공개키 암호방식을 응용한 사이버 머니(money) - 비트코인이란?

(요약 - 문봉교)

비트코인(Bitcoin)은 2009 년에 만들어진 디지털 통화로, 통화를 발행하고 관리하는 중앙 장치가 존재하지 않는 구조를 가지고 있다. 대신, 비트코인의 거래는 P2P 기반 분산 데이터베이스에 의해 이루어지며, 공개 키 암호 방식 기반으로 거래를 수행한다. 비트코인은 익명성과 공개성을 가지고 있다. 비트코인은 지갑 파일의 형태로 저장되며, 이 지갑에는 각각의 고유 주소가 부여되며, 그 주소를 기반으로 비트코인의 거래가 이루어진다.

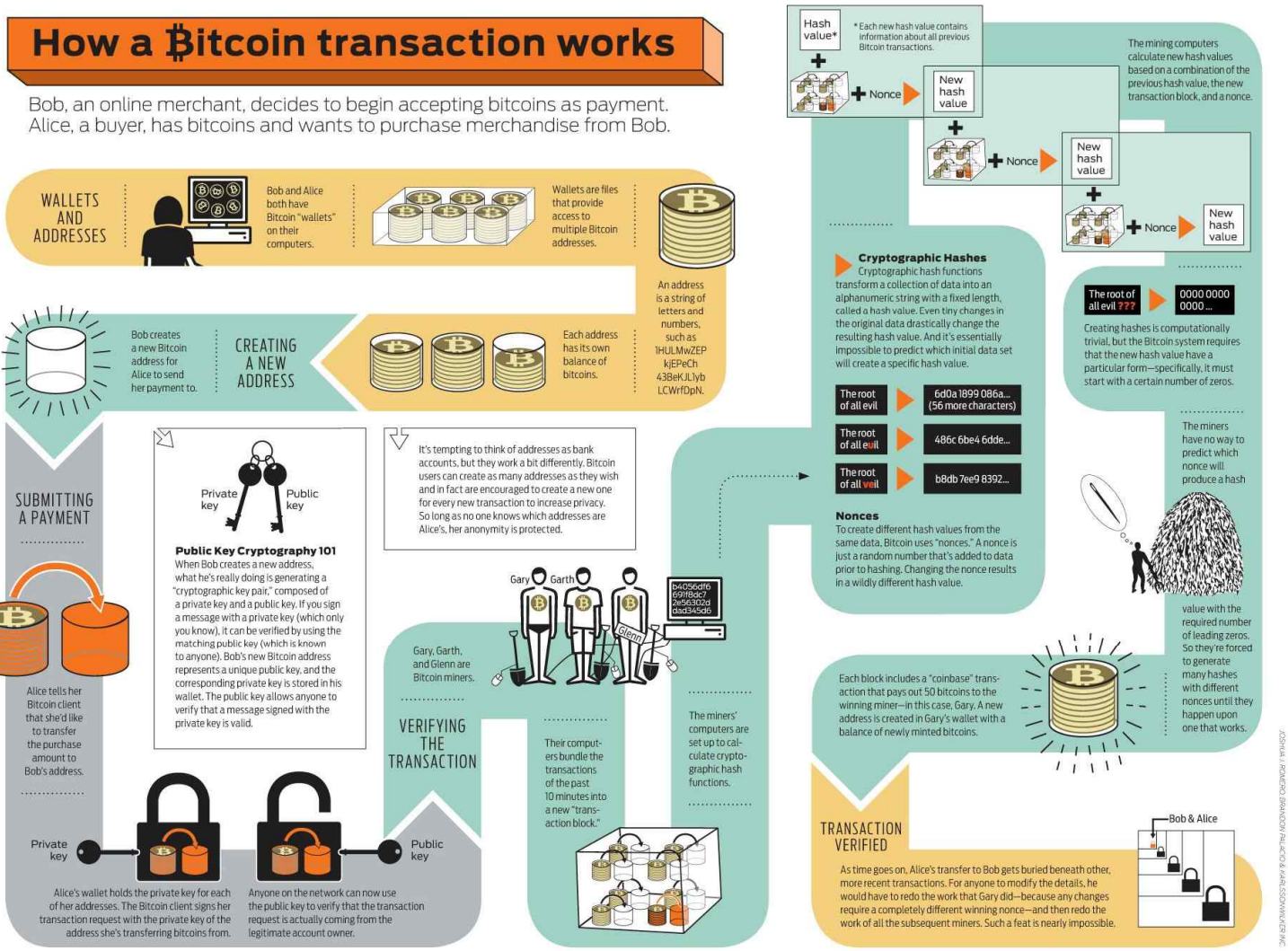

그림 1. 비트코인 결제 시스템의 작동 방식에 관하여 (출처: http://spectrum.ieee.org/)

밥은 비트코인을 받고 물건을 판다. 앨리스는 비트코인으로 물건을 구입한다.

지갑과 주소

1) 밥과 앨리스는 각각 자신의 컴퓨터에 비트코인 지갑을 가지고 있다.

2) 하나의 비트코인 지갑에는 여러 개의 주소들이 있다.

3) 각각의 주소명은 숫자와 글자의 조합(예를 들어 1LEdiñ83EnilFEb93)으로 이루어져 있다.

4) 각각의 주소에는 일정한 양의 비트코인이 보관되어 있다.

주소 생성하기

5) 밥은 앨리스에게서 비트코인을 받기 위해 주소를 하나 새로 만든다.

결제하기

6) 앨리스는 밥에게 밥이 방금 전 만든 주소로 비트코인을 보내고 싶다고 말한다.

7) 앨리스의 비트코인 지갑에는 각각의 주소들에 대한 비밀 키가 있다. 앨리스는 자기 지갑에 있는 여러 개의 주소들 중 비트코인이 송금되는 주소의 비밀 키를 가지고 자신의 결제 요청에 서명한다.

8) 앨리스의 결제 요청이 서명되면, 이제 비트코인 결제망에 있는 누구나 공개키(앨리스의 주소명)를 가지고 결제 요청의 서명에 사용된 개인 키와 비교함으로써, 이 결제 요청이 실제 계좌주로부터 왔는지 확인할 수 있다.

거래를 승인하기

9) 개리, 갈스, 글렌은 비트코인 광부(miner -_-)다.

10) 이들은 자신의 컴퓨터로 지난 10 분간 일어났던 모든 거래 기록들을 모아 '거래 블록'에 저장한다.

11) 광부들의 컴퓨터가 암호해쉬함수(Hash Function, 주어진 데이터에서 고정된 길이의 난수를 생성하는 연산기법)를 계산한다.

12) 암호해쉬함수는 '거래 블록'에 저장된 지난 10 분간의 모든 거래 데이터를 하나의 길다란 글자-숫자 조합으로 만들어버린다. 거래 데이터 값이 조금만 바뀌어도 해쉬계산값이 크게 달라지기 때문에 어떤 값이 나올지 예측이 힘들다. 동일한 거래 데이터에서 다른 해쉬값을 얻으려면 논스(Nonces)를 사용해야

한다. 논스란 거래 데이터를 해쉬하기 전에 더해지는 난수이다. 논스를 약간만 변형해도 해쉬값이 크게 달라진다.

13) 채광용 컴퓨터(mining computer -_-)들은 예전에 계산했던 해쉬값 + 거래 데이터가 저장된 '거래 블록' + 논스를 가지고 새로운 해쉬값을 만들어낸다. (참고로 예전에 계산했던 해쉬값에는 과거에 있었던 모든 거래 기록들이 짬뽕되어 있다)

14) 해쉬값을 계산하는 건 쉽지만, 문제는 비트코인 결제 시스템이 새로운 해쉬값에 대해 특정 조건을 요구한다는 점이다. (예를 들면 해쉬값이 몇 개의 0 으로 시작해야 한다든지)

15) 그런데 광부들은 해쉬 함수에 어떤 논스를 집어넣어야 비트코인 결제 시스템이 요구하는 해쉬값을 얻을 수 있는지 모른다. 따라서 광부들은 수많은 논스를 해쉬 함수에 일일히 대입하여 결제 시스템의 요구 조건을 만족하는 해쉬값이 나올 때까지 컴퓨터를 열나게 돌려야 한다(-_-)

16) 매우 운이 좋게도 이번에는 개리가 시스템의 요구 조건을 만족하는 해쉬값을 찾았다 (다시 말해서 새로운 거래 블록을 만들었다). 그에 대한 보답으로 개리의 비트코인 지갑에는 50 비트코인이 있는 새 주소가 생성된다.

계속되는 거래 승인 과정

17) 시간이 지날수록 밥과 앨리스 사이에 있었던 거래 자료는 다른 거래 자료들과 짬뽕된다. 만일 당신이 거래 데이터를 조작하고 싶다면 일단 개리가 해쉬값을 만드는데 썼던 논스를 알아내서 해쉬값을 다시 거래 데이터로 되돌려야 하고, 또 개리 이전에 광부들이 썼던 모든 논스를 알아내서 해쉬값을 원래 자료로 되돌려야 하는데 이는 거의 불가능한 일이다.