ОСОБЕННОСТИ ПРИМЕНЕНИЯ АЛГОРИТМА ЦИФРОВОЙ ПОДПИСИ ED25519 В РАСПРЕДЕЛЕННОЙ ПЛАТФОРМЕ SOLANA

О.Л. Крицкий к.ф.-м.н., доцент ОЭФ ТПУ
В.А. Карнаухов, студент гр. 0ВМ02
Томский политехнический университет
E-mail: vak65@tpu.ru

Введение

В данной статье будет рассматриваться Solana и алгоритм цифровой подписи ed25519. Solana – это развивающийся проект, который представляет из себя блокчейн и платформу для децентрализованного исполнения программ [1].

Одной из главных целей Solana является достижение высокой скорости работы сети, сопоставимой скорости централизованных сетей, а особенность алгоритма ed25519, которую использует Solana способствует ей в этом.

Аккаунты и on-chain программы

Одной из функциональных единиц Solana являются аккаунты. Аккаунт состоит из 32-байтной последовательности, называемой адресом и являющейся публичным ключом цифровой подписи ed25519, а также соответствующей структуры данных, существующей по этому адресу.

On-chain программа - программа, исполняемая распределено. Это означает, что при вызове она исполняется несколькими серверами (валидаторами) сети и истинный результат исполнения определяется путём принятия консенсуса. В других сетях эти программы называются смарт-контрактами.

Любой клиент, зная адрес программы, может собрать список аккаунтов, которые хочет передать ей, сформировать инструкцию и с помощью запроса к одному из серверов сети Solana «вызвать» программу с данными аргументами.

У передаваемых аккаунтов есть следующие флаги:

- is_signer - флаг, означающий факт подписания транзакции, вызывающий эту программу, приватным ключом данного аккаунта;
- is_writable - флаг, разрешающий программе менять поля этого аккаунта.

Обзор алгоритма ed25519

Алгоритм цифровой подписи ed25519 определяется выбором семи параметров:

1. Целое число $b \geqslant 10$;
2. Криптографическая хэш-функция H, дающая на выходе $2b$-битное число;
3. Простое число $q \approx 1 \pmod 4$, задающее поле Галуа $GF(q)$;
4. Число $d \in GF(q)$, не являющееся квадратом в этом поле (это также называют квадратичный невычет);
5. Точка $B \in E$, $B \neq (0, 1)$, где E – это, группа точек эллиптической кривой:
$$E = \{(x, y) \in GF(q) \cdot GF(q) : -x^2 + y^2 = 1 + dx^2y^2\}; \quad (1)$$
6. Простое $l \in [2^{b-4}; 2^{b-3}]$ такое, что $lB = (0,1)$ (нейтральный элемент группы);
7. Способ b-1-битного кодирования элементов $GF(q)$.

Данная эллиптическая кривая называется скрученной кривой Эдвардса (Twisted Edwards Curve) [2]. Групповой закон на ней выглядит следующим образом:

$$(x_1,y_1) + (x_2,y_2) = (\frac{x_1y_2 + x_2y_1}{1 + dx_1x_2y_1y_2}, \frac{x_1y_2 + x_2y_1}{1 - dx_1x_2y_1y_2}. \quad (2)$$

Далее определим некоторые числа из $GF(q)$, как «отрицательные». Для выбранного способа кодирования будем считать x отрицательным, если кодирование x лексикографически больше кодирования q-x. Например для $q = 13$ в поле $GF(q) = \{0, 1, 2, 3, ..., 12\}$ при little-endian кодировании отрицательными будут считаться $\{1, 3, 5, 7, 9, 11\}$. Таким образом для кодирования точки на кривой

(x,y) достаточно *b* бит: *b-1* бит для кодирования *y* и один бит для знака *x*, а сам *x* можно вычислить (в поле *GF(q)* , разумеется):

$$x = \pm\sqrt{(y^2 - 1)/(dy^2+1)}. \quad (3)$$

Реализация ключей, подписей и проверки

Сгенерируем случайным образом *b*-битное число *k*, которое будет являться приватным ключом. Возьмём от него хэш $H(k) = (h_0, h_1, h_2, \ldots, h_{2b-1})$, где h_i - биты вычисленного хэша. Вычислим число *a* следующим образом:

$$a = 2^{b-2} + \sum_{3 \le i \le b-3} 2^i h_i \quad (4)$$

Публичным ключом $\underline{A}$ будем считать кодирование точки $A = aB$ (далее кодирования точек будут обозначаться подчёркиванием). Ввиду групповой структуры эллиптической кривой точка *A* также лежит на этой кривой. Для подписи сообщения *M* необходимо вычислить число $r = H(h_b, h_{b+1}, \ldots, h_{2b-1}, M)$, затем найти точку $R = rB$ и число $S = (r + H(\underline{R}, \underline{A}, M)) \bmod l$. Сигнатурой будем считать *2b*-битную строку $(\underline{R}, \underline{S})$.

Для верификации подписи проверяющая сторона должна восстановить: точку $R \in E$, точку $A \in E$, целое число $S \in [0; l-1]$ и проверить равенство $SB = R + H(\underline{R}, \underline{A}, M) A$.

Выясним корректность такой проверки.

Домножим *B* на *S*:

$$SB = ((r + H(\underline{R}, \underline{A}, M)\, a) \bmod l)B. \quad (5)$$

Пусть $S^* = r + H(\underline{R}, \underline{A}, M)$ a, тогда для некоторого целого n справедливо:

$$S^* = nl + S^* \bmod l, \quad (6)$$

$$S^* \bmod l = S^* - nl. \quad (7)$$

В терминах S^* исходное выражение примет следующий вид:

$$SB = (S^* - nl)B \quad (8)$$

$$SB = S^* B - nlB \quad (9)$$

Но *l* и *B* выбраны так, что $lB = (0,1)$ - нейтральный элемент, следовательно:

$$SB = S^*B \quad (10)$$

$$SB = rB + H(\underline{R}, \underline{A}, M)\, aB \quad (11)$$

$$SB = R + H(\underline{R}, \underline{A}, M)\, A \quad (12)$$

Программно-производные адреса

On-chain программа P_1 может быть вызвана не только каким-то off-chain клиентом. Она также может быть вызвана другой on-chain программой P_2. При этом P_1 может передать в P_2 только те аккаунты, которые есть среди переданных в P_1. Представим, что в P_1был передан аккаунт A, поле is_signer которого равно false. Если этот аккаунт обладает особым адресом, то P_1 может передать его в P_2 с is_signer: true. Такие особые адреса в Solana называются PDA (Program derived address) - адреса, для которых не существует приватного ключа ed25519. Если точка *A* не лежит на кривой, с точки зрения ed25519 означает, что не существует приватного ключа такого, что подпись сообщения этим ключом проходила бы проверку относительно публичного ключа $\underline{A}$. Если для полученного адреса не существует приватного ключа, а также он зависит от адреса программы, то рантайм Solana может проверить, что адрес аккаунта с полем is_signer: false действительно вычисляется из адреса вызывающей программы и указанных последовательностей байтов, и передать вызываемой программе этот аккаунт с полем is_signer: true. Таким образом в Solana программа может быть в некотором смысле физическим владельцем аккаунта.

Заключение

В ходе данной работы был рассмотрен один из основополагающих механизмов платформы Solana, который позволяет программам самостоятельно, без участия оператора(человека) подписывать

и вызывать межпрограммные инструкции, на основе данного механизма строится все программы в Solana.

Список использованных источников:

1. Solana официальная документация [Электронный ресурс] https://docs.solana.com (дата обращения 15.02.2022).
2. О криптографии на эллиптических кривых [Электронный ресурс] https://habr.com/ru/post/335906/ (дата обращения 16.02.2022).