 mostra várias abordagens que EVMs nativas e projetos de EVMs personalizadas estão adotando para construir ZK-EVMs:

****

Na próxima parte, o livro apresenta as três grandes categorias de ZK-EVMs com base nas técnicas de interpretação e produção de provas, e o dilema entre desempenho e compatibilidade. Deseja que eu continue?

Dando continuidade à tradução literal do Capítulo 17 do livro *Mastering Blockchain - Fourth Edition*:

**Escalabilidade**

As ZK-EVMs podem ser divididas em três grandes categorias com base na técnica que usam para interpretar o código e produzir provas:

* **Baseadas em transpilação** – Nesse tipo, uma linguagem da EVM, como Solidity, é transcompilada (*transpiled*) para uma máquina virtual compatível com SNARK. StarkWare e Matter Labs usam essa abordagem.
* **Baseadas em interpretação direta** – Nessa abordagem, o bytecode da EVM é interpretado diretamente para produzir uma prova de execução correta do bytecode. Essa técnica é usada pelas ZK-EVMs da Hermez, Scroll e ConsenSys.
* **Equivalência total à EVM** – Essa abordagem busca equivalência total com a EVM no nível de consenso da camada 1 do Ethereum. Ela prova a validade das raízes de estado da Ethereum da camada 1. Ainda não há exemplo concreto dessa variante.

Também há vários tipos de ZK-EVMs com base em diferentes níveis de desempenho e compatibilidade com a EVM. Esses tipos foram propostos por Vitalik Buterin em:  
<https://vitalik.ca/general/2022/08/04/zkevm.html>

Existe um dilema aqui em termos de quão compatível uma ZK-EVM pode ser com a EVM. Podemos chamá-lo de *dilema do tradeoff das ZK-EVMs*. O dilema é que, quanto mais compatível uma ZK-EVM é com a EVM padrão, menos eficiente ela se torna. Em outras palavras, melhor desempenho (velocidade) significa menor compatibilidade com a EVM padrão. Esse dilema se tornará mais claro ao examinarmos os tipos de ZK-EVM descritos a seguir:

* **Tipo 1**: Esse tipo permite verificação de blocos do Ethereum nativamente. Esse tipo é totalmente equivalente (compatível) com o Ethereum sem que nenhuma mudança seja feita em qualquer componente do sistema Ethereum, incluindo árvores de estado, quaisquer estruturas de dados relevantes, lógica de computação, lógica de consenso, pré-compilações ou qualquer outro mecanismo. Equivalência com a EVM significa que, se o mesmo conjunto de transações for inserido tanto na EVM quanto na ZK-EVM, os estados resultantes produzidos por ambas seriam exatamente os mesmos, ou seja, a mesma raiz de estado, armazenamento, estado da conta, saldos e quaisquer outras estruturas de dados. Como tal, esse tipo é compatível com todos os aplicativos nativos do Ethereum. No entanto, não é eficiente, pois a geração de provas leva muito tempo nesse tipo.
* **Tipo 2**: Esse tipo é compatível com a EVM, mas não com o Ethereum, o que significa que é equivalente à EVM, mas difere das estruturas de dados centrais do Ethereum, como estrutura de blocos, tries de estado, consenso e outros mecanismos relevantes. É totalmente compatível com aplicativos existentes, mas abdica parcialmente da compatibilidade com o Ethereum ao modificar as estruturas de dados do Ethereum para tornar a geração de provas mais rápida e o desenvolvimento mais fácil. Por exemplo, o Ethereum usa a Árvore de Merkle Patricia (MPT), enquanto uma ZK-EVM poderia usar árvores de Merkle esparsas, que são estruturas de dados mais eficientes. Outro exemplo é que o Ethereum usa a função hash Keccak, enquanto a ZK-EVM poderia usar a função Poseidon, que é mais eficiente. Outra melhoria poderia ser usar apenas uma única *trie* no sistema da ZK-EVM, em vez dos múltiplos tipos de *tries* usados no Ethereum. É totalmente compatível com quase todos os aplicativos Ethereum e pode compartilhar a maior parte da infraestrutura, no entanto, a geração de provas ainda é lenta.
* **Tipo 2.5**: Esse tipo modifica a EVM apenas alterando os custos de gás. Ele possui tempos rápidos de geração de provas, mas introduz algumas incompatibilidades.

<https://vitalik.ca/general/2022/08/04/zkevm.html>  
<https://vitalik.ca/general/2022/08/04/zkevm.html>

* **Tipo 3**: Esse tipo é o mais equivalente à EVM, mas alguns tradeoffs são feitos para melhorar os tempos de geração de prova e facilitar o desenvolvimento. Esses tradeoffs incluem alguma incompatibilidade com aplicativos – por exemplo, se um dApp usa *precompiles*, ele teria que ser reescrito porque o tipo 3 não os suporta. Scroll e Polygon são dois exemplos principais de ZK-EVMs do tipo 3. A geração de prova é rápida nesse tipo e ele é totalmente compatível com a maioria dos dApps do Ethereum.
* **Tipo 4**: Esse tipo de ZK-EVM compila contratos de linguagens de alto nível escritas em Solidity ou outras linguagens em uma VM especializada. Não é compatível com alguns dApps do Ethereum e não pode compartilhar a maior parte de sua infraestrutura. No entanto, esse tipo tem os tempos de geração de prova mais rápidos, reduzindo assim os custos e o risco de centralização.

Alguns desafios-chave que as implementações de ZK-EVM devem resolver são complexidade e desempenho. As implementações de ZK-EVM são complexas porque criar e manter circuitos (aritmetização) das operações da EVM, ou seja, opcodes, pode ser desafiador. Além disso, as provas devem ser geradas o mais rápido possível e com custo mínimo, o que torna difícil alcançar um alto nível de desempenho. No entanto, é viável, e muitos exemplos razoáveis já existem. O tempo de geração de provas é geralmente linear ao tamanho do programa, enquanto o tempo do verificador é muito menor do que a execução do programa em si.

Com ZK-EVMs, é fácil migrar dApps da camada 1 para rollups zk da camada 2. Isso também resulta em redução das taxas de transação e aumento da taxa de execução dos programas, pois agora os programas são executados na camada 2.

Existem diferentes projetos de ZK-EVM, incluindo:

* Scroll ZK-EVM: <https://scroll.io/>
* ConsenSys ZK-EVM: <https://consensys.net/blog/news/consensys-launches-private-beta-zkevm-testnet-to-scale-ethereum/>
* zkSync: <https://zksync.io>
* Polygon ZK-EVM

**ZK-ZK-rollups**  
Esse é outro tipo de rollup, fundamentalmente ainda um ZK-rollup, mas fornecendo privacidade juntamente com escalabilidade. Um exemplo principal desse tipo é o Aztec, que introduziu o termo *ZK-ZK-rollup*, significando rollups totalmente privados.

**Rollups Otimistas vs ZK-rollups**

A tabela a seguir resume as diferenças entre rollups otimistas e ZK-rollups:

| **Propriedade** | **Rollups Otimistas** | **ZK-rollups** | **ZK-ZK-rollups** |
| --- | --- | --- | --- |
| Confidencialidade | Não | Não inerentemente; ZK apenas usado para escalabilidade | Sim |
| Anonimato | Não | Não inerentemente; ZK apenas usado para escalabilidade | Sim |
| Complexidade | Baixa | Alta | Alta |
| Execução de uso geral – Compatibilidade com EVM | Sim, mais fácil de fazer | Difícil de fazer | Difícil de fazer |
| Tempo de saque | Aprox. 1 semana, pois os participantes da rede devem ter tempo para verificar transações/lotes | Rápido, assim que o próximo lote é submetido pelo provedor de rollup, geralmente menos de 10 min | Rápido, assim que o próximo lote é submetido pelo provedor de rollup, geralmente menos de 10 min |
| Desempenho / eficiência | Alto | Inferior, comparativamente | Inferior, comparativamente |
| Eficiência de verificação | A prova de fraude é rápida de verificar uma vez submetida, no entanto, o sistema depende de verificadores para monitorar a cadeia e submeter provas de fraude, o que pode ser um processo lento | Rápida de verificar | Rápida de verificar |
| Segurança | Incentivos criptoeconômicos | Prova criptográfica | Prova criptográfica |
| Finalidade | Dias (confirmação única) | Minutos | Minutos |
| Usabilidade | Fácil de migrar aplicativos da L1 para a L2 | Difícil de construir aplicativos devido às complexidades matemáticas, mas compiladores e DSLs estão disponíveis para criar circuitos, por exemplo, Cairo | Difícil de construir aplicativos devido às complexidades matemáticas, mas compiladores e DSLs estão disponíveis para criar circuitos, por exemplo, Noir |
| Privacidade | Não | Não | Total |
| Custos de transação | Menores, pois geralmente não é necessário hardware de alto desempenho, e quantidade limitada de dados é postada on-chain. No entanto, observe que em escala, por exemplo, mais de 100 TPS, os rollups otimistas tendem a começar a custar mais, e os ZK-rollups começam a custar menos. | Maiores devido à verificação on-chain da prova e hardware de alto desempenho necessário para gerar provas | Maiores devido à verificação on-chain da prova e hardware de alto desempenho necessário para gerar provas |
| Configuração confiável (*Trusted setup*) | Não | Sim | Sim |

**Classificação dos rollups com base em provas de fraude e validade**

De outro ponto de vista, dependendo de se utilizam provas de validade ou provas de fraude e de qual mecanismo é usado para disponibilidade de dados, podemos dividir amplamente as soluções de camada 2 (L2s) nas seguintes quatro categorias:

* ZK-rollups – Utilizam provas de validade com os dados mantidos na cadeia Ethereum da camada 1 (L1) e utilizam ZK-SNARKs/STARKs.
* Rollups otimistas – Utilizam provas de fraude com dados postados na cadeia Ethereum da L1.
* Validium (validia) – *Validia* (ou *validiums*) são soluções de escalabilidade que usam provas de validade, como ZK-rollups utilizando ZK-SNARKs ou ZK-STARKs, para garantir a integridade das transações e não armazenam os dados das transações na cadeia L1 do Ethereum. Se os dados são mantidos off-chain, pode haver problemas de disponibilidade de dados, mas isso é visto como um tradeoff para alcançar um alto nível de escalabilidade – geralmente em torno de 10.000 transações por segundo. Além disso, várias cadeias podem ser executadas em paralelo para alcançar aumentos adicionais de taxa de transferência e escalabilidade.
* Plasma – Trata-se de uma blockchain separada ancorada à cadeia da camada 1 do Ethereum. Executa transações off-chain usando seu próprio mecanismo de validação de blocos e mantém os dados de transação off-chain. Usa provas de fraude para resolver disputas.

Podemos visualizar essa classificação na tabela a seguir:

| Onde estão os dados? | SNARK/STARK | Provas de Fraude |
| --- | --- | --- |
| Dados on-chain | ZK-rollup | Rollup otimista |
| Dados off-chain | Validium | Plasma |

Lembre-se: a principal diferença entre os modelos de segurança dos rollups e das sidechains é que os rollups herdam as garantias de segurança do Ethereum ao postar mudanças de estado e transações na cadeia principal do Ethereum, enquanto as sidechains são independentes e não postam mudanças de estado e dados de transação na cadeia Ethereum.

Existem muitas soluções que podem ser categorizadas como camada 2. A seguir, apresentamos uma seleção com uma breve descrição de cada uma:

* Arbitrum – Um rollup otimista que busca ter a mesma experiência de interação com o Ethereum, mas com uma fração do custo de transação da L1.

No futuro, esperamos ver soluções híbridas usando uma combinação das técnicas acima. Algumas são boas para determinados casos de uso específicos, e outras para outros.

Embora os ZK-ZK-rollups sejam basicamente ZK-rollups, eles também fornecem confidencialidade, razão pela qual são categorizados separadamente aqui.

* **Aztec** – Uma rede L2 de código aberto que oferece escalabilidade e privacidade para o Ethereum. Ela permite pagamentos cripto totalmente privados e de baixo custo usando provas de conhecimento zero.
* **Base** – Uma cadeia de camada 2 introduzida pela Coinbase com base no stack da Optimism (OP stack).
* **Boba Network** – Uma solução de escalabilidade e aprimoramento L2 para Ethereum baseada em rollups otimistas, derivada da Optimism.
* **DeversiFi** – Permite acesso a DeFi no Ethereum. Usuários podem investir, negociar e enviar tokens sem pagar taxas de gás.
* **Fuel v1** – Um rollup otimista com baixos custos de transação, alta velocidade e alta taxa de transferência.
* **Gluon** – Um mecanismo de negociação escalável de camada 2 construído sobre o Ethereum, oferecendo baixas taxas e negociação de alta frequência.
* **Immutable X** – Uma solução de camada 2 para NFTs no Ethereum com zero taxas de gás, negociações instantâneas e escalabilidade para jogos, aplicações e marketplaces.
* **Layer2 Finance** – Permite acesso a protocolos DeFi para todos, onde os usuários podem agregar seu uso de DeFi e economizar nas taxas do Ethereum.
* **Loopring** – As mesmas garantias de segurança da L1 do Ethereum, mas com escalabilidade e taxa de transferência aumentadas em 1000x e custos de transação reduzidos para 0,1% da L1.
* **Metis Andromeda** – Uma solução de escalabilidade compatível com EVM originalmente derivada da Optimism.
* **OMG Network** – Uma rede de transferência de valor para ETH e tokens ERC20. Escala através de processamento centralizado de transações e garante segurança ao descentralizar a segurança.
* **Optimism** – Ethereum otimista é uma cadeia de rollup otimista compatível com EVM. Tem como objetivo ser rápida, simples e segura.
* **Polygon** – Esta é uma solução de camada 2 composta por várias sidechains que são executadas em paralelo com a cadeia principal do Ethereum. Várias técnicas são utilizadas por essas sidechains para fornecer escalabilidade, incluindo ZK-rollups, cadeias Plasma e rollups otimistas. A Polygon possui um token nativo chamado **MATIC**, que é um token ERC-20 e é usado para governança, staking e taxas.

A Polygon é uma suíte composta pelas seguintes soluções de escalabilidade:

* **Polygon PoS** – Uma rede PoS pública que alcança alta taxa de transferência de transações e baixas taxas ao executar transações em sidechains. PoS garante segurança por meio da estrutura de ponte Plasma e descentralização por meio de uma rede de validadores PoS.
* **Polygon Edge** – Um framework que permite a construção de redes públicas e privadas compatíveis com Ethereum.
* **Polygon zkEVM** – Um ZK-rollup compatível com EVM.
* **Polygon Avail** – Uma solução de disponibilidade de dados que permite o ordenamento e armazenamento de dados de transações para outras blockchains.
* **Polygon Miden** – Um ZK-rollup baseado em STARK.
* **Polygon Zero** – Um ZK-rollup compatível com Ethereum baseado em provas recursivas.
* **Polygon Nightfall** – Um rollup otimista utilizando provas de conhecimento zero.
* **Polygon Hermez** – Um ZK-rollup otimizado para transferências de tokens seguras, de baixo custo e usáveis no Ethereum.
* **Polygon Supernets** – Permitem a criação de blockchains personalizadas e de alto desempenho.
* **STARKs, StarkNet e StarkEx** – StarkNet e StarkEx são duas soluções fornecidas pela StarkWare para escalar o Ethereum. StarkNet é uma camada 2 descentralizada, permissionless e de uso geral baseada em ZK-rollups com STARKs. StarkEx é um mecanismo de escalabilidade específico para aplicações desenvolvido com Cairo e SHARP. StarkNet, StarkEx e as tecnologias relacionadas são desenvolvidas pela StarkWare. Essas tecnologias de escalabilidade usam STARKs para permitir escalabilidade blockchain ao provar a integridade computacional das computações realizadas off-chain na camada 2.
* **ZKSpace** – Uma plataforma composta por três partes principais:
  1. uma exchange descentralizada de formador automático de mercado (AMM DEX) na camada 2 baseada em ZK-rollups chamada ZKSwap v3;
  2. um serviço de pagamentos chamado ZKSquare;
  3. um marketplace de NFTs chamado ZKSea.
* **ZKSwap** – É um fork do zkSync com funcionalidade AMM adicionada.
* **zkSync** – Uma plataforma de ZK-rollup centrada no usuário que oferece escalabilidade para o Ethereum. Suporta pagamentos, trocas de tokens e cunhagem de NFTs.
* **zkSync 2.0** – Esta é uma nova versão do zkSync que combina zk-rollup e zkPorter.

Em resumo, os rollups são a tecnologia mais promissora, mais ativamente desenvolvida e usada para escalar blockchains de camada 1.

**Exemplo**

Nesta seção, veremos como o Polygon funciona e como podemos implantar um contrato na rede de camada 2 Polygon.

**Polygon PoS**

Polygon é uma suíte de soluções de escalabilidade. Polygon PoS é uma cadeia de comprometimento proof-of-stake (prova de participação) ancorada à camada 1 do Ethereum por meio de checkpoints.

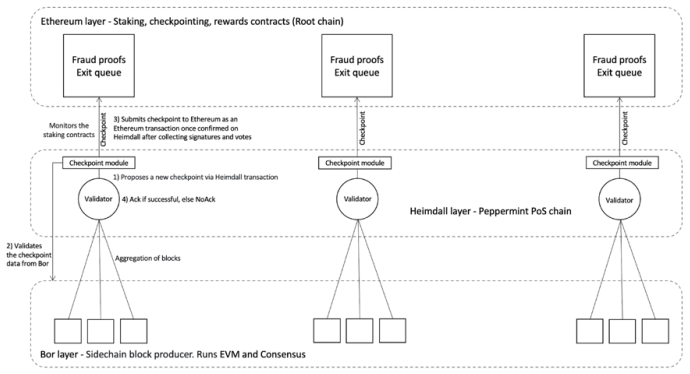
O Polygon PoS é composto por dois componentes principais: uma cadeia Heimdall e uma cadeia Bor. A cadeia Heimdall funciona em conjunto com o contrato stake manager implantado na camada 1 do Ethereum. O contrato stake manager é responsável pela gestão de validadores e participações. Ao ter um contrato stake manager na Ethereum, a segurança do mecanismo melhora, pois agora não é necessário confiar no validador executando na cadeia Heimdall, podendo-se confiar na segurança da camada 1. A cadeia Heimdall cria checkpoints na mainchain do Ethereum periodicamente. Esse mecanismo de checkpoints assegura a finalidade na cadeia principal do Ethereum.

O componente Bor é responsável pela produção de blocos, coletando transações em cada bloco. A produção de blocos é realizada por um subconjunto de validadores do Heimdall. Um subconjunto de validadores é selecionado para produzir blocos por um período de tempo ou número de blocos, chamado *span*. Após o término do *span*, o processo de seleção de validadores começa novamente. Os validadores são selecionados com base na quantidade de tokens MATIC que eles possuem em staking. Quanto maior o stake, maior o número de slots atribuídos ao validador. No entanto, a aleatorização do conjunto de validadores, combinada com um mecanismo de seleção de proponentes baseado no Tendermint, permite que o algoritmo selecione validadores de modo que todos tenham uma chance de produzir blocos em um *span*, mesmo que seu stake seja inferior ao dos outros.

Uma vez que os blocos são produzidos pela camada Bor, a camada Heimdall os agrega em uma única raiz de Merkle e a publica na cadeia da camada 1 do Ethereum em intervalos regulares. Os proponentes para os checkpoints são selecionados via Tendermint de forma ponderada e em ordem circular.

Polygon é, portanto, assegurado por seu próprio mecanismo de proof-of-stake e por seu token nativo chamado **MATIC**.

Uma arquitetura de alto nível do Polygon PoS é mostrada na Figura 17.9:

****

*Figure 17.9: High-level Polygon architecture*

O diagrama anterior mostra a cadeia principal do Ethereum (camada 1) no topo, com os nós PoS no meio e a sidechain Polygon na camada mais baixa.

Como exemplo, podemos implementar no Polygon o token ERC-20 chamado **MET** que construímos no Capítulo 15, *Tokenization*. Como o conjunto de ferramentas padrão do Ethereum funciona com o Polygon, tudo o que precisamos fazer é simplesmente conectar o MetaMask à rede de teste do Polygon e, então, implantá-lo por meio do Remix IDE. Não entrarei nesses passos detalhadamente, pois o processo é bastante direto e já vimos como configurar e usar o MetaMask nos capítulos anteriores. Tudo o que precisamos fazer é adicionar os detalhes da rede do Polygon sob a opção *custom RPC* no MetaMask e implantar o contrato como você faria normalmente no Ethereum. Os detalhes da rede de teste do Polygon estão mostrados abaixo:

* **Nome da rede:** Mumbai
* **URL da rede:** <https://matic-mumbai.chainstacklabs.com>
* **Chain ID:** 80001
* **Símbolo da moeda:** MATIC

Como alternativa, você pode acessar <https://chainlist.org/>, que hospeda uma lista de redes EVM e permite aos usuários adicionarem redes facilmente ao seu navegador MetaMask, sem exigir que esses detalhes sejam inseridos manualmente por meio de uma RPC personalizada no MetaMask.

Você também precisará de alguns tokens de teste para poder implantar os contratos inteligentes. Para obter tokens de teste MATIC, visite:  
<https://faucet.polygon.technology/> e insira o endereço da sua conta MetaMask.

Uma vez que o MetaMask esteja configurado e conectado, e que você tenha fundos suficientes do Faucet, abra o Remix IDE em <https://remix-project.org/> e escolha a opção **Injected provider – MetaMask** como ambiente, no item **DEPLOY & RUN TRANSACTIONS**. Depois que o Remix estiver conectado ao MetaMask, certifique-se de que a conta está corretamente configurada. Uma vez verificado, basta compilar e implantar da mesma maneira que foi feito no Capítulo 15, *Tokenization*, usando o Remix IDE. Como o MetaMask agora está conectado ao Polygon, a implantação será feita na testnet Mumbai da Polygon. Uma vez implantado, você pode explorar ainda mais usando o explorador de blocos em <https://mumbai.polygonscan.com>. Com este exemplo, completamos nossa introdução às tecnologias de camada 2.

Surge aqui uma pergunta – com todas essas soluções de escalabilidade de vários tipos, como será o futuro da escalabilidade blockchain? Ele será centrado em rollups, baseado em múltiplas cadeias, ou será o escalonamento da camada 1 o futuro? Apenas o tempo dirá, mas por enquanto os rollups estão na dianteira da corrida! Ao mesmo tempo, lembre-se de que não existe uma única melhor solução adequada a todos os casos de uso. É provável que o futuro contenha todas essas soluções de escalabilidade trabalhando juntas para formar um ecossistema global de blockchains eficiente e de alto desempenho.

**Layer 3 e além**

Soluções de múltiplas camadas devem existir no futuro, onde há uma cadeia pública de camada 1 no núcleo do ecossistema e uma camada 2 pública fornecendo escalabilidade. Existe a possibilidade de outra camada acima, a chamada **camada 3**, onde múltiplas cadeias específicas de aplicação podem ser executadas herdando a segurança da "cadeia núcleo" (ou seja, camada 1) por meio da camada 2 – por exemplo, uma cadeia que fornece privacidade, outra cadeia que seja outro rollup, outra cadeia usada para pagamentos, e assim por diante.

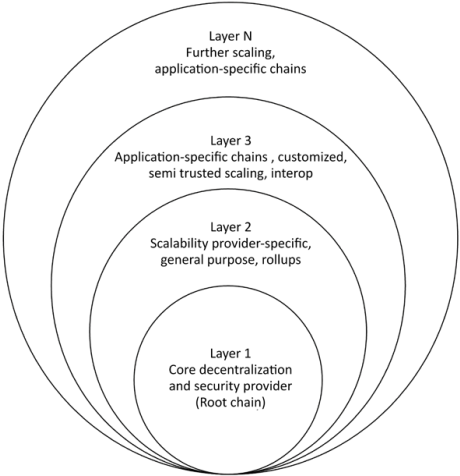
Além disso, podemos imaginar outra camada, também específica de aplicação ou outra camada de rollup, e esse ecossistema pode crescer camada por camada. A camada 3 é concebida como uma camada de execução off-chain adicional no topo das redes de camada 2 existentes. Pode até haver uma camada 4 adicional e camadas posteriores para fornecer ainda mais escalabilidade, apelidadas de “**hiperescalabilidade**”.

Essa configuração poderia permitir um ecossistema interoperável de cadeias e camadas que poderia dar origem a uma rede de redes em camadas múltiplas e heterogênea, criando um ambiente interoperável, inclusivo de serviços e economia. Portanto, a camada 3 também pode ser imaginada como uma camada de interoperabilidade entre diferentes blockchains.

Ideias como *fractal scaling* (escalonamento fractal) ou *hyperscaling* (hiperescalonamento) têm sido propostas pela StarkWare e outros com diferentes variações, mas o consenso geral é que é possível escalar além da camada 2 e que o futuro do ecossistema blockchain será amplamente baseado nessa visão.

<https://chainlist.org/>  
<https://faucet.polygon.technology/>  
<https://remix-project.org/>  
<https://mumbai.polygonscan.com>

Podemos visualizar esse conceito conforme ilustrado na Figura 17.10:

****

*Figure 17.10: A vision of a scalability ecosystem*

Com isso, concluímos nossa discussão sobre algumas das muitas melhorias de escalabilidade existentes e potenciais para blockchain. No próximo capítulo, discutiremos privacidade e algumas técnicas para alcançar privacidade em blockchain.

**Resumo**

Neste capítulo, introduzimos um grande desafio que o blockchain enfrenta: escalabilidade limitada, juntamente com vários métodos disponíveis para alcançar maior escalabilidade. Fundamentalmente, existem soluções de camada 0, camada 1 e camada 2. A camada 2 pode ser dividida em Plasma, canais de estado, rollups e validiums. As soluções de camada 1 são soluções on-chain, como cadeias baseadas em DAG e blockchains baseadas em mecanismos de consenso mais rápidos, como Solana. As soluções de camada 0 são soluções multichain, como o Polkadot.

Observe que, embora as soluções de camada 2 sejam bastante promissoras e haja muita pesquisa e desenvolvimento em andamento para torná-las ainda melhores, existem várias questões de segurança e desafios relevantes que precisam ser abordados para garantir que esses protocolos sejam seguros. Cobriremos alguns desafios de segurança chave dos protocolos de camada 2 no Capítulo 19, *Blockchain Security*. No próximo capítulo, exploraremos outro tópico muito interessante – **privacidade em blockchain**.