Rete Axelar:
Collegamento di applicazioni con ecosistemi Blockchain

Bozza 1.0

gennaio 2021

Astratto

Stanno emergendo molteplici ecosistemi blockchain che forniscono funzionalità uniche e distinte attraenti per utenti e sviluppatori di applicazioni. Tuttavia, la comunicazione tra gli ecosistemi è molto scarsa e frammentata. Per consentire alle applicazioni di comunicare senza attriti attraverso gli ecosistemi blockchain, proponiamo Axelar. Lo stack Axelar fornisce una rete decentralizzata, protocolli, strumenti e API che consentono una semplice comunicazione cross-chain. La suite di protocolli Axelar è composta da protocolli di routing e trasferimento transfrontalieri. Una rete aperta decentralizzata di validatori alimenta la rete; chiunque può iscriversi, usarlo e partecipare. Consenso bizantino, crittografia e meccanismi di incentivazione sono progettati per ottenere requisiti di sicurezza e vitalità elevati unici per le richieste cross-chain.

1. Introduzione

I sistemi blockchain stanno rapidamente guadagnando popolarità e attirano nuovi casi d'uso per la tokenizzazione delle risorse, la finanza decentralizzata e altre applicazioni distribuite. Diverse piattaforme importanti come Ethereum, Monero, EOS, Cardano, Terra, Cosmos, Avalanche, Algorand, Near, Celo e Polkadot offrono caratteristiche e ambienti di sviluppo distinti che li rendono attraenti per diverse applicazioni, casi d'uso e utenti finali [5, 11, 4, 21, 20, 23, 24, 19, 6, 14, 25]. Tuttavia, le funzionalità utili di ogni nuova piattaforma sono attualmente offerte a meno dell'1% degli utenti dell'ecosistema, ovvero i possessori del token nativo su quella piattaforma. Possiamo consentire agli sviluppatori di piattaforme di collegare facilmente i loro blockchain ad altri ecosistemi? Possiamo consentire ai costruttori di applicazioni di costruire sulla piattaforma migliore per le loro esigenze pur continuando a comunicare tra più ecosistemi blockchain? Possiamo consentire agli utenti di interagire con qualsiasi applicazione su qualsiasi blockchain direttamente dai loro portafogli?

Per collegare gli ecosistemi blockchain e consentire alle applicazioni di comunicare senza attriti attraverso di essi, proponiamo la rete Axelar. I validatori gestiscono collettivamente un protocollo di consenso bizantino ed eseguono i protocolli facilitando le richieste cross-chain. Chiunque può entrare a far parte della rete, partecipare e utilizzarla. La rete sottostante è ottimizzata per elevati requisiti di sicurezza e vitalità unici per le richieste cross-chain. La rete Axelar include anche una suite di protocolli e API. I protocolli fondamentali sono:

- Cross-Chain Gateway Protocol (CGP). Questo protocollo è analogo al Border Gateway Protocol su Internet. Questo protocollo viene utilizzato per connettere più ecosistemi blockchain autonomi ed è responsabile del routing attraverso di essi. Le blockchain non hanno bisogno di "parlare un linguaggio personalizzato", i loro sviluppatori di piattaforme non hanno bisogno di apportare modifiche personalizzate alle loro catene e le loro catene possono essere facilmente collegate alla rete globale.

- Cross-Chain Transfer Protocol (CTP). Questo protocollo è analogo ai protocolli a livello di applicazione File Transfer, Hypertext Transfer Protocol su Internet. È uno stack di protocolli a livello di applicazione che si trova in cima ai protocolli di routing (come CGP e altre tecnologie di routing). Gli sviluppatori di applicazioni possono connettere i loro dapp su qualsiasi catena per eseguire richieste cross-chain. Gli utenti possono utilizzare il protocollo CTP per interagire con le applicazioni su qualsiasi catena utilizzando semplici chiamate API analoghe alle richieste HTTP GET/POST. Gli sviluppatori possono bloccare, sbloccare e trasferire risorse tra due indirizzi qualsiasi su qualsiasi piattaforma blockchain, eseguire trigger di applicazioni cross-chain (ad esempio, un dapps sulla catena A, può aggiornare

il suo stato se qualche altra applicazione sulla catena B soddisfa alcuni criteri di ricerca (tasso di interesse $> X$)) ed eseguire richieste cross-chain generali tra app tra catene (uno smart contract sulla catena A può chiamare per aggiornare uno stato di uno smart contract sulla catena B). Questo protocollo consente la componibilità dei programmi negli ecosistemi blockchain.

La rete Axelar offre i seguenti vantaggi:

- *Per i costruttori di piattaforme blockchain*: Possibilità di collegare facilmente i loro blockchain a tutti gli altri ecosistemi blockchain. Solo un account di soglia deve essere impostato sulla catena per collegarsi alla rete.
- *Per i costruttori di dapps*: i costruttori di applicazioni possono ospitare le loro dapp ovunque, bloccare, sbloccare, trasferire risorse e comunicare con le applicazioni su qualsiasi altra catena tramite l'API CTP.
- *Per gli utenti*: gli utenti possono interagire con tutte le applicazioni nell'ecosistema direttamente dai loro portafogli.

Una piattaforma per costruttori. Infine, la rete Axelar è una piattaforma per sviluppatori e una comunità globale. Il suo modello di governance è aperto a chiunque. Gli sviluppatori possono proporre nuovi punti di integrazione, routing e protocolli a livello di applicazione e gli utenti possono decidere se adottarli votando le proposte e, se approvati, i validatori adotteranno le modifiche.

1.1 Soluzioni di interoperabilità esistenti

I precedenti tentativi di risolvere l'interoperabilità tra blockchain rientrano in una delle quattro categorie: scambi centralizzati, ecosistemi interoperabili, risorse avvolte e token bridge. Riassumiamo brevemente questi approcci di seguito.

Sistemi centralizzati. esigenze dell'ecosistema. Oggi i sistemi centralizzati sono le uniche soluzioni veramente scalabili per l'interoperabilità Possono elencare qualsiasi risorsa o integrare qualsiasi piattaforma in modo relativamente semplice. Tuttavia, è noto che i sistemi centralizzati presentano vari problemi di sicurezza e non sono abbastanza validi per alimentare il sistema finanziario decentralizzato emergente che richiede una solida sicurezza, trasparenza e governance aperta. Da soli non possono alimentare le applicazioni decentralizzate man mano che crescono.

Hub di interoperabilità. Progetti come Cosmos, Polkadot, Ava Labs affrontano l'interoperabilità tra *catene laterali* nativi dei loro ecosistemi utilizzando protocolli di comunicazione inter-catena personalizzati [23, 25, 24]. Ad esempio, è possibile avviare una catena laterale (una Cosmos Zone) in grado di comunicare con Cosmos Hub. La sidechain deve basarsi sul consenso di Tendermint e parlare il protocollo nativamente compreso dal Cosmos Hub. Le connessioni ad altri blockchain ed ecosistemi che parlano lingue diverse sono lasciate a tecnologie esterne.

Ponti a coppie. Gli asset avvolti (ad es. Bitcoin avvolti) cercano di colmare il divario di interoperabilità cross-chain mancante nell'ecosistema. Un esempio è tBTC [9], che è un protocollo personalizzato in cui una combinazione intelligente di contratti intelligenti e garanzie viene utilizzata per proteggere i trasferimenti. Queste soluzioni richiedono notevoli sforzi di progettazione: per ogni coppia di catene, gli sviluppatori devono creare un nuovo contratto intelligente sulla catena di destinazione che analizzi le prove di stato dalla catena di origine (simile a come ciascuna catena laterale potrebbe, in linea di principio, analizzare lo stato di altre catene) . Solo una manciata di bridge è stata implementata utilizzando questo approccio. Questi approcci non sono scalabili quando una delle blockchain sottostanti vuole aggiornare le sue regole di consenso o il formato della transazione. Questo perché tutti i contratti intelligenti che dipendono dallo stato di queste catene dovrebbero essere aggiornati. È inoltre necessario impostare validatori e richiedere loro di bloccare diverse attività al fine di garantire un'eccessiva garanzia di qualsiasi trasferimento di attività,

Abbiamo anche visto alcuni altri bridge monouso da parte di sviluppatori di piattaforme che riscrivono la logica di transizione dello stato nei contratti intelligenti per collegarsi ad altri ecosistemi [1, 7]. Soffrono di molteplici problemi di scalabilità, non consentono all'ecosistema di scalare in modo uniforme e introducono dipendenze aggiuntive per le applicazioni. Ad esempio, se una piattaforma cambia, tutti i contratti intelligenti su tutti i bridge dovranno essere aggiornati. Questo

alla fine metterà l'ecosistema in una situazione di stallo in cui nessuno sarà in grado di aggiornare. Infine, se un bridge monouso collega le piattaforme A e B e un secondo bridge monouso collega B e C, ciò non significa che le applicazioni su A saranno in grado di comunicare con le applicazioni su C. Potrebbe essere necessario creare un altro bridge single-purpose scopo bridge o ricablare la logica dell'applicazione.

Altri tentativi di affrontare l'interoperabilità includono oracoli federati (ad esempio, Ren [8]), e blockchain interoperabili specifiche dell'applicazione [10].

Per riassumere, le soluzioni esistenti per l'interoperabilità richiedono un intenso lavoro di progettazione sia da parte degli sviluppatori di piattaforme che dei costruttori di applicazioni che devono comprendere protocolli di comunicazione diversi per comunicare attraverso ogni coppia di ecosistemi. E quindi, l'interoperabilità è praticamente inesistente nello spazio blockchain di oggi. Alla fine della giornata, gli sviluppatori di piattaforme vogliono concentrarsi sulla creazione di piattaforme e ottimizzarle per i loro casi d'uso ed essere in grado di collegarsi facilmente ad altri blockchain. E gli sviluppatori di applicazioni vogliono creare dapp sulle piattaforme migliori per le loro esigenze, sfruttando al contempo utenti, liquidità e comunicare con altri dapp su altre catene.

2 La ricerca di una comunicazione cross-chain scalabile

Al centro, la comunicazione cross-chain richiede che le reti eterogenee trovino la capacità di comunicare usando la stessa lingua. Per risolvere questo problema, spieghiamo la suite di protocolli Axelar, ne descriviamo le proprietà di alto livello e spieghiamo come queste proprietà affrontano il nucleo della comunicazione scalabile a catena incrociata.

1. *“Integrazione “plug-and-play”.* Ai costruttori di piattaforme Blockchain non dovrebbe essere richiesto di eseguire pesanti lavori di ingegneria o integrazione per parlare un "linguaggio personalizzato" per supportare la catena incrociata. Il protocollo crosschain dovrebbe essere in grado di collegare qualsiasi blockchain esistente o nuovo senza attriti. Nuove risorse dovrebbero essere aggiunte con il minimo sforzo.
2. *Instradamento a catena incrociata.* Funzioni come il rilevamento di indirizzi di rete, percorsi di instradamento e reti sono al centro di Internet e facilitate da BGP e altri protocolli di instradamento. Allo stesso modo, per facilitare la comunicazione tra gli ecosistemi blockchain, dobbiamo supportare la scoperta di indirizzi attraverso di essi, applicazioni e routing.
3. *Supporto per l'aggiornabilità.* Se uno degli ecosistemi blockchain cambia, non dovrebbe influire sull'interoperabilità di altre blockchain. Il sistema deve riconoscere gli aggiornamenti e dovrebbe essere richiesto uno sforzo minimo per supportarli (ad esempio, nessuna "logica di transizione dello stato" dovrebbe essere riscritta e le applicazioni non dovrebbero interrompersi).
4. *Linguaggio uniforme per le applicazioni.* Le applicazioni necessitano di un protocollo semplice per bloccare, sbloccare, trasferire e comunicare con altre applicazioni, indipendentemente dalla catena in cui risiedono. Questo protocollo deve essere indipendente dalla catena e supportare chiamate semplici, analoghe ai protocolli HTTP/HTTPS che consentono a utenti e browser di comunicare con qualsiasi server web. Man mano che più reti e risorse si uniscono ai protocolli di routing di livello inferiore, le applicazioni dovrebbero essere in grado di utilizzarli per le comunicazioni senza riscrivere i loro stack software.

Successivamente, riassumiamo i requisiti di sicurezza che questi protocolli devono soddisfare.

1. *Fiducia decentralizzata.* La rete e i protocolli devono essere decentralizzati, aperti e consentire a tutti di partecipare in modo equo.
2. *Alta sicurezza.* Il sistema deve soddisfare elevate garanzie di sicurezza. Il sistema deve preservare la sicurezza delle risorse e dello stato mentre la rete a catena incrociata lo elabora.
3. *Elevata vivacità.* Il sistema deve soddisfare elevate garanzie di vivacità per supportare le applicazioni sfruttandone le caratteristiche cross-chain.

Soddisfare un sottoinsieme di queste proprietà è facile. Ad esempio, è possibile creare un account multisig federato con i propri amici e bloccare/sbloccare risorse sulle catene corrispondenti. Tali sistemi sono intrinsecamente vulnerabili agli attacchi di collusione e censura e mancano di incentivi adeguati affinché i validatori li proteggano. La creazione di una rete decentralizzata e di una suite di protocolli in cui chiunque possa partecipare pur essendo adeguatamente incentivato può consentire una comunicazione cross-chain senza attriti, ma risolverlo è un problema difficile che richiede un'attenta combinazione di consenso, crittografia e protocolli di progettazione del meccanismo.

3 Rete Axelar

La rete Axelar fornisce una soluzione uniforme per la comunicazione cross-chain che soddisfa le esigenze sia degli sviluppatori di piattaforme (non è richiesto alcun lavoro di integrazione da parte loro, sia dei costruttori di applicazioni): un semplice protocollo e un'API per accedere alla liquidità globale e comunicare con l'intero ecosistema.

La rete Axelar è costituita da una rete decentralizzata che collega ecosistemi blockchain che parlano lingue diverse e una suite di protocolli con API in cima, rendendo facile per le applicazioni eseguire richieste cross-chain. La rete collega blockchain indipendenti esistenti come Bitcoin, Stellar, Terra, Algorand e hub di interoperabilità come soluzioni come Cosmos, Avalanche, Ethereum e Polkadot. La nostra missione è consentire agli sviluppatori di applicazioni di creare tali app più facilmente utilizzando un protocollo e un'API universali senza implementare i loro protocolli cross-chain proprietari sotto o riscrivere le applicazioni man mano che vengono sviluppati nuovi bridge. A tal fine, abbiamo progettato una suite di protocolli che include il protocollo Cross-Chain Gateway (vedi Sezione6) e Cross-Chain Transfer Protocol (cfr 7).

Un componente fondamentale della rete sono i protocolli decentralizzati sottostanti. I validatori gestiscono collettivamente la rete Axelar ed eseguono i nodi che proteggono la blockchain di Axelar. Sono eletti attraverso un processo di delega dagli utenti. I validatori ricevono il potere di voto pro-quota in base alla quota loro delegata. I validatori raggiungono il consenso sullo stato di più blockchain a cui è connessa la piattaforma. La blockchain è responsabile del mantenimento e dell'esecuzione dei protocolli di trasferimento e routing cross-chain. Le regole di governance consentono ai partecipanti alla rete di adottare decisioni di protocollo come quali blockchain collegare e quali risorse supportare.

La blockchain di Axelar segue un modello Delegato Proof-of-Stake (DPoS) simile a Cosmos Hub. Gli utenti eleggono validatori che devono vincolare la loro partecipazione per partecipare al consenso e mantenere un servizio di alta qualità. Il modello DPoS consente il mantenimento di un ampio set di validatori decentralizzati e solidi incentivi per garantire che i validatori siano responsabili del mantenimento di ponti e condivisioni di schemi di soglia crittografica. Come parte del consenso, i validatori eseguono software light-client di altre blockchain, consentendo loro di verificare lo stato di altre blockchain. I validatori segnalano questi stati alla blockchain di Axelar e, una volta che un numero sufficiente di essi riportano, lo stato di Bitcoin, Ethereum e altre catene viene registrato su Axelar.

Successivamente, il livello base di Axelar è a conoscenza dello stato delle blockchain esterne in qualsiasi momento, creando i “ponti in entrata” da altre blockchain. I validatori mantengono collettivamente*account di firma di soglia* su altre blockchain (ad esempio, l'80% dei validatori deve approvare e firmare insieme qualsiasi transazione al di fuori di essa), il che consente loro di bloccare e sbloccare asset e stato attraverso le catene e di pubblicare lo stato su altre blockchain, i "ponti in uscita". Complessivamente, è possibile visualizzare la rete Axelar come a*Oracle decentralizzato di lettura/scrittura a catena incrociata*.

Il resto del documento descrive i preliminari e gli elementi costitutivi della rete (Sezione 4), alcuni dettagli tecnici della rete (Sez 5), protocollo gateway cross-chain (Sezione 6), e il protocollo di trasferimento cross-chain (Sezione 7).

4 preliminari

4.1 Notazione e ipotesi

Permettere v_R denotare l'insieme dei validatori Axelar a round R. Ogni validatore ha a *peso*, un numero in $(0, 1]$ che denota il potere di voto di quel particolare validatore. I pesi di tutti i validatori sommano fino a 1. Un validatore è *corretta* se esegue un nodo che è coerente con le regole del protocollo Axelar. Per finalizzare blocchi, o per firmare richieste cross-chain, Axelar richiede validatori di peso totale corretti $> F$. Chiamiamo parametro $F \in [0.5, 1]$ il *soglia del protocollo*.

Axelar può essere basato su un *finalità istantanea Delega-Proof-of-Stake* blockchain. I validatori corrono *Consenso Bizantino Fault Tolerant (BFT).* ad ogni round io per finalizzare il io_{th} bloccare. Una volta che il io_{th} blocco è finalizzato, viene eseguito un nuovo consenso BFT per finalizzare il $io + 1_{th}$ blocco e così via. I validatori sono eletti tramite delegazione di palo. Un utente con una partecipazione può scegliere di eseguire un nodo validatore o delegare il proprio potere di voto (partecipazione) a un validatore esistente, che quindi vota per suo conto. Il set di validatori può essere aggiornato, i validatori si uniscono/lasciano il set e gli utenti delegano/delegano il proprio potere di voto.

Blockchain differenti funzionano in base a diverse ipotesi di rete. *Comunicazione sincrona* significa che c'è un limite superiore fisso Δ sul tempo impiegato dai messaggi per essere consegnati, dove Δ è noto e può essere integrato nel protocollo. *Comunicazione asincrona* significa che i messaggi possono richiedere molto tempo per essere consegnati ed è noto che i protocolli BFT non possono essere creati per reti asincrone anche in presenza di un solo validatore dannoso. Si presuppone un compromesso realistico tra sincronia e asincronia *comunicazione parzialmente sincrona*. La rete può essere completamente asincrona fino a un tempo di stabilizzazione globale sconosciuto (GST), ma dopo che la comunicazione GST diventa sincrona con un limite superiore noto Δ [17].

I blockchain tipici funzionano presupponendo $> F$ validatori corretti. Per reti sincrone $F = 1/2$ è tipicamente impostato, ma per l'ipotesi più debole di una rete parzialmente sincrona $F = 2/3$. Bitcoin, i suoi fork e l'attuale versione Proof-of-Work di Ethereum funzionano solo presupponendo la sincronia. Altri come Algorand e Cosmos richiedono solo una sincronia parziale. Quando si collegano le catene tramite Axelar, la connessione funziona assumendo le ipotesi di rete più forti di queste catene, che è la sincronia nel caso della connessione di Bitcoin e Cosmos, ad esempio. La stessa blockchain di Axelar funziona in un ambiente parzialmente sincrono e quindi richiede $F = 2/3$, ma è possibile migliorare il requisito di soglia presupponendo che altre blockchain esistenti siano sicure e sfruttando la loro sicurezza.

4.2 Preliminari crittografici

Firme digitali. UN *schema di firma digitale* è una tupla di algoritmi (*Keygen, firma, verifica*). *Keygen* emette una coppia di chiavi (*PK, SK*). Solo il proprietario di *SK* può firmare messaggi, ma chiunque può verificare le firme data la chiave pubblica *PK*. La maggior parte dei sistemi blockchain oggi utilizza uno degli schemi di firma standard come ECDSA, Ed25519 o alcune delle loro varianti [2, 3].

Firme di soglia. UN *schema di firma di soglia* abilita un gruppo di n parti di dividere una chiave segreta per uno schema di firma in modo tale che qualsiasi sottoinsieme di $T + 1$ o più parti possono collaborare per produrre una firma, ma nessun sottoinsieme di T o meno parti possono produrre una firma o persino apprendere qualsiasi informazione sulla chiave segreta. Le firme prodotte dai protocolli di soglia per ECDSA ed EdDSA sembrano identiche alle firme prodotte dagli algoritmi standalone.

Uno schema di firma di soglia sostituisce il *Keygen* e *Cartello* algoritmi per uno schema di firma ordinaria con distribuzione n-protocolli di partito T.*Keygen*, T.*Segno*. Questi protocolli in genere richiedono sia un canale di trasmissione pubblico che canali privati a coppie tra le parti e in genere implicano diversi round di comunicazione. Dopo il completamento con successo di T.*Keygen* ogni utente detiene una condivisione s_{io} di una chiave segreta SK e della corrispondente chiave pubblica PK. Il T.*Segno* protocollo consente a queste parti di produrre una firma per a

messaggio dato che è valido sotto chiave pubblica PK. Questa firma può essere verificata da chiunque utilizzi il *Verificare* algoritmo dello schema di firma originale.

4.3 Proprietà delle firme di soglia

Esistono diverse proprietà che uno schema di soglia potrebbe avere particolarmente desiderabili per le reti decentralizzate:

Sicurezza contro una maggioranza disonesta. Alcuni schemi di soglia hanno la restrizione di essere sicuri solo quando la maggioranza del n le parti sono oneste. Quindi, il parametro di soglia T deve essere inferiore a $n/2$ [15]. Questa restrizione è tipicamente accompagnata dal fatto che $2T+$ Per firmare occorrono 1 parti oneste, anche se sole $T+1$ parti danneggiate possono collaborare per recuperare la chiave segreta. Si dice che gli schemi che non soffrono di questa restrizione lo siano *sicuro contro una maggioranza disonesta*.

Come discusso più avanti nella Sezione 5.2, le piattaforme cross-chain devono massimizzare la sicurezza delle loro reti ed essere in grado di tollerare il maggior numero possibile di parti corrotte. Pertanto, sono necessari schemi che possano tollerare la maggioranza disonesta.

Pre-firme, firma online non interattiva. Nel tentativo di ridurre il carico di comunicazione al momento della firma di un messaggio da parte delle parti, diversi protocolli recenti hanno individuato una parte significativa del lavoro per una firma che può essere eseguita "offline", prima che il messaggio da firmare sia noto [18, 13]. L'output di questa fase offline è chiamato a *pre-firma*. La produzione delle firme preliminari è vista come un protocollo separato $T.Presign$ distinto da $T.Keygen$ e $T.Segno$. Gli output del protocollo di pre-firma devono essere mantenuti privati dalle parti fino al loro utilizzo in fase di sottoscrizione. Successivamente, quando il messaggio da firmare diventa noto, resta solo una piccola quantità di lavoro "online" aggiuntivo da svolgere $T.Segno$ per completare la firma.

L'online $T.Segno$ fase non richiede alcuna comunicazione tra le parti. Ciascuna parte esegue semplicemente un calcolo locale sul messaggio e sulla pre-firma, quindi annuncia la sua quota S_{io} della firma. (Una volta pubbliche, queste firme condividono $S_1, \ldots, S_{T+1}$ sono facilmente combinati da chiunque per rivelare la firma effettiva S.) Questa proprietà è chiamata *firma online non interattiva*.

Robustezza. Gli schemi di soglia garantiscono solo che un sottoinsieme di soggetti malintenzionati non possa firmare messaggi o impara la chiave segreta. Questa garanzia, tuttavia, non preclude la possibilità che malintenzionati possano impedire a tutti gli altri di produrre chiavi o firme. In alcuni schemi, può causare comportamenti dannosi anche da parte di una singola parte $T.Keygen$ o $T.Segno$ per interrompere senza output utile. L'unico rimedio è riavviare il protocollo, possibilmente con soggetti diversi.

Invece, per le reti decentralizzate, vogliamo $T.Keygen$ e $T.Segno$ avere successo se almeno $T+1$ delle parti è onesto, anche se alcune parti malintenzionate inviano messaggi in formato non corretto o rilasciano messaggi nei protocolli. Questa proprietà è chiamata *robustezza*.

Attribuzione del difetto. La capacità di identificare i cattivi attori $T.Keygen$ o $T.Segno$ è chiamato *attribuzione di colpa*. Senza l'attribuzione della colpa è difficile escludere o punire in modo affidabile i cattivi attori, nel qual caso i costi imposti dai cattivi attori devono essere a carico di tutti. Questa proprietà è importante anche per le reti decentralizzate in cui il comportamento dannoso dovrebbe essere identificabile ed economicamente disincentivato tramite regole di taglio.

Sicurezza nelle impostazioni simultanee. Lo schema di firma deve essere sicuro in un ambiente simultaneo, dove più istanze di keygen e algoritmi di firma possono essere coinvolte in parallelo. (Drijvers et al. [16] ad esempio, ha mostrato un attacco contro gli schemi multifirma Schnorr in queste impostazioni). Esistono versioni degli schemi ECDSA e Schnorr che soddisfano queste proprietà [13, 22].

ECDSA ed EdDSA sono di gran lunga gli schemi di firma più diffusi nello spazio blockchain. In quanto tali, le versioni soglia di entrambi i regimi sono state al centro di una recente rinascita nella ricerca e sviluppo. I lettori interessati allo stato dell'arte possono fare riferimento a [22, 13, 18] e un recente documento di indagine [12].

Rete 5 Axelar

5.1 Progettazione di una rete a catena aperta

I bridge gestiti dalla rete Axelar sono supportati da account di soglia tali che (quasi) tutti i validatori devono autorizzare collettivamente qualsiasi richiesta cross-chain. La progettazione di una rete in cui chiunque possa partecipare per proteggere questi bridge richiede il rispetto dei seguenti requisiti tecnici:

- *Abbonamento aperto.* Qualsiasi utente dovrebbe essere in grado di diventare un validatore (seguendo le regole della rete).
- *Aggiornamenti sull'abbonamento.* Quando un validatore lascia il sistema onestamente, la sua chiave deve essere revocata in modo appropriato.
- *Incentivi e tagli.* I validatori dannosi dovrebbero essere identificabili e le loro azioni devono essere identificate e affrontate dal protocollo.
- *Consenso.* Gli schemi di soglia da soli sono definiti come protocolli autonomi. Per propagare i messaggi tra i nodi abbiamo bisogno di canali privati sia broadcast che point-to-point. Inoltre, i validatori devono concordare lo stato più recente di ciascuna chiamata di schemi di soglia poiché spesso hanno più cicli di interazioni.
- *Gestione delle chiavi.* Proprio come i validatori ordinari in qualsiasi sistema PoS devono custodire con cura le proprie chiavi, così anche i validatori Axelar devono custodire le proprie quote di soglia. Le chiavi devono essere ruotate, divise tra parti online e offline, ecc.

Axelar inizia con il modello Proof-of-Stake delegato, in cui la comunità elegge una serie di validatori per gestire il consenso. Si noti che gli schemi di soglia standard trattano ogni giocatore in modo identico e non hanno nozione di "peso" nel consenso. Pertanto, la rete deve adattarli per tenere conto del peso dei validatori. Un approccio semplice consiste nell'assegnare più condivisioni di soglia a validatori più grandi. Di seguito sono descritte tre funzioni di base che i validatori svolgono collettivamente.

- *Generazione di chiavi di soglia.* Gli algoritmi di generazione di chiavi di soglia esistenti per gli schemi di firma blockchain standard (ECDSA, Ed25519) sono protocolli interattivi tra più partecipanti (vedere la sezione 4). Una transazione speciale sulla rete Axelar ordina ai validatori di iniziare l'esecuzione di questo protocollo stateful. Ogni validatore esegue un processo daemon di soglia che è responsabile della conservazione sicura dello stato segreto. Per ogni fase del protocollo:

  1. Un validatore conserva lo stato del protocollo nella sua memoria locale.
  2. Chiama il demone segreto per generare i messaggi secondo la descrizione del protocollo per altri validatori.
  3. Propaga i messaggi tramite la trasmissione o tramite i canali privati ad altri validatori.
  4. Ciascun validatore esegue funzioni di transizione di stato per aggiornare il proprio stato, procedere alla fase successiva del protocollo e ripetere i passaggi precedenti.

  Al termine del protocollo, sulla catena Axelar viene generata una chiave pubblica di soglia, che può essere visualizzata nuovamente all'utente (es. per i depositi) o all'applicazione che ha generato la richiesta iniziale.

- *Firma della soglia.* Le richieste di firma sulla rete Axelar vengono elaborate in modo simile alle richieste di generazione delle chiavi. Questi vengono invocati, ad esempio, quando un utente vuole ritirare una risorsa da uno dei Catene. Questi sono protocolli interattivi e la transizione di stato tra i round viene attivata come funzione dei messaggi propagati tramite la vista blockchain di Axelar e la memoria locale di ogni validatore.

- *Gestione delle modifiche all'appartenenza a Validator.* Il set di validatori deve essere ruotato periodicamente per consentire a nuovi stakeholder di unirsi al set. Dopo un aggiornamento del set di validatori, è necessario aggiornare la chiave di soglia da condividere nel nuovo set. Pertanto, se consentiamo a qualcuno di partecipare in qualsiasi momento, dovremmo aggiornare la chiave di soglia molto frequentemente. Per evitare ciò, ruotiamo i validatori ogni T blocchi. Entro intervalli di T tondi, il set v_R e la chiave di soglia sono fisse. Ad ogni giro che è un multiplo intero del parametro T, aggiorniamo il set di validatori come segue:

1. In qualsiasi round R, lo stato Axelar tiene traccia del set di validatori corrente v_R. $v_{R+1} = v_R$ salvo che $R + 1$ è un multiplo di T.
2. Durante i round $((io - 1)T, iT]$, gli utenti pubblicano messaggi di collegamento/svincolo.
3. Alla fine del round *esso*, questi messaggi vengono applicati v_{iT-1} ottenere v_{esso}.

- *Generazione di chiavi di soglia e firma in presenza di validatori rotanti.* La blockchain di Axelar può emettere una richiesta per una nuova chiave o una firma di soglia a round R. Il processo di firma richiede più di un round e non vogliamo rallentare il consenso, quindi chiediamo che la firma venga prodotta prima del round $R + 10$ inizi. In particolare, i validatori iniziano il round $R + 10$ solo dopo aver visto un certificato per il round $R + 9$ e una firma per ogni keygen/richiesta di firma rilasciata a round R. Il risultato di tutto R le richieste devono essere incluse nel blocco $R + 11$. In altre parole, un giro R proposta di blocco che non contiene gli esiti di un round R-11 è considerato non valido e i validatori non lo votano. Per garantire che tutti i messaggi di soglia siano firmati prima di un aggiornamento del set di validatori, Axelar non emette richieste di soglia durante un round pari a $-1, -2, \ldots, -9$ modulo T.

5.2 Sicurezza della rete

La sicurezza dei sistemi blockchain si basa su vari protocolli crittografici e teorici dei giochi, nonché sul decentramento della rete. Ad esempio, nelle blockchain proof-of-stake, senza gli opportuni incentivi i validatori possono colludere e riscrivere la cronologia, rubando i fondi di altri utenti nel processo. Nelle reti proof-of-work, senza un decentramento sufficiente, è abbastanza facile creare fork lunghi e doppia spesa, come hanno dimostrato i molteplici attacchi a Bitcoin Gold ed Ethereum Classic.

La maggior parte della ricerca sulla sicurezza blockchain si è concentrata sulle catene sovrane. Ma una volta che le catene interagiscono, devono essere presi in considerazione nuovi vettori di attacco. Ad esempio, supponiamo che Ethereum parli con una piccola blockchain X attraverso un bridge diretto controllato da due smart contract, uno su Ethereum e uno su X. Oltre alle sfide ingegneristiche che abbiamo riassunto nella Sezione1.1, si deve decidere cosa succede quando le ipotesi di fiducia di X vengono violate. In questo caso, se ETH si è spostato su X, i validatori di X possono colludere per creare una cronologia di X in cui detengono tutti gli ETH, pubblicare le prove di consenso contraffatte su Ethereum e rubare l'ETH. La situazione è ancora peggiore quando X è connesso con più altre catene attraverso ponti diretti, dove se X si biforca gli effetti si propagano attraverso ogni ponte. L'impostazione delle linee guida di governance del ripristino per ogni bridge a coppie è un compito arduo per ogni singolo progetto.

La rete Axelar risolve i problemi di sicurezza utilizzando i seguenti meccanismi:

- *Massima sicurezza.* Axelar imposta la soglia di sicurezza al 90%, il che significa che quasi tutti i validatori dovranno collaborare per prelevare fondi bloccati dalla sua rete o falsificare prove di stato[1]. In pratica, è stato osservato che i validatori PoS hanno tempi di attività molto elevati (vicino al 100%), supponendo che siano adeguatamente incentivati. Quindi, la rete Axelar produrrà blocchi anche nonostante questa soglia elevata. Tuttavia, nel raro caso in cui qualcosa vada storto e la rete si blocchi, la rete necessita di robusti meccanismi di fallback per riavviare il sistema descritto di seguito.
- *Massimo decentramento.* Poiché la rete utilizza schemi di firma di soglia, il numero di validatori può essere il più grande possibile. La rete non è limitata dal numero di validatori che possiamo supportare, dai limiti di transazione o dalle commissioni che deriverebbero dall'utilizzo, ad esempio, di firme multiple su catene diverse in cui la complessità (e le commissioni) aumentano linearmente con il numero di validatori.[2]
- *Meccanismi di ripiego robusti.* La prima domanda che deve essere affrontata in una rete con soglie di sicurezza elevate come sopra è cosa succede quando la rete stessa va in stallo. Supponiamo che la stessa rete Axelar sia in stallo. Possiamo avere un meccanismo di riserva che permetta agli utenti di recuperare i propri fondi? Per affrontare qualsiasi potenziale stallo della rete Axelar stessa, ogni account bridge di soglia su una blockchain X che i validatori Axelar controllano collettivamente ha una "chiave di sblocco di emergenza". Questa chiave può essere condivisa

[1] Il parametro finale che verrà scelto per l'implementazione della rete può essere modificato.

[2] Per alcuni blockchain, le firme multiple offrono un'alternativa ragionevole in cui i gas fess sono piccoli e i formati di messaggio supportati sono appropriati. Ma non si adattano a due delle piattaforme più grandi come Bitcoin ed Ethereum.

tra migliaia di parti e potrebbe anche essere una chiave personalizzata per blockchain X condivisa dalla comunità di quella catena. Quindi, se la rete Axelar si blocca, questa chiave fungerà da ripiego e consentirà il ripristino delle risorse (vedi sotto per maggiori dettagli).

- *Massimo decentramento dei meccanismi di fall-back.* Questo meccanismo di fallback include un secondario *set di recupero* di utenti, a cui chiunque può partecipare senza alcun costo. Questi utenti non devono essere online, eseguire nodi o coordinarsi tra loro. Sono "chiamati in servizio" solo se la rete Axelar si blocca e non riesce a riprendersi. La sicurezza della rete è rafforzata da una soglia molto alta sul set di validatori primari e da un set di ripristino secondario decentralizzato al massimo.

- *Governo Condiviso.* Un protocollo comune governa la rete Axelar. Collettivamente, gli utenti possono votare su quale catena dovrebbe essere supportata attraverso la sua rete. La rete destinerà inoltre un pool di fondi che potranno essere utilizzati per rimborsare gli utenti in caso di emergenze impreviste, controllate anche attraverso i protocolli di governance.

Di seguito vengono discussi vari meccanismi di sicurezza.

Meccanismi di ricaduta. Quando Axelar va in stallo a causa della soglia alta, una "chiave di sblocco di emergenza" prende il controllo della rete. Esistono diversi modi per creare un'istanza di questa chiave di sblocco e alcune catene/applicazioni possono scegliere di utilizzare una variazione diversa per il "set di ripristino" o annullare completamente:[3]

- *Opzione a.* Condividi la chiave tra le fondamenta dei progetti blockchain e le persone rispettabili nella comunità.

- *Opzione b.* Condividi i partiti eletti attraverso il meccanismo PoS delegato.

- *Opzione c.* Per gli account che gestiscono le risorse e le informazioni per la catena/applicazione X, condividi una chiave personalizzata tra le parti interessate/validatori di X. Supponendo che X disponga di meccanismi di governance in atto, gli stessi meccanismi di governance possono essere applicati per determinare una linea d'azione se Axelar si blocca.

Ora, date le identità degli utenti di ripristino e le loro chiavi pubbliche, un semplice protocollo genera condivisioni della chiave di ripristino che nessuno conosce. Inoltre, gli utenti del set di ripristino non devono essere online fino a quando non vengono chiamati a recuperare tramite i meccanismi di governance. Seguendo i protocolli standard di generazione di chiavi distribuite, ogni validatore Axelar condivide un valore casuale. La chiave segreta di ripristino viene generata sommando questi valori. Invece di fare le somme in chiaro, tutte le condivisioni vengono crittografate con le chiavi pubbliche degli utenti di ripristino e quindi sommate in modo omomorfico (questo presuppone una crittografia additiva omomorfa e un ulteriore livello di conoscenza zero, entrambi facilmente ottenibili). Il risultato di questo protocollo è una chiave pubblica di ripristino *RPK* e potenzialmente migliaia di crittografie (sotto le chiavi pubbliche degli utenti di ripristino) delle condivisioni della chiave segreta corrispondente $Enc_{io}(S_{io})$ che vengono distribuiti ai rispettivi proprietari (ad es. affissi su catena). I contratti bridge Axelar includono un'opzione per recuperare i fondi utilizzando *RPK* a determinate condizioni. Infine, è anche possibile aggiornare questa chiave di ripristino e persino modificare l'insieme degli utenti che detengono le sue azioni senza richiedere alcun intervento da parte degli azionisti partecipanti.

Se la catena X collegata ad Axelar si interrompe, ci sono un paio di opzioni:

- Imporre limiti al valore in USD degli asset che possono essere spostati dentro/fuori X in un singolo giorno. Quindi una catena X dannosa può rubare solo una piccola parte di tutte le risorse che sono collegate ad essa prima che i validatori Axelar lo rilevino e i meccanismi di governance dei seguenti proiettili si attivino.

- Il modulo di governance di Axelar può essere utilizzato per votare su ciò che accade in quelle situazioni. Ad esempio, se è presente un bug benigno e la community riavvia X, la governance di Axelar può determinare di riavviare la connessione da dove era stata interrotta.

- Se ETH fosse passato a X, una chiave di ripristino Ethereum personalizzata può determinare cosa succede alle risorse ETH.

[3] L'implementazione finale sulla rete Axelar sarà finalizzata in prossimità del lancio della rete.

6 Protocollo gateway incrociato (CGP)

In questa sezione, spieghiamo il protocollo del gateway cross-chain e i meccanismi di routing su due esempi fondamentali comuni tra le esigenze di molte applicazioni:

Sincronizzazione degli stati (Sezione 6.2). Pubblica informazioni sullo stato di una blockchain di origine *S* dentro stato di una blockchain di destinazione *D*.

(Ad esempio, pubblica un'intestazione di blocco Bitcoin sulla blockchain di Ethereum.)

Trasferimento di asset (Sezione 6.3). Trasferisci una risorsa digitale da *S* a *D* e ritorno.

(Ad esempio, trasferisci bitcoin dalla blockchain di Bitcoin alla blockchain di Ethereum e poi di nuovo alla blockchain di Bitcoin.)

Per semplicità assumiamo quella catena *D* ha almeno un supporto minimo per i contratti intelligenti ma *S* può essere qualsiasi blockchain.

6.1 Conti su altre catene

Per collegare diverse catene, su ciascuna catena vengono creati account di soglia che controllano il flusso di valore e informazioni attraverso di esse. Per catena *Catena*, denota il conto di $Catena_{Axelar}$.

Conto Bitcoin. Per Bitcoin e altre catene di contratti non intelligenti, i validatori Axelar creano una chiave ECDSA soglia come da sezione 5.1. Questa chiave controlla l'account ECDSA su Bitcoin ed è l'indirizzo di destinazione a cui gli utenti inviano i depositi. È possibile creare chiavi soglia personalizzate per richiesta dell'utente. La chiave può essere aggiornata periodicamente e la chiave più recente e le chiavi personalizzate possono essere trovate interrogando un nodo Axelar.

Soglia conto ponte su catene con smart contract. Indichiamo la catena con SC. i validatori creano una chiave di soglia ECDSA o ED25519 come da sezione5.1, a seconda del tipo di chiave supportata dalla catena. Indichiamo questa chiave con PK_{Axelar}, quando non c'è ambiguità su quale catena ci riferiamo. Questa chiave controlla un account smart contract su SC, indicato con SC_{Axelar} e qualsiasi applicazione su SC può eseguire query SC_{Axelar} per apprendere l'indirizzo PK di quella chiave. In questo modo, qualsiasi applicazione SC può riconoscere i messaggi firmati da SK_{Axelar}. Il protocollo deve anche tenere conto dei valori rotanti di PK_{Axelar}. Ciò avviene come segue:

1. Inizializzare SC_{Axelar} su SC. Memorizza PK_{Axelar} come parte del suo stato, che viene inizializzato come valore di genesi su Axelar. SC_{Axelar} include anche le regole per l'aggiornamento del PK.
2. Per aggiornare PK_{Axelar}, una transazione del formato (*aggiornamento,* PK_{nuovo}) deve essere presentata con una firma dell'attuale SK_{Axelar}. Poi il contratto si stabilisce $PK_{Axelar} = PK_{nuovo}$.
3. Ogni volta che i validatori aggiornano la chiave di soglia per SC da PK_{io} a PK_{io+1}, Axelar richiede che i validatori utilizzino SK_{io} firmare (*aggiornamento,* PK_{io+1}). Successivamente questa firma viene affissa a SC_{Axelar} che aggiorna PK_{Axelar}.

6.2 Sincronizzazione degli stati

Permettere Q_S denota una domanda arbitraria sullo stato della catena *S*. Esempi di tali domande includono:

- *"In quale round di blocco, se presente, è apparsa una transazione tx?"*
- *"Qual è il valore di un determinato campo di dati?"*
- *"Qual è l'hash radice Merkle dell'intero stato di S al blocco round 314159?"*

Permettere un_S indicare la risposta corretta a Q_S e supponiamo che un utente finale o un'applicazione lo richieda un_S essere inviato alla catena *D*. La rete Axelar soddisfa questa richiesta come segue:

1. L'utente invia una richiesta Q_S su uno dei conti ponte (che successivamente vengono prelevati dai validatori) o direttamente sulla blockchain di Axelar.

2. Come parte del consenso Axelar, ogni validatore deve eseguire il software del nodo per le catene S, D. I validatori Axelar interrogano l'API della loro catena S software del nodo per la risposta un_S e riporta la risposta alla catena Axelar.

3. Una volta $> F$ i validatori pesati riportano la stessa risposta a round R, Axelar chiede ai validatori di firmare un_S.

4. Usando la crittografia a soglia i validatori firmano un_S. La firma è inclusa nel blocco $R + 11$.

5. Chiunque può prendere il valore firmato un_S dal blocco $R + 11$ e postalo su D.

6. La richiesta è stata evasa. Qualsiasi applicazione su D può ora assumere il valore con segno un_S, interrogare D_{Axelar} per l'ultimo PK_{Axelar}, e verificare che la firma di un_S corrisponde a PK_{Axelar}. Postano anche i validatori un_S al conto ponte sulla catena D, che le applicazioni possono recuperare.

6.3 Trasferimento di asset cross-chain

La rete consente trasferimenti cross-chain di risorse digitali estendendo il flusso di lavoro di sincronizzazione dello stato di Section 6.2.

Una fornitura sufficiente di pegged-S gettoni viene stampato e controllato da D_{Axelar} alla sua inizializzazione. Supponiamo che un utente richieda di scambiare X quantità di token sulla catena di origine S per X quantità di pegging-S token sulla catena di destinazione D, da depositare presso a D-indirizzo w_D a scelta dell'utente. Presentiamo il flusso di lavoro completamente generale, che supporta catene di sorgenti arbitrarie S—anche catene come Bitcoin che non supportano i contratti intelligenti:

1. L'utente (o un'applicazione che agisce per conto dell'utente) invia una richiesta di trasferimento (x, w_D) al conto ponte soglia che viene successivamente instradato alla rete Axelar.

2. I validatori Axelar utilizzano la crittografia a soglia per creare collettivamente un nuovo indirizzo di deposito D_S per S. Pubblicano D_S alla blockchain di Axelar.

3. L'utente (o un'applicazione che agisce per conto dell'utente) apprende D_S monitorando la blockchain di Axelar. L'utente invia X quantità di S-token da indirizzare D_S tramite un ordinario S-transazione tx_S usando il suo software preferito per la catena S.

   (A causa della proprietà di soglia di D_S, i gettoni non possono essere spesi da D_S a meno che un numero di soglia dei validatori non coordini per farlo.)

4. tx_S è pubblicato su Axelar. I validatori interrogano l'API della loro catena S software del nodo per l'esistenza di tx_S e, se la risposta è "vera", riportare la risposta alla catena Axelar.

5. Una volta $> F$ i validatori pesati riportano "vero" per tx_S a tondo R, Axelar chiede ai validatori di firmare una transazione un_D che invia X quantità di pegging-S gettoni da D_{Axelar} a w_D.

6. Usando la crittografia a soglia i validatori firmano un_D. La firma è inclusa nel blocco $R + 11$.

7. Chiunque può prendere il valore firmato un_D dal blocco $R + 11$ e postalo su D.

8. La richiesta è stata evasa, una volta un_D è pubblicato D il trasferimento viene elaborato.

Supponiamo ora che un utente richieda di riscattare X' quantità di avvolto-S gettoni dalla catena D torna alla catena S, da depositare presso a S-indirizzo w_S a scelta dell'utente. Il flusso di lavoro è il seguente:

1. L'utente avvia una richiesta di trasferimento (X', w_S) depositando X' quantità di avvolto-S gettoni in C_D tramite un ordinario D-transazione utilizzando il suo software preferito per la catena D.

2. (X', w_S) è pubblicato su Axelar. I validatori interrogano l'API della loro catena D software del nodo per l'esistenza di (X', w_S) e, se la risposta è "vera", riportare la risposta alla catena Axelar.

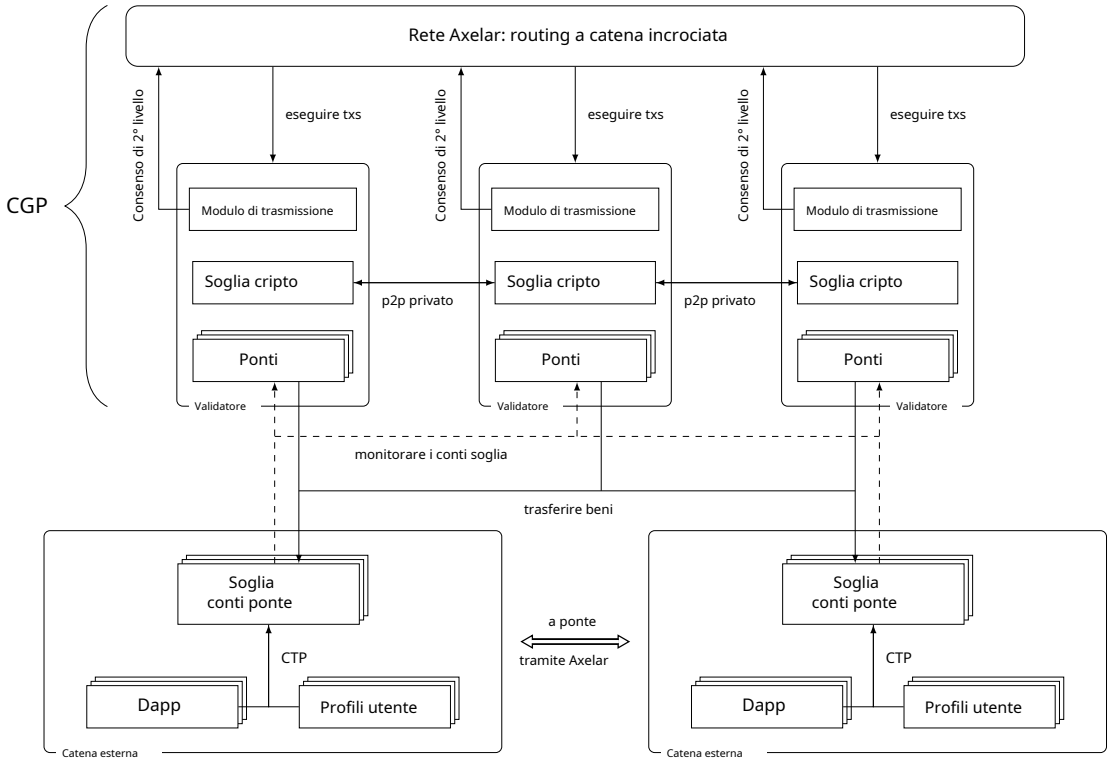

Figura 1: diagramma dei componenti

3. Una volta $> F$ i validatori pesati riportano "vero" per (X, ws) in tondo R, Axelar chiede ai validatori di firmare una transazione uns che invia X quantità di S gettoni da S_{Axelar} a ws.
4. Usando la crittografia a soglia i validatori firmano uns. La firma è inclusa nel blocco $R + 11$.
5. Chiunque può prendere il valore firmato uns dal blocco $R + 11$ e postalo su S.
6. La richiesta è stata evasa, una volta uns è pubblicato S il trasferimento viene elaborato.

Ulteriori richieste supportate dal livello di routing CGP includono il blocco, lo sblocco o il trasferimento di risorse tra catene.

Raggiungere il flusso di transazioni atomiche a catena incrociata. A seconda del tipo di richiesta cross-chain, Axelar cerca di garantire che le transazioni corrispondenti vengano eseguite su più catene o nessuna. A tal fine, ogni richiesta può trovarsi in uno dei seguenti stati nella blockchain di Axelar:*(inizializzato, in attesa, completato, scaduto)*. Se un*tempo scaduto* nella fase di attesa viene attivata, la richiesta restituisce un codice di errore. Iniziano anche alcuni eventi di timeout a*rimborso* evento: ad esempio, se un bene di una catena deve essere trasferito in un bene di un'altra catena, se la catena ricevente non ha elaborato la transazione, il bene viene rimborsato all'utente originario.

7 Protocollo di trasferimento a catena (CTP)

CTP è un protocollo a livello di applicazione che consente alle applicazioni di sfruttare facilmente le funzionalità cross-chain. Spieghiamo l'integrazione concentrandoci sulle funzionalità di trasferimento delle risorse (ad esempio, utilizzate in DeFi). Queste applicazioni sono in genere costituite da tre componenti principali: GUI front-end, contratti intelligenti su una catena e un nodo intermedio che pubblica le transazioni tra il front-end e i contratti intelligenti. I front-end interagiscono con i portafogli dell'utente per accettare depositi, elaborare prelievi, ecc. Le applicazioni possono sfruttare le funzionalità cross-chain

chiamando query CTP analoghe ai metodi HTTP/HTTPS GET/POST. Queste query vengono successivamente prelevate dal livello CGP per l'esecuzione e i risultati vengono restituiti agli utenti.

- *Query CTP*. Gli sviluppatori di applicazioni possono ospitare le loro applicazioni su qualsiasi catena e integrare i loro contratti intelligenti con account bridge di soglia per eseguire query CTP.
- *Conti ponte di soglia*. Supponiamo che uno sviluppatore di applicazioni costruisca i propri contratti sulla catena A. Quindi, farebbero riferimento ai contratti bridge con soglia per ottenere il supporto cross-chain. Questo contratto consente alle applicazioni di:
  - Registra una blockchain con cui vorrebbe comunicare.
  - Registra le risorse su quella blockchain che vorrebbe sfruttare.
  - Esegui operazioni sulle risorse come accettare depositi, elaborare prelievi e altre funzioni (simili, ad esempio, alle chiamate di contratti ERC-20).

Supponiamo che un'importante applicazione DeFi, MapleSwap, risieda nativamente sui registri della catena A con un account bridge di soglia. I validatori Axelar gestiscono collettivamente il contratto stesso sulla catena corrispondente. Supponiamo che un utente desideri inviare un deposito in una coppia di scambio tra gli asset X e Y che risiedono rispettivamente attraverso le due catene. Quindi, quando un utente invia tale richiesta, viene instradata tramite l'account bridge di soglia alla rete Axelar per l'elaborazione. Da lì, vengono eseguiti i seguenti passaggi:

1. La rete Axelar comprende che questa applicazione è stata registrata per il supporto cross-chain attraverso le risorse. Genera la chiave di deposito sfruttando la crittografia a soglia e il consenso per l'utente sulle corrispondenti catene A e B.
2. Le chiavi pubbliche associate vengono restituite all'applicazione e visualizzate all'utente che può utilizzare i propri portafogli preferiti per effettuare depositi. La chiave segreta corrispondente è condivisa tra tutti i validatori Axelar.
3. Quando i depositi sono confermati, Axelar aggiorna la sua directory cross-chain per registrare che l'utente sulle catene corrispondenti ha depositato questi asset.
4. I validatori Axelar eseguono protocolli multiparti per generare una firma di soglia che consenta di aggiornare l'account bridge di soglia sulla catena A in cui risiede l'applicazione.
5. La query CTP viene quindi restituita agli smart contract dell'applicazione DeFi, che possono aggiornarne lo stato, aggiornare le formule di rendimento, i tassi di cambio o eseguire altre condizioni relative allo stato dell'applicazione.

Durante questo processo, la rete Axelar, ad alto livello, agisce come un oracolo di lettura/scrittura cross-chain decentralizzato, CGP è il livello di instradamento tra le catene e CTP è il protocollo dell'applicazione.

Richieste aggiuntive a catena. CTP supporta una catena incrociata più generale tra applicazioni su blockchain come:

- Eseguire Public Key Name Services (PKNS). Questa è una directory universale per la mappatura delle chiavi pubbliche a numeri di telefono/handle di Twitter (alcuni progetti, come Celo, forniscono queste funzionalità all'interno delle loro piattaforme).
- Trigger di applicazioni a catena incrociata. Un'applicazione della catena A può aggiornare il suo stato se qualche altra applicazione della catena B soddisfa un criterio di ricerca (tasso di interesse $< X$).
- Componibilità smart contract. Lo smart contract sulla catena A può aggiornare il proprio stato in base allo stato dei contratti sulla catena B o attivare un'azione per aggiornare uno smart contract sulla catena B.

Ad alto livello, queste richieste possono essere elaborate poiché collettivamente, i protocolli CTP, CGP e rete Axelar possono trasmettere e scrivere informazioni di stato verificabili arbitrarie attraverso blockchain.

8 Riepilogo

Nei prossimi anni, applicazioni e risorse significative saranno costruite su più ecosistemi blockchain. La rete Axelar può essere utilizzata per collegare queste blockchain in uno strato di comunicazione cross-chain uniforme. Questo livello fornisce protocolli di routing e a livello di applicazione che soddisfano le esigenze dei costruttori di piattaforme e degli sviluppatori di applicazioni. Gli sviluppatori di applicazioni possono costruire sulle migliori piattaforme per le loro esigenze e sfruttare un semplice protocollo e API per accedere alla liquidità globale cross-chain, agli utenti e comunicare con altre catene.

Riferimenti

[1] Althea peggy. https://github.com/cosmos/peggy. [Citato a pagina 2.]

[2] Utilizzo deterministico dell'algoritmo di firma digitale (dsa) e dell'algoritmo di firma digitale a curva ellittica (ecdsa). https://tools.ietf.org/html/rfc6979. [Citato a pagina 5.]

[3] Algoritmo di firma digitale della curva di Edwards (eddsa). https://tools.ietf.org/html/rfc8032. [Citato su pagina 5.]

[4] White paper tecnico Eos.io v2. https://github.com/EOSIO/Documentazione/blob/master/TechnicalWhitePaper.md. [Citato a pag 1.]

[5] Ethereum: un registro delle transazioni generalizzato decentralizzato sicuro. https://ethereum.github.io/yellowpaper/paper.pdf. [Citato a pagina 1.]

[6] Il libro quasi bianco. https://near.org/papers/the-official-near-white-paper/. [Citato a pagina 1.]

[7] Ponte arcobaleno. https://github.com/near/rainbow-bridge. [Citato a pagina 2.]

[8] Ren: una macchina virtuale che preserva la privacy che alimenta applicazioni finanziarie a conoscenza zero. https://whitepaper.io/document/419/ren-litepaper. [Citato a pagina 3.]

[9] tbtc: un token erc-20 riscattabile decentralizzato supportato da btc. https://docs.keep.network/tbtc/index.PDF. [Citato a pagina 2.]

[10] Thorchain: una rete di liquidità decentralizzata. https://thorchain.org/. [Citato a pagina 3.]

[11] Kurt M. Alonso. Zero a Monero.https://www.getmonero.org/library/Zero-to-Monero-1-0-0.PDF. [Citato a pagina 1.]

[12] Jean-Philippe Aumasson, Adrian Hamelink e Omer Shlomovits. Un'indagine sulla firma della soglia ecdsa. Archivio ePrint di crittografia, rapporto 2020/1390, 2020.https://eprint.iacr.org/2020/1390. [Citato su pagina 6.]

[13] Ran Canetti, Nikolaos Makriyannis e Udi Peled. Archivio Uc non interattivo, proattivo, soglia ecdsa. ePrint di crittografia, rapporto 2020/492, 2020. https://eprint.iacr.org/2020/492. [Citato su pagina 6.]

[14] White paper di cLabs. https://celo.org/papers. [Citato a pagina 1.]

[15] Ivan Damgård, Thomas Pelle Jakobsen, Jesper Buus Nielsen, Jakob Illeborg Pagter e Michael Bæksvang Østergård. ECDSA a soglia rapida con maggioranza onesta. In*SCN*, volume 12238 di *Appunti delle lezioni in Informatica*, pagine 382–400. Primavera, 2020.[Citato a pagina 6.]

[16] Manu Drijvers, Kasra Edalatnejad, Bryan Ford, Eike Kiltz, Julian Loss, Gregory Neven e Igors Stepanovs. Sulla sicurezza delle firme multiple a due round. In*Simposio IEEE su sicurezza e privacy*, pagine 1084–1101. IEEE, 2019.[Citato a pagina 6.]

[17] Cynthia Dwork, Nancy Lynch e Larry Stockmeyer. Consenso in presenza di sincronia parziale. https://groups.csail.mit.edu/tds/papers/Lynch/jacm88.pdf. [Citato a pagina 5.]

[18] Rosario Gennaro e Steven Goldfeder. Una soglia rotonda ecdsa con interruzione identificabile. Archivio ePrint di crittografia, rapporto 2020/540, 2020.https://eprint.iacr.org/2020/540. [Citato a pagina 6.]

[19] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos e Nickolai Zeldovich. Algorand: ridimensionamento degli accordi bizantini per le criptovalute. Atti del 26° Simposio sui principi dei sistemi operativi, 2017.https://dl.acm.org/doi/pdf/10.1145/3132747.3132757. [Citato a pagina 1.]

[20] Evan Kereiakes, Do Kwon, Marco Di Maggio e Nicholas Platias. Soldi di Terra: stabilità e adozione.

https://terra.money/Terra_White_paper.pdf. [Citato a pagina 1.]

[21] Aggelos Kiayias, Alexander Russell, Bernardo David e Roman Oliynykov. Ouroboros: un protocollo blockchain proof-of-stake dimostrabilmente sicuro.https://eprint.iacr.org/2016/889.pdf. [Citato a pagina 1.]

[22] Chelsea Komlo e Ian Goldberg. Frost: firme di soglia schnorr flessibili ottimizzate per il round. Archivio ePrint di crittografia, rapporto 2020/852, 2020.https://eprint.iacr.org/2020/852. [Citato a pagina 6.]

[23] Jae Kwon e Ethan Buchman. Cosmos: una rete di libri mastri distribuiti. https://cosmos.network/resources/whitepaper. [Citato nelle pagine 1 e 2.]

[24] Squadra valanga. Piattaforma da valanga. https://www.avalabs.org/whitepapers. [Citato nelle pagine 1 e 2.]

[25] Gavin Wood. Polkadot: Visione per un framework multi-catena eterogeneo. https://polkadot.network/PolkaDotPaper.pdf. [Citato nelle pagine 1 e 2.]