Bitcoin: A Peer-to-Peer Electronic Cash System

บิตคอยน์: ระบบเงินสดอิเล็กทรอนิกส์แบบ Peer-to-Peer

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

Thai translation by Peeraphat Hankongkaew, Bitcoin Addict
แปลโดย พีรพัฒน์ หาญคงแก้ว (Peeraphat Hankongkaew) จาก Bitcoin Addict ร่วมกับ Claude Fable 5.1 (Anthropic)
bitcoinaddict.com/articles/bitcoin-whitepaper-thai

บทคัดย่อ (Abstract). เงินสดอิเล็กทรอนิกส์ที่เป็นแบบ peer-to-peer[1] ที่แท้จริงนั้นจะสามารถทำให้การจ่ายเงินออนไลน์จากคนหนึ่งไปอีกคนเกิดขึ้นได้โดยไม่จำเป็นต้องมีสถาบันทางการเงินมาเป็นตัวกลาง[2] Digital Signature เป็นส่วนหนึ่งของคำตอบ แต่ประโยชน์หลักจะหายไปถ้าสุดท้ายระบบยังต้องการตัวกลางที่น่าเชื่อถือมาคอยป้องกัน Double spending[3] เราขอนำเสนอวิธีการแก้ไขปัญหา Double spending โดยใช้เครือข่ายแบบ peer-to-peer เครือข่ายนี้จะประทับเวลาให้ธุรกรรมโดยการ hash[4] ธุรกรรมนั้นเข้าไปต่อกับห่วงโซ่ของ Proof-of-work[5] ที่ต่อยาวไปเรื่อยๆ กลายเป็นบันทึกที่แก้ไขไม่ได้ถ้าไม่ทำ Proof-of-work ใหม่ทั้งหมด ห่วงโซ่ที่ยาวที่สุดไม่ใช่แค่หลักฐานของลำดับเหตุการณ์ที่เครือข่ายเป็นพยานเท่านั้น แต่ยังบอกว่าห่วงโซ่นั้นมาจากกลุ่มที่มีกำลังประมวลผลของ CPU[6] มากที่สุดอีกด้วย ตราบใดที่กำลัง CPU ส่วนใหญ่อยู่กับ Node[7] ที่ไม่ได้ร่วมมือกันโจมตีระบบ Node เหล่านั้นจะสร้างห่วงโซ่ที่ยาวที่สุดและทิ้งห่างผู้โจมตี ระบบของเครือข่ายนี้ไม่จำเป็นต้องมีโครงสร้างที่ซับซ้อน วิธีการสื่อสารภายในเครือข่ายจะเป็นแบบ best effort[8] แต่ละ Node สามารถออกจากเครือข่ายและกลับเข้ามาเมื่อไรก็ได้ โดยยอมรับห่วงโซ่ Proof-of-work ที่ยาวที่สุดเป็นหลักฐานว่าเกิดอะไรขึ้นบ้างในช่วงที่ Node เหล่านั้นหายไป

1. บทนำ (Introduction)

โดยทั่วไปการค้าขายแลกเปลี่ยนในอินเทอร์เน็ตแทบทั้งหมดต้องพึ่งพาสถาบันทางการเงินเป็นเสมือนบุคคลที่สามที่น่าเชื่อถือ ในการประมวลผลการจ่ายเงินอิเล็กทรอนิกส์ แม้ว่าระบบตัวกลางนี้จะทำงานได้ดีสำหรับธุรกรรมส่วนใหญ่ แต่มันก็ยังมีจุดอ่อนที่ติดมากับโมเดลที่ตั้งอยู่บน trust[9] ธุรกรรมที่ย้อนกลับไม่ได้เลยนั้นเป็นไปไม่ได้จริง เพราะ

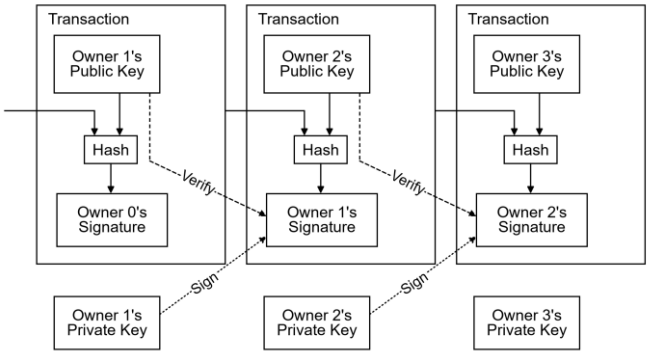
[1] peer-to-peer: เครือข่ายที่คอมพิวเตอร์ของผู้ใช้ติดต่อกันโดยตรง ไม่มีตัวกลาง ทำนองเดียวกับ BitTorrent
[2] ต้นฉบับใช้คำว่า financial institution สถาบันการเงินในที่นี้หมายรวมถึงธนาคาร ผู้ให้บริการบัตร และตัวกลางชำระเงินทุกแบบ
[3] Double spending: การใช้เงินซ้ำ คือเอาเงินหน่วยเดียวกันไปจ่ายมากกว่าหนึ่งครั้ง ปัญหาหลักของเงินดิจิทัลเพราะข้อมูลคัดลอกได้
[4] hash: การย่อยข้อมูลใดๆ ให้เป็นค่าความยาวคงที่ ข้อมูลเปลี่ยนแม้นิดเดียวค่า hash ก็เปลี่ยนทั้งหมด และย้อนกลับหาข้อมูลต้นทางไม่ได้
[5] Proof-of-work: หลักฐานว่าได้ลงแรงคำนวณไปจริง ในที่นี้คือการหาค่าที่ทำให้ hash ของบล็อกขึ้นต้นด้วยศูนย์ตามจำนวนที่กำหนด คนทั่วไปเรียกว่าการขุด
[6] ต้นฉบับปี 2008 ใช้คำว่า CPU เพราะขณะนั้นขุดด้วยซีพียูคอมพิวเตอร์ทั่วไป ปัจจุบันหมายถึงกำลังประมวลผลรวมของเครือข่าย (hash rate)
[7] Node: คอมพิวเตอร์ที่รันซอฟต์แวร์ Bitcoin และเชื่อมต่ออยู่ในเครือข่าย
[8] best effort: ส่งให้ดีที่สุดเท่าที่ทำได้ โดยไม่รับประกันว่าจะถึงผู้รับทุกครั้ง
[9] trust: ความไว้วางใจ ในที่นี้คือการที่ทั้งระบบเดินได้เพราะทุกคนต้องเชื่อว่าตัวกลางจะทำหน้าที่ถูกต้อง

สถาบันการเงินเลี่ยงไม่ได้ที่จะต้องเข้ามาไกล่เกลี่ยข้อพิพาท[10] การมีตัวกลางไกล่เกลี่ยนั้นทำให้ต้นทุนในการทำธุรกรรมเพิ่มขึ้น และค่าใช้จ่ายนี้ทำให้ไม่สามารถทำธุรกรรมขนาดเล็กๆ ได้[11] และยังมีต้นทุนที่ใหญ่กว่านั้น คือเราสูญเสียความสามารถในการจ่ายเงินแบบย้อนกลับไม่ได้ ให้กับบริการที่ย้อนกลับไม่ได้[12] เมื่อการจ่ายเงินมีสิทธิ์ถูกย้อนกลับความจำเป็นต้องพึ่งความน่าเชื่อถือก็ลามไปทั่ว ผู้ค้าขายต้องระแวงลูกค้า และขอข้อมูลต่างๆ ที่มากเกินความจำเป็น[13] และต้องยอมรับว่าจะมีการโกงจำนวนหนึ่งที่เลี่ยงไม่ได้ ต้นทุนและความไม่แน่นอนเหล่านี้เลี่ยงได้เมื่อเจอหน้ากันแล้วใช้เงินสดที่จับต้องได้ แต่ยังไม่มีกลไกใดที่ทำให้จ่ายเงินผ่านช่องทางสื่อสารได้โดยไม่ต้องมีตัวกลางที่น่าเชื่อถือ[14]

สิ่งที่เราต้องการคือระบบการจ่ายเงินแบบอิเล็กทรอนิกส์ที่ตั้งอยู่บนหลักฐานทาง cryptographic[15] แทนที่จะใช้ความน่าเชื่อถือของตัวกลางทางการเงิน ซึ่งมันจะทำให้บุคคลสองคนที่สมัครใจทำธุรกรรมต่อกันได้โดยตรงโดยไม่ต้องมีตัวกลาง ธุรกรรมที่ย้อนกลับได้ยากมากในทางการคำนวณจะช่วยปกป้องผู้ขายจากการหลอกลวง และกลไก escrow[16] แบบทั่วไปก็นำมาใช้ปกป้องผู้ซื้อได้อย่างง่ายดาย ในเอกสารนี้เราได้นำเสนอวิธีการป้องกันการใช้เงินซ้ำ (Double spending) โดยใช้ timestamp server แบบกระจายศูนย์บนระบบ peer-to-peer เพื่อสร้างหลักฐานทางการคำนวณของลำดับเวลาที่ธุรกรรมเกิดขึ้น ระบบจะปลอดภัยตราบใดที่ Node ที่ซื่อสัตย์รวมกันมีกำลัง CPU มากกว่ากลุ่ม Node ผู้โจมตีกลุ่มใดๆ ที่ร่วมมือกัน

2. ธุรกรรม (Transactions)

เรากำหนดให้เหรียญเงินอิเล็กทรอนิกส์นั้นเป็นห่วงโซ่ของลายเซ็นดิจิทัล (digital signature) โดยเจ้าของเหรียญแต่ละคนจะส่งเหรียญไปให้คนถัดไปโดยการลงนามแบบดิจิทัล (digitally signing) บนเลข hash ของธุรกรรมที่ผ่านมาและกุญแจสาธารณะ (public key)[17] ของคนถัดไป แล้วเพิ่มสิ่งเหล่านี้ลงที่ส่วนท้ายของเหรียญ ซึ่งผู้รับเงินจะสามารถตรวจสอบลายเซ็นเพื่อตรวจสอบห่วงโซ่ของความเป็นเจ้าของ (chain of ownership)



[10] ตัวอย่างข้อพิพาทที่ตัวกลางต้องเข้ามาตัดสิน เช่น ลูกค้าปฏิเสธรายการบัตรเครดิต เช็คเด้ง หรือขอเงินคืนหลังได้รับของแล้ว

[11] เพราะต้นทุนในการทำธุรกรรมแพงกว่าจำนวนเงินในการทำธุรกรรม

[12] เช่น บริการหรือเนื้อหาดิจิทัลที่ส่งมอบแล้วเรียกคืนไม่ได้ ถ้าผู้ซื้อยกเลิกการจ่ายเงินภายหลังได้ ผู้ขายจะเสียของไปฟรี เหมือนให้บริการไปแล้ว เช็คเด้ง

[13] เช่น ชื่อ ที่อยู่ เลขบัตร และเอกสารยืนยันตัวตน ซึ่งการเก็บข้อมูลเหล่านี้ไว้ก็กลายเป็นความเสี่ยงของทั้งลูกค้าและร้านค้าเองถ้ารั่วไหล

[14] เงินสดจ่ายมือต่อมือได้โดยไม่ต้องมีใครมารับรองว่าธุรกรรมเกิดขึ้นจริงและย้อนกลับไม่ได้ทันทีที่ส่งมอบ แต่พอต้องจ่ายผ่านอินเทอร์เน็ต ก่อนมี Bitcoin ยังไม่มีอะไรที่ทำหน้าที่แบบเงินสดได้

[15] cryptographic proof: การพิสูจน์ด้วยคณิตศาสตร์การเข้ารหัส ที่ใครก็ตรวจสอบได้เองโดยไม่ต้องเชื่อคำของใคร

[16] escrow: การพักเงินไว้กับคนกลางจนกว่าเงื่อนไขจะครบ เช่น ผู้ซื้อได้รับของแล้ว จึงปล่อยเงินให้ผู้ขาย ใน Bitcoin ทำได้ด้วยธุรกรรมที่ต้องใช้หลายลายเซ็น

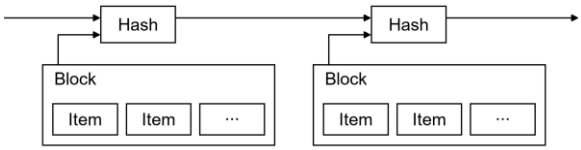
[17] public key / private key: คู่กุญแจเข้ารหัส public key เปิดเผยได้ ใช้เป็นที่อยู่รับเงิน ส่วน private key ใช้เซ็นอนุมัติการโอน ต้องเก็บเป็นความลับ

ปัญหาคือผู้รับเงินไม่สามารถตรวจสอบได้ว่าเจ้าของคนก่อนๆ ไม่ได้นำเหรียญนั้นไปใช้ซ้ำ (double-spend) ซึ่งวิธีที่ทั่วไปในการแก้ปัญหานี้คือการใช้ตัวกลางที่เชื่อถือได้แบบรวมศูนย์ (trusted central authority) หรือ Mint[18] ในการตรวจสอบทุกๆ ธุรกรรมว่ามีการใช้ซ้ำ (double-spend) หรือไม่ ซึ่งในทุกๆ ธุรกรรมเหรียญนั้นจะต้องกลับไปสู่ตัวกลางเพื่อสร้างเหรียญใหม่ และเฉพาะเหรียญที่ถูกสร้างจากตัวกลางโดยตรงเท่านั้นที่เราจะเชื่อได้ว่ามันไม่ได้ถูกนำไปใช้ซ้ำ (Double-spend) ซึ่งปัญหาของวิธีนี้คือชะตากรรมของระบบการเงินทั้งระบบจะขึ้นอยู่กับบริษัทที่ดำเนินการตัวกลางนั้น ซึ่งทุกๆ ธุรกรรมจะต้องผ่านตัวกลางนี้เหมือนกับระบบธนาคาร

เราต้องการวิธีที่ผู้รับเงินสามารถรู้ได้ว่าเจ้าของคนก่อนๆ ไม่ได้เซ็นธุรกรรมอื่นใดไปก่อนหน้านี้ สำหรับระบบของเรา ธุรกรรมที่เกิดก่อนสุดคือธุรกรรมที่นับ เราจึงไม่สนใจความพยายามใช้ซ้ำที่มาทีหลัง ซึ่งวิธีเดียวที่เราจะยืนยันได้ว่าไม่มีธุรกรรมอื่นอยู่ก่อนคือการรับรู้ธุรกรรมทั้งหมด โดยในระบบที่ใช้รูปแบบตัวกลาง ตัวกลางจะเห็นการทำธุรกรรมทั้งหมดและตัดสินว่าอันไหนมาก่อน การจะทำให้สำเร็จได้โดยไม่ต้องใช้ตัวกลางที่น่าเชื่อถือ การทำธุรกรรมทั้งหมดจะต้องถูกประกาศแก่สาธารณะ [1] และเราต้องการระบบที่ทำให้ผู้เข้าร่วมเห็นพ้องกันในประวัติเพียงชุดเดียวของลำดับที่ธุรกรรมถูกรับเข้ามา โดยผู้รับเงินต้องมีหลักฐานว่า ณ เวลาที่แต่ละธุรกรรมเกิดขึ้น Node ส่วนใหญ่เห็นด้วยว่านี่เป็นธุรกรรมที่ได้รับเป็นครั้งแรก

3. เซิร์ฟเวอร์ประทับเวลา (Timestamp Server)

ในการแก้ปัญหานี้เราจะเริ่มจาก timestamp server[19] โดย timestamp server จะเอาเลข Hash ของ Block รายการที่ต้องการบันทึกเวลา แล้วกระจายเลขนี้ออกไปในวงกว้าง เหมือนกับลงในหนังสือพิมพ์[20] หรือ Usenet post[21] [2-5] โดย timestamp พิสูจน์ว่าข้อมูลนั้นต้องมีอยู่จริง ณ เวลานั้น เพราะไม่เช่นนั้นจะเข้าไปอยู่ในเลข Hash ไม่ได้ ซึ่ง timestamp แต่ละอันจะรวม timestamp อันก่อนหน้าไว้ในเลข hash ของตัวเอง เกิดเป็นสายโซ่ โดย timestamp แต่ละอันที่เพิ่มเข้ามาจะยิ่งตอกย้ำอันก่อนหน้านี้ให้มั่นคงขึ้น



4. Proof-of-Work

ในการสร้างระบบ timestamp server แบบกระจายศูนย์[22] ที่เป็นแบบ peer-to-peer เราต้องใช้ระบบ Proof-of-work ซึ่งคล้ายกับ Adam Back's Hashcash [6][23] แทนที่จะใช้รูปแบบหนังสือพิมพ์ หรือ Usenet posts โดย Proof-of-work คือการไล่หาค่าค่าหนึ่ง ที่เมื่อนำไป Hash เช่นด้วยอัลกอริทึม SHA-256 แล้ว ค่า Hash จะเริ่มต้น

[18] Mint: โรงกษาปณ์ ในที่นี้คือหน่วยงานกลางที่ออกเหรียญและตรวจทุกธุรกรรม

[19] timestamp server: เซิร์ฟเวอร์ประทับเวลา ทำหน้าที่ยืนยันว่าข้อมูลชุดหนึ่งมีอยู่จริง ณ เวลาหนึ่ง

[20] หนังสือพิมพ์กับ Usenet ถูกยกมาในฐานะสื่อที่พิมพ์หรือกระจายออกไปหลายที่พร้อมกัน จนแก้ย้อนหลังไม่ได้ ไม่ใช่ในฐานะสื่อสำหรับอ่าน ใครก็ตรวจสอบได้ว่าข้อความนั้นมีอยู่จริงในวันนั้น ผู้เขียนงานวิจัยที่อ้างถึง [3][4] เคยตั้งบริษัทที่ทำแบบนี้จริง โดยลงค่า hash เป็นประกาศย่อยใน The New York Times ทุกสัปดาห์

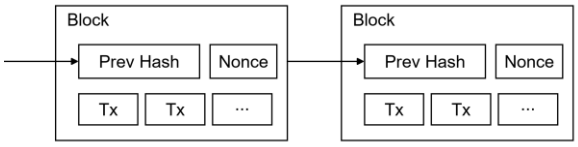
[21] Usenet: กระดานข่าวสาธารณะบนอินเทอร์เน็ตยุคก่อนเว็บ โพสต์แล้วถูกกระจายเก็บไว้หลายเซิร์ฟเวอร์ จึงใช้เป็นหลักฐานสาธารณะได้ว่ามีข้อความนั้น ณ เวลานั้น

[22] กระจายศูนย์ (distributed): ไม่มีเครื่องใดเครื่องหนึ่งเป็นศูนย์กลาง งานกระจายอยู่กับทุกเครื่องในเครือข่าย

[23] Hashcash: ระบบที่ Adam Back เสนอเมื่อปี 1997 ให้ผู้ส่งอีเมลต้องคำนวณ hash ที่ยากพอสมควรก่อนส่ง เพื่อให้การส่งสแปมจำนวนมากมีต้นทุน แนวคิดเดียวกันใช้ป้องกันการโจมตีแบบ denial-of-service ได้ด้วย Bitcoin นำหลักการนี้มาใช้กับการสร้างบล็อก

ด้วยบิต 0 จำนวนหนึ่ง โดย work หรือกำลังประมวลผลเฉลี่ยที่ต้องใช้จะเพิ่มขึ้นแบบ Exponential ตามจำนวนบิต 0 ที่ต้องการ แต่ตรวจสอบได้ด้วยการ Hash เพียงครั้งเดียว

ซึ่งสำหรับเครือข่าย timestamp ของเรา เราจะสร้าง Proof-of-work โดยเพิ่มค่า Nonce[24] ใน Block ไปเรื่อยๆ จนกว่าจะเจอค่าที่ทำให้ Block hash มีบิต 0 นำหน้าครบตามที่กำหนด เมื่อได้ใช้กำลัง CPU จนบล็อกผ่าน Proof-of-work แล้ว Block นั้นจะไม่สามารถเปลี่ยนแปลงได้โดยปราศจากการประมวลผลซ้ำ และเมื่อมี Block ถัดๆ ไปมาต่อโซ่ข้างหลัง กำลังประมวลผลที่ใช้ในการเปลี่ยนแปลง Block นั้นจะต้องรวมการประมวลผล Block ทั้งหมดหลังจากนั้นใหม่ด้วย



ระบบ Proof-of-work นั้นยังช่วยแก้ปัญหาในเรื่องการกำหนดตัวแทนในการตัดสินใจด้วยเสียงส่วนใหญ่ด้วย ถ้าการนับเสียงส่วนใหญ่นั้นเป็นระบบ 1 IP 1 vote มันอาจจะมีโอกาสที่ใครซักคนจะสร้าง IP จำนวนมากขึ้นมาเพื่อควบคุมระบบ แต่ระบบ Proof-of-work นั้นเป็นระบบ 1 CPU 1 vote เสียงส่วนใหญ่ในระบบนั้นจะแสดงออกผ่านห่วงโซ่ที่ยาวที่สุด ซึ่งเป็นห่วงโซ่ที่มีการลงแรง Proof-of-work มากที่สุด ถ้ากำลัง CPU ส่วนใหญ่ยังถูกควบคุมโดย Node ที่ซื่อสัตย์ ห่วงโซ่ที่ซื่อสัตย์ก็จะโตเร็วที่สุดและทิ้งห่างห่วงโซ่คู่แข่งอื่นๆ ในการที่จะแก้ไข Block ที่ผ่านไปแล้ว Attacker จะต้องทำ Proof-of-work ของ Block นั้นและ Block ทั้งหมดหลังจากนั้นใหม่ แล้วยังต้องไล่ให้ทันและแซงงานของ Node ที่ซื่อสัตย์ ซึ่งเราจะอธิบายในภายหลังว่าความเป็นไปได้ที่ Attacker ซึ่งช้ากว่าจะไล่ทันนั้นจะลดลงแบบ Exponential ในทุกๆ Block ที่เพิ่มขึ้นมาในห่วงโซ่

และด้วยการที่ความเร็วของฮาร์ดแวร์เพิ่มขึ้นเรื่อยๆ และความสนใจในการรัน Node เปลี่ยนไปตามเวลา ค่าความยาก (difficulty)[25] ในการทำ Proof-of-work จะถูกกำหนดจากค่าเฉลี่ยเคลื่อนที่ที่ตั้งเป้าจำนวน Block เฉลี่ยต่อชั่วโมง ถ้ามันถูกสร้างเร็วเกินไป ค่าความยากก็จะเพิ่มขึ้น

5. เครือข่าย (Network)

ระบบเครือข่ายของ Bitcoin จะทำตามขั้นตอนดังนี้

1) เมื่อธุรกรรมเกิดขึ้นมันจะถูกกระจายไปยังทุกๆ Node
2) แต่ละ Node จะนำธุรกรรมใหม่ๆ เหล่านี้ลงไปเก็บใน Block
3) แต่ละ Node จะพยายามหา Proof-of-work ที่ยากสำหรับ Block ของตัวเอง
4) เมื่อ Node สามารถหา Proof-of-work ได้แล้ว Node จะกระจาย Block ของตัวเองไปยังทุก Node
5) Node ที่เหลือจะยอมรับ Block ที่ส่งมาก็ต่อเมื่อทุกธุรกรรมใน Block นั้นถูกต้องและไม่เคยถูกใช้มาก่อน
6) Node แสดงการยอมรับ Block ที่ส่งมาด้วยการเริ่มสร้าง Block ถัดไปต่อจากมัน โดยใช้เลข Hash ของ Block ที่ยอมรับเป็นเลข previous hash

Node ในระบบนั้นจะถือว่าสายโซ่ที่ยาวที่สุดเป็นสายโซ่ที่ถูกต้องเสมอและจะสร้าง Block ถัดไปๆ จากสายโซ่นั้น ถ้ามี Node 2 แห่งกระจาย Block ถัดไปคนละเวอร์ชันออกมาพร้อมกัน Node บางแห่งอาจจะได้รับเวอร์ชันหนึ่งก่อน

[24] Nonce: ตัวเลขที่ใส่ไว้ในบล็อกเพื่อให้ไล่เปลี่ยนค่าได้ ผู้ขุดเปลี่ยน Nonce ไปเรื่อยๆ จนกว่า Hash ของบล็อกจะผ่านเงื่อนไข

[25] difficulty: ค่าที่กำหนดว่า Hash ต้องขึ้นต้นด้วย 0 กี่บิต ปัจจุบันเครือข่าย Bitcoin ปรับค่านี้ทุก 2,016 บล็อก ราวสองสัปดาห์ เพื่อรักษาเวลาเฉลี่ยราว 10 นาทีต่อบล็อก

อีกเวอร์ชันหนึ่ง ในกรณีนี้ Node จะทำงานต่อจาก Block ที่ตัวเองได้รับก่อน แต่เก็บอีกกิ่งหนึ่งไว้เผื่อว่ามันจะยาวกว่า ความแตกต่างนี้จะจบลงเมื่อมีคนหา Proof-of-work ถัดไปเจอและกิ่งหนึ่งยาวกว่าขึ้นมา Node ที่ทำงานอยู่บนอีกกิ่งก็จะย้ายมากิ่งที่ยาวกว่า

เวลาที่มีธุรกรรมใหม่เกิดขึ้นและกระจายไปในระบบนั้น ธุรกรรมเหล่านั้นไม่จำเป็นต้องกระจายไปถึง Node ทุก Node ตราบใดที่มันไปถึง Node จำนวนมากพอ อีกไม่นานมันก็จะถูกใส่เข้า Block การกระจาย Block ก็ทนต่อข้อความที่ตกหล่นได้เช่นกัน[26] ถ้ามีกรณีที่ Node ไม่ได้รับ Block ที่ส่งมา Node นั้นจะขอ Block ที่หายไปในตอนที่ได้รับ Block ถัดไป เพราะเมื่อ Node รับ Block ใหม่มา Node นั้นจะรู้ว่ามี Block ที่หายไป

6. แรงจูงใจ (Incentive)

โดยปกติแล้วธุรกรรมแรกใน Block จะเป็นธุรกรรมพิเศษที่สร้างเหรียญใหม่ให้แก่ผู้ที่สร้าง Block นั้น[27] การมอบเหรียญที่สร้างขึ้นใหม่นี้เป็นการสร้างแรงจูงใจให้ Node ทั้งหลายสนับสนุนระบบ และยังเป็นวิธีที่จะแจกจ่ายเหรียญเข้าสู่ระบบในช่วงเริ่มต้นอีกด้วย เนื่องจากในระบบนี้ไม่มีตัวกลางที่จะออกเหรียญ การเพิ่มเหรียญใหม่ในปริมาณคงที่อย่างสม่ำเสมอนี้จะคล้ายคลึงกับการขุดทองที่นักขุดจะต้องมีต้นทุนในการเพิ่มทองเข้าไปในระบบ ซึ่งในกรณีนี้คือเวลาของ CPU และพลังงานไฟฟ้าที่จะต้องใช้เป็นต้นทุน

แรงจูงใจนี้ยังมาจากค่าธรรมเนียมของธุรกรรมได้อีกด้วย โดยถ้ามูลค่าขาออกของธุรกรรมนั้นน้อยกว่ามูลค่าขาเข้า จำนวนเงินที่เป็นส่วนต่างนั้นคือค่าธรรมเนียมที่จะถูกบวกเป็นรางวัลเพิ่มเติมของ Block ที่มีธุรกรรมนั้น เมื่อเหรียญเข้าสู่ระบบครบตามจำนวนที่กำหนดไว้ล่วงหน้าแล้ว[28] รางวัลก็เปลี่ยนไปเป็นค่าธรรมเนียมของธุรกรรมทั้งหมดได้ และระบบจะปลอดจากเงินเฟ้อโดยสิ้นเชิง

ซึ่งรางวัลเหล่านี้เป็นสิ่งที่ช่วยกระตุ้นให้ Node ทั้งหลายยังทำงานอย่างซื่อสัตย์ ถ้ามี Attacker ที่โลภมากรวบรวมกำลัง CPU ได้มากกว่า Node ที่ซื่อสัตย์ทั้งหมด เขาก็ต้องเลือกว่าจะใช้มันโกงคนอื่นโดยการดึงเงินที่เขาจ่ายไปแล้วกลับคืน หรือใช้กำลังประมวลผลที่เขามีในการสร้างเหรียญใหม่ๆ ซึ่งเขาน่าจะพบว่าการเล่นตามกติกา ที่ให้เหรียญใหม่กับเขามากกว่าคนอื่นทั้งหมดรวมกัน ได้กำไรมากกว่าการทำลายระบบและความชอบธรรมของความมั่งคั่งของตัวเอง

7. การเรียกคืนพื้นที่ดิสก์ (Reclaiming Disk Space)

เมื่อธุรกรรมล่าสุดของเหรียญนั้นถูกฝังอยู่ใต้ Block มากพอแล้ว ธุรกรรมที่ถูกใช้ไปแล้วก่อนหน้านั้นสามารถถูกนำออกไปได้เพื่อประหยัดพื้นที่ในการจัดเก็บข้อมูล ซึ่งการที่จะทำสิ่งนี้ได้โดยไม่ทำให้เลข Hash ของ Block เสียหาย ธุรกรรมต่างๆ จะต้องถูก Hash ในรูปแบบของ Merkle Tree [7][2][5][29] โดยมีเฉพาะ Root เท่านั้นที่ถูกรวมไว้ในเลข hash ของ Block สำหรับ Block เก่าๆ นั้นสามารถบีบอัดข้อมูลได้โดยตัดกิ่งของข้อมูลออกจาก Tree ซึ่ง Hash ภายในต้นไม้ไม่จำเป็นต้องเก็บไว้

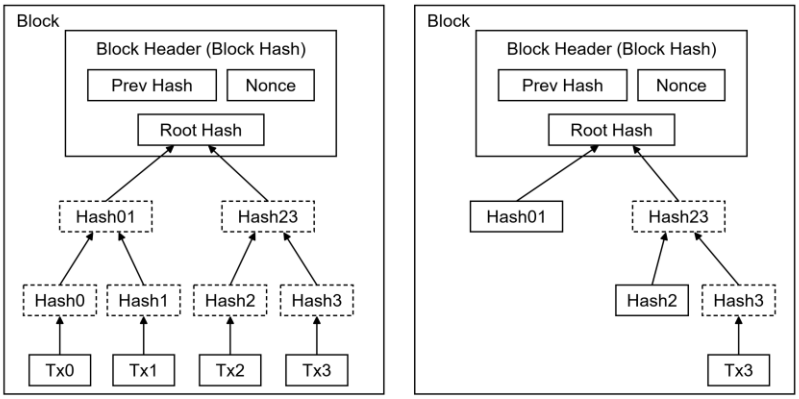
---

[26] ในทางปฏิบัติ Node จะไม่ประมวลผล Block หรือธุรกรรมที่เคยได้รับแล้วซ้ำอีก จึงส่งซ้ำหรือส่งหลายทางได้โดยไม่เกิดปัญหา

[27] ธุรกรรมนี้เรียกกันในภายหลังว่า coinbase transaction เหรียญที่ได้เรียกว่า block reward เริ่มต้นที่ 50 BTC ต่อบล็อก และลดลงครึ่งหนึ่งทุก 210,000 บล็อก ราวสี่ปี ที่เรียกว่า halving

[28] จำนวนที่กำหนดไว้ล่วงหน้าคือ 21 ล้าน BTC ซึ่งตามกำหนดจะออกครบราวปี 2140

[29] Merkle Tree: การจับคู่ข้อมูลแล้ว Hash ซ้อนขึ้นไปทีละชั้นแบบต้นไม้จนเหลือค่าเดียวที่ยอด (Root) ทำให้พิสูจน์ได้ว่าธุรกรรมหนึ่งอยู่ในบล็อกโดยไม่ต้องมีธุรกรรมทั้งบล็อก ตั้งชื่อตาม Ralph Merkle ผู้คิดค้น



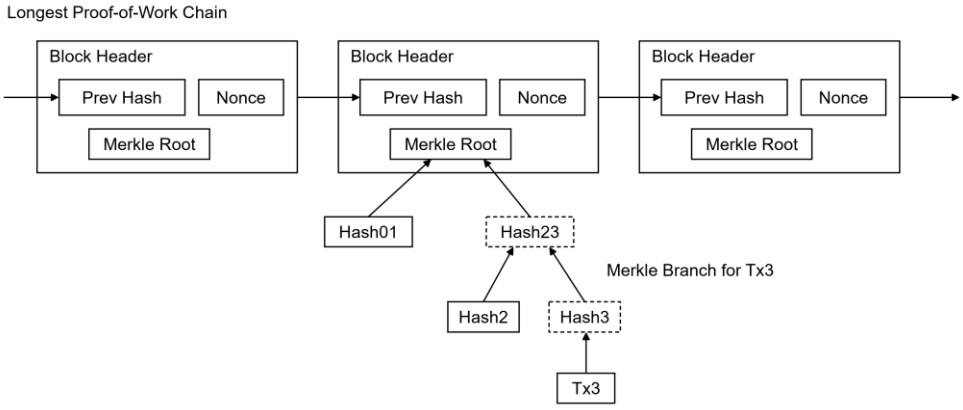
ธุรกรรมถูก Hash ในรูปแบบ Merkle Tree

หลังตัด Tx0-2 ออกจาก Block

ในส่วน Header[30] ของ Block ที่ไม่มีธุรกรรมนั้นจะมีขนาดประมาณ 80 bytes ถ้าทุก Block นั้นถูกสร้างขึ้นในเวลาทุกๆ 10 นาทีก็เท่ากับ 80 bytes * 6 * 24 * 365 = 4.2MB ต่อปี ด้วยระบบคอมพิวเตอร์ที่ปกติจะมีขนาดของ RAM ที่ 2GB ในปี 2008 และตามกฎของ Moore[31] ซึ่งประมาณการณ์ไว้ว่ามันจะเพิ่มขึ้นปีละ 1.2 GB ทำให้ขนาดพื้นที่ของการเก็บข้อมูลนั้นไม่ใช่ปัญหาแม้ว่า Header ของ Block จะต้องถูกเก็บในหน่วยความจำ

8. การตรวจสอบการชำระเงินแบบง่าย (Simplified Payment Verification)

การตรวจสอบการจ่ายเงินนั้นสามารถทำได้โดยไม่จำเป็นต้องรัน Full Node[32] ผู้ใช้งานเพียงแค่เก็บสำเนา Header ของห่วงโซ่ Proof-of-work ที่ยาวที่สุด ซึ่งผู้ใช้งานสามารถหาได้จากการดึงข้อมูลจาก Node ในเครือข่าย จนกว่าเขาจะมั่นใจว่าได้ห่วงโซ่ที่ยาวที่สุด และขอกิ่งของ Merkle[33] ที่เชื่อมธุรกรรมไปยัง Block ที่มีการบันทึกเวลาไว้ (timestamped) ผู้ใช้งานจะไม่สามารถตรวจสอบธุรกรรมด้วยตัวเองได้ แต่เมื่อเชื่อมโยงมันไปยังตำแหน่งหนึ่งในห่วงโซ่ ผู้ใช้งานจะสามารถเห็นได้ว่า Node ในเครือข่ายยอมรับธุรกรรมนั้นแล้ว และ Block ที่เพิ่มเข้ามาหลังจากนั้นก็ยิ่งยืนยันว่าเครือข่ายยอมรับธุรกรรมแล้ว



[30] Header: ข้อมูลส่วนหัวของ Block ที่มี hash ของบล็อกก่อนหน้า Nonce และ Root ของ Merkle Tree แต่ไม่มีตัวธุรกรรม

[31] กฎของมัวร์: ข้อสังเกตของ Gordon Moore ว่าจำนวนทรานซิสเตอร์บนวงจรรวมเพิ่มเป็นเท่าตัวประมาณทุกสองปี ตัวเลข 1.2 GB ต่อปีเป็นการประมาณของผู้เขียนในปี 2008

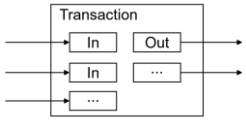
[32] Full Node: Node ที่เก็บและตรวจสอบบล็อกทั้งหมดของเครือข่ายครบถ้วน

[33] Merkle branch: ชุด hash ไม่กี่ค่าที่เชื่อมธุรกรรมหนึ่งขึ้นไปถึง Root ใช้พิสูจน์ว่าธุรกรรมนั้นอยู่ในบล็อกโดยไม่ต้องโหลดทั้งบล็อก วิธีนี้เรียกว่า SPV และเป็นหลักการของกระเป๋าเงินบนมือถือส่วนใหญ่

การตรวจสอบแบบนี้จะเชื่อถือได้ตราบเท่าที่ Node ที่ซื่อสัตย์ยังควบคุมเครือข่ายอยู่ แต่จะเปราะบางกว่าถ้าเครือข่ายถูก Attacker ที่มีกำลังมากกว่าครอบงำ ในขณะที่ Node ในเครือข่ายสามารถตรวจสอบธุรกรรมได้ด้วยตนเอง วิธีแบบง่ายนี้อาจถูกหลอกด้วยธุรกรรมปลอมที่ Attacker สร้างขึ้นตราบเท่าที่พวกเขายังครอบงำเครือข่ายได้ และวิธีหนึ่งที่จะป้องกันได้คือรับการแจ้งเตือนจาก Node ในเครือข่ายเมื่อตรวจพบ Block ที่ไม่ถูกต้อง ซึ่งจะกระตุ้นให้ซอฟต์แวร์ของผู้ใช้งานดาวน์โหลดข้อมูล Block ทั้งหมดและธุรกรรมที่ถูกแจ้งเตือนมาเพื่อยืนยันความผิดปกตินั้น ธุรกิจที่มีจำนวนการจ่ายเงินมากๆ น่าจะยังต้องการมี Node เป็นของตัวเองเพื่อความปลอดภัยที่เป็นอิสระกว่าและการตรวจสอบธุรกรรมที่รวดเร็วขึ้น

9. การรวมและแบ่งมูลค่า (Combining and Splitting Value)

ถึงแม้ว่าระบบสามารถจัดการเหรียญเป็นรายเหรียญได้ แต่มันคงจะเทอะทะเกินไปถ้าเราต้องแตกธุรกรรมออกเป็นหลายๆ ธุรกรรมสำหรับเงินทุกเซ็นต์ที่เราจะส่ง การที่เราจะทำให้มูลค่าถูกแบ่งและรวมกันได้ ธุรกรรมจะต้องมีขาเข้า (input) และขาออก (output) ได้หลายรายการ[34] ปกติแล้วจะมีขาเข้ารายการเดียวจากธุรกรรมก่อนหน้าที่มีมูลค่ามากกว่า หรือมีขาเข้าหลายรายการที่รวมเงินก้อนเล็กๆ หลายๆ จำนวน และมีขาออกอย่างมากสองรายการ คือส่วนที่เราจ่ายเงิน และส่วนที่เป็นเงินทอน (ถ้ามี) ส่งกลับไปหาผู้ส่ง



พึงสังเกตว่า fan-out[35] ที่ธุรกรรมขึ้นกับธุรกรรมย่อยๆ จำนวนมาก และธุรกรรมเหล่านั้นยังขึ้นกับธุรกรรมอื่นๆ จำนวนมากเช่นกัน ไม่ใช่ปัญหาในระบบนี้ เพราะไม่มีความจำเป็นต้องดึงประวัติทั้งหมดของธุรกรรมใดออกมาเป็นสำเนาเดี่ยวๆ เลย

10. ความเป็นส่วนตัว (Privacy)

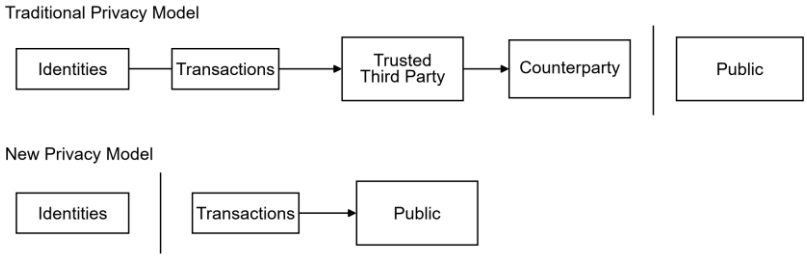
ในรูปแบบธนาคารทั่วไป ความเป็นส่วนตัวได้มาจากการจำกัดให้เฉพาะคู่ธุรกรรมและตัวกลางที่น่าเชื่อถือเท่านั้นที่เข้าถึงข้อมูล การประกาศทุกธุรกรรมสู่สาธารณะทำให้ใช้วิธีนั้นไม่ได้ แต่ความเป็นส่วนตัวนั้นยังสามารถรักษาได้โดยการตัดการไหลของข้อมูลที่จุดอื่นแทน และวิธีนั้นคือการทำให้ Public key นั้นกลายเป็นสิ่งที่ไม่มีการระบุตัวตน[36] ถ้ามองจากมุมมองข้างนอกทุกคนจะสามารถเห็นได้ว่ามีใครบางคนกำลังส่งเงินจำนวนหนึ่งให้กับอีกคนหนึ่ง แต่จะไม่มีข้อมูลที่จะเชื่อมโยงธุรกรรมไปยังบุคคลใดๆ วิธีนี้มันก็คล้ายๆ กับข้อมูลที่ตลาดหลักทรัพย์เปิดเผย คือเวลาและขนาดของการซื้อขายแต่ละรายการ หรือ "tape"[37] ถูกเปิดเผยต่อสาธารณะ แต่จะไม่บอกว่าใครเป็นคู่ค้า

[34] input / output: ขาเข้าคือเงินที่ธุรกรรมนี้หยิบมาใช้จากขาออกของธุรกรรมก่อนหน้า ขาออกคือปลายทางที่เงินถูกส่งไป เหมือนเอาแบงก์หลายใบมารวมจ่าย แล้วได้เงินทอนกลับเป็นแบงก์ใบใหม่

[35] fan-out: การแตกกิ่งของการอ้างอิง ธุรกรรมหนึ่งอ้างถึงหลายธุรกรรม และธุรกรรมเหล่านั้นอ้างต่อไปอีกมาก

[36] เนื่องจากไม่มีการใส่ข้อมูลส่วนตัวใดๆ ลงใน public key

[37] tape: บันทึกราคาและปริมาณซื้อขายแบบต่อเนื่องของตลาดหุ้น มาจากยุคที่พิมพ์ลงแถบกระดาษ ticker tape



และเพื่อเป็นกำแพงกันอีกชั้น ผู้ใช้งานควรสร้างคู่กุญแจใหม่ทุกครั้งในทุกๆ การโอน[38] เพื่อไม่ให้มีข้อมูลที่เชื่อมโยงธุรกรรมเหล่านั้นไปสู่เจ้าของคนเดียวกัน แต่มันก็จะมีข้อมูลเชื่อมโยงบางอย่างที่ไม่สามารถปกปิดได้ เช่นธุรกรรมที่มีขาเข้าหลายรายการ ซึ่งย่อมเปิดเผยว่าขาเข้าเหล่านั้นเป็นของเจ้าของคนเดียวกัน ซึ่งความเสี่ยงนั้นคือหากมีข้อมูลที่เปิดเผยว่าคนๆ นั้นเป็นเจ้าของกุญแจ มันอาจจะเชื่อมโยงไปยังธุรกรรมอื่นๆ อีกว่ามันเป็นของคนๆ นั้น[39]

11. การคำนวณ (Calculations)

ถ้าเรามองว่าการโจมตีของ Attacker คือการที่เขาพยายามสร้างห่วงโซ่ทางเลือกให้เร็วกว่าห่วงโซ่ที่ถูกต้อง แม้ว่าเขาจะทำสำเร็จ มันก็ไม่ได้แปลว่าเขาจะสามารถเปลี่ยนแปลงระบบได้ตามใจชอบ เช่น Attacker ก็ไม่สามารถเสกเงินขึ้นมาจากอากาศหรือขโมยเงินที่ไม่เคยเป็นของ Attacker ได้ เนื่องจาก Node จะไม่ยอมรับธุรกรรมที่ไม่ถูกต้องเป็นการจ่ายเงิน[40] และ Node ที่ซื่อสัตย์จะไม่มีวันยอมรับ Block ที่มีธุรกรรมแบบนั้น สิ่งเดียวที่ Attacker สามารถทำได้คือพยายามแก้ไขธุรกรรมของตัวเองรายการหนึ่ง เพื่อเอาเงินที่เขาเพิ่งใช้ไปกลับคืนมา

ซึ่งการแข่งขันระหว่างห่วงโซ่ที่ถูกต้องและห่วงโซ่ของ Attacker จะเป็นลักษณะของ Binomial Random Walk[41] เหตุการณ์สำเร็จคือห่วงโซ่ที่ถูกต้องสร้าง Block ได้ ทำให้ระยะนำเพิ่มขึ้น +1 และเหตุการณ์ล้มเหลวคือห่วงโซ่ของ Attacker สร้าง Block ได้ ทำให้ช่องว่างลดลง -1

โอกาสที่ Attacker จะไล่ทันจากระยะที่ตามหลังอยู่นั้นเทียบได้กับ Gambler's Ruin problem[42] สมมติว่ามีนักพนันที่มีเครดิตไม่จำกัด เริ่มจากสถานะติดลบ และเล่นได้ไม่จำกัดจำนวนครั้งเพื่อพยายามให้ถึงจุด Breakeven หรือก็คือจุดที่ Attacker จะสร้างห่วงโซ่ทันห่วงโซ่ที่ถูกต้อง ซึ่งคำนวณได้ดังนี้ [8]

p = ความน่าจะเป็นที่ Node ที่ซื่อสัตย์จะเจอ Block ถัดไป

q = ความน่าจะเป็นที่ Attacker จะเจอ Block ถัดไป

qz = ความน่าจะเป็นที่ Attacker จะไล่ทันได้ในที่สุด โดย z คือจำนวน Block ที่ตามหลังอยู่

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

ถ้าเราตั้งสมมติฐานว่า p>q หมายความว่า ความน่าจะเป็นจะตกลงแบบ Exponential ตามจำนวน Block ที่ Attacker ต้องตามให้ทัน ซึ่งมันทำให้ Attacker นั้นเสียเปรียบ ถ้าเขาไม่ได้ดวงดีพุ่งแซงตั้งแต่แรก โอกาสที่เขาจะตามทันจะน้อยมากเมื่อเขายิ่งตามหลังไกลออกไป

[38] ในทางปฏิบัติคือใช้ address ใหม่ทุกครั้ง

[39] เช่นถ้ารู้ว่า Address หนึ่งมีใครเป็นเจ้าของ ก็อาจสืบต่อไปได้ว่าเขาได้เงินจากใครหรือส่งเงินให้ใครบ้าง

[40] เช่นเขียนว่ามีเงินเข้ากระเป๋าเราหรือเอาเงินจากคนอื่นเข้ากระเป๋าเราแต่ Digital signature นั้นไม่ถูกต้อง Node อื่นก็จะไม่รับ

[41] Binomial Random Walk: แบบจำลองการเดินสุ่มที่แต่ละก้าวมีสองผลลัพธ์ บวกหนึ่งหรือลบหนึ่ง ด้วยความน่าจะเป็นคงที่

[42] Gambler's Ruin: โจทย์คลาสสิกในทฤษฎีความน่าจะเป็น ว่านักพนันที่เล่นเกมที่เสียเปรียบจะหมดตัวด้วยความน่าจะเป็นเท่าใด

ต่อไปเราลองพิจารณาว่าผู้รับจะต้องรอนานเท่าไหร่ถึงจะมั่นใจได้ว่าผู้ส่งจะเปลี่ยนแปลงธุรกรรมไม่ได้แล้ว ถ้าเราลองคิดว่า Attacker คือผู้ส่งเงินที่ต้องการทำให้ผู้ที่รับเงินเชื่อว่าเขาส่งเงินแล้วสักพัก หลังจากนั้นค่อยสลับให้ธุรกรรมนั้นจ่ายกลับคืนตัวเองหลังจากเวลาผ่านไป แม้ผู้รับจะได้รับการแจ้งเตือนเมื่อเกิดขึ้น แต่ Attacker ก็จะหวังว่ามันคงจะสายเกินไป

ผู้รับสร้างคู่กุญแจใหม่และเอา Public key ให้แก่ผู้ส่งไม่นานก่อนที่จะมีการ Sign[43] มันจะช่วยป้องกันไม่ให้ผู้ส่งสร้างห่วงโซ่เตรียมไว้ล่วงหน้า ด้วยการขุดไปเรื่อยๆ จนโชคดีนำไปไกลพอ แล้วค่อยทำธุรกรรมในจังหวะนั้น เมื่อธุรกรรมถูกส่งไปแล้ว Attacker ที่เป็นผู้ส่งจึงจะเริ่มแอบสร้างห่วงโซ่อีกสายหนึ่งที่มีธุรกรรมของเขาในเวอร์ชันอื่น

ผู้รับรอจนธุรกรรมของเขาถูกใส่เข้า Block และมี Block ต่อไปอีก z Block ซึ่งเขาไม่รู้ว่า Attacker นั้นสามารถสร้าง Block ของตัวเองไปถึงเท่าไหร่แล้ว แต่ถ้าสมมติว่า Block ที่ถูกสร้างอย่างถูกต้องนั้นใช้เวลาในการสร้างตามค่าเฉลี่ยที่คาดไว้ ความคืบหน้าที่เป็นไปได้ของ Attacker จะเป็นการแจกแจงความน่าจะเป็นแบบปัวซง (Poisson distribution)[44] ที่มีค่าคาดหมายเท่ากับ

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

ในการที่จะได้ค่าความน่าจะเป็นที่ Attacker ยังจะไล่ทันได้ ณ ตอนนี้ เราต้องคูณค่าความหนาแน่น Poisson ของความคืบหน้าแต่ละระดับที่ Attacker อาจทำได้ ด้วยความน่าจะเป็นที่เขาจะสามารถไล่ทันได้จากจุดๆ นั้น

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

จัดเรียงสูตรใหม่เพื่อเลี่ยงการบวกหางอนันต์ของการแจกแจง

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

เปลี่ยนเป็นภาษา C

```
#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;
}
```

ลองรันดูผลลัพธ์ แล้วเราจะเห็นว่าความน่าจะเป็นนั้นจะตกลงแบบ exponential ตามค่า z

[43] Sign: การลงลายเซ็นดิจิทัลยืนยันธุรกรรมโดยผู้ส่ง

[44] Poisson distribution: การแจกแจงของจำนวนเหตุการณ์ที่เกิดในช่วงเวลาหนึ่ง เมื่อเหตุการณ์เกิดอย่างอิสระด้วยอัตราเฉลี่ยคงที่ เหมาะกับการนับจำนวนบล็อกที่ขุดได้ในช่วงเวลาหนึ่ง

```
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
```

หาค่า z ที่ทำให้ P น้อยกว่า 0.1%

```
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. บทสรุป (Conclusion)

เราได้นำเสนอระบบสำหรับธุรกรรมอิเล็กทรอนิกส์ที่ไม่ต้องพึ่งพาความน่าเชื่อถือ เราเริ่มจากกรอบเดิมของเหรียญที่สร้างจาก Digital signature ซึ่งทำให้มันมีจุดแข็งในด้านการควบคุมความเป็นเจ้าของ (ownership) แต่ยังไม่สมบูรณ์เพราะไม่สามารถป้องกัน Double spending ได้ ซึ่งในการแก้ไขในจุดนี้เราจึงขอนำเสนอเครือข่ายแบบ peer-to-peer ที่ใช้ระบบ Proof-of-work ในการบันทึกประวัติธุรกรรมแบบสาธารณะ ซึ่งจะกลายเป็นสิ่งที่ Attacker เปลี่ยนแปลงได้ยากมากในทางการคำนวณอย่างรวดเร็ว ถ้า Node ที่ซื่อสัตย์ยังควบคุมกำลังของ CPU ส่วนใหญ่อยู่ ระบบนี้แข็งแกร่งด้วยความเรียบง่ายที่ไม่มีโครงสร้าง[45] Node ทั้งหมดจะทำงานพร้อมกันโดยประสานงานกันเพียงเล็กน้อย การระบุตัวตนไม่จำเป็น เพราะข้อความไม่ได้ถูกส่งไปที่จุดใดจุดหนึ่งเป็นการเฉพาะ และเพียงต้องส่งถึงแบบ best effort Node สามารถออกหรือเข้าร่วมระบบเมื่อไหร่ก็ได้ตามที่ต้องการ เพราะห่วงโซ่ Proof-of-work จะเป็นสิ่งที่บอกว่าเกิดอะไรขึ้นบ้างในช่วงที่ Node หายไป Node โหวตด้วยกำลัง CPU ของตัวเอง โดยยอมรับ Block ที่ถูกต้องด้วยการต่อยอดจากมัน และปฏิเสธ Block ที่ไม่ถูกต้องด้วยการไม่ทำงานต่อจากมัน ซึ่งกฎและแรงจูงใจต่างๆ ที่จำเป็นจะถูกบังคับใช้ผ่านกลไก Consensus[46] นี้

[45] unstructured simplicity: เครือข่ายไม่มีลำดับชั้น ไม่มี Node หัวหน้า ไม่ต้องลงทะเบียน ทุก Node เท่ากันและทำงานด้วยกติกาง่ายๆ ชุดเดียว ความเรียบง่ายนี้เองที่ทำให้มันทนทานต่อการล้มของ Node ใด Node หนึ่ง

[46] Consensus: ฉันทามติ

อ้างอิง (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.

หมายเหตุจากผู้แปล

ผู้แปลฉบับนี้คือผู้แปลคนเดียวกับฉบับภาษาไทยที่เผยแพร่อยู่บน bitcoin.org ตั้งแต่ปี 2562 (ค.ศ. 2019) ฉบับนี้จึงเป็นการกลับมาปรับปรุงงานแปลของตัวเอง โดยยึดตัวบทและสำนวนเดิมเป็นฐาน เพื่อให้ตัวบทตรงกับต้นฉบับของ Satoshi Nakamoto มากที่สุด สิ่งที่ปรับในฉบับนี้มีดังนี้

- แก้จุดที่แปลคลาดเคลื่อน ประโยคที่ตกหล่น และคำสะกด โดยคงสำนวนเดิมไว้
- ย้ายคำอธิบายศัพท์และบริบทที่เดิมแทรกอยู่ในเนื้อความ ไปไว้เป็นเชิงอรรถท้ายหน้า
- วาดภาพประกอบทั้ง 7 ภาพใหม่ตามต้นฉบับ โดยคงป้ายภาษาอังกฤษไว้

ฉบับแปลภาษาไทย ปรับปรุง พ.ศ. 2569

แปลโดย พีรพัฒน์ หาญคงแก้ว (Peeraphat Hankongkaew) · Bitcoin Addict

ร่วมกับ Claude Fable 5.1 (Anthropic) · ผู้แปลตรวจทานและรับผิดชอบเนื้อหาทั้งหมด

ต้นฉบับ: Bitcoin: A Peer-to-Peer Electronic Cash System · Satoshi Nakamoto · 31 ตุลาคม 2008

bitcoin.org/bitcoin.pdf · bitcoinaddict.com/articles/bitcoin-whitepaper-thai