Red Axelar :
Conectar las aplicaciones a los ecosistemas de blockchain

Draft 1.0

Enero de 2021

Traduccion de https://static1.squarespace.com/static/5f7679246f2afb311bbb67dd/t/602d3503ad35b40d1bdc209c/1613575428020/axelar_whitepaper_v1.pdf

Resumen

Están surgiendo muchos ecosistemas de blockchain, que ofrecen características únicas y distintas atractivas para los usuarios y los desarrolladores de aplicaciones. Sin embargo, la comunicación entre los ecosistemas es muy escasa y fragmentada. Para permitir que las aplicaciones se comuniquen sin problemas a través de los ecosistemas de blockchain, proponemos Axelar. La pila Axelar proporciona una red descentralizada, protocolos, herramientas y APIs que permiten una comunicación sencilla entre cadenas. El conjunto de protocolos Axelar consta de protocolos de enrutamiento y transferencias transfronterizas. Una red abierta y descentralizada de validadores impulsa la red; cualquiera puede unirse, utilizar y participar. El consenso bizantino, la criptografía y los mecanismos de incentivos están diseñados para cumplir con altos requisitos de seguridad y facilitación, únicos para las solicitudes entre cadenas.

1 Introducción

Los sistemas de blockchain están ganando rápidamente popularidad y atrayendo nuevos casos de uso para la tokenización de activos, las finanzas descentralizadas y otras aplicaciones distribuidas. Varias plataformas importantes como Ethereum, Monero, EOS, Cardano, Terra, Cosmos, Avalanche, Algorand, Near, Celo y Polkadot ofrecen cada una sus propias características y entornos de desarrollo que las hacen atractivas para una variedad de aplicaciones, casos de uso y usuarios finales [5, 11, 4, 21, 20, 23, 24, 19, 6, 14, 25]. Sin embargo, las características útiles de cada nueva plataforma están actualmente disponibles para menos del 1% de los usuarios del ecosistema, es decir, los titulares de fichas nativas en esa plataforma. ¿Podemos permitir que los desarrolladores de plataformas conecten fácilmente sus blockchains con otros ecosistemas? ¿Podemos permitir a los creadores de aplicaciones que construyan en la plataforma que mejor se adapte a sus necesidades, al tiempo que se comunican con múltiples ecosistemas de blockchains? ¿Podemos permitir que los usuarios interactúen con cualquier aplicación en cualquier blockchain directamente desde su wallet ?

Para vincular los ecosistemas de blockchain y permitir que las aplicaciones se comuniquen sin problemas entre sí, proponemos la red Axelar. Los validadores ejecutan colectivamente un protocolo de consenso bizantino y ejecutan protocolos que facilitan las peticiones entre cadenas. Cualquiera puede unirse, participar y utilizar la red. La red subyacente está optimizada para una alta requisitos de seguridad y vitalidad, únicos para las solicitudes entre canales.

La red Axelar también incluye un conjunto de protocolos y API. Los protocolos básicos son los siguientes:

- Cross-chain Gateaway Protocol (CGP). Este protocolo es análogo al Border Gateway Protocol de Internet. Este protocolo se utiliza para conectar múltiples ecosistemas autónomos de blockchain y se encarga de enrutar a través de ellos. Las blockchains no necesitan "hablar un lenguaje especial". Los desarrolladores de sus plataformas no necesitan hacer ningún cambio específico en sus canales y sus canales pueden conectarse fácilmente a la red mundial.
- Cross-chain Transfer Protocol (CTP). Este protocolo es análogo a los protocolos de aplicación FTP y HTTP de Internet. Se trata de una pila de protocolos a nivel de aplicación sobre protocolos de enrutamiento (como CGP y otras tecnologías de enrutamiento). Los desarrolladores de aplicaciones pueden conectar sus dapps a cualquier cadena para realizar consultas entre cadenas. Los usuarios pueden utilizar el protocolo CTP para interactuar con las aplicaciones de cualquier cadena mediante sencillas llamadas a la API similares a las peticiones HTTP GET/POST. Los desarrolladores pueden bloquear, desbloquear y transferir recursos entre dos direcciones cualesquiera en cualquier plataforma de blockchain, realizar activaciones de aplicaciones entre cadenas (por ejemplo, una dapp en la cadena A puede actualizar su estado si otra aplicación en la cadena B cumple ciertos criterios de búsqueda (tipo de interés > X)), y realizar solicitudes genéricas entre aplicaciones en cadenas (un contrato inteligente en la cadena A puede hacer una llamada para actualizar el estado de un contrato inteligente en la cadena B). Este protocolo permite la composibilidad de los programas a través de los ecosistemas de blockchain.

La red Axelar ofrece las siguientes ventajas:

- *Para los desarrolladores de plataformas de blockchain :* capacidad de conectar fácilmente sus blockchains con todos los demás ecosistemas. Sólo es necesario crear una *cuenta de umbral (threshold account)* en la blockchain para conectarse a la red.
- *Para los autores de dapp : Los* creadores de aplicaciones pueden alojar sus dapps en cualquier lugar, bloquear, desbloquear, transferir recursos y comunicarse con las aplicaciones de cualquier otra cadena a través de la API CTP.
- *Para los usuarios :* Los usuarios pueden interactuar con todas las aplicaciones del ecosistema directamente desde su cartera.

Una plataforma para desarrolladores. Por último, la red Axelar es una plataforma para desarrolladores y una comunidad global. Su modelo de gobierno está abierto a todos. Los desarrolladores pueden proponer nuevos puntos de integración, nuevas rutas y nuevos protocolos a nivel de aplicación, y los usuarios pueden decidir adoptarlos votando las propuestas y, si se aprueban, los validadores adoptarán los cambios.

1.1 Soluciones de interoperabilidad existentes

Los intentos anteriores de resolver el problema de la interoperabilidad entre las blockchains se encuadran en una de estas cuatro categorías : intercambios centralizados, ecosistemas interoperables, activos *envueltos* y puentes de tokens. A continuación resumimos brevemente estos enfoques.

Sistemas centralizados. En la actualidad, los sistemas centralizados son las únicas soluciones realmente escalables para las necesidades de interoperabilidad del ecosistema. Pueden listar cualquier activo o incrustar cualquier plataforma con relativa facilidad. Sin embargo, se sabe que los sistemas centralizados tienen varios problemas de seguridad y su calidad no es suficiente para alimentar el emergente sistema financiero descentralizado que requiere una fuerte seguridad, transparencia y gobernanza abierta. Por sí solas no pueden alimentar las aplicaciones descentralizadas a medida que se desarrollan.

Centros de interoperabilidad. Proyectos como Cosmos, Polkadot y Avalanche abordan la interoperabilidad entre *cadenas laterales* nativas en sus ecosistemas utilizando protocolos de comunicación entre cadenas personalizados [23, 25, 24]. Por ejemplo, se puede ejecutar una cadena lateral (una zona Cosmos) que puede comunicarse con el Hub Cosmos. La cadena lateral debe estar basada en el consenso de Tendermint y debe hablar el protocolo que entiende de forma nativa el Cosmos Hub. Las conexiones con otros blockchains y ecosistemas que hablan otros idiomas se dejan en manos de tecnologías externas.

Puentes por parejas. Los activos envueltos (por ejemplo, los bitcoins envueltos) intentan salvar la falta de interoperabilidad entre las cadenas del ecosistema. Un ejemplo es tBTC [9], que es un protocolo personalizado en el que se utiliza una ingeniosa combinación de contratos inteligentes y garantías para asegurar las transferencias. Estas soluciones requieren un considerable esfuerzo de ingeniería ; para cada par de cadenas, los desarrolladores deben establecer un nuevo contrato inteligente en la cadena de destino que analice las pruebas de estado de la cadena de origen (ya que cada sidechain podría, en principio, analizar el estado de las otras cadenas). Sólo unos pocos puentes se han desplegado con este enfoque. Estos enfoques no son adecuados cuando una de las blockchains subyacentes quiere mejorar sus reglas de consenso o el formato de las transacciones. De hecho, todos los contratos inteligentes que dependen del estado de estas cadenas deben ser actualizados. También hay que poner validadores y pedirles que bloqueen diferentes activos para sobreproveer cualquier transferencia de activos, lo que limita la eficiencia económica de dichas transferencias.

También hemos visto algunos otros puentes de propósito único creados por desarrolladores de plataformas que reescriben la lógica de transición de estados en los contratos inteligentes para hacer de puente con otros ecosistemas [1, 7]. Sufren múltiples problemas de escalabilidad, no permiten que el ecosistema se adapte de manera uniforme e introducen dependencias

para las aplicaciones. Por ejemplo, si una plataforma cambia, todos los contratos inteligentes de todas las pasarelas tendrán que actualizarse. Esto acabará colocando el ecosistema en un punto muerto en el que nadie podrá mejorarlo. Por último, si un puente de un solo uso conecta las plataformas A y B, y un segundo puente de un solo uso conecta B y C, esto no significa que las aplicaciones en A puedan hablar con las aplicaciones en C. Puede que haya que crear otro puente de un solo uso o recablear la lógica de la aplicación.

Otros intentos de abordar la interoperabilidad incluyen oráculos federados (por ejemplo, Ren [8]), y blockchains interoperables para aplicaciones específicas [10].

En resumen, las soluciones de interoperabilidad existentes requieren un importante trabajo de ingeniería por parte de los desarrolladores de plataformas y diseñadores de aplicaciones, que deben entender los diferentes protocolos de comunicación para comunicarse entre cada par de ecosistemas. Así, la interoperabilidad es prácticamente inexistente en el espacio actual de la blockchain. En última instancia, los desarrolladores de plataformas quieren centrarse en la construcción de plataformas y en optimizarlas para sus casos de uso y poder conectarse fácilmente a otras blockchains. Y los desarrolladores de aplicaciones quieren escribir dapps en las mejores plataformas para sus necesidades sin dejar de aprovechar a los usuarios, la liquidez y la comunicación con otras dapps en otras cadenas.

2 La búsqueda de una comunicación multicanal escalable

En el centro de la comunicación entre cadenas está la necesidad de que redes heterogéneas puedan comunicarse utilizando el mismo lenguaje. Por lo tanto, explicamos el conjunto de protocolos Axelar, describimos sus propiedades de alto nivel y explicamos cómo estas propiedades abordan el núcleo de la comunicación escalable entre cadenas.

1 *Integración plug-and-play.* A los desarrolladores de plataformas de blockchains no se les debería exigir un gran trabajo de ingeniería o de integración para hablar un "lenguaje especial" que soporte la "cross-chain". El protocolo de "cross-chain" debe ser capaz de conectarse sin problemas a cualquier blockchain existente o nueva. Los nuevos activos deben añadirse con un esfuerzo mínimo.

2 *Enrutamiento entre canales.* Funciones como el descubrimiento de direcciones, rutas y redes están en el corazón de Internet y son facilitadas por BGP y otros protocolos de enrutamiento. Del mismo modo, para facilitar la comunicación entre los ecosistemas de blockchain, se debe apoyar el descubrimiento de direcciones, el descubrimiento de aplicaciones y el enrutamiento entre ellos.

3 *Apoyo a las actualizaciones.* Si uno de los ecosistemas de la cadena de producción cambia, esto no debería afectar a la interoperabilidad de las demás cadenas. El sistema debe reconocer las actualizaciones, y se debe requerir un esfuerzo mínimo para apoyarlas (es decir, no es

necesario reescribir la "lógica de transición de estados", y las aplicaciones no tienen que romperse como resultado).

4 *Un lenguaje uniforme para las aplicaciones.* Las aplicaciones necesitan un protocolo sencillo para bloquear, desbloquear, transferir y comunicarse entre ellas, independientemente de la cadena en la que se encuentren. Este protocolo debe ser independiente de la cadena y soportar llamadas sencillas, como los protocolos HTTP/HTTPS que permiten a los usuarios y navegadores comunicarse con cualquier servidor web. A medida que más y más redes y activos se unen a los protocolos de enrutamiento de bajo nivel, las aplicaciones deberían poder utilizarlos para comunicarse sin reescribir sus bases de software.

A continuación, resumimos los requisitos de seguridad que deben cumplir estos protocolos.

1 *Confianza descentralizada.* La red y los protocolos deben ser descentralizados, abiertos y permitir que todos participen de forma equitativa.

2 *Alta seguridad.* El sistema debe cumplir con altos estándares de seguridad. El sistema debe mantener la seguridad de los activos y el estado mientras la red de la "cross-chain" lo procesa.

3 *Alta vitalidad.* El sistema debe cumplir las garantías de vitalidad para poder soportar aplicaciones que aprovechen sus características de "cross-chain". Un subconjunto de estas propiedades puede satisfacerse fácilmente. Por ejemplo, uno puede crear una cuenta federada *multisig* con sus amigos y bloquear/desbloquear activos en las cadenas correspondientes. Estos sistemas son intrínsecamente vulnerables a los ataques de colusión y censura, y los validadores no tienen suficientes incentivos para protegerlos. La creación de una red descentralizada y un conjunto de protocolos en los que todos puedan participar y estén debidamente incentivados permite una comunicación fluida entre cadenas, pero se trata de un problema difícil que requiere una compleja combinación de protocolos de consenso, criptografía y diseño de sistemas.

3 Red Axelar

La red Axelar ofrece una solución uniforme de comunicación entre canales que satisface las necesidades de los desarrolladores de plataformas -no se requiere ningún trabajo de integración por su parte- y de los creadores de aplicaciones -un protocolo y una API sencillos para acceder a la liquidez global y comunicarse con todo el ecosistema-.

La red Axelar consiste en una red descentralizada que conecta ecosistemas de blockchains que hablan diferentes idiomas y un conjunto de protocolos con APIs en la parte superior, que permite a las aplicaciones realizar fácilmente peticiones entre cadenas. La red conecta blockchains independientes existentes como Bitcoin, Stellar, Terra, Algorand, y centros de interoperabilidad como Cosmos, Avalanche, Ethereum y Polkadot. Nuestra misión es facilitar a los desarrolladores de aplicaciones la escritura de las mismas utilizando un protocolo y una API universales sin tener

que desplegar sus protocolos propietarios de "cross-chain" por debajo o reescribir las aplicaciones a medida que se desarrollan nuevos puentes. Para ello, hemos diseñado un conjunto de protocolos que incluye el Cross-Chain Gateaway Protocol (véase la sección 6) y el Cross-Chain Transfer Protocol (véase la sección 7).

Los protocolos descentralizados subyacentes son una parte esencial de la red. Los validadores mantienen colectivamente la red Axelar y operan los nodos que aseguran la blockchain Axelar. Son elegidos por los usuarios a través de un proceso de delegación. Los validadores tienen un poder de voto proporcional a la participación que se les delega. Los validadores llegan a un consenso sobre el estado de las múltiples blockchains a las que está conectada la plataforma. La blockchain se encarga de mantener y ejecutar los protocolos de enrutamiento y transferencia entre cadenas. Las reglas de gobierno permiten a los participantes de la red tomar decisiones sobre los protocolos, como qué blockchains conectar y qué activos apoyar.

La blockchain de Axelar sigue un modelo de prueba de participación delegada (DPoS) similar al de Cosmos Hub. Los usuarios eligen validadores que deben bloquear su participación en el consenso y mantener un servicio de alta calidad. El modelo DPoS permite mantener un gran conjunto de validadores descentralizados y fuertes incentivos para garantizar que los validadores sean responsables de mantener los puentes y las acciones de los sistemas de umbral criptográfico. Como parte del consenso, los validadores utilizan el software thin client de otras blockchains, lo que les permite comprobar el estado de otras blockchains. Los validadores informan de estos estados a la blockchain de Axelar, y una vez que un número suficiente de validadores informa, el estado de Bitcoin, Ethereum y otras cadenas queda registrado en Axelar.

Posteriormente, la capa central de Axelar conoce en todo momento el estado de las blockchains externas, creando los "puentes de entrada" de otras blockchains. Los validadores gestionan colectivamente *cuentas de firma de umbral en* otras blockchains (por ejemplo, el 80% de los validadores deben aprobar y cofirmar cualquier transacción fuera de estas cadenas), lo que les permite bloquear y desbloquear activos y estados en las cadenas y publicar estados en otras blockchains, los "puentes de salida". En general, la red Axelar puede considerarse como un *oráculo de lectura/escritura descentralizado entre cadenas*.

El resto del documento describe los preliminares y los componentes de la red (sección 4), algunos detalles técnicos de la red (sección 5), el protocolo de pasarela entre canales (sección 6) y el protocolo de transferencia entre canales (sección 7).

4 Preliminares

4.1 Calificación y supuestos

Sea V^{r} el conjunto de validadores Axelar en la ronda R. Cada validador tiene un *peso*, un número en $(0, 1)$ que indica el poder de voto de ese validador en particular. La suma de los pesos de todos los validadores es igual a 1. Un validador es *correcto* si ejecuta un nodo que se ajusta a las reglas del protocolo Axelar. Para finalizar los bloques, o para firmar las solicitudes entre cadenas, Axelar necesita de validadores correctos con un peso total $> F$. Llamamos al parámetro $F \in [0{,}5, 1]$ el *umbral de protocolo*.

Axelar puede basarse en una blockchain de *finalidad instantánea en proof-of-stake delegada.* Los validadores ejecutan el *consenso Byzantine Fault Tolerant (BFT)* en cada ronda i para finalizar el bloque i. Una vez finalizado el bloque i, se ejecuta un nuevo consenso BFT para finalizar el bloque $i + 1$, y así sucesivamente. Los validadores son elegidos por delegación de las partes interesadas. Un usuario con una determinada participación puede optar por ejecutar un nodo validador o delegar su poder de voto (participación) a un validador existente, que vota en su nombre. El conjunto de validadores puede actualizarse, los validadores se unen o abandonan el juego, y los usuarios delegan o retiran su poder de voto.

Las distintas blockchains operan bajo diferentes supuestos de red. La *comunicación sincrónica* significa que hay un límite superior fijo Δ en el tiempo que los mensajes tardan en ser entregados, donde Δ es conocido y puede ser incorporado en el protocolo. La *comunicación asíncrona* significa que los mensajes pueden tardar cualquier tiempo en ser entregados, y se sabe que los protocolos BFT no pueden ser implementados para redes asíncronas incluso en presencia de un único validador malicioso. Un compromiso realista entre sincronía y asincronía es la asunción de una *comunicación parcialmente sincrónica*. La red puede ser completamente asíncrona hasta un cierto tiempo de establecimiento global (GST) desconocido, pero después del GST la comunicación se vuelve síncrona con un límite superior conocido Δ [17].

Las blockchains típicas funcionan bajo el supuesto de $> F$ validadores correctos. Para las redes síncronas, se suele fijar $F = 1/2$, pero para el supuesto más débil de una red parcialmente síncrona, $F = 2/3$. Bitcoin, sus *bifurcaciones* y la versión actual de la prueba de trabajo de Ethereum sólo funcionan bajo el supuesto de la sincronía. Otros, como Algorand y Cosmos, sólo requieren una sincronización parcial. Cuando se conectan cadenas a través de Axelar, la conexión funciona asumiendo los supuestos de red más fuertes entre estas cadenas, que es la sincronía en el caso de la conexión de Bitcoin y Cosmos, por ejemplo. La propia blockchain de Axelar opera en un marco parcialmente síncrono y, por tanto, requiere $F = 2/3$, pero es posible mejorar el umbral requerido asumiendo que otras blockchains existentes son seguras y aprovechando su seguridad.

4.2 Preliminares criptográficos

Firmas digitales. Un *esquema de firma digital* es un conjunto de algoritmos (*Keygen, Sign, Verify*). *Keygen* produce un par de claves (*PK, SK*). Sólo el propietario de *SK* puede firmar mensajes, pero cualquiera puede verificar las firmas con la clave pública *PK*. La mayoría de las blockchains actuales utilizan uno de los esquemas de firma estándar, como ECDSA, Ed25519, o algunas de sus variantes [2, 3].

Firmas de umbral. Un *esquema de firma de umbral* permite a un grupo de *n* partes dividir una clave secreta para un esquema de firma de tal manera que cualquier subconjunto de $t + 1$ o más partes puede colaborar para producir una firma, pero ningún subconjunto de *t* o menos partes puede producir una firma o incluso aprender información sobre la clave secreta. Las firmas producidas por los protocolos de umbral para ECDSA y EdDSA son idénticas a las firmas producidas por los algoritmos convencionales.

Un sistema de firma umbral sustituye los algoritmos *Keygen* y *Sign* de un sistema de firma ordinario por protocolos distribuidos de *n partes T.Keygen*, *T.Sign*. Estos protocolos suelen requerir un canal de difusión público y canales privados por parejas entre las partes, y suelen implicar varios ciclos de comunicación. Tras el éxito de *T.Keygen,* cada usuario tiene una parte *s ide* una clave secreta SK y la correspondiente clave pública PK. El protocolo *T.Sign* permite a estas partes producir una firma para un mensaje dado que es válida bajo la clave pública PK. Esta firma puede ser verificada por cualquier persona utilizando el algoritmo de *verificación del* sistema de firma original.

4.3 Propiedades de los umbrales de las firmas

Hay varias propiedades para un sistema de umbral que son particularmente deseables para las redes descentralizadas:

Seguridad contra una mayoría deshonesta. Algunos sistemas de umbral tienen la restricción de que

sólo son seguros cuando la mayoría de las *n* partes son honestas. Por lo tanto, el parámetro de umbral *t* debe ser inferior a *n/2* [15]. Esta restricción suele ir acompañada del hecho de que se necesitan $2t + 1$ partes honestas para firmar, aunque sólo $t + 1$ partes corruptas puedan ponerse de acuerdo para recuperar la clave secreta. Se supone que los sistemas que no sufren esta restricción están *a salvo de una mayoría deshonesta*.

Como veremos más adelante en la sección 5.2, las plataformas de "cross-chain" deben maximizar la seguridad de sus redes y ser capaces de tolerar el mayor número posible de partes corruptas. Por lo tanto, es necesario establecer sistemas que puedan tolerar una mayoría deshonesta.

Firma previa, firma en línea no interactiva. Con el fin de reducir la carga de comunicación para las partes que firman un mensaje, varios protocolos recientes han identificado una parte importante del trabajo de firma que puede realizarse "fuera de línea", antes de que el mensaje

para firmar se conoce [18, 13]. El resultado de esta fase offline se denomina *pre-firma*. La producción de prefirmas se considera un protocolo *T.Presign* separado, distinto de T.*Keygen* y *T.Sign*. Los resultados del protocolo de pre-firma deben permanecer privados por las partes hasta que los utilicen en la fase de firma. Más tarde, cuando se conozca el mensaje que se va a firmar, sólo habrá que realizar una pequeña cantidad de trabajo adicional
"Esta es la única manera de completar la firma en *T.Sign.*

La fase de *T.Sign en línea* no requiere ninguna comunicación entre las partes. Cada parte simplemente hace un cálculo local sobre el mensaje y la presignatura, y luego anuncia su cuota de firma s_i. *(Una vez hecha pública, estas cuotas de firma s ,...,s son fácilmente combinadas por cualquiera para revelar la firma real s).* (Una vez hechas públicas, estas acciones de firma $s_{1},...,s_{t+1}$son fácilmente combinadas por cualquiera para revelar la firma real *s.*) Esta propiedad se denomina *firma en línea no interactiva*.

Robustez. Los sistemas de umbral sólo garantizan que un subconjunto de partes malintencionadas no pueda firmar mensajes ni conocer la clave secreta. Sin embargo, esta garantía no excluye la posibilidad de que actores malintencionados puedan impedir que todos produzcan claves o firmas. En algunos sistemas, el comportamiento malicioso de una sola persona puede hacer que *T.Keygen* o *T.Sign* se interrumpan sin ningún resultado útil. El único recurso es reiniciar el protocolo, posiblemente con otras partes.

En cambio, para las redes descentralizadas, queremos que *T.Keygen* y *T.Sign* tengan éxito si al menos $t + 1$ de las partes son honestas, incluso si algunas partes maliciosas envían mensajes malformados o dejan caer mensajes en los protocolos. Esta propiedad se llama *robustez*.

Atribución de fallos. La capacidad de identificar a los malos actores en *T.Keygen* o *T.Sign* se denomina *atribución de fallos*. Sin la atribución de la culpa, es difícil excluir o castigar de forma fiable a los malos actores, en cuyo caso los costes impuestos por los malos actores deben ser asumidos por todos. Esta propiedad también es importante para las redes descentralizadas, en las que los comportamientos maliciosos deben ser identificables y desincentivados económicamente mediante reglas de castigo.

Seguridad en configuraciones concurrentes. El esquema de firma debe ser seguro en un entorno concurrente, en el que se puedan utilizar múltiples instancias de los algoritmos de *keygen* y de firma en paralelo. (Drijvers et al [16], por ejemplo, han demostrado un ataque contra los esquemas Schnorr multifirma en tales configuraciones). Existen versiones de los esquemas ECDSA y Schnorr que satisfacen estas propiedades [13, 22].

ECDSA y EdDSA son, con diferencia, los sistemas de firma más extendidos en el ámbito de la blockchain. Por ello, las versiones de umbral de estos dos sistemas han sido objeto de investigaciones y desarrollos recientes. Los lectores interesados en el estado del arte pueden consultar [22, 13, 18] y un reciente documento de estudio [12].

5 Red Axelar

5.1 Diseño de una red transversal abierta

Los puentes que mantiene la red Axelar están respaldados por cuentas umbral, de modo que (casi) todos los validadores deben autorizar colectivamente cualquier solicitud de "cross-chain". Diseñar una red en la que todos puedan participar para asegurar estos puentes requiere cumplir los siguientes requisitos técnicos:

- *Afiliación abierta.* Cualquier usuario debería poder convertirse en validador (siguiendo las normas de la red).
- *Actualización de la afiliación.* Cuando un validador abandona el sistema de forma honesta, su clave debe ser revocada adecuadamente.
- *Incentivos y reducciones.* Los validadores maliciosos deben ser identificables y sus acciones deben ser identificadas y tratadas por el protocolo.
- *Consenso.* Los sistemas de umbral se definen como protocolos autónomos. Para propagar los mensajes entre los nodos, necesitamos canales privados de difusión y punto a punto. Además, los validadores deben acordar el último estado de cada invocación de los sistemas de umbral, ya que suelen tener varias iteraciones de interacciones.
- *Gestión de llaves.* Al igual que los validadores ordinarios de cualquier sistema de punto de venta deben proteger sus claves cuidadosamente, los validadores de Axelar deben proteger sus acciones de umbral. Hay que renovar las llaves, dividirlas entre las partes online y offline, etc.

Axelar comienza con un modelo de *prueba* de participación *delegada*, en el que la comunidad elige un conjunto de validadores para dirigir el consenso. Nótese que los modelos de umbral estándar tratan a cada actor de forma idéntica y no tienen ninguna noción de "peso" en el consenso. Por lo tanto, la red debe adaptarlos para tener en cuenta el peso de los validadores. Un enfoque sencillo es asignar varias cuotas de umbral a los validadores más grandes. A continuación se describen las tres funciones básicas que realizan los validadores en conjunto.

- *Generación de la clave del umbral.* Los algoritmos de generación de claves de umbral existentes para los esquemas de firma estándar de blockchain (ECDSA, Ed25519) son protocolos interactivos entre múltiples participantes (véase la sección 4). Una transacción especial en la red Axelar ordena a los validadores que empiecen a ejecutar este protocolo de estado. Cada validador ejecuta un proceso daemon de umbral que se encarga de mantener el estado secreto de forma segura. Para cada fase del protocolo :
    1. Un validador mantiene el estado del protocolo en su memoria local.
    2. Llama al demonio de los secretos para que genere mensajes según la descripción del protocolo para los demás validadores.
    3. Propaga los mensajes por difusión o a través de canales privados a otros validadores.

4 Cada validador realiza funciones de transición de estado para actualizar su estado, pasar a la siguiente fase del protocolo y repetir los pasos anteriores.

Al final del protocolo, se genera una clave pública umbralizada en la cadena Axelar, que puede ser mostrada al usuario (por ejemplo, para los repositorios) o a la aplicación que generó la solicitud original.

- *Firma de umbral.* Las solicitudes de firma en la red Axelar se gestionan de la misma manera que las solicitudes de generación de claves. Se invocan, por ejemplo, cuando un usuario desea eliminar un activo de una de las cadenas. Se trata de protocolos interactivos, y la transición de estado entre rondas se desencadena en base a los mensajes que se propagan a través de la vista de la blockchain de Axelar y la memoria local de cada validador.

- *Gestión de los cambios en la pertenencia al Validador.* El conjunto de validadores debe rotar periódicamente para permitir la incorporación de nuevas partes interesadas. Cuando se actualiza un conjunto de validadores, tenemos que actualizar la clave de umbral para compartirla con el nuevo conjunto. Así que si permitimos que cualquiera se una en cualquier momento, tendríamos que actualizar la clave de umbral con mucha frecuencia. Para evitarlo, rotamos los validadores cada T bloques. En intervalos de T bloques, el conjunto V^R y la clave de umbral son fijos. En cada rotación, que es un múltiplo entero del parámetro T, actualizamos el conjunto de validadores como sigue:
  1 En cada ronda R, el estado Axelar mantiene un registro del conjunto actual de validadores V^R. $V^{R+1} = V^R$ a menos que $R + 1$ sea un múltiplo de T.
  2 Durante las rondas $((i - 1)T, iT)$, los usuarios publican mensajes de compromiso y desenganche.
  3 Al final del ciclo iT, estos mensajes se aplican a V^{iT} para obtener V^{iT}.

- *Generación de claves de umbral y firma en presencia de validadores rotativos.* La blockchain Axelar puede emitir una solicitud de una nueva clave o una firma de umbral en la ronda R. El proceso de firma lleva más de una ronda y no queremos ralentizar el consenso, por lo que exigimos que la firma se genere antes del inicio de la ronda $R + 10$. En concreto, los validadores comienzan la ronda $R + 10$ sólo después de haber visto un certificado para la ronda $R + 9$ y una firma para cada solicitud de clave/firma emitida en la ronda R. El resultado de todas las peticiones de ronda R debe incluirse en el bloque $R + 11$. En otras palabras, una propuesta de bloque R que no contenga los resultados de una ronda $R - 11$ se considera inválida, y los validadores no la votan. Para garantizar que todos los mensajes umbralizados se firmen antes de que se actualice un conjunto de validadores, Axelar no emite una solicitud umbralizada durante una ronda igual a $-1, -2, \ldots -9$ módulo T.

5.2 Seguridad de la red

La seguridad de los sistemas blockchain se basa en varios protocolos criptográficos y de teoría de juegos, así como en la descentralización de la red. Por ejemplo, en las blockchains de prueba de apuestas, sin los incentivos adecuados, los validadores pueden confabularse y reescribir la historia, robando fondos de otros usuarios en el proceso. En las redes de prueba de trabajo

sin la suficiente descentralización, es bastante fácil crear *bifurcaciones* largas y duplicar el gasto, como han demostrado los múltiples ataques a Bitcoin Gold y Ethereum Classic.

La mayoría de las investigaciones sobre la seguridad de las blockchains se han centrado en las cadenas soberanas. Pero una vez que las cadenas interoperan, hay que tener en cuenta nuevos vectores de ataque. Por ejemplo, supongamos que Ethereum habla con una pequeña blockchain X a través de un puente directo controlado por dos contratos inteligentes, uno en Ethereum y otro en X. Además de las cuestiones de ingeniería que resumimos en la sección 1.1, tenemos que decidir qué ocurre cuando se violan los supuestos de confianza de X. En este caso, si ETH se ha trasladado a X, los validadores de X pueden acordar forjar una historia de X en la que tengan todos los ETH, mostrar las pruebas de consenso forjadas en Ethereum y robar los ETH. La situación es aún peor cuando X está conectada a varias otras cadenas por puentes directos, donde si X *se forja*, los efectos se propagan a través de cada puente. Establecer las directrices de gobernanza de la renovación para cada puente de pares es una tarea abrumadora para cualquier proyecto individual.

La red Axelar aborda los problemas de seguridad mediante los siguientes mecanismos :

- *Máxima seguridad.* Axelar fija el umbral de seguridad en el 90%, lo que significa que casi todos los validadores tendrían que aceptar retirar fondos bloqueados por su red o falsificar la prueba de estado[1]. En la práctica, se ha observado que los validadores de puntos de venta tienen un tiempo de actividad muy elevado (cercano al 100%), siempre que se les incentive adecuadamente. Por lo tanto, la red Axelar producirá bloques incluso a pesar de este alto umbral. Sin embargo, en el raro caso de que algo vaya mal y la red se detenga, la red necesita sólidos mecanismos de respaldo para reiniciar el sistema que se describen a continuación.
- *Máxima descentralización.* Como la red utiliza sistemas de firma de umbral, el número de validadores puede ser lo más grande posible. La red no está limitada por el número de validadores que podemos admitir, ni por los límites de las transacciones, ni por los costes que se derivarían de utilizar, por ejemplo, varias firmas en diferentes cadenas, cuya complejidad (y costes) aumenta linealmente con el número de validadores[2].
- *Mecanismos robustos de retroceso.* La primera cuestión que hay que abordar en una red con umbrales de seguridad elevados como la anterior es qué ocurre cuando la propia red se paraliza. Supongamos que la propia red Axelar se detiene. ¿Podemos tener un mecanismo de reserva que permita a los usuarios recuperar sus fondos? Para hacer frente a una posible caída de la propia red Axelar, cada cuenta de puente umbral en una blockchain X que los validadores de Axelar controlan colectivamente tiene una "llave de desbloqueo de emergencia". Esta clave puede ser compartida por miles de partes e incluso puede ser una clave específica para la cadena de bloques X que se comparte en toda la comunidad de esa cadena.

1 Se puede ajustar el parámetro final que se elegirá para el despliegue de la red.

2 Para algunas blockchains, las multifirmas ofrecen una alternativa razonable cuando los gases son pequeños y los formatos de mensaje admitidos son adecuados. Pero no son adecuados para dos de las plataformas más grandes como Bitcoin y Ethereum.

Por lo tanto, si la red Axelar se bloquea, esta clave servirá como solución alternativa y permitirá la recuperación de los activos (véase más abajo para más detalles).

- *Máxima descentralización de los mecanismos de emergencia.* Este mecanismo de reserva incluye *un conjunto de superposiciones de* usuarios, en las que cualquiera puede participar sin coste alguno. Estos usuarios no necesitan estar conectados, ejecutar nodos o coordinarse entre sí. Sólo son "llamados al servicio" si la red Axelar se bloquea y no puede recuperarse. La seguridad de la red se ve reforzada por un umbral muy alto en el conjunto primario de validadores y un conjunto secundario de superposición máximamente descentralizado.
- *Gobernanza compartida.* Un protocolo común rige la red Axelar. Colectivamente, los usuarios pueden votar qué canal debe ser apoyado por su red. La red también asignará una reserva de fondos que podrá utilizarse para reembolsar a los usuarios en caso de emergencia imprevista, también controlada por los protocolos de gobernanza.

A continuación se analizan varios mecanismos de seguridad.

Mecanismos de retroceso. Cuando Axelar se paraliza debido al alto umbral, una "llave de desbloqueo de emergencia" toma el control de la red. Existen múltiples formas de instanciar esta clave de liberación, y algunos canales/aplicaciones pueden optar por utilizar una variante diferente de Este es el caso del "paquete de recogida", o de desvincularse completamente[3]:

- *Opción a.* Compartir la clave entre las fundaciones de los proyectos de blockchain y las personas reputadas de la comunidad.
- *Opción b.* Compartir información entre los partidos elegidos a través del mecanismo de PdS delegado.
- *Opción c.* Para las cuentas que gestionan activos e información para la cadena/aplicación X, comparta una clave personalizada entre los interesados/validadores de X. Suponiendo que X disponga de mecanismos de gobernanza, los mismos mecanismos pueden aplicarse para determinar un curso de acción si Axelar se estanca.

Ahora, conociendo la identidad de los usuarios de la recuperación y sus claves públicas, un sencillo protocolo genera acciones de la clave de recuperación que nadie conoce. Además, los usuarios del conjunto de recuperación no necesitan estar en línea antes de ser llamados a recuperarse a través de los mecanismos de gobierno. Siguiendo los protocolos estándar de generación de claves distribuidas, cada validador de Axelar comparte un valor aleatorio. La clave secreta de recuperación se genera sumando estos valores. En lugar de realizar las sumas en claro, todas las acciones se encriptan con las claves públicas de los usuarios de recuperación y luego se suman homomórficamente (esto supone una encriptación homomórfica aditiva y una capa adicional de conocimiento nulo, ambas fácilmente disponibles). El resultado de este protocolo es una clave pública de *recuperación (RPK*) y potencialmente miles de encriptaciones (mediante las claves públicas de los usuarios de recuperación) de las correspondientes acciones de clave secreta $Enc_i(s_i)$ que se distribuyen a sus propietarios (por ejemplo, se muestran en la cadena).

3 El despliegue definitivo en la red Axelar se determinará más adelante.

Los contratos puente de Axelar incluyen una opción para recuperar los fondos utilizando el *RPK* bajo ciertas condiciones. Por último, también es posible actualizar esta clave de recuperación e incluso cambiar el conjunto de usuarios titulares de sus acciones sin requerir ningún trabajo de los accionistas participantes.

Si la cadena X que está conectada a Axelar se detiene, hay varias opciones:

- Imponer límites al valor en USD de los activos que pueden transferirse a o desde X en un solo día. Así, una cadena X maliciosa sólo puede robar una pequeña fracción de todos los activos vinculados a ella antes de que los validadores de Axelar la detecten, y entren en juego los mecanismos de gobernanza de los siguientes elementos.
- El módulo de gobernanza de Axelar puede utilizarse para votar lo que ocurre en estas situaciones. Por ejemplo, si hay un error menor y la comunidad reinicia X, el gobierno de Axelar puede decidir retomar la conexión donde la dejó.
- Si ETH se ha trasladado a X, una clave de recuperación específica de Ethereum puede determinar lo que ocurre con los activos ETH.

6 Cross-Chain Gateaway Protocol (CGP)

En esta sección, explicamos el protocolo de pasarela intercanal y los mecanismos de enrutamiento con dos ejemplos básicos comunes a las necesidades de muchas aplicaciones:

Sincronización del estado (apartado 6.2). Envío de información de estado desde una blockchain de origen *S*

en el estado de una blockchain *D de* destino.

(Por ejemplo, el envío de una cabecera de bloque de Bitcoin en la blockchain de Ethereum).

Transferencia de activos (sección 6.3). Transferir un activo digital de *S* a *D* y viceversa.

(Por ejemplo, transferir bitcoins de la blockchain de Bitcoin a la blockchain de Ethereum, y luego de vuelta a la blockchain de Bitcoin).

Para simplificar, asumimos que la cadena *D* tiene al menos un soporte mínimo para los contratos inteligentes, pero que la cadena *S* puede ser cualquier blockchain.

6.1 Cuentas en otros canales

Para vincular las diferentes cadenas, se crean cuentas umbral en cada una de ellas que controlan el flujo de valor e información entre ellas. En el caso de la *Cadena*, la cuenta está anotada por *ChainAxelar*.

Cuenta Bitcoin. Para Bitcoins y otras cadenas de contratos no inteligentes, los validadores de Axelar crean una clave ECDSA de umbral según la sección 5.1. Esta clave controla la cuenta ECDSA en Bitcoin, y es la dirección de destino donde los usuarios envían los depósitos. Se pueden crear llaves con

umbrales específicos a petición del usuario. La clave puede actualizarse periódicamente, y la última clave y las claves personalizadas pueden encontrarse consultando un nodo Axelar.

Cuenta puente de umbral en cadenas con contratos inteligentes. La cadena SC está anotada. Los validadores crean una clave umbral ECDSA o ED25519 según la sección 5.1, dependiendo del tipo de clave que admita la cadena. Denotamos esta clave *PKAxelar*, cuando no hay ambigüedad sobre a qué cadena nos referimos. Esta clave controla una cuenta de contrato inteligente en SC, denotada por *SCAxelar,* y cualquier aplicación en SC puede consultar a *SCAxelar* la dirección PK de esta clave. Así, cualquier aplicación en SC puede reconocer los mensajes firmados por *SKAxelar*. El protocolo también debe tener en cuenta los valores rotativos de *PKAxelar*. Esto sucede de la siguiente manera:

1 Inicialización del *SCAxelar* en SC. Almacena *PKAxelar* como parte de su estado, que se inicializa como su valor de génesis en Axelar. *SCAxelar* también incluye reglas para actualizar el PK.

2 Para actualizar *el PKAxelar*, se debe presentar una transacción del formato (*update, PKnew*) con una firma del *SKAxelar* actual. Entonces el contrato define $PK_{Axelar} = PK_{new}$.

3 Cada vez que los validadores actualizan la clave umbral para SC de PK^{i} a PK^{i+1}, Axelar solicita que los validadores utilicen SK^{i} para firmar (actualización, PK^{i+1}). Posteriormente, esta firma se envía a *SCAxelar*, que actualiza *PKAxelar*.

6.2 Sincronización de estados

Sea q_S una pregunta cualquiera sobre el estado de la cadena *S*. He aquí algunos ejemplos de estas preguntas:

- *"¿En qué ronda de bloques, si es que hay alguna, se produjo una transacción tx?"*
- *"¿Cuál es el valor de un determinado campo de datos?"*
- *"¿Cuál es el hash de la raíz Merkle de todo el estado de S en el bloque 314159?"*

Dejemos que a_S sea la respuesta correcta a q_S y supongamos que un usuario final o una aplicación solicita que a_S se publique en el canal *D*. La red Axelar responde de la siguiente manera:

1 El usuario envía una solicitud q_S a una de las cuentas puente (que luego son recuperadas por los validadores) o directamente a la blockchain de Axelar.

2 Bajo el consenso Axelar, cada validador debe ejecutar un software de nodo para las cadenas *S*, *D*. Los validadores Axelar consultan la API de su software de nodo de *la cadena S para la* respuesta a_S y reportan la respuesta a la cadena Axelar.

3 Una vez que $> F$ validadores ponderados informan de la misma respuesta en la ronda *R*, Axelar pide a los validadores que firmen un $_S$.

4 Utilizando la criptografía de umbral, los validadores firman un_S. La firma se incluye en el $R + 11$.

5 Cualquiera puede tomar el valor firmado a_S del bloque $R + 11$ y enviarlo a D.

6 La solicitud ha sido procesada. Cualquier solicitud en D puede ahora tomar el valor firmado a_S, consultar $_{DAxelar}$ para el último *PKAxelar*, y verificar que la firma de a_S coincide con *PKAxelar*. Los validadores también muestran *un* $_S$ en el recuento de puentes de la cadena D, que las solicitudes pueden recuperar.

6.3 Transferencia de activos entre cadenas

La red permite la transferencia de activos digitales entre canales ampliando el flujo de sincronización de estados del apartado 6.2.

Se emite una cantidad suficiente de fichas S de forma consecutiva y controlada por $_{DAxelar}$ en su inicialización. Supongamos que un usuario solicita intercambiar x cantidad de tokens en la cadena de origen S por x cantidad de tokens S adscritos a la cadena de destino D, para ser depositados en una dirección Dw_D *de su* elección. Presentamos el flujo de trabajo general completo, que soporta cualquier cadena de origen S, incluso cadenas como Bitcoin que no soportan contratos inteligentes:

1 El usuario (o una aplicación que actúe en su nombre) envía una solicitud de transferencia (x, w_D) a la cuenta del puente del umbral, que se dirige a la red Axelar.

2 Los validadores de Axelar utilizan la criptografía de umbral para crear colectivamente una nueva dirección de depósito d_S para S. Envían d_S a la blockchain de Axelar.

3 El usuario (o una aplicación que actúe en su nombre) se da cuenta de d_S mediante la monitorización de la blockchain de Axelar. El usuario envía una cantidad x de S-tokens para tener en cuenta d_S a través de una transacción S ordinaria tx_S utilizando su software habitual para la *cadena S*.

(Debido a la propiedad de umbral de la d_S*, no se pueden gastar fichas de la* d_S *a menos que un número umbral de validadores se coordinen para hacerlo).*

4 tx_S se publica en Axelar. Los validadores consultan a la API del software de su nodo de *la cadena S* la existencia de tx_S y, si la respuesta es "verdadera", envían la respuesta a la cadena Axelar.

5 Una vez que $> F$ validadores ponderados declaran "true" para tx_S en la ronda R, Axelar pide a los validadores que firmen una transacción a_D que envía una cantidad de x tokens S adjuntos de d_{Axelar} a wD.

6 Utilizando la criptografía de umbral, los validadores firman a_D. La firma se incluye en el R +11.

7 Cualquiera puede tomar el valor firmado a_D del bloque $R + 11$ y enviarlo a D.

8 La solicitud ha sido procesada y *una* vez que aparece a_D en D, la transferencia es procesada.

Supongamos ahora que un usuario solicita canjear una cantidad x' de fichas S envueltas de la cadena D a la cadena S, para depositarlas en una dirección Sw_S de su elección. El flujo de trabajo es el siguiente:

1 El usuario inicia una solicitud de transferencia (x', w_S) depositando la cantidad x' de fichas S envuelto en la *cadena D* a través de una *transacción D* ordinaria utilizando su software habitual de *cadena D*.

2 (x', wS) se publica en Axelar. Los validadores consultan a la API del software de su nodo de la cadena D la existencia de (x', w_S) y, si la respuesta es "verdadera", informan de la respuesta a la cadena Axelar.

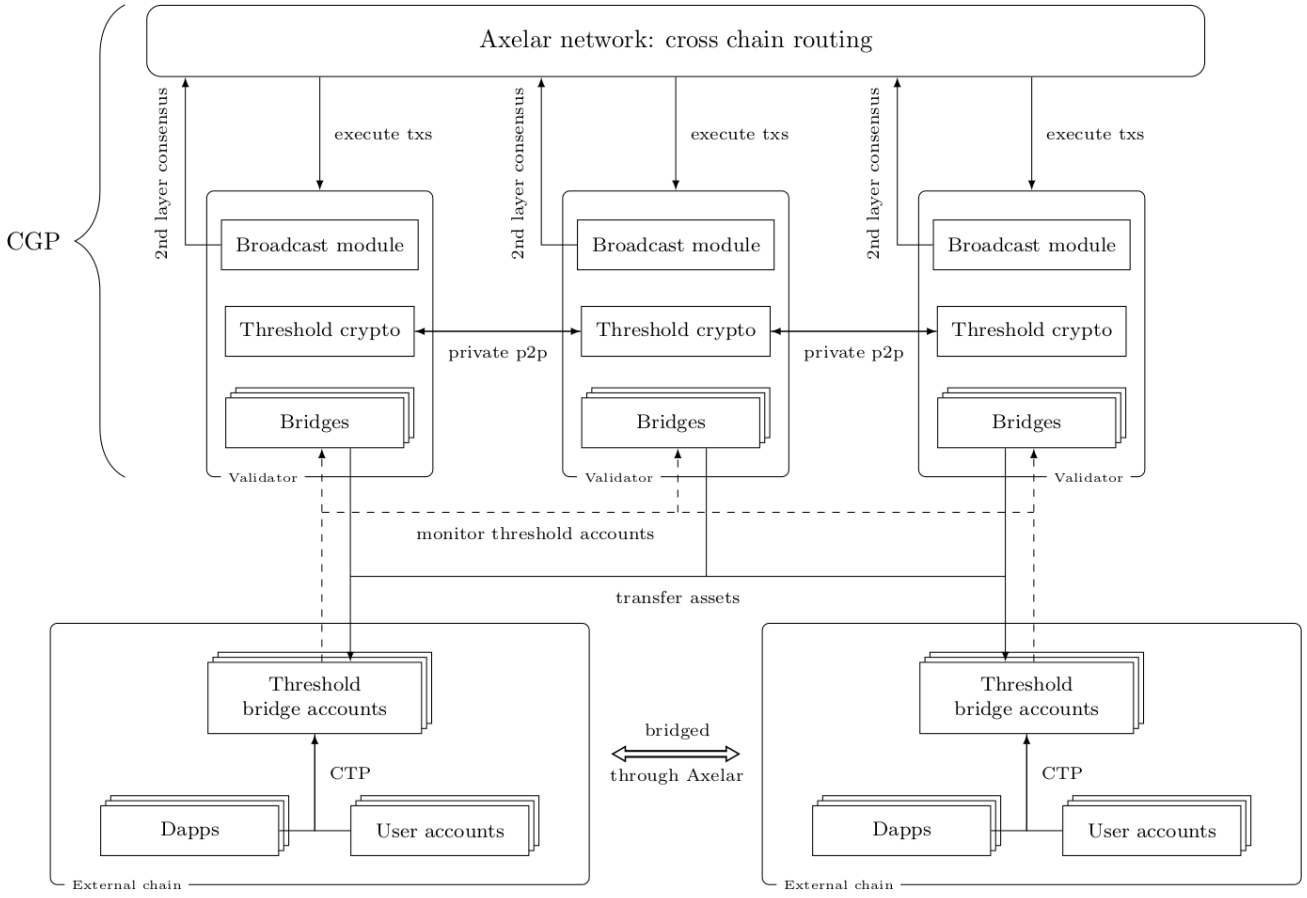

Figura 1: Diagrama de componentes

3 Una vez que > F validadores ponderados declaran "verdadero" p para (x', w_S) en la ronda R, Axelar pide a los validadores que firmen una transacción $a_S$que envíe una cantidad x' de fichas $_{Saxelar}S$ a wS.

4 Utilizando la criptografía de umbral, los validadores firman un_S. La firma se incluye en el $R + 11$.

5 Cualquiera puede tomar el valor con signo $a_S$del bloque $R + 11$ y enviarlo a S.

6 La solicitud *ha* sido procesada y una vez que la a_S aparece en la *S,* la transferencia es procesada.

Las solicitudes adicionales que admite la capa de enrutamiento CGP incluyen el bloqueo, el desbloqueo o la transferencia de activos entre cadenas.

Flujo de transacciones atómicas entre cadenas. Dependiendo del tipo de solicitud entre cadenas, Axelar intenta que las transacciones correspondientes se ejecuten en varias cadenas o en ninguna. Para ello, cada solicitud puede estar en uno de los siguientes estados en la blockchain de Axelar: *(inicializada, pendiente, completada, agotada)*. Si se produce un *tiempo de espera*, la solicitud devuelve un código de error. Algunos eventos de *tiempo de espera* también desencadenan un evento de *reembolso: por* ejemplo, si un activo de una cadena debe ser transferido a un activo de otra cadena, si la cadena receptora no ha procesado la transacción, el activo se reembolsa al usuario original.

7 Protocolo de transferencia en cadena (CTP)

El CTP es un protocolo a nivel de aplicación que permite a las aplicaciones explotar fácilmente la funcionalidad entre canales. Explicamos la integración centrándonos en las características de la transferencia de activos (por ejemplo, la utilizada en DeFi). Estas aplicaciones suelen constar de tres elementos principales: una interfaz gráfica de usuario (front-end), contratos inteligentes en una cadena y un nodo intermedio que registra las transacciones entre el front-end y los contratos inteligentes. El front-end interactúa con el monedero del usuario para aceptar depósitos, procesar retiros, etc. Las aplicaciones pueden aprovechar la funcionalidad entre cadenas llamando a las peticiones CTP de forma análoga a los métodos GET/POST de HTTP/HTTPS. A continuación, la capa CGP recupera estas solicitudes para su ejecución y devuelve los resultados a los usuarios.

- *Solicitudes de CTP*. Los desarrolladores de aplicaciones pueden alojar sus aplicaciones en cualquier canal e integrar sus contratos inteligentes con las cuentas puente del umbral para ejecutar las solicitudes CTP.
- *Cuentas puente de umbral*. Supongamos que un desarrollador de aplicaciones implementa sus contratos en la cadena A. A continuación, hará referencia a los contratos puente para obtener el apoyo de toda la cadena. Este contrato permite que las aplicaciones :
  - Registrar una blockchain con la que desea comunicarse.
  - Registrar los activos en esta blockchain que desea operar.
  - Realizar transacciones de activos, como aceptar depósitos, procesar retiros y otras funciones (similares, por ejemplo, a las llamadas de contratos ERC-20).

Supongamos que una importante aplicación DeFi, MapleSwap, reside de forma nativa en los registros de la cadena A con una cuenta puente de umbral. Los validadores de Axelar gestionan colectivamente el propio contrato en la cadena correspondiente. Supongamos que un usuario quiere hacer un depósito

en un par comercial entre los activos X e Y que residen en las dos cadenas respectivamente. A continuación, cuando un usuario envía una solicitud correspondiente, ésta se dirige a través de la cuenta del puente de umbral a la red Axelar para su procesamiento. A partir de ahí, se realizan los siguientes pasos:

1 La red Axelar entiende que esta aplicación se ha registrado para el apoyo entre cadenas entre los activos. Genera la clave de depósito explotando la criptografía de umbral y el consenso para el usuario en las cadenas A y B correspondientes.

2 Las claves públicas asociadas se devuelven a la aplicación y se muestran al usuario, que puede utilizar su monedero habitual para realizar depósitos. La clave secreta correspondiente se comparte entre todos los validadores de Axelar.

3 Cuando se confirman los depósitos, Axelar actualiza su directorio multicanal para dejar constancia de que el usuario de los canales correspondientes ha depositado esos activos.

4 Los validadores de Axelar ejecutan protocolos multipartitos para generar una firma de umbral que actualiza el recuento de puentes de umbral en el canal A donde reside la aplicación.

5 La solicitud de CTP se envía de vuelta a los contratos inteligentes de la aplicación DeFi, que pueden actualizar su estado, actualizar sus fórmulas de rendimiento, tipos de cambio o ejecutar otras condiciones relacionadas con el estado de las aplicaciones.

En todo este proceso, la red Axelar, a alto nivel, actúa como un oráculo descentralizado de lectura/escritura entre cadenas, CGP es la capa de enrutamiento entre cadenas y CTP es el protocolo de aplicación.

Solicitudes adicionales de cadena cruzada. El CTP admite solicitudes más generales entre aplicaciones a través de blockchains como :

- Realizar servicios de *nombres de clave pública* (*PKNS*). Se trata de un directorio universal que permite emparejar las claves públicas con los números de teléfono o los handles de Twitter (algunos proyectos, como Celo, ofrecen esta funcionalidad en sus plataformas).
- Activadores de aplicaciones en cadena. Una aplicación en la cadena A puede actualizar su estado si otra aplicación en la cadena B cumple un criterio de búsqueda (tipo de interés < *X).*
- Composibilidad de los contratos inteligentes. El contrato inteligente de la cadena A puede actualizar su estado en función del estado de los contratos de la cadena B, o desencadenar una acción para actualizar un contrato inteligente de la cadena B.

Estas solicitudes pueden ser manejadas a un alto nivel ya que, colectivamente, los protocolos de red CTP, CGP y Axelar pueden transmitir y escribir información de estado arbitrariamente verificable a través de blockchains.

8 Resumen

En los próximos años se desarrollarán importantes aplicaciones y activos, además de múltiples ecosistemas de blockchain. La red Axelar puede utilizarse para integrar estas blockchains en una capa de comunicación uniforme entre cadenas. Esta capa proporciona protocolos de enrutamiento y de nivel de aplicación que satisfacen los requisitos tanto de los constructores de plataformas como de los desarrolladores de aplicaciones. Los desarrolladores de aplicaciones pueden aprovechar las mejores plataformas para sus necesidades y sacar provecho de un protocolo y una API sencillos para acceder a la liquidez global entre cadenas, a los usuarios y a la comunicación con otras cadenas.

Referencias

1 Althea Peggy. https://github.com/cosmos/peggy. [Citado en la p. 2.]

2 Uso determinista del algoritmo de firma digital (dsa) y del algoritmo de firma digital de curva elíptica (ecdsa). https://tools.ietf.org/html/rfc6979. [Citado en la p. 5.]

3 Algoritmo de firma digital de curva de Edwards (eddsa). https://tools.ietf.org/html/rfc8032. [Citado en la p. 5.]

4 Libro blanco técnico de Eos.io v2. https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md. [Citado en la p. 1.]

5 Ethereum: Un libro mayor de transacciones descentralizado y seguro. https://ethereum.github.io/yellowpaper/paper.pdf. [Citado en la p. 1.]

6 El libro casi blanco. https://near.org/papers/the-official-near-white-paper/. [Citado en la p. 1.]

7 Puente del Arco Iris. https://github.com/near/rainbow-bridge. [Citado en la p. 2.]

8 Ren: A privacy preserving virtual machine powering zero-knowledge financial applications. https://whitepaper.io/document/419/ren-litepaper. [Citado en la p. 3.]

9 tbtc: Un token descentralizado canjeable respaldado por `btc-20`. https://docs.keep.network/tbtc/index.pdf. [Citado en la p. 2.]

10 Thorchain: Una red de liquidez descentralizada. https://thorchain.org/. [Citado en la p. 3.]

11 Kurt M. Alonso. Cero a monero. https://www.getmonero.org/library/Zero-to-Monero-1-0-0.pdf. [Citado en la p. 1.]

12 Jean-Philippe Aumasson, Adrian Hamelink y Omer Shlomovits. Un estudio sobre la firma del umbral de la ecdsa. Cryptology ePrint Archive, Report 2020/1390, 2020. https://eprint.iacr.org/2020/1390. [Citado en la p. 6.]

13 Ran Canetti, Nikolaos Makriyannis y Udi Peled. Uc no interactiva, proactiva, umbral ecdsa. Cryptology ePrint Archive, Report 2020/492, 2020. https://eprint.iacr.org/2020/492. [Citado en la p. 6.]

14 Libros blancos de cLabs. https://celo.org/papers. [Citado en la p. 1.]

15 Ivan Damgård, Thomas Pelle Jakobsen, Jesper Buus Nielsen, Jakob Illeborg Pagter y Michael Bæksvang Østergård. ECDSA de umbral rápido con mayoría honesta. En SCN, volumen12238 de Lecture Notes in Computer Science, páginas 382-400. Springer, 2020. [Citado en la p. 6.]

16 Manu Drijvers, Kasra Edalatnejad, Bryan Ford, Eike Kiltz, Julian Loss, Gregory Neven e Igors Stepanovs. Sobre la seguridad de las multifirmas de dos rondas. En IEEE Symposium on Security and Privacy, páginas 1084-1101. IEEE, 2019. [Citado en la p. 6.]

17 Cynthia Dwork, Nancy Lynch y Larry Stockmeyer. Consenso en presencia de sincronía parcial. https://groups.csail.mit.edu/tds/papers/Lynch/jacm88.pdf. [Citado en la p. 5.]

18 Rosario Gennaro y Steven Goldfeder. Un umbral de ronda ecdsa con aborto identificable. Cryptology ePrint Archive, Report 2020/540, 2020. https://eprint.iacr.org/2020/540. [Citado en la p. 6.]

19 Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos y Nickolai Zeldovich. Algorand: Escalando acuerdos bizantinos para criptomonedas. Actas del 26º Simposio sobre Principios de Sistemas Operativos, 2017. https://dl.acm.org/doi/pdf/10.1145/3132747.3132757. [Citado en la p. 1.]

20 Evan Kereiakes, Do Kwon, Marco Di Maggio y Nicholas Platias. Terra money: Estabilidad y adopción. https://terra.money/Terra_White_paper.pdf. [Citado en la p. 1.]

21 Aggelos Kiayias, Alexander Russell, Bernardo David y Roman Oliynykov. Ouroboros: Un protocolo de blockchain de prueba de apuestas con seguridad demostrable. https://eprint.iacr.org/2016/889.pdf. [Citado en la p. 1.]

22 Chelsea Komlo e Ian Goldberg. Frost: Firmas de umbral de schnorr flexibles y optimizadas para la ronda. Cryptology ePrint Archive, Report 2020/852, 2020. https://eprint.iacr.org/2020/852. [Citado en la p. 6.]

23 Jae Kwon y Ethan Buchman. Cosmos: una red de libros de contabilidad distribuida. https://cosmos.network/resources/whitepaper. [Citado en las páginas 1 y 2.]

24 Equipo Avalancha. Plataforma Avalancha. https://www.avalabs.org/whitepapers. [Citado en las páginas 1 y 2.]

25 Gavin Wood. Polkadot: visión de un marco multicadena heterogéneo. https://polkadot.network/PolkaDotPaper.pdf. [Citado en las páginas 1 y 2.]