Sieć Axelar:
Łączenie aplikacji z ekosystemami Blockchain

Szkic 1.0

styczeń 2021

Abstrakcyjny

Powstaje wiele ekosystemów blockchain, które zapewniają unikalne i wyróżniające się funkcje atrakcyjne dla użytkowników i twórców aplikacji. Jednak komunikacja między ekosystemami jest bardzo rzadka i fragmentaryczna. Aby umożliwić aplikacjom bezproblemową komunikację między ekosystemami blockchain, proponujemy Axelar. Stos Axelar zapewnia zdecentralizowaną sieć, protokoły, narzędzia i interfejsy API, które umożliwiają prostą komunikację między łańcuchami. Pakiet protokołów Axelar składa się z transgranicznych protokołów routingu i transferu. Zdecentralizowana otwarta sieć walidatorów zasila sieć; każdy może dołączyć, używać go i uczestniczyć. Bizantyjski konsensus, kryptografia i mechanizmy motywacyjne mają na celu osiągnięcie wysokich wymagań dotyczących bezpieczeństwa i żywotności, unikalnych dla żądań cross-chain.

1. Wstęp

Systemy Blockchain szybko zyskują na popularności i przyciągają nowe przypadki użycia do tokenizacji aktywów, zdecentralizowanych finansów i innych aplikacji rozproszonych. Kilka głównych platform, takich jak Ethereum, Monero, EOS, Cardano, Terra, Cosmos, Avalanche, Algorand, Near, Celo i Polkadot, oferuje różne funkcje i środowiska programistyczne, które czynią je atrakcyjnymi dla różnych aplikacji, przypadków użycia i użytkowników końcowych [5, 11, 4, 21, 20, 23, 24, 19, 6, 14, 25]. Jednak przydatne funkcje każdej nowej platformy są obecnie oferowane mniej niż 1% użytkowników ekosystemu, a mianowicie posiadaczom natywnego tokena na tej platformie. Czy możemy pozwolić programistom platform na łatwe podłączenie ich łańcuchów bloków do innych ekosystemów? Czy możemy umożliwić twórcom aplikacji tworzenie na najlepszej platformie dla ich potrzeb, jednocześnie komunikując się w wielu ekosystemach blockchain? Czy możemy pozwolić użytkownikom na interakcję z dowolną aplikacją na dowolnym blockchainie bezpośrednio z ich portfeli?

Aby połączyć ekosystemy blockchain i umożliwić aplikacjom bezproblemową komunikację między nimi, proponujemy sieć Axelar. Walidatory wspólnie uruchamiają bizantyjski protokół konsensusu i uruchamiają protokoły ułatwiające żądania międzyłańcuchowe. Każdy może dołączyć do sieci, uczestniczyć i korzystać z niej. Podstawowa sieć jest zoptymalizowana pod kątem wysokich wymagań bezpieczeństwa i żywotności, unikalnych dla żądań cross-chain. Sieć Axelar zawiera również pakiet protokołów i API. Podstawowe protokoły to:

- Protokół bramy krzyżowej (CGP). Protokół ten jest analogiczny do protokołu Border Gateway w Internecie. Protokół ten służy do łączenia wielu autonomicznych ekosystemów blockchain i odpowiada za routing między nimi. Blockchainy nie muszą „mówić żadnym niestandardowym językiem”, programiści ich platform nie muszą wprowadzać żadnych niestandardowych zmian w swoich łańcuchach, a ich łańcuchy można łatwo podłączyć do globalnej sieci.
- Protokół CTP (Cross-Chain Transfer Protocol). Protokół ten jest analogiczny do protokołów poziomu aplikacji File Transfer, Hypertext Transfer Protocols w Internecie. Jest to stos protokołów na poziomie aplikacji, który znajduje się na szczycie protokołów routingu (takich jak CGP i inne technologie routingu). Twórcy aplikacji mogą łączyć swoje dappy w dowolnym łańcuchu, aby wykonywać żądania międzyłańcuchowe. Użytkownicy mogą używać protokołu CTP do interakcji z aplikacjami w dowolnym łańcuchu za pomocą prostych wywołań API, analogicznych do żądań HTTP GET/POST. Deweloperzy mogą blokować, odblokowywać i przenosić zasoby między dowolnymi dwoma adresami na dowolnych platformach blockchain, wykonywać cross-chainowe wyzwalacze aplikacji (np. dapps w łańcuchu A, mogą aktualizować

jego stan, jeśli inna aplikacja w łańcuchu B spełnia pewne kryteria wyszukiwania (stopa procentowa) *> x*)) i wykonuj ogólne żądania między łańcuchami między aplikacjami w łańcuchach (inteligentna umowa w łańcuchu A może wywołać aktualizację stanu inteligentnej umowy w łańcuchu B). Protokół ten umożliwia komponowanie programów w ekosystemach blockchain.

Sieć Axelar oferuje następujące zalety:

- *Dla twórców platform blockchain*: Możliwość łatwego podłączenia swoich łańcuchów bloków do wszystkich innych ekosystemów łańcucha bloków. Aby podłączyć się do sieci, należy skonfigurować tylko konto progowe w łańcuchu.
- *Dla twórców dappów*: Twórcy aplikacji mogą hostować swoje aplikacje w dowolnym miejscu, blokować, odblokowywać, przenosić zasoby i komunikować się z aplikacjami w dowolnym innym łańcuchu za pośrednictwem interfejsu API CTP.
- *Dla użytkowników*: Użytkownicy mogą wchodzić w interakcję ze wszystkimi aplikacjami w całym ekosystemie bezpośrednio ze swoich portfeli.

Platforma dla budowniczych. Wreszcie, sieć Axelar to platforma dla programistów i globalnej społeczności. Jego model zarządzania jest otwarty dla każdego. Deweloperzy mogą proponować nowe punkty integracji, routing i protokoły na poziomie aplikacji, a użytkownicy mogą zdecydować, czy je przyjąć, głosując nad propozycjami, a jeśli zostaną zatwierdzone, walidatorzy przyjmą zmiany.

1.1 Istniejące rozwiązania interoperacyjne

Poprzednie próby rozwiązania problemu interoperacyjności między łańcuchami bloków należą do jednej z czterech kategorii: scentralizowane wymiany, interoperacyjne ekosystemy, opakowane aktywa i mosty tokenów. Poniżej krótko podsumowujemy te podejścia.

Systemy scentralizowane. potrzeby ekosystemu. Obecnie scentralizowane systemy są jedynymi naprawdę skalowalnymi rozwiązaniami zapewniającymi interoperacyjność Mogą stosunkowo łatwo wymienić dowolny zasób lub na pokładzie dowolnej platformy. Jednakże, Wiadomo, że scentralizowane systemy mają różne problemy z bezpieczeństwem i nie są wystarczająco dobre, aby zasilać powstający zdecentralizowany system finansowy, który wymaga solidnego bezpieczeństwa, przejrzystości i otwartego zarządzania. Na własną rękę nie mogą zasilać zdecentralizowanych aplikacji w miarę ich rozwoju.

Centra interoperacyjności. Projekty takie jak Cosmos, Polkadot, Ava labs dotyczą interoperacyjności między *łańcuchy boczne* natywny dla ich ekosystemów przy użyciu niestandardowych protokołów komunikacyjnych między łańcuchami [23, 25, 24]. Na przykład można rozkręcić łańcuch boczny (strefę Cosmos), który może komunikować się z centrum Cosmos. Łańcuch boczny musi opierać się na konsensusie Tendermint i posługiwać się protokołem natywnie rozumianym przez Cosmos Hub. Połączenia z innymi blockchainami i ekosystemami, które mówią różnymi językami, pozostawia się zewnętrznym technologiom.

Mosty parami. Opakowane aktywa (np. opakowane bitcoiny) próbują wypełnić brakującą w ekosystemie lukę w interoperacyjności między łańcuchami. Jednym z przykładów jest tBTC [9], który jest niestandardowym protokołem, w którym do zabezpieczenia przelewów wykorzystuje się sprytne połączenie inteligentnych kontraktów i zabezpieczeń. Te rozwiązania wymagają znacznych wysiłków inżynieryjnych w celu zbudowania – dla każdej pary łańcuchów programiści muszą zbudować nowy inteligentny kontrakt w łańcuchu docelowym, który analizuje dowody stanu z łańcucha pochodzenia (podobnie jak każdy łańcuch boczny mógłby w zasadzie analizować stan innych łańcuchów) . Tylko kilka mostów zostało wdrożonych przy użyciu tego podejścia. Te podejścia nie podlegają skalowaniu, gdy jeden z podstawowych łańcuchów bloków chce uaktualnić swoje zasady konsensusu lub format transakcji. Dzieje się tak, ponieważ wszystkie inteligentne kontrakty, które zależą od stanu tych łańcuchów, musiałyby zostać zaktualizowane. Należy również skonfigurować walidatory i wymagać od nich blokowania różnych aktywów w celu nadzabezpieczenia jakiegokolwiek transferu aktywów,

Widzieliśmy również kilka innych mostów jednofunkcyjnych przez twórców platform, którzy przepisują logikę przejścia stanu w inteligentnych kontraktach, aby połączyć się z innymi ekosystemami [1, 7]. Cierpią z powodu wielu problemów ze skalowalnością, nie pozwalają na jednolite skalowanie ekosystemu i wprowadzają dodatkowe zależności dla aplikacji. Na przykład, jeśli zmieni się jedna platforma, wszystkie inteligentne kontrakty na wszystkich mostach będą musiały zostać zaktualizowane. Ten

ostatecznie wprowadzi ekosystem w impas, w którym nikt nie będzie mógł go ulepszyć. Wreszcie, jeśli jeden most jednofunkcyjny łączy platformy A i B, a drugi most jednofunkcyjny łączy B i C, nie oznacza to, że aplikacje na A będą w stanie komunikować się z aplikacjami na C. Być może trzeba będzie utworzyć inny jednokierunkowy celu mostu lub zmiany logiki aplikacji.

Inne próby rozwiązania problemu interoperacyjności obejmują federacyjne wyrocznie (np. Ren [8]) oraz interoperacyjne łańcuchy bloków specyficzne dla aplikacji [10].

Podsumowując, istniejące rozwiązania w zakresie interoperacyjności wymagają ciężkiej pracy inżynierskiej zarówno od twórców platform, jak i twórców aplikacji, którzy muszą rozumieć różne protokoły komunikacyjne, aby komunikować się w każdej parze ekosystemów. Tak więc interoperacyjność praktycznie nie istnieje w dzisiejszej przestrzeni blockchain. Ostatecznie programiści platform chcą skupić się na budowaniu platform i optymalizowaniu ich pod kątem ich zastosowań, a także w łatwym łączeniu się z innymi łańcuchami bloków. Twórcy aplikacji chcą tworzyć aplikacje na najlepszych platformach dla swoich potrzeb, jednocześnie wykorzystując użytkowników, płynność i komunikację z innymi aplikacjami w innych sieciach.

2 W poszukiwaniu skalowalnej komunikacji międzyłańcuchowej

W istocie komunikacja międzyłańcuchowa wymaga, aby heterogeniczne sieci znalazły możliwość komunikowania się przy użyciu tego samego języka. Aby rozwiązać ten problem, wyjaśniamy pakiet protokołów Axelar, opisujemy jego właściwości wysokiego poziomu i wyjaśniamy, w jaki sposób te właściwości odnoszą się do rdzenia skalowalnej komunikacji międzyłańcuchowej.

1. *"Integracja typu „plug and play".* Konstruktorzy platformy Blockchain nie powinni być zobowiązani do wykonywania ciężkich prac inżynieryjnych lub integracyjnych, aby mówić „niestandardowym językiem" w celu obsługi łańcucha krzyżowego. Protokół crosschain powinien być w stanie bezproblemowo podłączyć dowolny istniejący lub nowy blockchain. Nowe zasoby należy dodawać przy minimalnym wysiłku.
2. *Routing krzyżowy.* Funkcje takie jak wykrywanie adresów sieciowych, ścieżek routingu i sieci są rdzeniem Internetu i są wspierane przez BGP i inne protokoły routingu. Podobnie, aby ułatwić komunikację między ekosystemami blockchain, musimy wspierać wykrywanie adresów w nich, aplikacjach i routingu.
3. *Wsparcie dla możliwości rozbudowy.* Jeśli jeden z ekosystemów blockchain ulegnie zmianie, nie powinno to wpłynąć na interoperacyjność innych blockchainów. System musi rozpoznawać aktualizacje, a ich obsługa wymaga minimalnego wysiłku (tj. nie należy przepisywać „logiki przejścia między stanami", a aplikacje nie powinny się psuć).
4. *Jednolity język aplikacji.* Aplikacje potrzebują prostego protokołu do blokowania, odblokowywania, przesyłania i komunikowania się z innymi aplikacjami, niezależnie od tego, w którym łańcuchu się znajdują. Protokół ten musi być niezależny od łańcucha i obsługiwać proste wywołania, analogicznie do protokołów HTTP/HTTPS, które umożliwiają użytkownikom i przeglądarkom komunikację z dowolnym serwerem sieciowym. Ponieważ coraz więcej sieci i zasobów dołącza do protokołów routingu niższego poziomu, aplikacje powinny być w stanie używać ich do komunikacji bez przepisywania stosów oprogramowania.

Następnie podsumowujemy wymagania bezpieczeństwa, które muszą spełniać te protokoły.

1. *Zdecentralizowane zaufanie.* Sieć i protokoły muszą być zdecentralizowane, otwarte i umożliwiać wszystkim uczciwy udział.
2. *Wysokie bezpieczeństwo.* System musi spełniać wysokie gwarancje bezpieczeństwa. System musi zachować bezpieczeństwo aktywów i stanu, gdy przetwarza je sieć międzyłańcuchowa.
3. *Wysoka żywotność.* System musi spełniać gwarancje wysokiej żywotności, aby wspierać aplikacje wykorzystujące jego funkcje cross-chain.

Spełnienie podzbioru tych właściwości jest łatwe. Na przykład można utworzyć sfederowane konto multisig ze swoimi przyjaciółmi i zablokować/odblokować zasoby w odpowiednich łańcuchach. Takie systemy są z natury podatne na ataki zmowy i cenzury i nie mają odpowiednich zachęt dla walidatorów do ich ochrony. Stworzenie zdecentralizowanej sieci i zestawu protokołów, w których każdy może uczestniczyć, a jednocześnie jest odpowiednio motywowany, może umożliwić bezproblemową komunikację między łańcuchami, ale rozwiązanie tego problemu jest trudnym problemem, który wymaga starannego połączenia protokołów konsensusu, kryptografii i projektowania mechanizmów.

3 Sieć Axelar

Sieć Axelar zapewnia jednolite rozwiązanie do komunikacji międzyłańcuchowej, które spełnia potrzeby zarówno deweloperów platform – nie wymaga od nich żadnych prac integracyjnych, jak i twórców aplikacji – jeden prosty protokół i API, aby uzyskać dostęp do globalnej płynności i komunikować się z całym ekosystemem.

Sieć Axelar składa się ze zdecentralizowanej sieci, która łączy ekosystemy blockchain, które mówią różnymi językami, oraz pakiet protokołów z interfejsami API na górze, ułatwiając aplikacjom wykonywanie żądań międzyłańcuchowych. Sieć łączy istniejące samodzielne łańcuchy bloków, takie jak Bitcoin, Stellar, Terra, Algorand, oraz węzły interoperacyjności, takie jak rozwiązania takie jak Cosmos, Avalanche, Ethereum i Polkadot. Naszą misją jest umożliwienie twórcom aplikacji łatwiejszego tworzenia takich aplikacji przy użyciu uniwersalnego protokołu i API bez rozwijania ich zastrzeżonych protokołów cross-chain pod spodem lub przepisywania aplikacji w miarę opracowywania nowych mostów. W tym celu zaprojektowaliśmy pakiet protokołów, który zawiera Cross-Chain Gateway Protocol (patrz rozdział6) i Cross-Chain Transfer Protocol (patrz rozdział 7).

Podstawowym elementem sieci są podstawowe zdecentralizowane protokoły. Walidatory wspólnie utrzymują sieć Axelar i uruchamiają węzły, które zabezpieczają blockchain Axelar. Są wybierani przez użytkowników w procesie delegacji. Walidatorzy otrzymują prawo głosu proporcjonalnie do delegowanego im stawki. Walidatorzy osiągają konsensus co do stanu wielu łańcuchów bloków, z którymi platforma jest połączona. Blockchain jest odpowiedzialny za utrzymanie i działanie protokołów routingu i transferu między łańcuchami. Reguły zarządzania umożliwiają uczestnikom sieci podejmowanie decyzji dotyczących protokołów, takich jak łańcuchy bloków, które należy połączyć i które zasoby wspierać.

Blockchain Axelar jest zgodny z modelem Delegated Proof-of-Stake (DPoS), podobnym do Cosmos Hub. Użytkownicy wybierają walidatorów, którzy muszą związać swój udział, aby uczestniczyć w konsensusie i utrzymać wysoką jakość usług. Model DPoS umożliwia utrzymanie dużego zdecentralizowanego zestawu walidatorów i solidne zachęty, aby zagwarantować, że walidatorzy są odpowiedzialni za utrzymywanie mostów i udziałów kryptograficznych schematów progowych. W ramach konsensusu walidatorzy uruchamiają lekkie oprogramowanie klienckie innych łańcuchów blokowych, co pozwala im weryfikować stan innych łańcuchów blokowych. Walidatorzy zgłaszają te stany do łańcucha blokowego Axelar, a gdy zgłoszą wystarczającą ich liczbę, stan Bitcoin, Ethereum i innych łańcuchów jest rejestrowany na Axelar.

Następnie warstwa podstawowa Axelar jest świadoma stanu zewnętrznych blockchainów w dowolnym momencie, tworząc „przychodzące mosty" z innych blockchainów. Walidatorzy wspólnie utrzymują*konta z podpisem progowym* na innych blockchainach (np. 80% walidatorów musi zatwierdzić i wspólnie podpisać każdą z nich transakcję), co pozwala im blokować i odblokowywać aktywa i stany w łańcuchach oraz publikować stany na innych blockchainach, „mostach wychodzących". W sumie sieć Axelar można postrzegać jako:*zdecentralizowana wyrocznia odczytu/zapisu crosschain*.

Pozostała część dokumentu opisuje etapy wstępne i elementy składowe sieci (sekcja 4), niektóre szczegóły techniczne sieci (sekcja 5), protokół bramy międzyłańcuchowej (sekcja 6) oraz protokół transferu międzyłańcuchowego (sekcja 7).

4 eliminacje

4.1 Notacja i założenia

Pozwalać V_r oznaczają zestaw walidatorów Axelar w rundzie r. Każdy walidator ma *waga*, liczba w (0, 1] oznaczający siłę głosu tego konkretnego walidatora. Wagi wszystkich walidatorów sumują się do 1. Walidator to *prawidłowy* jeśli uruchomi węzeł zgodny z zasadami protokołu Axelar. Aby sfinalizować bloki lub podpisać żądania cross-chain, Axelar wymaga odpowiednich walidatorów całkowitej wagi $> F$. Nazywamy parametr $F \in$. [0.5, 1] *próg protokołu*.

Axelar może być oparty na *natychmiastowa ostateczność Delegowany dowód stawki* blockchain. Walidatory działają *Konsensus dotyczący bizantyjskiej tolerancji błędów (BFT)* w każdej rundzie i sfinalizować i_{ten} blok. Kiedyś i_{ten} blok jest sfinalizowany, uruchamiany jest nowy konsensus BFT w celu sfinalizowania $i + 1_{ten}$ blokować i tak dalej. Walidatorzy są wybierani poprzez delegację palika. Użytkownik z pewną stawką może zdecydować się na uruchomienie węzła walidatora lub przekazać swoją moc głosu (stawkę) istniejącemu walidatorowi, który następnie głosuje w jego imieniu. Zestaw walidatorów można aktualizować, walidatory dołączają do zestawu/opuszczają go, a użytkownicy delegują/oddelegowują swoje uprawnienia do głosowania.

Różne łańcuchy bloków działają przy różnych założeniach sieciowych. *Komunikacja synchroniczna* oznacza, że istnieje ustalona górna granica Δ czasu dostarczenia wiadomości, gdzie Δ jest znane i może być wbudowane w protokół. *Komunikacja asynchroniczna* oznacza, że dostarczenie wiadomości może zająć dowolnie dużo czasu, a wiadomo, że protokołów BFT nie można zbudować dla sieci asynchronicznych, nawet w obecności tylko jednego złośliwego walidatora. Realistycznym kompromisem między synchronią a asynchronią jest założenie *częściowo synchroniczna komunikacja*. Sieć może być całkowicie asynchroniczna do pewnego nieznanego czasu stabilizacji globalnej (GST), ale po komunikacji GST staje się synchroniczna ze znaną górną granicą Δ [17].

Typowe blockchainy działają przy założeniu $> F$ poprawne walidatory. Dla sieci synchronicznych $F = 1/2$ jest zwykle ustawiane, ale dla słabszego założenia sieci częściowo synchronicznej $F = 2/3$. Bitcoin, jego widełki i aktualna wersja Ethereum typu Proof-of-Work działają tylko przy założeniu synchronizacji. Inne, takie jak Algorand i Cosmos, wymagają tylko częściowej synchronizacji. Przy łączeniu łańcuchów przez Axelara połączenie działa przy założeniu najsilniejszych założeń sieciowych z tych łańcuchów, czyli synchroniczności w przypadku połączenia np. Bitcoina i Cosmos. Sam blockchain Axelar działa w częściowo synchronicznym ustawieniu i dlatego wymaga: $F = 2/3$, ale możliwe jest poprawienie wymagań progowych poprzez założenie, że inne istniejące łańcuchy bloków są bezpieczne i wykorzystanie ich bezpieczeństwa.

4.2 Wstępy kryptograficzne

Podpisy cyfrowe. A *schemat podpisu cyfrowego* jest krotką algorytmów (*Keygen, znak, w eryfikacja*). *Generator kluczy* wyprowadza parę kluczy (*PK, SK*). Tylko właściciel *SK* może podpisywać wiadomości, ale każdy może zweryfikować podpisy podając klucz publiczny *PK*. Większość dzisiejszych systemów blockchain używa jednego ze standardowych schematów podpisów, takich jak ECDSA, Ed25519 lub kilku ich wariantów [2, 3].

Podpisy progowe. A *schemat sygnatury progowej* umożliwia grupie n strony podzielą tajny klucz dla schematu podpisu w taki sposób, aby każdy podzbiór T + Jedna lub więcej stron może współpracować w celu stworzenia podpisu, ale bez podzbioru T lub mniej stron może złożyć podpis, a nawet uzyskać informacje o tajnym kluczu. Podpisy tworzone przez protokoły progowe dla ECDSA i EdDSA wyglądają identycznie jak podpisy generowane przez algorytmy autonomiczne.

Schemat sygnatury progowej zastępuje *Generator kluczy* oraz *Podpisać* algorytmy dla zwykłego schematu podpisu z rozproszonym n-protokoły imprezowe $T.Keygen$, $T.Znak$. Protokoły te zazwyczaj wymagają zarówno publicznego kanału rozgłoszeniowego, jak i prywatnych kanałów parami między stronami i zazwyczaj obejmują kilka rund komunikacji. Po pomyślnym zakończeniu $T.Keygen$ każdy użytkownik posiada udział s_i tajnego klucza SK i odpowiedniego klucza publicznego PK. ten $T.Znak$ protokół pozwala tym stronom na złożenie podpisu dla

podana wiadomość, która jest ważna pod kluczem publicznym PK. Ten podpis może zweryfikować każdy, kto korzysta z*Zweryfikować* algorytm oryginalnego schematu podpisu.

4.3 Właściwości sygnatur progowych

Istnieje kilka właściwości, które może mieć schemat progowy, które są szczególnie pożądane w przypadku sieci zdecentralizowanych:

Zabezpieczenie przed nieuczciwą większością. Niektóre systemy progowe mają ograniczenie, że są bezpieczne tylko wtedy, gdy większość *n* imprezy są uczciwe. Zatem parametr progowy *T* musi być mniejszy niż *n*/2 [15]. Ograniczeniu temu zazwyczaj towarzyszy fakt, że 2*T* + Do podpisania wystarczy 1 uczciwa strona, chociaż tylko *T* + 1 skorumpowana strona może zmówić się w celu odzyskania tajnego klucza. Mówi się, że systemy, które nie podlegają temu ograniczeniu, są*zabezpieczyć się przed nieuczciwą większością*.

Jak omówiono w dalszej części rozdziału 5.2platformy cross-chain muszą zmaksymalizować bezpieczeństwo swoich sieci i być w stanie tolerować jak największą liczbę stron skorumpowanych. Dlatego konieczne są schematy, które mogą tolerować nieuczciwą większość.

Podpisy wstępne, nieinteraktywne podpisywanie online. W celu zmniejszenia ciężaru komunikacji po podpisaniu wiadomości przez strony, kilka ostatnich protokołów określiło znaczną część pracy nad podpisem, którą można wykonać „offline", zanim wiadomość do podpisania stanie się znana [18, 13]. Wyjście tej fazy offline nazywa się a*pre-podpis*. Produkcja podpisów wstępnych jest traktowana jako osobny protokół *T* .*Presign* odróżnić od *T* .*Keygen* oraz *T* .*Znak*. Dane wyjściowe protokołu wstępnego podpisu muszą być utrzymywane w tajemnicy przez strony, dopóki nie użyją ich w fazie podpisywania. Później, gdy wiadomość do podpisania stanie się znana, do wykonania pozostaje tylko niewielka ilość dodatkowej pracy „online" w*T* .*Znak* w celu uzupełnienia podpisu.

Internet *T* .*Znak* faza nie wymaga żadnej komunikacji między stronami. Każda ze stron po prostu wykonuje lokalne obliczenia dotyczące wiadomości i wstępnego podpisu, a następnie ogłasza swój udziałs_i podpisu. (Po upublicznieniu te akcje podpisu$s_1, \ldots, s_{T+1}$ są łatwo łączone przez każdego, aby ujawnić rzeczywisty podpis *s*.) Ta właściwość nazywa się *nieinteraktywne podpisywanie online*.

Krzepkość. Schematy progowe gwarantują tylko, że podzbiór złośliwych stron nie może podpisywać wiadomości lub poznaj tajny klucz. Ta gwarancja nie wyklucza jednak możliwości, że źli aktorzy mogą zablokować wszystkim innym możliwość tworzenia kluczy lub podpisów. W niektórych schematach złośliwe zachowanie nawet jednej strony może spowodować:*T* .*Keygen* lub *T* .*Znak* przerwać bez użytecznego wyjścia. Jedynym wyjściem jest ponowne uruchomienie protokołu, być może z różnymi stronami.

Zamiast tego w przypadku zdecentralizowanych sieci chcemy *T* .*Keygen* oraz *T* .*Znak* odnieść sukces, jeśli przynajmniej *T* + Jedna ze stron jest uczciwa, nawet jeśli niektóre złośliwe strony wysyłają zniekształcone wiadomości lub upuszczają wiadomości w protokołach. Ta właściwość nazywa się*krzepkość*.

Przypisanie błędu. Umiejętność identyfikowania złych aktorów w *T* .*Keygen* lub *T* .*Znak* jest nazywany *przypisywanie winy*. Bez przypisania winy trudno jest wiarygodnie wykluczyć lub ukarać złych aktorów, w którym to przypadku koszty narzucone przez złych aktorów muszą ponieść wszyscy. Ta właściwość jest również ważna w przypadku zdecentralizowanych sieci, w których złośliwe zachowanie powinno być możliwe do zidentyfikowania i zniechęcenia ekonomicznego za pomocą reguł slashingu.

Bezpieczeństwo w ustawieniach współbieżnych. Schemat podpisu musi być bezpieczny w warunkach współbieżnych, gdzie wiele wystąpień algorytmów generowania kluczy i podpisywania może być zaangażowanych równolegle. (Drijvers i in. [16] pokazał atak na schematy multisygnaturowe Schnorra w tych ustawieniach). Istnieją wersje schematów ECDSA i Schnorr, które spełniają te właściwości [13, 22].

ECDSA i EdDSA są zdecydowanie najszerzej stosowanymi schematami podpisów w przestrzeni blockchain. Jako takie, progowe wersje obu programów stały się przedmiotem niedawnego odrodzenia badań i rozwoju. Czytelnicy zainteresowani najnowszym stanem wiedzy mogą zapoznać się z [22, 13, 18] i niedawny artykuł ankietowy [12].

5 Sieć Axelar

5.1 Projektowanie otwartej sieci krzyżowej

Mosty utrzymywane przez sieć Axelar są wspierane przez konta progowe, tak że (prawie) wszyscy walidatorzy muszą wspólnie autoryzować każde żądanie cross-chain. Projektowanie sieci, w której każdy może uczestniczyć w zabezpieczaniu tych mostów, wymaga spełnienia następujących wymagań technicznych:

- *Otwarte członkostwo.* Każdy użytkownik powinien mieć możliwość zostania walidatorem (zgodnie z zasadami sieci).
- *Aktualizacje członkostwa.* Kiedy walidator uczciwie opuszcza system, jego klucz musi zostać odpowiednio unieważniony.
- *Zachęty i cięcia.* Złośliwe walidatory powinny być możliwe do zidentyfikowania, a ich działania muszą być zidentyfikowane i uwzględnione w protokole.
- *Zgoda.* Same schematy progowe są definiowane jako protokoły samodzielne. Aby propagować komunikaty między węzłami, potrzebujemy zarówno prywatnych kanałów rozgłoszeniowych, jak i prywatnych. Co więcej, walidatorzy muszą uzgodnić najnowszy stan każdego wywołania schematów progowych, ponieważ często mają wiele rund interakcji.
- *Zarządzanie kluczami.* Tak jak zwykłe walidatory w dowolnym systemie PoS muszą starannie strzec swoich kluczy, tak samo walidatory Axelar muszą strzec swoich udziałów progowych. Klucze muszą być obracane, dzielone na części online i offline itp.

Axelar zaczyna od modelu Delegated Proof-of-Stake, w którym społeczność wybiera zestaw walidatorów do przeprowadzenia konsensusu. Zwróć uwagę, że standardowe schematy progowe traktują każdego gracza identycznie i nie mają pojęcia „wagi" w konsensusie. Dlatego sieć musi je dostosować, aby uwzględnić wagę walidatorów. Prostym podejściem jest przypisanie wielu udziałów progowych do większych walidatorów. Poniżej opisano trzy podstawowe funkcje, które walidatory wspólnie wykonują.

- *Generowanie klucza progowego.* Istniejące algorytmy progowego generowania kluczy dla standardowych schematów podpisów blockchain (ECDSA, Ed25519) to protokoły interaktywne między wieloma uczestnikami (patrz rozdział 4). Specjalna transakcja w sieci Axelar nakazuje walidatorom rozpoczęcie wykonywania tego protokołu stanowego. Każdy walidator uruchamia proces demona progu, który jest odpowiedzialny za bezpieczne przechowywanie stanu tajnego. Dla każdej fazy protokołu:

  1. Walidator przechowuje stan protokołu w swojej pamięci lokalnej.
  2. Wywołuje tajnego demona do generowania komunikatów zgodnie z opisem protokołu dla innych walidatorów.
  3. Propaguje komunikaty za pośrednictwem transmisji lub kanałów prywatnych do innych walidatorów.
  4. Każdy walidator wykonuje funkcje zmiany stanu, aby zaktualizować swój stan, przejść do następnej fazy protokołu i powtórzyć powyższe kroki.

  Na końcu protokołu w łańcuchu Axelar generowany jest progowy klucz publiczny, który może zostać wyświetlony z powrotem użytkownikowi (np. w przypadku depozytów) lub aplikacji, która wygenerowała początkowe żądanie.

- *Podpisywanie progowe.* Żądania podpisania w sieci Axelar są przetwarzane podobnie jak żądania generowania kluczy. Są one wywoływane, na przykład, gdy użytkownik chce wycofać zasób z jednego z więzy. protokołów interaktywnych, a zmiana stanu między rundami jest wyzwalana jako funkcja komunikatów propagowanych za pośrednictwem widoku blockchain Axelar i lokalnej pamięci każdego walidatora.

- *Obsługa zmian w członkostwie walidatora.* Zestaw walidatorów musi być okresowo zmieniany, aby umożliwić nowym interesariuszom dołączenie do zestawu. Po aktualizacji zestawu walidatorów musimy zaktualizować klucz progu, który ma być udostępniany w nowym zestawie. Tak więc, gdybyśmy w dowolnym momencie pozwolili komukolwiek dołączyć, musielibyśmy bardzo często aktualizować klucz progu. Aby temu zapobiec, zmieniamy walidatory co T Bloki. W odstępach T rundy, zestaw V_r a klawisz progu jest stały. W każdej rundzie, która jest całkowitą wielokrotnością parametru T, aktualizujemy zestaw walidatorów w następujący sposób:

1. W każdej rundzie r, stan Axelar śledzi bieżący zestaw walidatorów V_r. $V_{r+1} = V_r$ chyba że $r + 1$ jest wielokrotnością T.

2. Podczas rund $((ja - 1)T, toT]$, użytkownicy publikują wiadomości o wiązaniu/odłączaniu.

3. Pod koniec rundy to, te wiadomości są stosowane do V_{iT-1} dostać V_{to}.

- *Generowanie klucza progowego i podpisywanie w obecności walidatorów rotacyjnych.* Blockchain Axelar może wystawić prośbę o nowy klucz lub sygnaturę progową w rundzie r. Proces podpisywania trwa dłużej niż jedną rundę i nie chcemy spowalniać konsensusu, dlatego prosimy o złożenie podpisu przed rundą $r + 10$ startów. W szczególności rozpoczynają rundę walidatory $r + 10$ dopiero po obejrzeniu certyfikatu na rundę $r + 9$ i podpis dla każdego żądania wygenerowania klucza / podpisu wydanego w rundzie r. Wynik całej rundy r żądania muszą być zawarte w bloku $r + 11$. Innymi słowy, runda r propozycja blokowa, która nie zawiera wyników rundy $R - 11$ jest uważana za nieważną, a walidatorzy nie głosują na nią. Aby upewnić się, że wszystkie komunikaty progowe są podpisane przed aktualizacją zestawu walidatorów, Axelar nie wysyła żadnych żądań progowych podczas rundy równej $-1, -2, \ldots, -9$ modułów T.

5.2 Bezpieczeństwo sieci

Bezpieczeństwo systemów blockchain opiera się na różnych protokołach kryptograficznych i teorii gier, a także decentralizacji sieci. Na przykład w blockchainach typu proof-of-stake, bez odpowiednich zachęt, walidatorzy mogą zmówić się i przepisać historię, kradnąc przy tym środki innych użytkowników. W sieciach typu proof-of-work, bez wystarczającej decentralizacji, dość łatwo jest tworzyć długie forki i podwójne wydatki, co udowodniło wielokrotne ataki na Bitcoin Gold i Ethereum Classic.

Większość badań dotyczących bezpieczeństwa blockchain koncentruje się na suwerennych łańcuchach. Jednak po współdziałaniu łańcuchów należy wziąć pod uwagę nowe wektory ataków. Załóżmy na przykład, że Ethereum komunikuje się z małym blockchainem X przez bezpośredni most kontrolowany przez dwa inteligentne kontrakty, jeden na Ethereum, a drugi na X. Oprócz wyzwań inżynieryjnych, które podsumowaliśmy w rozdziale1,1, należy zdecydować, co się stanie, gdy naruszone zostaną założenia zaufania X. W takim przypadku, jeśli ETH przeniosło się do X, walidatorzy X mogą zmówić się, aby sfałszować historię X, w której przechowują wszystkie ETH, opublikować sfałszowane dowody konsensusu na Ethereum i ukraść ETH. Sytuacja jest jeszcze gorsza, gdy X jest połączony z wieloma innymi łańcuchami przez bezpośrednie mosty, gdzie jeśli X rozwidla się, efekty rozchodzą się przez każdy most. Skonfigurowanie wytycznych dotyczących zarządzania odzyskiwaniem dla każdego pomostu w parach jest przytłaczającym zadaniem dla każdego indywidualnego projektu.

Sieć Axelar rozwiązuje problemy związane z bezpieczeństwem za pomocą następujących mechanizmów:

- *Maksymalne bezpieczeństwo.* Axelar ustawia próg bezpieczeństwa na 90%, co oznacza, że prawie wszyscy walidatorzy będą musieli zmówić się, aby wypłacić wszelkie środki zablokowane przez jego sieć lub sfałszować dowody stanu[1]. W praktyce zaobserwowano, że walidatory PoS mają bardzo długi czas działania (bliski 100%), zakładając, że są odpowiednio motywowani. W związku z tym sieć Axelar będzie produkować bloki nawet pomimo tego wysokiego progu. Jednak w rzadkim przypadku, gdy coś pójdzie nie tak i sieć się zawiesi, sieć potrzebuje solidnych mechanizmów awaryjnych do ponownego uruchomienia opisanego dalej systemu.

- *Maksymalna decentralizacja.* Ponieważ sieć wykorzystuje schematy sygnatur progowych, liczba walidatorów może być tak duża, jak to możliwe. Sieć nie jest ograniczona liczbą walidatorów, które możemy obsługiwać, limitami transakcyjnymi czy opłatami, które wynikałyby z używania np. multipodpisów w różnych sieciach, gdzie złożoność (i opłaty) wzrastają liniowo wraz z liczbą walidatorów.[2]

- *Solidne mechanizmy awaryjne.* Pierwszym pytaniem, które należy rozwiązać w sieci o wysokich progach bezpieczeństwa, jak powyżej, jest to, co dzieje się, gdy sama sieć przestaje działać. Załóżmy, że sama sieć Axelar zatrzymuje się. Czy możemy mieć mechanizm awaryjny, który umożliwiłby użytkownikom odzyskanie środków? Aby rozwiązać wszelkie potencjalne przestoje samej sieci Axelar, każde progowe konto pomostowe w blockchain X, nad którym walidatory Axelar wspólnie kontrolują, ma „awaryjny klucz odblokowujący". Ten klucz można udostępnić

[1] Ostateczny parametr, który zostanie wybrany do wdrożenia sieci, można dostosować.

[2] W przypadku niektórych łańcuchów bloków multipodpisy stanowią rozsądną alternatywę tam, gdzie opłaty za gaz są małe, a obsługiwane formaty wiadomości są odpowiednie. Ale nie skalują się dla dwóch największych platform, takich jak Bitcoin i Ethereum.

na tysiącach stron, a nawet może być niestandardowym kluczem do blockchain X, który jest udostępniany całej społeczności tego łańcucha. W związku z tym, jeśli sieć Axelar utknie, ten klucz będzie działał jako rezerwa i umożliwi odzyskanie zasobów (więcej szczegółów poniżej).

- *Maksymalna decentralizacja mechanizmów awaryjnych.* Ten mechanizm awaryjny obejmuje wtórny *zestaw do odzyskiwania* użytkowników, w której każdy może uczestniczyć bez żadnych kosztów. Ci użytkownicy nie muszą być online, uruchamiać węzłów ani koordynować się nawzajem. Są „wezwani do służby" tylko wtedy, gdy sieć Axelar przestaje działać i nie może się zregenerować. Bezpieczeństwo sieci jest wzmocnione przez bardzo wysoki próg podstawowego zestawu walidatorów i maksymalnie zdecentralizowany dodatkowy zestaw odzyskiwania.
- *Wspólne zarządzanie.* Wspólny protokół rządzi siecią Axelar. Wspólnie użytkownicy mogą głosować, który łańcuch powinien być wspierany przez jego sieć. Sieć przydzieli również pulę środków, które można wykorzystać do zwrotu kosztów użytkownikom w przypadku nieoczekiwanych sytuacji awaryjnych, również kontrolowaną za pomocą protokołów zarządzania.

Poniżej omówiono różne mechanizmy bezpieczeństwa.

Mechanizmy awaryjne. Gdy Axelar zatrzymuje się z powodu wysokiego progu, kontrolę nad siecią przejmuje „awaryjny klucz odblokowujący". Istnieje wiele sposobów utworzenia instancji tego klucza odblokowującego, a niektóre sieci/aplikacje mogą zdecydować się na wykorzystanie innej odmiany „zestawu odzyskiwania" lub całkowicie zrezygnować:[3]

- *Opcja* Udostępnij klucz między fundamentami projektów blockchain i szanowanymi osobami w społeczności.
- *Opcja b.* Udostępnij między stronami wybranymi za pomocą mechanizmu delegowanego PoS.
- *Opcja c.* W przypadku kont zarządzających aktywami i informacjami dla łańcucha/aplikacji X, udostępnij niestandardowy klucz udziałowcom/walidatorom X. Zakładając, że X ma mechanizmy zarządzania, te same mechanizmy zarządzania można zastosować do określenia sposobu działania, jeśli Axelar przestanie działać.

Teraz, biorąc pod uwagę tożsamości użytkowników odzyskiwania i ich klucze publiczne, prosty protokół generuje udziały klucza odzyskiwania, których nikt nie zna. Co więcej, użytkownicy zestawu do odzyskiwania nie muszą być online, dopóki nie zostaną wezwani do odzyskiwania za pośrednictwem mechanizmów zarządzania. Zgodnie ze standardowymi protokołami rozproszonego generowania kluczy, każdy walidator Axelar dzieli losową wartość. Sekretny klucz odzyskiwania jest generowany przez zsumowanie tych wartości. Zamiast robić podsumowania w sposób jawny, wszystkie udziały są szyfrowane za pomocą kluczy publicznych użytkowników odzyskiwania, a następnie dodawane homomorficznie (zakłada to dodatkowo homomorficzne szyfrowanie i dodatkową warstwę wiedzy zerowej, z których oba są łatwo dostępne). Wynikiem tego protokołu jest odzyskanie klucza publicznego *RPK* i potencjalnie tysiące szyfrów (pod kluczami publicznymi użytkowników odzyskiwania) udziałów odpowiedniego tajnego klucza *Enc*(*si*), które są dystrybuowane do ich właścicieli (np. umieszczane w sieci). Kontrakty pomostowe Axelar obejmują opcję odzyskania środków za pomocą *RPK* pod pewnymi warunkami. Wreszcie, możliwe jest również zaktualizowanie tego klucza odzyskiwania, a nawet zmiana zestawu użytkowników posiadających jego udziały bez konieczności jakiejkolwiek pracy ze strony uczestniczących udziałowców.

Jeśli łańcuch X połączony z Axelar pęknie, istnieje kilka możliwości:

- Nałóż limity na wartość aktywów w USD, które można przenosić do/z X każdego dnia. W ten sposób złośliwy łańcuch X może ukraść tylko niewielki ułamek wszystkich zasobów, które są z nim połączone, zanim weryfikatory Axelar to wykryją i uruchomią mechanizmy zarządzania z poniższych pocisków.
- Moduł zarządzania Axelar może być używany do głosowania nad tym, co dzieje się w takich sytuacjach. Na przykład, jeśli pojawi się łagodny błąd i społeczność zrestartuje X, zarządzanie Axelarem może zdecydować o ponownym uruchomieniu połączenia od miejsca, w którym zostało przerwane.
- Jeśli ETH zostało przeniesione do X, niestandardowy klucz odzyskiwania Ethereum może określić, co stanie się z zasobami ETH.

[3] Ostateczne wdrożenie w sieci Axelar zostanie sfinalizowane bliżej uruchomienia sieci.

6 Protokół bramy krzyżowej (CGP)

W tej sekcji wyjaśnimy protokół bramy krzyżowej i mechanizmy routingu na dwóch podstawowych przykładach wspólnych dla potrzeb wielu aplikacji:

Synchronizacja stanu (sekcja 6,2). Opublikuj informacje o stanie łańcucha bloków źródłowych *S* do stan docelowego łańcucha bloków *D*.

(Na przykład opublikuj nagłówek bloku Bitcoin w łańcuchu blokowym Ethereum.)

Przeniesienie aktywów (sekcja 6,3). Przenieś zasób cyfrowy z *S* do *D* iz powrotem.

(Na przykład przenieś bitcoiny z łańcucha bloków Bitcoin do łańcucha bloków Ethereum, a następnie z powrotem do łańcucha bloków Bitcoin.)

Dla uproszczenia zakładamy, że łańcuch *D* ma przynajmniej minimalne wsparcie dla inteligentnych kontraktów, ale *S* może być dowolnym blockchainem.

6.1 Konta w innych sieciach

Aby połączyć różne łańcuchy, w każdym łańcuchu tworzone są konta progowe, które kontrolują przepływ wartości i informacji przez nie. Do łańcucha*Łańcuch*, oznacza konto przez *Łańcuch*$_{Axelar}$.

Konto Bitcoin. W przypadku Bitcoin i innych nieinteligentnych łańcuchów kontraktów walidatory Axelar tworzą klucz progowy ECDSA zgodnie z sekcją 5.1. Ten klucz kontroluje konto ECDSA na Bitcoin i jest adresem docelowym, na który użytkownicy wysyłają depozyty. Spersonalizowane klucze progowe mogą być tworzone na żądanie użytkownika. Klucz może być okresowo aktualizowany, a najnowszy klucz i spersonalizowane klucze można znaleźć, odpytując węzeł Axelar.

Progowe konto pomostowe w sieciach z inteligentnymi kontraktami. Oznacz łańcuch przez SC. walidatory tworzą próg ECDSA lub ED25519 klucz zgodnie z sekcją5.1, w zależności od typu klucza obsługiwanego przez łańcuch. Oznaczamy ten klucz przezPK_{Axelar}, gdy nie ma dwuznaczności, do którego łańcucha się odnosimy. Ten klucz kontroluje konto inteligentnej umowy na SC, oznaczone przezSC_{Axelar}, a każda aplikacja w SC może wysyłać zapytania SC_{Axelar} aby poznać adres PK tego klucza. W ten sposób każda aplikacja SC może rozpoznać wiadomości podpisane przezSK_{Axelar}. Protokół musi również uwzględniać wartości rotacyjnePK_{Axelar}. Dzieje się to w następujący sposób:

1. Zainicjuj SC_{Axelar} na SC. PrzechowujePK_{Axelar} jako część jego stanu, który jest inicjowany jako jego wartość genezy na Axelarze. SC_{Axelar} zawiera również zasady aktualizacji PK.
2. Aktualizacja PK_{Axelar}, transakcja w formacie (*aktualizacja,* PK_{Nowy}) należy złożyć z podpisem z aktualnego SK_{Axelar}. Wtedy umowa się ustala$PK_{Axelar} = PK_{Nowy}$.
3. Za każdym razem, gdy walidatory aktualizują klucz progu dla SC z PK_i do PK_{i+1}, Axelar żąda, których używają walidatorzy SK_i podpisać (*aktualizacja,* PK_{i+1}). Następnie podpis ten jest wysyłany doSC_{Axelar} które aktualizacje PK_{Axelar}.

6.2 Synchronizacja stanu

Pozwalać Q_s oznaczają arbitralne pytanie o stan łańcucha *S*. Przykłady takich pytań obejmują:

- *"W jakiej rundzie bloku, jeśli w ogóle, pojawiła się transakcja tx?"*
- *"Jaka jest wartość określonego pola danych?"*
- *"Czym jest hash korzenia Merkle dla całego stanu S w rundzie bloku 314159?"*

Pozwalać a_s oznaczają poprawną odpowiedź na Q_s i załóżmy, że użytkownik końcowy lub aplikacja wymaga tego a_s być wysłana do łańcucha *D*. Sieć Axelar spełnia to zapotrzebowanie w następujący sposób:

1. Użytkownik wysyła prośbę Q_S na jednym z kont pomostowych (które są następnie pobierane przez walidatory) lub bezpośrednio na blockchainie Axelar.

2. W ramach konsensusu Axelar każdy walidator musi uruchomić oprogramowanie węzła dla łańcuchów *S*, *D*. Walidatory Axelar odpytują API swojego łańcucha S oprogramowanie węzła dla odpowiedzi a_S i zgłoś odpowiedź do sieci Axelar.

3. Raz $> F$ walidatory ważone zgłaszają tę samą odpowiedź w rundzie r, Axelar prosi walidatorów o podpisanie a_S.

4. Używając kryptografii progowej, walidatory podpisują a_S. Podpis jest zawarty w bloku $r + 11$.

5. Każdy może przyjąć wartość ze znakiem a_S z bloku $r + 11$ i wyślij do *D*.

6. Żądanie zostało obsłużone. Dowolna aplikacja włączona D może teraz przyjmować wartość ze znakiem a_S, zapytanie D_{Axelar} na najnowsze PK_{Axelar} i sprawdzić, czy podpis a_S koresponduje z PK_{Axelar}. Walidatorzy również publikują a_S na konto pomostowe na łańcuchu *D*, które aplikacje mogą pobierać.

6.3 Międzyłańcuchowy transfer aktywów

Sieć umożliwia międzyłańcuchowe transfery zasobów cyfrowych poprzez rozszerzenie przepływu pracy synchronizacji stanu Sekcji 6,2.

Wystarczająca podaż sztywnych S tokeny są drukowane i kontrolowane przez D_{Axelar} po jego inicjalizacji. Załóżmy, że użytkownik żąda wymiany x ilość tokenów w łańcuchu źródłowym *S* dla x ilość zakotwiczonych-S tokeny w łańcuchu docelowym *D*, do złożenia w *D*-adres w_D do wyboru użytkownika. Przedstawiamy w pełni ogólny przepływ pracy, który obsługuje dowolne łańcuchy źródeł S—nawet sieci takie jak Bitcoin, które nie obsługują inteligentnych kontraktów:

1. Użytkownik (lub aplikacja działająca w jego imieniu) przesyła żądanie przelewu (x, w_D) na progowe konto pomostowe, które jest następnie kierowane do sieci Axelar.

2. Walidatory Axelar wykorzystują kryptografię progową do wspólnego tworzenia nowego adresu depozytowego D_S dla *S*. publikują D_S do łańcucha bloków Axelar.

3. Użytkownik (lub aplikacja działająca w jego imieniu) uczy się D_S monitorując blockchain Axelar. Użytkownik wysyła x ilość *S*-tokeny do zaadresowania D_S przez zwykły *S*-transakcja tx_S używa swojego ulubionego oprogramowania do sieci *S*.

   (Ze względu na właściwość progu D_S, żetony nie mogą być wydawane od D_S chyba że liczba progowa walidatorów koordynuje, aby to zrobić.)

4. tx_S jest publikowany na Axelar. Walidatorzy odpytują API swojego łańcucha S oprogramowanie węzła dla istnienia tx_S a jeśli odpowiedź jest „prawda”, zgłoś odpowiedź do łańcucha Axelar.

5. Raz $> F$ walidatory ważone raportują „prawda” dla tx_S w rundzie r, Axelar prosi walidatorów o podpisanie transakcji a_D to wysyła x ilość zakotwiczonych-S tokeny z D_{Axelar} do w_D.

6. Używając kryptografii progowej, walidatory podpisują a_D. Podpis jest zawarty w bloku $r + 11$.

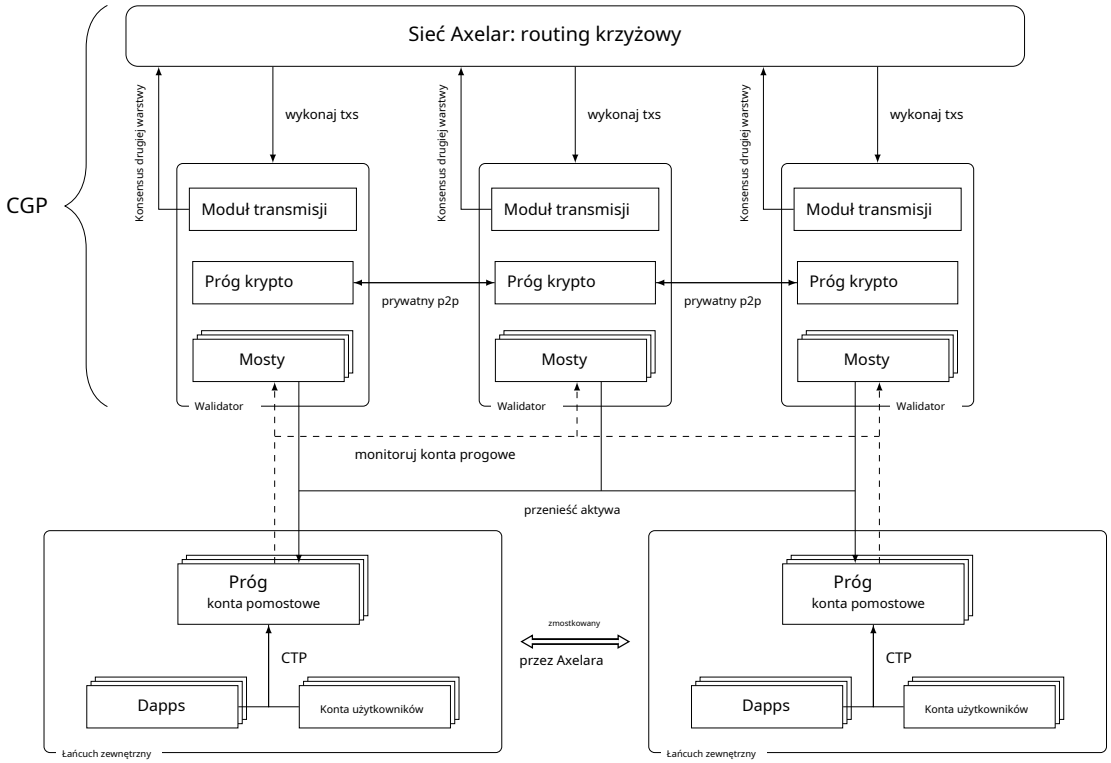
7. Każdy może przyjąć wartość ze znakiem a_D z bloku $r + 11$ i wyślij do *D*.

8. Żądanie zostało obsłużone, raz a_D jest opublikowany D przelew jest przetwarzany.

Załóżmy teraz, że użytkownik żąda umorzenia $x_?$ ilość owiniętych-S tokeny z łańcucha D z powrotem do łańcucha *S*, do złożenia w *S*-adres w_S do wyboru użytkownika. Przepływ pracy wygląda następująco:

1. Użytkownik inicjuje żądanie przelewu ($x_?$, w_S) deponując $x_?$ ilość owiniętych-S tokeny do C_D przez zwykły *D* -transakcja przy użyciu jej ulubionego oprogramowania dla sieci *D*.

2. ($x_?$, w_S) jest publikowany na Axelar. Walidatorzy odpytują API swojego łańcucha D oprogramowanie węzła dla istnienia ($x_?$, w_S) i jeśli odpowiedź jest „prawda”, zgłoś odpowiedź do łańcucha Axelar.



Rysunek 1: Schemat komponentów

3. Raz *> F* walidatory ważone zgłaszają „prawda” dla (*x?, ws*) w rundzie *r*, Axelar prosi walidatorów o podpisanie transakcji *as* to wysyła *x?* ilość *S* tokeny z *SAxelar* do *ws*.
4. Używając kryptografii progowej, walidatory podpisują *as*. Podpis jest zawarty w bloku *r* + 11.
5. Każdy może przyjąć wartość ze znakiem *as* z bloku *r* + 11 i wyślij do *S*.
6. Żądanie zostało obsłużone, raz *as* jest opublikowany *S* przelew jest przetwarzany.

Dodatkowe żądania obsługiwane przez warstwę routingu CGP obejmują blokowanie, odblokowywanie lub przesyłanie zasobów między łańcuchami.

Osiągnięcie przepływu transakcji między łańcuchami atomowymi. W zależności od typu żądania cross-chain, Axelar stara się zapewnić, że odpowiednie transakcje są wykonywane w wielu łańcuchach lub w żadnym. W tym celu każde żądanie może znajdować się w jednym z następujących stanów w blockchainie Axelar:*(zainicjowane, oczekujące, zakończone, przekroczony limit czasu)*. Jeśli *koniec czasu* na etapie oczekiwania jest wyzwalane, żądanie zwraca kod błędu. Niektóre zdarzenia z limitem czasu również zaczynają się *zwrot* zdarzenie: na przykład, jeśli aktywa z jednego łańcucha muszą zostać przeniesione do aktywów w innym łańcuchu, jeśli łańcuch otrzymujący nie przetworzył transakcji, aktywa są zwracane pierwotnemu użytkownikowi.

7 Protokół transferu między łańcuchami (CTP)

CTP to protokół na poziomie aplikacji, który ułatwia aplikacjom korzystanie z funkcji cross-chain. Wyjaśniamy integrację, koncentrując się na funkcjach transferu aktywów (np. używanych w DeFi). Aplikacje te zazwyczaj składają się z trzech głównych komponentów: interfejsu graficznego interfejsu użytkownika, inteligentnych kontraktów w jednym łańcuchu oraz węzła pośredniczącego, który publikuje transakcje między interfejsem użytkownika a inteligentnymi kontraktami. Interfejsy front-end wchodzą w interakcję z portfelami użytkownika, aby akceptować wpłaty, przetwarzać wypłaty itp. Aplikacje mogą wykorzystywać funkcje cross-chain

wywołując zapytania CTP analogicznie do metod HTTP/HTTPS GET/POST. Zapytania te są następnie pobierane przez warstwę CGP w celu wykonania, a wyniki są zwracane użytkownikom.

- *Zapytania CTP*. Deweloperzy aplikacji mogą hostować swoje aplikacje w dowolnym łańcuchu i integrować swoje inteligentne kontrakty z kontami mostu progowego w celu wykonywania zapytań CTP.
- *Progowe konta pomostowe*. Załóżmy, że twórca aplikacji buduje swoje kontrakty w łańcuchu A. Następnie odwołuje się do kontraktów pomostowych progowych, aby uzyskać wsparcie międzyłańcuchowe. Niniejsza umowa umożliwia aplikacjom:
  - Zarejestruj łańcuch bloków, z którym chciałby się komunikować.
  - Zarejestruj aktywa w tym blockchainie, które chciałby wykorzystać.
  - Wykonuj operacje na aktywach, takie jak przyjmowanie wpłat, przetwarzanie wypłat i inne funkcje (podobne do, powiedzmy, wywołań kontraktów ERC-20).

Załóżmy, że znana aplikacja DeFi, MapleSwap, natywnie znajduje się w rejestrach łańcucha A z progowym kontem pomostowym. Walidatorzy Axelar wspólnie zarządzają samą umową w odpowiednim łańcuchu. Załóżmy, że użytkownik chce złożyć depozyt w parze handlowej między aktywami X i Y, które znajdują się odpowiednio w dwóch łańcuchach. Następnie, gdy użytkownik prześle takie żądanie, jest ono kierowane przez konto mostu progowego do sieci Axelar w celu przetworzenia. W tym formularzu wykonywane są następujące kroki:

1. Sieć Axelar rozumie, że ta aplikacja została zarejestrowana do obsługi cross-chain w aktywach. Generuje klucz depozytów wykorzystując kryptografię progową i konsensus dla użytkownika w odpowiednich łańcuchach A i B.
2. Powiązane klucze publiczne są zwracane do aplikacji i wyświetlane użytkownikowi, który może używać swoich ulubionych portfeli do dokonywania wpłat. Odpowiedni tajny klucz jest współdzielony przez wszystkie walidatory Axelar.
3. Po potwierdzeniu depozytów Axelar aktualizuje swój katalog międzyłańcuchowy, aby odnotować, że użytkownik w odpowiednich sieciach zdeponował te aktywa.
4. Walidatory Axelar wykonują protokoły wielostronne w celu wygenerowania sygnatury progu, która umożliwia aktualizację konta pomostowego progu w łańcuchu A, w którym znajduje się aplikacja.
5. Zapytanie CTP jest następnie zwracane do inteligentnych kontraktów aplikacji DeFi, które mogą aktualizować jego stan, aktualizować formuły zysku, kursy wymiany lub wykonywać inne warunki związane ze stanem aplikacji.

Podczas tego procesu sieć Axelar działa na wysokim poziomie jako zdecentralizowana, międzyłańcuchowa wyrocznia odczytu/zapisu, CGP jest warstwą routingu pomiędzy łańcuchami, a CTP jest protokołem aplikacji.

Dodatkowe żądania cross-chain. CTP obsługuje bardziej ogólne cross-chain między aplikacjami w blockchainach, takie jak:

- Wykonaj usługi nazw kluczy publicznych (PKNS). Jest to uniwersalny katalog do mapowania kluczy publicznych na numery telefonów/uchwyty na Twitterze (kilka projektów, takich jak Celo, udostępnia te funkcje na swoich platformach).
- Wyzwalacze aplikacji cross-chain. Aplikacja w łańcuchu A może zaktualizować swój stan, jeśli inna aplikacja w łańcuchu B spełnia kryteria wyszukiwania (stopa procentowa $< x$).
- Inteligentne komponowanie kontraktów. Inteligentny kontrakt w łańcuchu A może zaktualizować swój stan na podstawie stanu kontraktów w łańcuchu B lub wywołać akcję aktualizacji inteligentnego kontraktu w łańcuchu B.

Na wysokim poziomie żądania te mogą być przetwarzane, ponieważ łącznie protokoły CTP, CGP i Axelar mogą przekazywać i zapisywać dowolne, weryfikowalne informacje o stanie w łańcuchach bloków.

8 Podsumowanie

W ciągu najbliższych lat znaczące aplikacje i zasoby zostaną zbudowane na wielu ekosystemach blockchain. Sieć Axelar może być wykorzystana do podłączenia tych łańcuchów bloków do jednolitej warstwy komunikacji między łańcuchami. Ta warstwa zapewnia protokoły routingu i na poziomie aplikacji, które spełniają wymagania zarówno konstruktorów platform, jak i twórców aplikacji. Twórcy aplikacji mogą budować na najlepszych platformach dla swoich potrzeb i korzystać z prostego protokołu i interfejsu API, aby uzyskać dostęp do globalnej płynności między łańcuchami, użytkowników i komunikować się z innymi łańcuchami.

Bibliografia

[1] Peggy Althei. https://github.com/cosmos/peggy. [Cytowane na stronie 2.]

[2] Deterministyczne wykorzystanie algorytmu podpisu cyfrowego (dsa) i algorytmu podpisu cyfrowego z krzywą eliptyczną (ecdsa). https://tools.ietf.org/html/rfc6979. [Cytowane na stronie 5.]

[3] Algorytm podpisu cyfrowego opartego na krzywej Edwardsa (eddsa). https://tools.ietf.org/html/rfc8032. [Cyt strona 5.]

[4] Dokumentacja techniczna Eos.io v2. https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md. [Cytowane na stronie 1.]

[5] Ethereum: bezpieczna zdecentralizowana uogólniona księga transakcji. https://ethereum.github.io/yellowpaper/paper.pdf. [Cytowane na stronie 1.]

[6] Niemal biały papier. https://near.org/papers/the-official-near-white-paper/. [Cytowane na stronie 1.]

[7] Tęczowy most. https://github.com/near/rainbow-bridge. [Cytowane na stronie 2.]

[8] Ren: Zachowująca prywatność maszyna wirtualna obsługująca aplikacje finansowe o zerowej wiedzy. // https: whitepaper.io/document/419/ren-litepaper. [Cytowane na stronie 3.]

[9] tbtc: zdecentralizowany, wymienialny token erc-20 wspierany przez btc. https://docs.keep.network/tbtc/index.pdf. [Cytowane na stronie 2.]

[10] Thorchain: zdecentralizowana sieć płynności. https://thorchain.org/. [Cytowane na stronie 3.]

[11] Kurt M. Alonso. Zero do monety.https://www.getmonero.org/library/Zero-to-Monero-1-0-0.pdf. [Cytowane na stronie 1.]

[12] Jean-Philippe Aumasson, Adrian Hamelink i Omer Szlomowits. Badanie podpisywania progów ecdsa. Archiwum e-druku kryptologicznego, Raport 2020/1390, 2020.https://eprint.iacr.org/2020/1390. [Cyt strona 6.]

[13] Ran Canetti, Nikolaos Makriyannis i Udi Peled. Archiwum e- Uc nieinteraktywne, proaktywne, progowe druku kryptologicznego, Raport 2020/492, 2020. ecdsa. https://eprint.iacr.org/2020/492. [Cyt strona 6.]

[14] Białe księgi cLabs. https://celo.org/papers. [Cytowane na stronie 1.]

[15] Ivan Damgård, Thomas Pelle Jakobsen, Jesper Buus Nielsen, Jakob Illeborg Pagter i Michael Bæksvang Østergård. Szybki próg ECDSA ze uczciwą większością. w*SCN*, tom 12238 z *Notatki z wykładów z informatyki*, strony 382-400. Springer, 2020.[Cytowane na stronie 6.]

[16] Manu Drijvers, Kasra Edalatnejad, Bryan Ford, Eike Kiltz, Julian Loss, Gregory Neven i Igors Stepanovs. O bezpieczeństwie dwurundowych multipodpisów. w*Sympozjum IEEE na temat bezpieczeństwa i prywatności*, strony 1084-1101. IEEE, 2019.[Cytowane na stronie 6.]

[17] Cynthia Dwork, Nancy Lynch i Larry Stockmeyer. Konsensus w obecności częściowej synchronizacji. https://groups.csail.mit.edu/tds/papers/Lynch/jacm88.pdf. [Cytowane na stronie 5.]

[18] Rosario Gennaro i Steven Goldfeder. Jeden okrągły próg ecdsa z możliwym do zidentyfikowania przerwaniem. Archiwum e-druku kryptologicznego, Raport 2020/540, 2020.https://eprint.iacr.org/2020/540. [Cytowane na stronie 6.]

[19] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos i Nickolai Zeldovich. Algorand: Skalowanie bizantyjskich umów dla kryptowalut. Materiały XXVI Sympozjum Zasad Systemów Operacyjnych, 2017.https://dl.acm.org/doi/pdf/10.1145/31327477.3132757. [Cytowane na stronie 1.]

[20] Evan Kereiakes, Do Kwon, Marco Di Maggio i Nicholas Platias. Terra money: stabilność i adopcja. https://terra.money/Terra_White_paper.pdf. [Cytowane na stronie 1.]

[21] Aggelos Kiayias, Alexander Russell, Bernardo David i Roman Oliynykov. Ouroboros: Bezpieczny protokół blockchain typu „proof-of-stake".https://eprint.iacr.org/2016/889.pdf. [Cytowane na stronie 1.]

[22] Chelsea Komlo i Ian Goldberg. Frost: Elastyczne sygnatury progowe schnorr zoptymalizowane pod kątem zaokrągleń. Archiwum e-druku kryptologicznego, Raport 2020/852, 2020.https://eprint.iacr.org/2020/852. [Cytowane na stronie 6.]

[23] Jae Kwon i Ethan Buchman. Cosmos: sieć rozproszonych ksiąg. https://cosmos.network/resources/whitepaper. [Cytowane na stronach 1 oraz 2.]

[24] Zespół lawinowy. Platforma lawinowa. https://www.avalabs.org/whitepapers. [Cytowane na stronach 1 oraz 2.]

[25] Gavin Wood. Polkadot: Wizja heterogenicznej struktury wielołańcuchowej. https://polkadot.network/PolkaDotPaper.pdf. [Cytowane na stronach 1 oraz 2.]