비트코인: 개인간(peer to peer) 전자 화폐 시스템

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org
번역: Park HeeJin

요약. 순수한 개인 간(peer to peer) 전자화폐시스템으로도 금융기관의 개입 없이 직접 개인 대 개인으로 송금하는 온라인 지불이 가능하다. 전자 서명을 통해 이러한 것이 부분적으로 가능한데, 이중 지불을 막기 위해 여전히 중앙 기관이 필요하다면 주된 이점이 사라지게 된다.

이 논문에서 개인 간 네트워크에서 일어날 수 있는 이중 지불 문제에 대한 해결책을 제시하고자 한다. 네트워크 내에서 이루어지는 거래는 해싱(hashing)에 의해 타임 스탬프가 찍히게 되고, 해시(hash) 기반의 작업 증명(proof of work)에 의해 생성되는 연속된 체인의 일부로 포함되어, 이러한 작업 증명이 다시 진행되지 않는 한 변하지 않는 기록으로 남게 된다. 최장길이의 체인은 단순히 이 체인이 담고 있는 사건들의 시간 순서에 대한 증명일 뿐 아니라, 가장 큰 CPU 파워 풀(pool)에 의해 생성되었다는 증거이기도 하다. 과반수의 CPU 파워가 선량한 목적을 가진 네트워크의 노드(node)로부터 나온다면, 이 노드들은 네트워크 공격자들을 압도하며, 가장 긴 체인을 생성할 것이다.

네트워크 자체로는 최소한의 구조가 요구된다. 메시지들은 네트워크 상에서 최대한의 노력에 의해 브로드캐스트 되고, 노드들은 자유로이 네트워크를 이탈할 수 있으며, 이탈해 있는 동안 생성된 가장 긴 체인을 그사이 발생한 것들에 대한 증거로 수용함으로써 다시 네트워크에 합류할 수 있다.

1. 서론

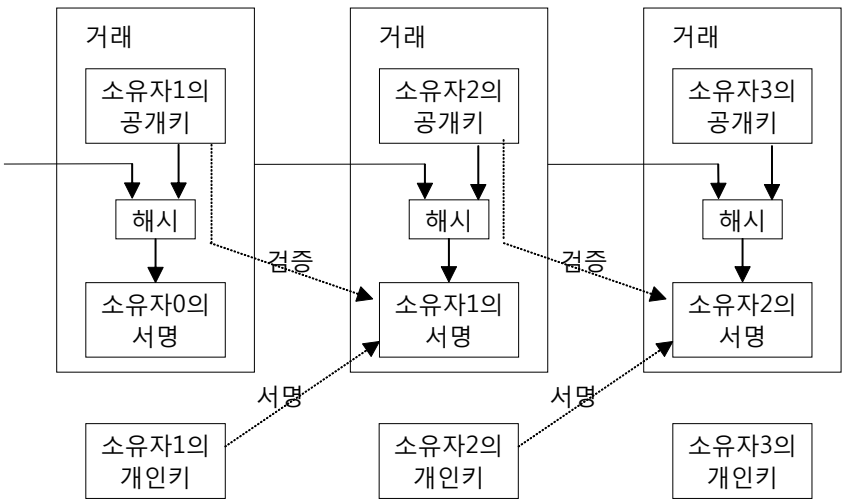
인터넷 기반 상거래는, 보증된 제삼자로서 전자지불 프로세스를 수행하는 금융기관에 전적으로 의존해 왔다. 이 시스템은 대부분의 거래에 대해 잘 동작하고 있지만, 신뢰 기반 모델이기에 안고 있는 약점이 있다. 금융기관들이 분쟁에 대한 중재에서 자유로울 수 없기에, 완벽하게 철회 불가한 거래라는 것은 실재하지 않는다. 이러한 중재비용은 결국 거래비용을 상승시켜서, 거래 가능한 실질적인 최소거래금액을 제한하여 소액거래가 일어나기 힘들게 하고, 철회 불가한 속성을 가진 거래까지 철회 가능하도록 하려다 보니 더 큰 비용이 발생한다. 철회 가능성 때문에 더 많은 신용을 요구하게 된다. 판매자는 구매자에게 구매에 필요한 최소 정보보다 더 많은 정보를 귀찮게 요청하게 되고 경계하게 된다. 어느 정도 거래 사

기를 감내할 수밖에 없다는 것이 현실이다. 이러한 비용과 거래의 불확실성은 실제 사람들에 의해 물리적 화폐가 사용될 때는 회피될 수 있는 사항이나, 디지털 통신상에서 거래가 일어날 때는, 믿을 수 있는 기관이 개입되지 않는 한 해결할 방법이 없다.

신뢰보다는 암호학적인 증명에 기반을 둔 전자 지불 시스템이 필요하다. 이 시스템은 거래 의사가 있는 두 당사가가 신뢰기관의 도움 없이 직접 거래를 수행하게 한다. 계산의 복잡성에 기인한 비가역적 거래는, 판매자가 사기당할 위험성이 없게 해주고, 구매자를 위해서는 이미 널리 사용되고 있는 에스크로 방법이 사용될 수 있다. 이 논문에서 우리는 거래들의 시간 순서에 대한 계산적 증거를 생성하는 피어 투 피어 기반의 분산 타임스탬프 서버를 써서 이중 지불 문제를 해결하는 방법을 제시한다. 이 시스템은 참여하고 있는 건전한 노드들의 CPU 파워가 공격자들의 파워보다 높은 한 안전하게 동작한다.

2. 거래

우리는 전자 코인을 "디지털 서명의 체인"으로 정의한다. 각각의 전자 코인 소유자는, 이전(以前) 거래의 해시와 받을 이의 공개키(public key)에 대해 디지털 서명하고 이 서명값을 코인의 끝에 붙여서, 다른 이에게 코인을 넘긴다. 받는 이는 서명에 대한 검증을 통해 해당 체인의 소유권을 확인할 수 있다.



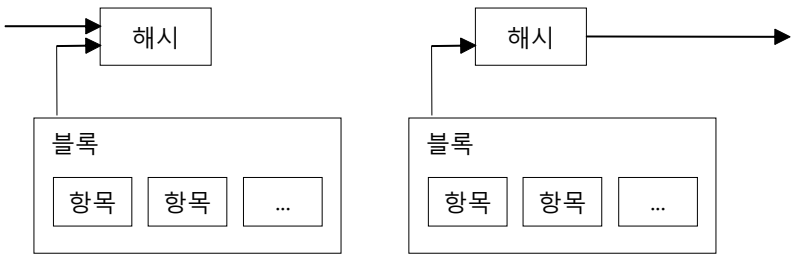
이 과정상 문제는, 대금을 송금한 소유자가 해당 코인을 중복으로 사용하지 않았다는 것을 수금 자(=받는 이)가 확인할 수 없다는 데 있다. 이 문제를 해결하는 일반적인 방법은 중앙의 신뢰기관 혹은 조폐국(造幣局)을 두고, 거래마다 이중 지불인지 여부를 확인하는 것이다. 거래가 끝난 후에 해당 코인이 조폐국으로 회수되어 다시 발행되게 하고, 조폐국에서 직접 발행된 코인만이 이중 지불되지 않는 코인으로써 신뢰성을 갖게 하는 것이다. 이러한 해결책의 문제점은, 모든 거래가 조폐국을 운영하는 회사를 거쳐 가게 되고, 전체 통화체계의 운명이 – 마치 은행과 유사한 – 이 운영회사에 달려있다는 것이다.

결국, 먼젓번 소유자가 현재의 거래 이전에 어떤 거래에도 서명하지 않았음을, 수금자가

알게 할 방법이 필요하다. 이를 위해서는, 현재의 거래가 가장 첫 번째 거래라는 것만 입증되면 되고, 그 이후에 이중 지불이 시도되는지는 상관없다. 거래의 앞에 다른 거래가 없었다는 것을 확인하는 유일한 방법은 모든 거래 내역을 인지하는 방법밖에 없다. 조폐국 기반의 모델에서는, 조폐국이 모든 거래에 대해 인지하면서 어떤 거래가 가장 처음 발생한 것이지 판단했었다. 이러한 것을 신뢰기관을 이용하지 않고 가능하게 하려면, 거래들이 공개되어야 하고[1], 거래에 참여하는 참가자들이, 그들이 받은 순서의 단일 이력에 합의하는 시스템이 필요하다. 수금인들에게는 거래 때마다, 대다수의 노드가 "이 거래가 처음 거래"라고 합의한 증거가 필요하다.

3. 타임스탬프 서버

우리가 제안하는 해결책은 타임스탬프 서버로부터 시작된다. 타임스탬프 서버는, 거래 항목들로 이루어진 블록들에 대한 해시를 취함으로써 타임스탬핑을 하고, 신문이나 유즈넷 포스트처럼[2-5] 그 해시들을 널리 배포하는 작업을 수행한다. 타임스탬프는 그 데이터들이 그 시점에 분명히 해시값들의 입력값으로 존재했었다는 것을 보증해준다. 각 타임스탬프는 그 해시값 안에 이전 타임스탬프를 포함하고, 각각 추가되는 타임스탬프에 의해 강화되는 구조를 가지는 체인을 형성한다.

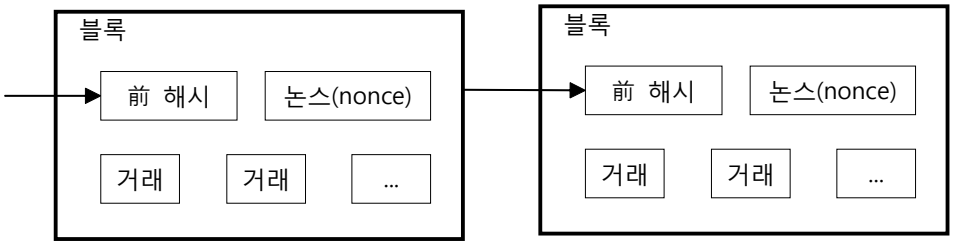


4. 작업 증명

피어 투 피어 기반으로 분산된 타임스탬프 서버를 구축하기 위해서는, 신문이나 유즈넷 포스트가 아닌 아담 백(Adam Back)의 해시캐시[6]와 유사한 작업 증명(PoW, Proof of Work) 시스템이 필요하다. 작업증명은 sha256같은 해시함수를 사용해서, 연속된 0(zero) 값으로 이루어지는 비트열로 시작되는 해시값을 찾는 과정이 포함된다. 이러한 과정에 걸리는 평균작업량은 요구되는 0 비트 수에 따라 지수적으로 증가하며, 반면에 이에 대한 검증은 한 번의 해시 계산으로 가능하다.

타임스탬프 네트워크의 작업증명은, 블록 해시 결과가 요구되는 연속된 0비트 수를 만족할 때까지, 블록에 포함되는 논스(nonce)값을 증가시키는 과정을 통해 구현된다. 일단 CPU 연산에 의해 작업증명을 만족하는 값을 찾은 경우, 해당 블록은 고정된다. 만약 해당 블록을 수정하려면 다시 작업증명을 만족하는 노력이 수반되어야 하고, 블록들이 체인처럼 연결되

이전 블록의 해시를 입력으로해서, 그 다음 블록의 해시가 계산되기에,
어 있기에, 수정한 블록의 다음 블록들에 대해서 동일하게 작업 증명을 해야 한다.



이러한 작업 증명 방법은, 다수결에 의해 결정되는 시스템에서의 대표성 문제도 해결한다. 만약 IP주소 하나당 한 표로 하는 대표성을 부여하는 경우, 다수의 IP를 보유할 수 있는 한 개인에 의해 시스템이 장악될 수 있다. 작업 증명은 CPU 한 개에 한 표이다. 다수결에 의한 결과는, 가장 많은 작업증명 노력이 투여되어야 만들어지는, 가장 긴 체인으로 표현된다. 만약 대다수의 CPU 파워가 정직한 노드들에 의해 사용된다면, 정직한 체인이 다른 경쟁 체인들을 압도하면서 가장 빠르게 성장할 것이다. 공격자가 예전 블록을 수정하기 위해서는, 수정하려는 그 블록과 그 블록 뒤에 있는 모든 블록의 작업증명을 재작업 해야만 하고, 그런 이후에 다시 정직한 노드들의 작업을 추월해야만 가능하다. 블록이 하나씩 더 추가될 때마다, 추격이 늦은 공격자가 따라잡을 확률은 기하급수적으로 낮아진다. 이 부분은 이 논문의 후반부에서 보여줄 것이다

시간이 흐름에 따른 하드웨어 속도 증가와 노드들의 참여도 증가율을 보상하기 위해서, 작업증명의 난이도는 시간당 평균 블록 생성수를 목표로 하는 이동평균값에 의해 결정된다. 블록들이 너무 빨리 생성되면, 난이도는 증가 된다.

5. 네트워크

네트워크의 동작은 다음과 같은 과정으로 이루어진다:
1) 새로운 거래들은 모든 노드에 브로드캐스트된다.
2) 각 노드는 새로운 거래들을 모아 블록에 넣는다.
3) 각 노드는 해당 블록에 대한 작업증명을 찾는 과정을 수행한다.
4) 어떤 노드가 작업증명을 찾았다면, 해당 블록을 모든 노드에게 브로드캐스트한다.
5) 노드들은, 브로드캐스트 받은 블록에 대해서, 블록 내 모든 거래가 유효하고 중복 사용되지 않았을 때, 해당 블록을 승인한다.
6) 노드들은, 그들이 해당 블록을 승인했다는 것을, 해당 블록의 해시를 이용해서 그다음 블록생성을 시작함으로써 암묵적으로 나타낸다.

노드들은 항상 최장길이 체인을 옳은 것으로 간주하며, 그 체인을 대상으로 확장하는 작업을 수행한다. 만약 두 개의 노드가, 다음 블록(next block)에 대해서 서로 다른 버전을 동시에 전파한 경우, 노드들은 둘 중 하나를 먼저 받을 것이다. 이 경우, 먼저 받은 블록을 기준

으로 작업을 수행하지만, 다른 갈래의 체인이 더 길어질 경우를 대비하여 다른 갈래도 저장해 둔다. 이후에 체인의 어느 한쪽 갈래가 더 길게 되는 작업증명이 발견되면 두 갈래의 체인은 더는 동등하지 않게 되고, 짧은 쪽 갈래를 기준으로 작업하던 노드들은 더 긴 다른 갈래로 작업을 전환한다.

새로운 거래에 대한 브로드캐스트가, 반드시 모든 노드에게 이루어질 필요는 없다. 가능한 많은 노드들에게 전파되기만 하면, 그 거래들은 머지않아 블록 안에 포함될 것이다. 블록 브로드캐스트 또한 누락 메시지에 대한 내성이 있다. 만약 한 노드가 특정 블록을 못 받았다 하더라고, 그다음 블록을 받았을 때 블록 누락을 인식해서 누락 블록을 요청해서 받는다.

6. 인센티브

관례상, 블록에 포함된 거래 중 첫 번째 거래는, 해당 블록 생성자에게 새로운 코인이 발행돼서 주어지는, 특별한 거래로 이루어진다. 이렇게 함으로써, 네트워크를 지탱하고 있는 노드들에게 인센티브를 줄 수 있고 또한, 코인을 발행하는 중앙 기관 없이도 신규 코인을 발행해서 유통될 수 있게 된다. 지속해서 일정한 양의 신규 코인을 발행하는 것은, 금을 유통할 수 있게 광부들이 자원을 소비하는 것과 유사하다. 이 경우에는 CPU 동작과 전력이 소비된다.

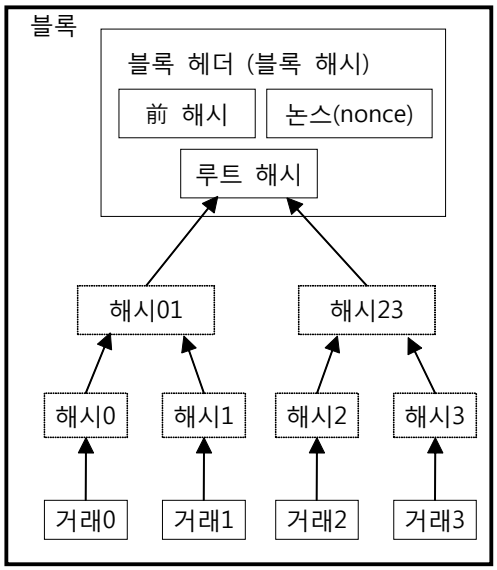
인센티브는 거래수수료에 의해서도 발생한다. 거래 내역에서 출금(out)되는 값이 입금(in)되는 값보다 작다면, 그 차액은 거래수수료가 되며, 해당 거래를 포함하는 블록생성 인센티브에 더해진다. 코인 유통량이 어느 수준에 이르면, 인센티브는 거래수수료에 의해서만 이루어지며, 인플레이션으로부터 완전히 자유롭게 된다. 새로운 코인이 발행되지 않기에, 신규코인 발행에 의한 코인가치 하락(=인플레이션)이 발생하지 않음

인센티브는 노드들의 정직한 참여를 유도하는 데도 도움이 된다. 만약 욕심 많은 공격자가, 모든 정직한 노드들의 CPU 파워 보다 더 많은 파워를 가지게 되었다면 그는 이 파워를 이용해서, 자신이 송금했던 거래를 조작하여 사취(詐取)하거나 혹은, (블록생성 경쟁에서 이겨) 새로운 코인을 발급 받을지를 선택하게 될 것이다. 여기서 그는, 시스템을 붕괴시키고 또한 그가 보유한 부의 유효성이 사라지는 것보다는, 시스템의 규칙을 따르면서 그 규칙에 따라 다른 누구보다도 더 많은 신규 코인을 받는 것이, 그 자신에게 더 이득이라는 것을 알게 될 것이다.

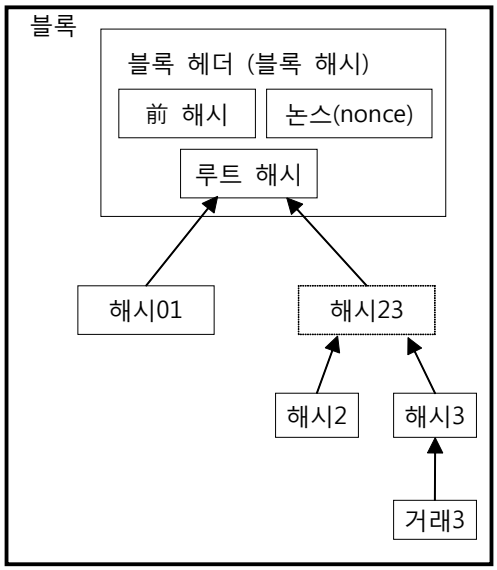
7. 디스크 공간 회수

* A->B, B->C 거래가 있을 때, B->C 거래가 블록5에 있었고, 현재 블록15정도라면, A->B거래는 폐기할 수 있다는 얘기

코인에 대한 가장 최근 거래가 충분한 수의 블록에 묻히면, 디스크 공간을 절약하기 위해, 그 거래 이전에 발생했던 거래들은 삭제해도 된다. 이미 계산된 블록 해시값이 훼손되지 않으면서 이러한 것이 가능하게 하도록, 거래들의 해시값으로 머클트리(Merkle Tree)를 구성하고[7][2][5], 블록의 해시에는 머클트리의 최상부 루트(root)만 포함되게 한다. 이렇게 함으로써 오래된 블록의 경우는, 트리의 일부 가지를 쳐 낼 수 있기에 크기가 작아질 수 있다. 자식 노드의 해시들은 저장될 필요가 없다. * 머클트리의 특성상 루트에 대해서만 검증하면 되기에 하위 해시는 저장 불필요



거래들의 해시로 이루어진 머클트리



블록에서 거래0-2에 대해 가지치기 수행

거래데이터가 없이 블록헤더 단독으로는 크기가 80바이트이다. 만약 블록이 매 10분마다 생성한다고 가정하면, 80바이트 x 6 x 24 x 365 = 4.2MB 정도의 데이터가 매년 생성된다. 2008년 기준으로 했을 때 일반적인 컴퓨터의 메모리(RAM)가 2GB 정도이고, 무어의 법칙 (Moore's Law)에 따른다면 매년 1.2G정도씩 커질 것이기에, 블록헤더가 메모리에 로드되어야 한다고 가정하더라도 문제 될 것은 없을 것이다.

* 반도체 집적회로의 성능이 18개월마다 2배씩 늘어난다는 이론

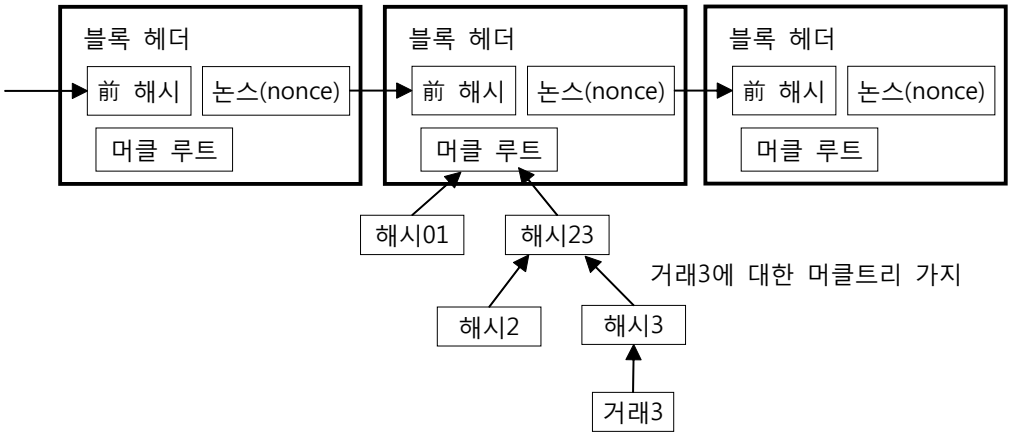
8. 간소화된 지불 검증

전체 네트워크 노드를 사용하지 않고도 지불에 대한 검증이 가능하다. 사용자가 가장 긴 작업증명 체인의 헤더블록에 대한 복사본을 가지고 있기만 하면 된다. 사용자는 해당 체인이 가장 길다고 확신될 때까지 네트워크 노트에 질의를 보내 그 복사본을 얻을 수 있고, 그 거래에 대한 타임스탬핑이 되어 있는 블록과 연결시키면서 머클트리의 가지를 얻을 수 있다. 이를 통해 거래에 대한 직접 검증은 할 수 없으나, 체인상에 연결해봄으로써, 네트워크 노드가 그 거래를 받아들였고, 해당 블록 다음에 다른 블록들이 추가되어 이어지고 있다는 사실로부터, 네트워크가 그 거래를 승인했다는 것을 확인할 수 있다.

* 아래그림에서 거래3이 위쪽으로 올라가면서 머클루트까지 가는 과정 참조

* 다음 블록의 해시를, 해당 거래를 포함한 블록의 해시를 가지고 계산해 본다는 얘기

최장 작업증명 체인



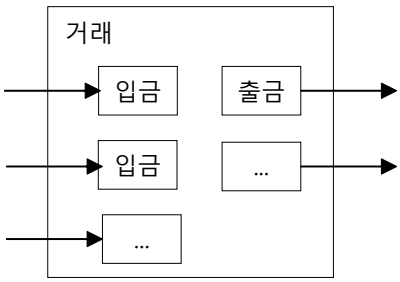
이러한 검증방식은 네트워크가 정직한 노드들에 의해 통제되는 경우에는 신뢰할 수 있지만, 공격자에 의한 파워가 더 큰 경우에는 더 취약하다. 네트워크 노드들은 스스로 거래들을 검증할 수 있는 반면 간소화된 방법은, 공격자들이 네트워크를 계속 과점하고 있으면서 만들어내는 조작된 거래에 의해 농락당할 수 있다. 이를 막는 한 가지 전략은, 네트워크 노드들이 잘못된 블록을 발견했을 경고를 보내고, 이 경고를 받은 사용자의 소프트웨어는 전체 블록과 경고된 거래를 다운로드 받아 불일치를 확인하게 하는 것이다. 빈번하게 거래를 받아야하는 사업의 경우에는, 보다 독립적인 보안성과 빠른 검증을 위해서, 자체 노드를 운영하는 것을 선호할 것이다.

* 공격자 파워가 크면 일반 full node도 취약하지만, 간편노드는 더 취약하다는 얘기
* 간편 노드의 소프트웨어가 이렇게 하라는 것
* full node를 운영하는게 좋다는.

9. 금액의 결합과 분할

코인들을 개별로 처리할 수도 있겠으나 그런 경우, 송금되는 모든 단위별로 거래를 분리해야 하기에 현실적이지 않다. 금액에 대해 쪼개고 합치는 것이 가능하게 하도록, 거래는 복수의 입금과 출금으로 구성된다. 일반적으로 입금은, 이전 거래로부터 받은 큰 금액의 단일 입금이거나, 소액으로 이루어진 여러 입금 값들로 구성되고, 출금은 최대 2개의 출금 값으로 구성된다. 출금 값 중 하나는 상대방에게 주는 지불이고, 만약 입력값대비 잔액이 발생한다면, 송금자 자신에게 되돌리는 출금이 발생한다.

* 잔액 = 입금 – (상대방에게 지불하는 금액)
* 만약 (입금 > 출금)이면, 그 차액은 거래 수수료가 되어서, 이 거래를 블록화하는 노드에게 돌아간다.



한 거래가 여러 거래들과 연관되고, 그 거래들은 더 많은 거래들과 연관되는, 마치 부채꼴

모양의 의존관계가 생기게 되지만, 이는 문제가 되지 않는다. 거래 이력에 대한 완전히 독립된 사본을 추출할 필요가 전혀 없다.

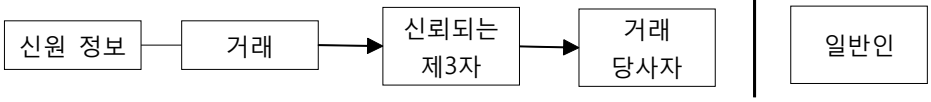
* 마지막 거래에서의 출금만 의미있기 때문이다. 즉, 입금값을 가지고, 새로운 출금이 '생성'되는 개념이기에, 앞 부분의 입금에 연결된 출금값을 거슬러가면서, 계산할 필요가 없다.

10. 프라이버시

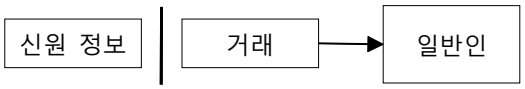
전통적인 뱅킹 모델에서는, 관련된 당사자와 신뢰할 수 있는 제3자에게만 정보 접근을 제한함으로써, 개인정보 보호가 가능하도록 하고 있다. 그러나, 이 방법은 거래를 모두 공개해야 하는 한 사용될 수 없다. 대신, 개인에 대한 공개키를 익명화함으로써 정보에 대한 추적을 막아서, 개인의 프라이버시를 보호할 수 있다. 공개된 거래를 통해서, 누군가가 또 다른 누군가에게 얼마 만큼의 금액을 송금했다는 것은 알 수 있으나, 그 거래에 연결된 누군가에 대한 정보는 없다. 이것은 마치 증권거래소에서 공개되는 테이프(tape)에 의한 정보공개와 비슷한 수준으로, 개별 주식거래에 대한 시각과 규모는 공개되지만, 그 거래에 누가 관여되어 있는지는 공개되지 않는 것과 유사하다.

* ticker tape: 예전에 증권거래소에서 텔레그램을 통해, 주식가격을 찍어 보내던 테이프

전통적인 프라이버시 모델



새로운 프라이버시 모델



* key pair: 개인키(public key) + 공개키(public key)

추가적인 안전장치로, 거래마다 새로운 키 쌍이 생성되어 사용되고, 이때 그 키 쌍이 동일한 소유자와 연결되어 있다는 것이 유지된다. 그러나, 복수의 입금값을 갖은 거래에서, 그 입금들이 동일한 소유자에게 연결되었다는 정보 노출은 피할 수 없다. 만약 그 키에 대한 소유자가 노출된 경우, 연결정보를 통해 동일한 소유자의 다른 거래정보들이 노출될 위험이 있다.

* 새로운 개인키/공개키를 생성하고, 그 공개키로부터 새로운 주소를 유도해서 사용하라는 것

* 하나의 거래에 있는 입금값들은, 동일한 주소를 출금대상 주소로해서 만들어진 복수개으이 다른 거래에서 온 것이기에

11. 계산

공격자가 정직한 체인보다 더 빨리 대체체인을 만들어내는 경우를 고려해보자. 만일 이런 시도가 가능하더라도, 아무것도 없던 것에서 코인을 새로 만들어 내거나, 공격자가 전혀 소유한 적도 없는 코인을 얻을 수 있다거나 하는 식의, 시스템에 대한 무단 수정이 된다는 것은 아니다. 노드들이 이러한 유효하지 않은 거래를 결제로 받아들이지 않으며, 정직한 노드들은 이러한 거래를 담은 블록을 절대 받아들이지 않는다. 공격자는 자신의 거래에서, 그가 최근에 지출한 돈에 대한 회수만을 시도해볼 수 있을 뿐이다.

* 모든이의 개인키를 알아낸다는 것을 얘기하는 것은 아님.
해시경쟁의 우위를 통해, 불록생성이 빠른 것 뿐임

* 이중 지불 시도

* 블록체인에서 '체인' 개념은, '이중 지불'을 막는게 주목적.
'거래'는 기존 비대칭키에 의한 서명/검증에 의해서 유효성 보장

* 랜덤한 확률값을 가지고 양쪽 방향으로 1씩 증가하는.

정직한 체인과 공격자 체인의 경주는 이항 임의 보행(Bionomial Random Walk) 성격을 띤다. 성공 이벤트는 정직한 체인을 +1 증가시키고, 실패 이벤트는 공격자 체인을 증가시켜서 정직한 체인과의 갭을 −1로 줄이는 것이다.

* 유한한 초기 자산을 가지고 일련의 공평한 도박을 하는 도박꾼은 거의 확실하게 자산이 0이 되어 파산하게 된다는 정리

공격자가 주어진 열세를 따라잡을 확률은 도박꾼의 파산(Gambler's Ruin) 문제와 유사한다. 도박꾼이, 적자인 상태에서 무제한의 자금을 가지고, 손익분기점에 도달하기 위해 무제한의 시도를 한다고 가정해 보자. 우리는 도박꾼이 손익분기점에 도달할 확률 즉, 공격자가 정직한 체인을 따라잡을 확률을 다음과 같이 계산할 수 있다.

$p =$ 정직한 노드가 다음 블록을 찾을 확률
$q =$ 공격자가 다음 블록을 찾을 확률
$q_z =$ 공격자가 z블록 뒤에서부터 따라잡을 확률

$$q_z = \begin{cases} 1 & \text{if } p \le q \\ \left(\frac{q}{p}\right)^z & \text{if } p > q \end{cases}$$

* p>q이기에 q/p < 1. 따라서 z가 커질수록 $(q/p)^z$는 지수적으로 작아짐

p > q인 경우, 공격자가 정직한 체인을 따라잡을 확률은, 공격자가 따라잡아야 할 블록수가 많을수록 지수적으로 감소한다. 공격자에게 주어진 조건상, 만일 그가 초기에 운좋게 앞으로 치고 나가지 못한다면, 그의 기회는 그가 뒤처질수록 보이지 않을 만큼 작아질 것이다.

이제, 새로운 거래에 대해 수금인이, 그 거래의 내용을 송금인이 수정할 수 없다고 믿으려면, 얼마나 기다려야 하는지를 고려해보자. 송금자는 공격자라고 가정하고, 공격자는 수금인이 금액을 받았다고 인지하도록 하고, 얼마의 시간이 흐른 뒤, 지불했던 금액을 다시 자신에게 회수하려 한다고 가정하자. 이런 일이 발생할 경우, 해당 수금자는 경고를 받겠지만, 이미 늦어버린 상황이 되길 공격자는 바란다.

* 송금인이 미리 거래를 만들고, 그 거래를 포함한 블록을 충분한 시간을 두고 만들지 못하게 하겠다는 것. 그럴려면, 수금인 공개키를 미리 주지 않아야 함

수금인은 새로운 키 쌍을 생성하고, 송금인의 서명 직전에 공개키가 송금인에게 보내지도록 한다. 이렇게 함으로써, 송금인이 미리 거래에 대한 블록생성을 시작해서 블록생성에 성공한 후 거래를 진행하는 것을 막을 수 있다. 일단 거래가 생성되어 보내지면, 부정직한 송금자는 그 거래를 조작하고, 조작된 거래를 포함한 또 다른 블록생성을 비밀리에 수행한다.

* 정직한 노드들에 의해 생성되는 블록체인과 다른 블록을 만든다는 것

수금인은, 해당 거래가 포함된 블록이 생성되고, 또한 z개의 블록이 그 뒷부분에 추가 연결되어 생성되기까지 기다린다. 그는 공격자가 얼마만큼의 작업을 진행했는지 정확히 모르지만, 정직한 블록들이 평균적으로 기대되는 블록생성시간을 거쳐 생성된다고 가정하면, 공격자의 잠재적 진척도는 푸아송(Poisson) 분포를 따를 것이고 그 기댓값은 아래와 같이 될 것이다.

* 단위 시간 안에 어떤 사건이 몇 번 발생할 것인지를 표현하는 확률분포

$$\lambda = z\frac{q}{p}$$

공격자가 현재의 블록까지 따라잡을 확률을 구하기 위해서는, 그가 만들어 낼 수 있는 진척도에 대한 푸아송 밀도와, 그 시점에서 따라잡을 수 있는 확률값을 곱하면 된다.

$$\sum_{k=0}^{\infty} \frac{\lambda^k e^{-\lambda}}{k!} \cdot \begin{cases} (q/p)^{(z-k)} & \text{if } k \le z \\ 1 & \text{if } k > z \end{cases}$$

분포의 무한급수를 더하지 않기 위해, 식을 정리하면...

$$1-\sum_{k=0}^{z}\frac{\lambda^k e^{-\lambda}}{k!}\left(1-(q/p)^{(z-k)}\right)$$

C 언어 코드로 구현하면...

```
#include <math.h >
double AttackerSuccessProbability(double q, int z)
{
    double p =1.0 - q;
    double lambda = z * (q / p);
    double sum =1.0;
    int i, k;
    for (k =0; k <= z; k ++)
    {
        double poisson = exp(-lambda);
        for (i =1; i <= k; i ++)
            poisson *= lambda / i;
        sum -= poisson * (1 - pow(q / p, z - k));
     }
     return sum;
}
```

실행 결과를 보면, 확률값이 z에 따라서 지수적으로 감소함을 알 수 있다.

```
q=0.1
z=0  P=1.0000000
z=1  P=0.2045873
z=2  P=0.0509779
z=3  P=0.0131722
z=4  P=0.0034552
z=5  P=0.0009137
z=6  P=0.0002428
z=7  P=0.0000647
z=8  P=0.0000173
z=9  P=0.0000046
z=10  P=0.0000012

q=0.3
z=0  P=1.0000000
z=5  P=0.1773523
z=10  P=0.0416605
z=15  P=0.0101008
z=20  P=0.0024804
z=25  P=0.0006132
z=30  P=0.0001522
z=35  P=0.0000379
z=40  P=0.0000095
z=45  P=0.0000024
z=50  P=0.0000006
```

P가 0.1% 미만의 경우에 대해 풀어보면...

P < 0.001
q=0.10 z=5
q=0.15 z=8
q=0.20 z=11
q=0.25 z=15
q=0.30 z=24
q=0.35 z=41
q=0.40 z=89
q=0.45 z=340

12. 결론

이 논문에서 우리는 신용에 기반을 두지 않는 전자 거래 시스템을 제안했다. 소유권을 강력히 관리할 수 있으나, 이중 지불 문제는 해결하지 못하는, 전자서명을 이용하는 일반적인 화폐 구조로부터 이야기를 시작했다. 이중지불 문제를 해결하기 위해서 작업증명을 사용하는 개인 간(peer to peer) 네트워크를 제안했다. 그 네트워크에서는 거래에 대한 공개된 이력을 저장하게 되는데, 한번 저장되면, 네트워크를 구성하는 정직한 노드들이 대다수의 CPU 파워를 가지는 한, 공격자들이 조작하는 것이 계산적으로 불가능하게 되는 특성을 가지도록 했다. 네트워크는 구조화되지 않은 단순구조이기에 견고하다. 노드들은 최소한의 조정을 통해 한꺼번에 동작한다. 메시지가 어떤 특정 위치에 전달돼야 하는 것이 아니라, 최선을 기반으로 전달되기만 하면 되기 때문에, 노드들의 신원이 인식될 필요도 없다. 노드들은 자유로이 네트워크를 떠났다가 재합류할 수 있으며, 재합류할 때는 작업증명 체인을 그동안 발생했던 거래에 대한 증명으로 받아들이면 된다. 노드들은 그들의 CPU 파워를 갖고 투표를 한다. 유효한 블록에 대해서는 그 블록을 확장하는 작업을 수행함으로써 찬성을 표시하고, 유효하지 않은 블록에 대해서는 그 블록에 대한 작업을 수행하지 않음으로써 거부를 나타낸다. 이러한 합의 메커니즘과 함께 어떤 규칙과 인센티브가 시행될 수 있다.

* peer to peer이기에, 전달된 메시지를 가까운 노드들에게 최선을 다해 배포하기만 하면 됨

References

[1] W. Dai, "b-money," http://www.weidai.com/bmoney.txt, 1998.

[2] H. Massias, X.S. Avila, and J.-J. Quisquater, "Design of a secure timestamping service with minimal trust requirements," In 20th Symposium on Information Theory in the Benelux, May 1999.

[3] S. Haber, W.S. Stornetta, "How to time-stamp a digital document," In Journal of Cryptology, vol 3, no 2, pages 99-111, 1991.

[4] D. Bayer, S. Haber, W.S. Stornetta, "Improving the efficiency and reliability of digital time-stamping," In Sequences II: Methods in Communication, Security and Computer Science, pages 329-334, 1993.

[5] S. Haber, W.S. Stornetta, "Secure names for bit-strings," In Proceedings of the 4th ACM Conference on Computer and Communications Security, pages 28-35, April 1997.

[6] A. Back, "Hashcash - a denial of service counter-measure," http://www.hashcash.org/papers/hashcash.pdf, 2002.

[7] R.C. Merkle, "Protocols for public key cryptosystems," In Proc. 1980 Symposium on Security and Privacy, IEEE Computer Society, pages 122-133, April 1980.

[8] W. Feller, "An introduction to probability theory and its applications," 1957. 9