Виды ЭЦП: Кольцевая подпись. Концепт и применимость.

Для начала о групповых подписях

- В первую очередь стоит сказать, что **кольцевые подписи** (Ring Signature) можно рассматривать как **продолжение идеи групповых подписей**, поэтому сначала о них.

- На уровне концепта **механизм групповых подписей** позволяет члену определенной группы **подписать сообщение** от **имени всей группы** в целом и **без раскрытия** своей **личности**.

- Ну а если же **более формально**, то групповые подписи обладают **следующими свойствами**:

    1) **Только член группы** может подписать сообщение;

    2) Получатель подписи может убедиться в том, что он **получил валидную подпись** от имени группы. Однако **узнать, кто именно поставил** эту подпись, он **не сможет**;

    3) Подпись **может быть открыта** с помощью Менеджера группы. Это может, например, понадобиться, когда группа хочет узнать личность подписанта.

- Если **первые 2** свойства нам в целом **никак не могут навредить**, то вот последнее выглядит не очень хорошо. У нас явно есть **зависимость от действий Менеджера**.

  Более того, лишь у Менеджера есть прерогатива добавления новых участников в группу (в некоторых системах это право передают **Membership Manager'**у - но всё равно проблема та же).

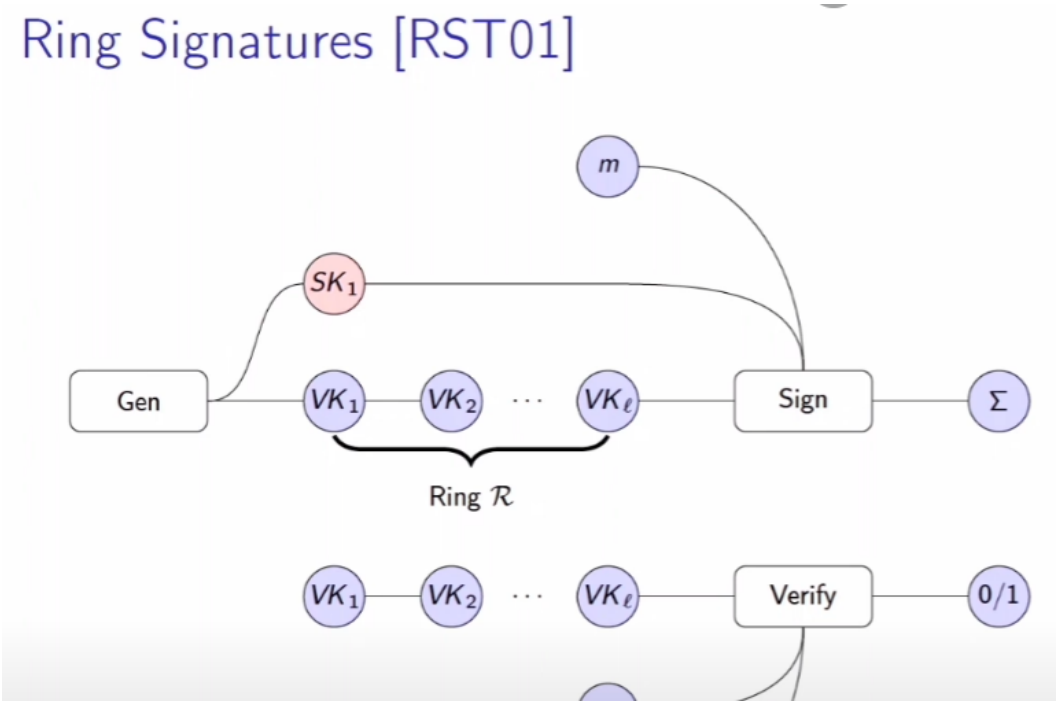
- Но это **не единственная проблема** с групповыми подписями: у них **нет гибкости** при подписывании. **В момент подписания** все члены группы фактически **фиксированы** и **статичны** - группы не могут формироваться на разовой основе, по подписям.

Так в чем преимущество кольцевых?

- Во-первых, они-то как раз **дают** нам **нужную гибкость**: подписывающий может **сам определить кольцо** (то есть подгруппу) в момент подписания. После чего эта подпись будет **обладать** теми самыми **2 "хорошими" свойствами**, которые нам нравятся в групповых подписях. (***No setup)***

- Ну и во-вторых, кольцевые подписи **выносят Менеджера за скобки** - здесь он не нужен. Ведь каждый пользователь может **сам набрать** в своё кольцо **кого угодно**, при этом он **не обязан повторяться** и выбирать одних и тех же. (***True anonymity!)***

- Таким образом, **единственное требование** кольцевых подписей к системе состоит в том, что **каждый участник** должен иметь **опубликованный открытый ключ**.



Детали реализации и использования

- Допустим, у нас есть **группа**, каждый участник которой имеет **пару публичного** и **секретного** ключа: ***(P1, S1), (P2, S2), ..., (Pn, Sn).***

- Если i-тый участник захочет подписать сообщение (обозначим ***message***), то он будет использовать только следующий набор данных

  (***message***, ***Si***, ***P1***, ***P2***, ***..., Pn***) - здесь ***Si*** его секретный ключ.

- Теперь **что делать** конкретно по пунктам:

  1) **Взять** сообщение, **найти хэш** и **обозначить** его **как ключ**: **k** = ***Hash(message)***
     Этот ключ (***Ek = encryption key***) будет использоваться с для **шифрования каждого из элементов** кольца.

  2) Каким-то образом сгенерировать **рандомное число *V*** (на деле генерируется другое число ***u***, мы просто ещё пропускаем его через Ek, но не суть важно).

  3) Далее **для каждого участника** кольца (кроме того, кто реально ставит подпись) делается следующее:

     a) Мы берем **якобы секретный ключ** i-того участника = ***Si*** (**в реальности** мы его точно так же **генерируем**) и вычисляем для него значение ***e***:

$$e = s_i^{P_i} \pmod{N_i}$$

(к слову, здесь ***Ni*** скорее всего является каким-то **большим простым числом**, но об этом точно не сказано)

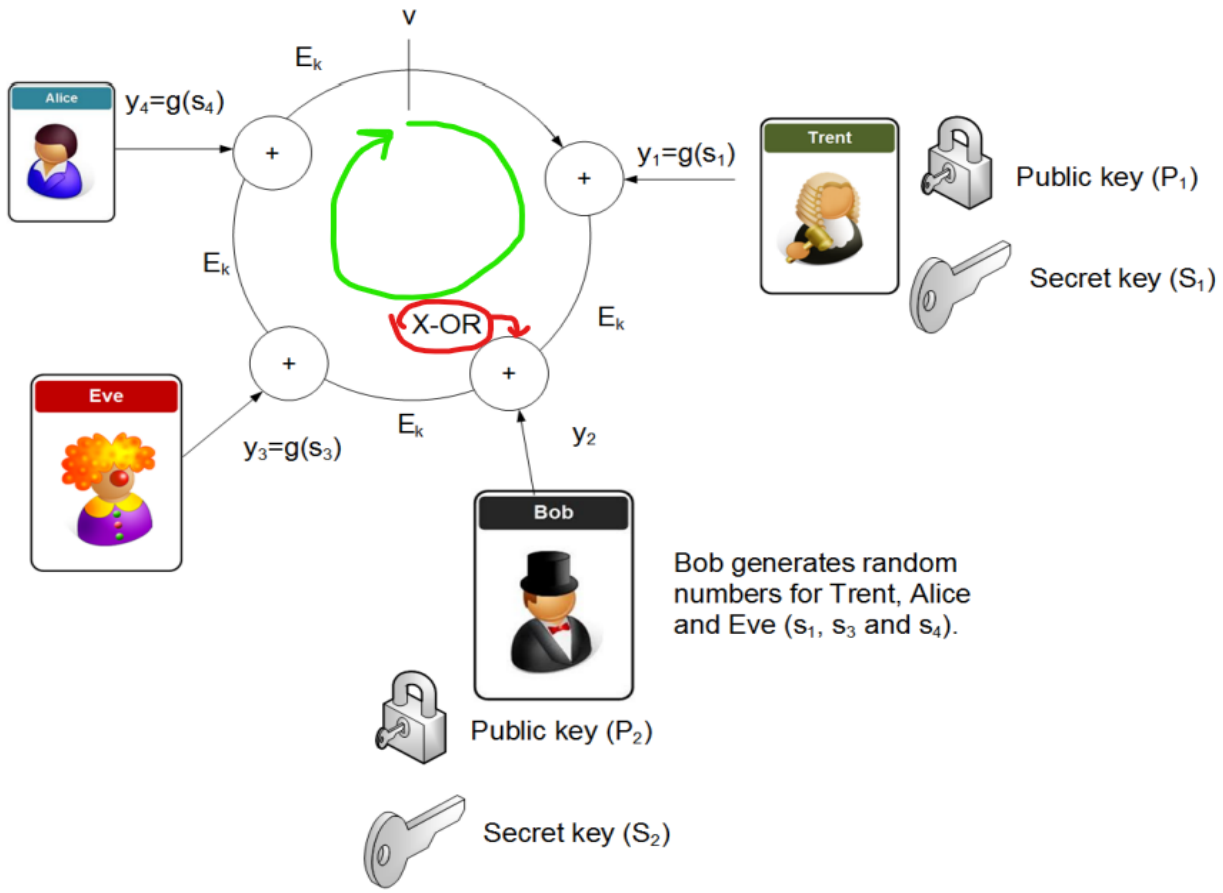
b) Далее мы берем это значение **e** и **XOR**-им с **V**. При этом этот результат дальше и считаем как V.

4) На последнем этапе мы получаем, наконец, нужное значение ***V***, используя свой **приватный ключ *d*** и то самое число, которое мы **изначально генерировали** (***u***):

$$v = E_k(u \oplus v)^d$$

Да, то есть ***d*** - это **единственный реальный секретный ключ**, все **остальные фейковые,** мы сами их генерировали в процессе.

- Это и есть алгоритм генерации подписи. Если в виде отображать **его в виде картинки**, то получится что-то примерно такое:

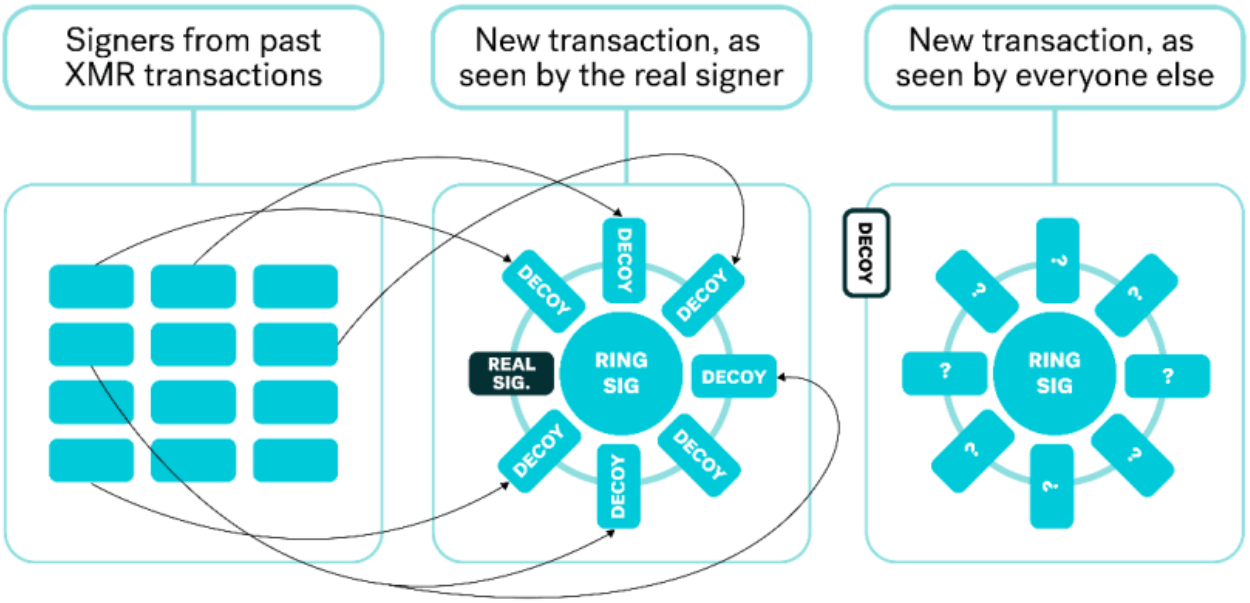


- Выше **зелёным** я показал **направление обхода** (то есть в каком именно порядке мы совершаем операции), а красным - обозначил, что на каждом участнике мы выполняем **XOR** с нашим текущим **v**.

Переходим к главному применению - **Monero**!

- Итак, в Монеро кольцевые подписи используются в качестве **механизма конфиденциальности отправителя** транзакции. Как именно это работает?

  Для начала кратко: мы просто **набираем адресов** другого народа и это используется **как обманка,** то есть отправителем **потенциально мог быть любой из них**.



Теперь подробнее:

- Прежде всего мы создаем транзакцию, **подписываем** её **одноразовым ключом** и привязываемся к нашему **реальному выходу**.

- **Остальные выходы** мы **набираем из чейна** (в оф. доке написано, что делается это треугольным распределением). К слову, эти же выходы могут **использоваться** потом **ещё и не раз** (***от себя***: *на деле странное решение, но ладно*).

- **Все выходы** кольцевой подписи после этого вместе составляют **вход транзакции**. Процесс **завершён**. Со стороны отличить реальный перевод от фейковых сложно, однако в теории возможно (если **других участников** банально **было мало**).

- То есть ***реально*** понять, какие средства были потрачены, а какие нет - просто так нельзя. Звучит хорошо, но **только это** порождает классическую **проблему,** которая называется ***double spend :)***

История о том, как **Monero** победила ***double spend***

- (***от себя**: Формально это не относится к основной теме доклада, но **не рассмотреть** это **непростительно***).

- Итак, для **борьбы с двойной тратой** Монеро вводит такое понятие как ***key image (KI)***. Ещё точнее ***private key image***.

- Формально (из определения) задается этот образ так: `I = x * hash(P)`

  В этой формуле у нас `I` - тот самый ***KI***; `x` - **одноразовый приватный ключ**, который используется для анлока не потраченного выхода, а `P` - **одноразовый публичный ключ** также для не потраченного выхода.

- То есть уже из формулы понятно, что ***key image*** точно так же **является одноразовым**. Он создаётся с **каждым не потраченным выводом**, но он **не виден** в чейне, **пока** этот вывод **не будет израсходован**.

- **Список** всех ***KI*** при этом **доступен** всем желающим и майнеры с его помощью спокойно могут отслеживать, была ли попытка двойной траты или нет.

Список источников:

1) Ring Signatures + Explanation Video by Bill Buchanan
2) Group Signatures

3) An Exploration of Group and Ring Signatures
4) Moneropedia: Ring Signatures

5) What Are Ring Signatures? Providing Privacy for Cryptocurrency
6) Brief Dive into Ring Signatures

7) Monero Private Key Image