बिटकॉइन: परस्परांमधील इलेक्ट्रॉनिक चलनाची प्रणाली

सतोशी नाकामोटो
satoshin@gmx.com
www.bitcoin.org

शिवाजी आंबेडकर यांनी मराठीत अनुवादित केले आहे

सारांश. परस्परांमधील इलेक्ट्रॉनिक चलनाची शुद्ध आवृत्ती कोणत्याही वित्तीय संस्थेशिवाय ऑनलाइन पेमेंट एका पक्षाकडून दुसऱ्या पक्षाकडे पाठवण्याची परवानगी देईल. डिजिटल स्वाक्षरी हा उपायाचा एक भाग आहे, परंतु दुहेरी खर्च टाळण्यासाठी विश्वासार्ह तृतीय पक्षाची आवश्यकता असल्यास डिजिटल स्वाक्षरीचा मुख्य फायदा गमावला जातो. आम्ही दुहेरी-खर्चाच्या समस्येवर परस्परांचा नेटवर्क वापरून एक उपाय मांडतो. हा नेटवर्क हॅश-आधारित कामाच्या-पुराव्याच्या चालू साखळीत व्यवहारांना हॅश करून टाईमस्टॅम्प करतो, ज्यांनी कि एक रेकॉर्ड तयार होतो जो कामाचा-पुरावा पुन्हा केल्याशिवाय बदलला जाऊ शकत नाही. सर्वात लांब साखळी केवळ घटनांच्या क्रमाचा पुरावा म्हणून काम करत नाही, तर ती CPU शक्तीच्या सर्वात मोठ्या वंशातून आली याचा देखील पुरावा देते. जोपर्यंत बहुमत CPU शक्ती नेटवर्कवर हल्ला करण्यासाठी समर्थन न देणाऱ्या नोड्सद्वारे नियंत्रित केले जातात, तोपर्यंत ते सर्वात लांब साखळी उत्पन्न करतील आणि हल्लेखोरांना पराभूत करतिल. या नेटवर्कसाठी कमित कमी संरचनेची आवश्यकता आहे. संदेश सर्वोत्तम प्रयत्न आधारावर प्रसारित केले जातात, आणि नोड्स इच्छेनुसार नेटवर्क सोडू शकतात आणि पुन्हा सामील होऊ शकतात, ते गेल्यावर काय घडले याचा पुरावा म्हणून ते सर्वात लांब कामाच्या-पुराव्याची साखळी स्वीकारतात.

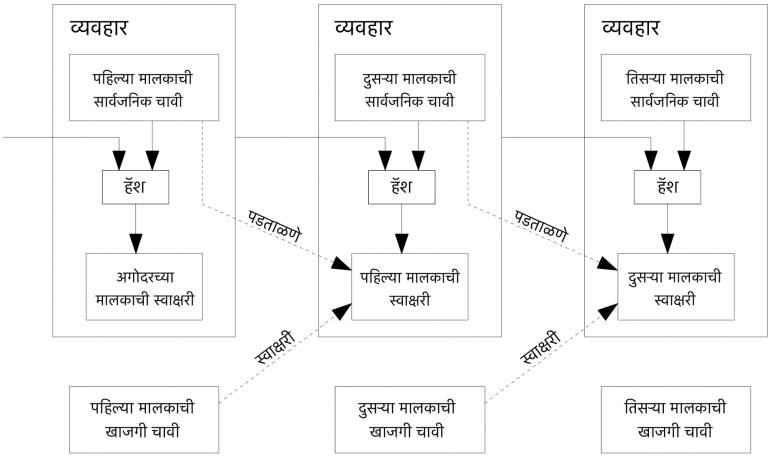
१. प्रस्तावना

इंटरनेटवरील वाणिज्य इलेक्ट्रॉनिक पेमेंट्सवर प्रक्रिया करण्यासाठी विश्वासार्ह तृतीय पक्ष म्हणून सेवा देणाऱ्या वित्तीय संस्थांवर जवळजवळ पूर्णपणे अवलंबून आहे. जरी ही प्रणाली बहुतेक व्यवहारांसाठी पुरेशी असली, तरीही ती विश्वास आधारित मॉडेलच्या अंतर्निहित कमकुवतपणामुळे ग्रस्त आहे. पूर्णपणे न-परतता येणारे व्यवहार खरोखरच शक्य नाहीत, कारण वित्तीय संस्था मध्यस्थी करणारे विवाद टाळू शकत नाहीत. आकार मर्यादित करून आणि लहान आपत्कालीन व्यवहारांची शक्यता काढून टाकून व्यवहारांची किंमत वाढवते आणि उलट न करता येण्याजोग्या सेवांसाठी अपरिवर्तनीय पेमेंट करण्याची क्षमता गमावण्यामध्ये एक व्यापक किंमत आहे. पायमेन्ट उलटता येतात म्हणून विश्वाशी मध्यस्थांची गरज वाढते. व्यापाऱ्यांनी त्यांच्या ग्राहकांपासून सावधानी बाळगावी लागते, व ग्राहकांना आवश्यकतेपेक्षा अधिक माहितीसाठी त्रास द्यावा लागतो. थोडी फसवणूक अपरिहार्य मानली जाते. भौतिक चलन वापरून हे घरचं व पेमेंट अनिश्चितता टाळता येऊ शकते, परंतु विश्वसनीय पक्षाशिवाय संप्रेषण चॅनेलवर पेमेंट करण्यासाठी कोणतीही यंत्रणा अस्तित्वात नाही.

विश्वासाऐवजी क्रिप्टोग्राफिक पुराव्यावर आधारित इलेक्ट्रॉनिक पेमेंट सिस्टमची गरज आहे, जी कोणत्याही दोन इच्छुक पक्षांना विश्वासार्ह तृतीय पक्षाची गरज न घेता एकमेकांशी थेट व्यवहार करू देते. संगणकीयदृष्ट्या अव्यवहार्य असलेले अपरिवर्तनीय व्यवहार विक्रेत्यांची फसवणुकीपासून संरक्षण करतील आणि खरेदीदारांचे संरक्षण करण्यासाठी नियमित एस्क्रो यंत्रणा सहजपणे लागू केली जाऊ शकतात. या पेपर मध्ये, आम्ही परस्परांचा वितरित टाइमस्टॅम्प सर्व्हर वापरून व्यवहारांच्या कालक्रमानुसार संगणकीय पुरावा व्युत्पन्न करून दुहेरी-खर्चाच्या समस्येवर एक उपाय सुचवतो. जोपर्यंत प्रामाणिक नोड्स एकत्रितपणे आक्रमणकर्त्या नोड्सच्या कोणत्याही सहयोगी गटापेक्षा अधिक CPU शक्ती नियंत्रित करतात तोपर्यंत ही प्रणाली सुरक्षित आहे.

२. व्यवहार

आम्ही इलेक्ट्रॉनिक नाणे डिजिटल स्वाक्षरीची साखळी म्हणून परिभाषित करतो. प्रत्येक मालक नाण्याच्या मागील व्यवहार व पुढच्या मालकाच्या सार्वजनिक चावी यांच्या एकत्रित हॅशवर डिजिटल स्वाक्षरी करून आणि मग नाण्याच्या शेवटी जोडून ते नाणे पुढीलकडे हस्तांतरित करतो. मालकीच्या साखळीची पडताळणी करण्यासाठी स्वीकारणारा स्वाक्षरी सत्यापित करू शकतो.

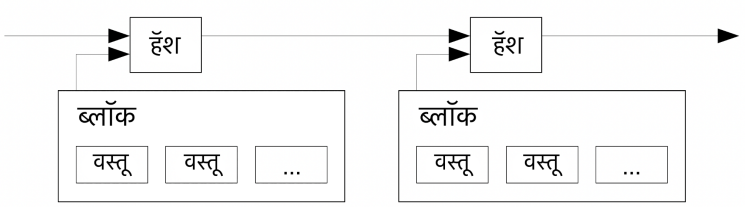


अर्थातच अडचण अशी आहे की प्राप्तकर्ता हे सत्यापित करू शकत नाही की मालकांपैकी एकाने नाणे दुप्पट खर्च केले नाही. एक सामान्य उपाय म्हणजे विश्वासार्ह केंद्रीय प्राधिकरण, किंवा टांकसाळ, जी प्रत्येक व्यवहार दुप्पट खर्चासाठी तपासते. प्रत्येक व्यवहारानंतर, नवीन नाणे जारी करण्यासाठी नाणे टांकसाळीकडे परत केले जाणे आवश्यक आहे आणि केवळ टांकसाळीतून थेट जारी केलेली नाणी दुप्पट खर्च झाली नाहीत यावर विश्वास ठेवला जातो. या उपायातील समस्या अशी आहे की संपूर्ण चलन प्रणालीचे भवितव्य टांकसाळ चालवणाऱ्या कंपनीवर अवलंबून असते, प्रत्येक व्यवहार बँकेप्रमाणेच त्यांच्यामार्फत करावा लागतो.

आधीच्या मालकांनी पूर्वीच्या कोणत्याही व्यवहारांवर स्वाक्षरी केलेली नाही हे जाणून घेण्यासाठी आम्हाला प्राप्तकर्त्यासाठी मार्ग हवा आहे. आमच्या उद्देशांसाठी, सर्वात पहिला व्यवहार हा महत्वाचाहे आहे, त्यामुळे दुप्पट खर्च करण्याच्या नंतरच्या प्रयत्नांची आम्ही पर्वा करत नाही. व्यवहाराच्या अनुपस्थितीची पुष्टी करण्याचा एकमेव मार्ग म्हणजे सर्व व्यवहारांची माहिती असणे. टांकसाळीवर आधारित मॉडेलमध्ये, टांकसाळीला सर्व व्यवहारांची माहिती होती आणि त्यावरून कोणता व्यवहार प्रथम आला हे ठरवले. विश्वासार्ह पक्षाशिवाय हे पूर्ण करण्यासाठी, व्यवहार सार्वजनिकपणे घोषित केले जाणे आवश्यक आहे [१], आणि आम्हाला सहभागींनी प्राप्त केलेल्या व्यवहारांच्या क्रमाच्या एका इतिहासावर सहमती देण्यासाठी एक प्रणाली आवश्यक आहे. प्राप्तकर्त्याला पुराव्याची आवश्यकता आहे की प्रत्येक व्यवहाराच्या वेळी, बहुसंख्य नोड्स सहमत होते की तो व्यवहार प्रथम प्राप्त झाला होता.

३. टाईमस्टँप सर्व्हर

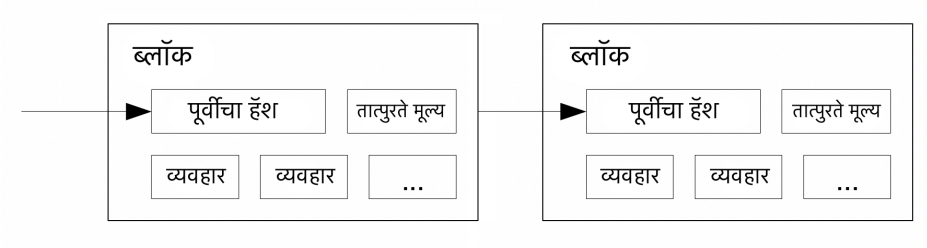
आम्ही सुचवलेला उपाय टाइमस्टॅम्प सर्व्हरपासून सुरू होतो. टाइमस्टॅम्प सर्व्हर टाइमस्टँप करण्यासाठी वस्तूंच्या ब्लॉकचा हॅश घेतो आणि हॅश मोठ्या प्रमाणावर प्रकाशित करतो, जसे की वर्तमानपत्र किंवा युजनेट पोस्ट [२-५]. टाइमस्टॅम्प हे सिद्ध करतो की हॅशवर जाण्यासाठी डेटा त्यावेळी अस्तित्वात होता. प्रत्येक टाइमस्टॅम्पच्या हॅशमध्ये त्याचा मागील टाइमस्टॅम्प असतो, अशा प्रकारे एक साखळी बनत जाते, प्रत्येक अतिरिक्त टाइमस्टॅम्प त्याच्या आधीच्या टाइमस्टॅम्पस्ना अधिक मजबूत करतो.



४. कामाचा-पुरावा

परस्परांच्या आधारावर वितरित टाइमस्टॅम्प सर्व्हर कार्यान्वित करण्यासाठी, आम्हाला वृत्तपत्र किंवा युजनेट पोस्ट ऐवजी ॲडम बॅकच्या हॅशकॅश [६] प्रमाणेच कामाच्या-पुराव्याची प्रणाली वापरावी लागेल. कामाच्या पुराव्यासाठी असे मूल्य स्कॅन करणे समाविष्ट आहे जे हॅश केल्यावर, जसे की SHA-256 हॅश सह, तो हॅश शून्य बिट्सच्या संख्येने सुरू होईल. संख्येच्या सुरवातीला जितके आवश्यक शून्य बिट्स असतील तितकेच सरासरीने कार्य घातांकीय असणार आणि एकल हॅश कार्यान्वित करून सत्यापित केले जाऊ शकतो.

आमच्या टाइमस्टॅम्प नेटवर्कसाठी, ब्लॉकच्या हॅशला आवश्यक शून्य बिट्स देणारे मूल्य सापडेपर्यंत आम्ही ब्लॉकमध्ये तात्पुरती वाढ करून कामाचा-पुरावा कार्यान्वित करतो. एकदा CPU ने प्रयत्न खर्च करून कामाच्या-पुराव्याची पूर्तता केल्यावर, ते काम पुन्हा केल्याशिवाय ब्लॉक बदलता येणार नाही. नंतरचे ब्लॉक्स त्याच्या नंतर साखळीने बांधलेले असल्यामुळे एक ब्लॉक बदलण्यासाठी, त्यानंतरचे सर्व ब्लॉक्स बदलावे लागतील.



बहुमताच्या निर्णयामध्ये प्रतिनिधित्व निश्चित करण्यासाठी सुद्धा कामाच्या-पुराव्याचा उपयोग होतो. जर बहुमत एक-IP-पत्ता-एक-मतावर आधारित असेल, तर अनेक IP वाटप करू शकणाऱ्या कोणीही ते विघटित करू शकतात. कामाचा पुरावा म्हणजे मूलत: एक-CPU-एक-मत. बहुमत निर्णय हे सर्वात लांब साखळीद्वारे दर्शविले जाते, ज्यामध्ये कामाचा सर्वात मोठा पुरावा आहे. CPU ची बहुतेक शक्ती प्रामाणिक कामगारांद्वारे नियंत्रित केली असल्यास, प्रामाणिक शृंखला सर्वात वेगवान चालेल आणि कोणत्याही संघर्ष साखळीला मागे टाकेल. मागील ब्लॉक बदलण्यासाठी, आक्रमणकर्त्याला त्या ब्लॉकच्या व नंतरच्या सर्व ब्लॉक्सच्या कामाची पुन्हा पुनरावृत्ती करावी लागेल आणि मग त्याला प्रामाणिक नोडस्च्या कामापर्यंत पोहोचावे लागेल आणि नंतर त्यांना मागे टाकावे लागेल. आम्ही नंतर दर्शवू की पुढील ब्लॉक्स जोडले गेल्याने हळूवार आक्रमणकर्त्याला प्रामाणिक शृंखले पर्यंत पोहोचण्याची संभाव्यता वेगाने कमी होते.

हार्डवेअरची वाढती गती आणि नोडस् चालवण्याच्या बदलत्या स्वारस्या मुळे, प्रति तास ब्लॉक्सची सरासरी संख्या लक्ष्यित करत चालणारी सरासरी, ही कामाच्या-पुराव्याची कठीणता ठरवते. जर ब्लॉक्स खूप लवकर तयार होत असतील तर कठीणता वाढते.

५. नेटवर्क

नेटवर्क चालवण्याच्या पायऱ्या खालीलप्रमाणे आहेत:

1) नवीन व्यवहार सर्व नोड्सवर प्रसारित केले जातात.
2) प्रत्येक नोड नवीन व्यवहार एका ब्लॉकमध्ये गोळा करतो.
3) प्रत्येक नोड त्याच्या ब्लॉकसाठी कामाचा कठीण पुरावा शोधण्याचे काम करतो.
4) जेव्हा नोडला कामाचा पुरावा सापडतो तेव्हा तो ब्लॉकला सर्व नोड्सवर प्रसारित करतो.
5) नोड्स ब्लॉक फक्त तेव्हाच स्वीकारतात जेव्हा त्यातील सर्व व्यवहार वैध असतील आणि आधीच खर्च केलेले नसतील.
6) नोड्स ब्लॉकची स्वीकृती व्यक्त करण्यासाठी स्वीकृत ब्लॉकचा हॅश मागील हॅश म्हणून वापरून, साखळीतील पुढील ब्लॉक तयार करण्यावर काम करतात.

नोड्स नेहमी सर्वात लांब साखळी योग्य मानतात आणि ती वाढवण्याचे काम करत राहतात. जर दोन नोड्स एकाच वेळी पुढील ब्लॉकच्या वेगवेगळ्या आवृत्त्या प्रसारित करत असतील, तर काही नोड्सना पहिला ब्लॉक मिळेल आणि काहींना दुसरा ब्लॉक मिळेल. अशावेळी, त्यांना मिळालेल्या पहिल्या शाखेवर ते काम करतात, परंतु दुसरी शाखा लांब होण्याच्या शक्यतेमुळे ती सुद्धा जतन करतात. जेव्हा एका शाखेला पुढच्या कामाचा पुरावा सापडेल तेव्हा ही बरोबरी तुटेल; जे नोड्स इतर शाखांवर काम करत आहेत ते त्या सोडतील आणि सर्वात लांब शाखेत येतील.

नवीन व्यवहार प्रसारणास सर्व नोड्सपर्यंत पोहोचणे आवश्यक नाही. जोपर्यंत ते अनेक नोड्सपर्यंत पोहोचतात, तोपर्यंत ते ब्लॉकमध्ये जातील. ब्लॉक प्रसारण हरवलेल्या संदेशांच्या प्रति पण सहनशील आहे. जर नोडला एखादा ब्लॉक नाही मिळाला, तर जेव्हा त्याच्या पुढचा ब्लॉक त्याला मिळेल आणि आधीचा ब्लॉक चुकल्याचे लक्षात आल्यावर तो नोड त्याची विनंती करेल.

६. प्रोत्साहन

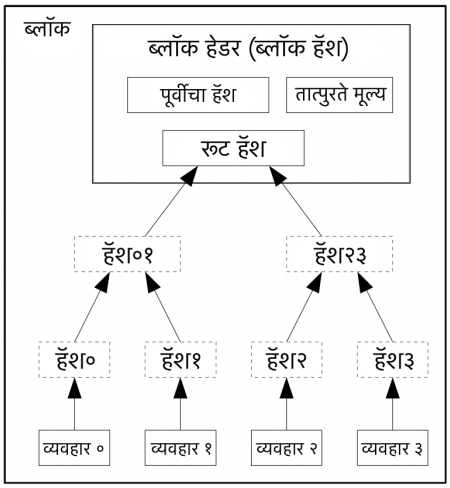
नियमानुसार, ब्लॉकमधील पहिला व्यवहार हा एक विशेष व्यवहार असतो जो ब्लॉकच्या निर्मात्याच्या मालकीचे नवीन नाणे सुरू करतो. हे नेटवर्कला समर्थन देण्यासाठी नोदेस्ना प्रवृत्त करते, आणि सुरुवातीला नाणी चलनात आण्याचा मार्ग प्रदान करेल, कारण ती जारी करण्यासाठी कोणतेही केंद्रीय अधिकार नाहीत. सतत नवीन नाण्यांची भर घालणे हे सोन्याच्या खाणकामगारांनी सोन्याचे परिसंचरण करण्यासाठी संसाधने खर्च करण्यासारखे आहे. आमच्या बाबतीत, CPU वेळ आणि वीज खर्च होते.

प्रोत्साहनासाठी व्यवहारावर शुल्क सुद्धा लावला जाऊ शकतो. जर व्यवहाराचे आउटपुट मूल्य त्याच्या इनपुट मूल्यापेक्षा कमी असेल, तर फरक हा व्यवहार शुल्क आहे जो व्यवहार असलेल्या ब्लॉकच्या प्रोत्साहन मूल्यामध्ये जोडला जातो. एकदा नाण्यांची पूर्वनिर्धारित संख्या चलनात आली की, प्रोत्साहन पूर्णपणे व्यवहार शुल्कात बदलू शकते आणि पूर्णपणे चलनवाढ मुक्त होऊ शकते.

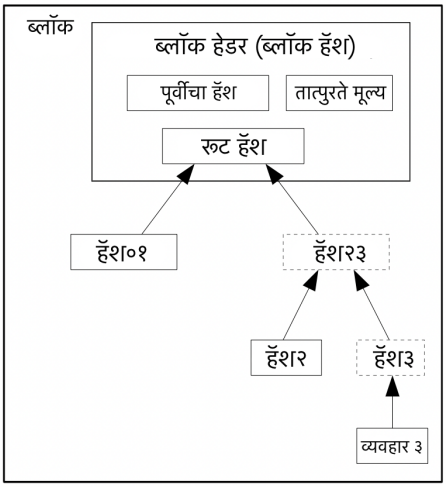
हे नोड्सना प्रामाणिक राहण्यासाठी प्रोत्साहित करू शकते. जर एखादा लोभी हल्लेखोर सर्व प्रामाणिक नोड्सपेक्षा जास्त CPU पॉवर एकत्र करू शकत असेल, तर त्याला त्याची देयके चोरून लोकांची फसवणूक करण्यासाठी किंवा नवीन नाणी तयार करण्यासाठी वापरणे यापैकी निवड करावी लागेल. त्याला नियमांनुसार खेळणे अधिक फायदेशीर वाटले पाहिजे, असे नियम जे व्यवस्थेची आणि त्याच्या स्वतःच्या संपत्तीची वैधता कमी करण्यापेक्षा त्याला इतर सर्वांच्या एकत्रिततेपेक्षा अधिक नवीन नाणी देतील.

७. डिस्क स्पेसची पुनर्प्राप्ती

एकदा नाण्यातील नवीनतम व्यवहार पुरेशा ब्लॉक्सखाली दबला गेला की, डिस्क स्पेस वाचवण्यासाठी खर्च केलेले व्यवहार टाकून दिले जाऊ शकतात. ब्लॉकचा हॅश न तोडता हे सुलभ करण्यासाठी, मर्कल ट्री [७][२][५] मध्ये व्यवहार हॅश केले जातात, ब्लॉकच्या हॅशमध्ये फक्त रूट समाविष्ट केला जातो. जुने ब्लॉक नंतर मर्कल ट्रीच्या ब्रँचेस काढून संक्षिप्त केले जाऊ शकतात. आतील हॅश संग्रहित करणे आवश्यक नाही.



मर्कल ट्री मध्ये हॅश केलेले व्यवहार

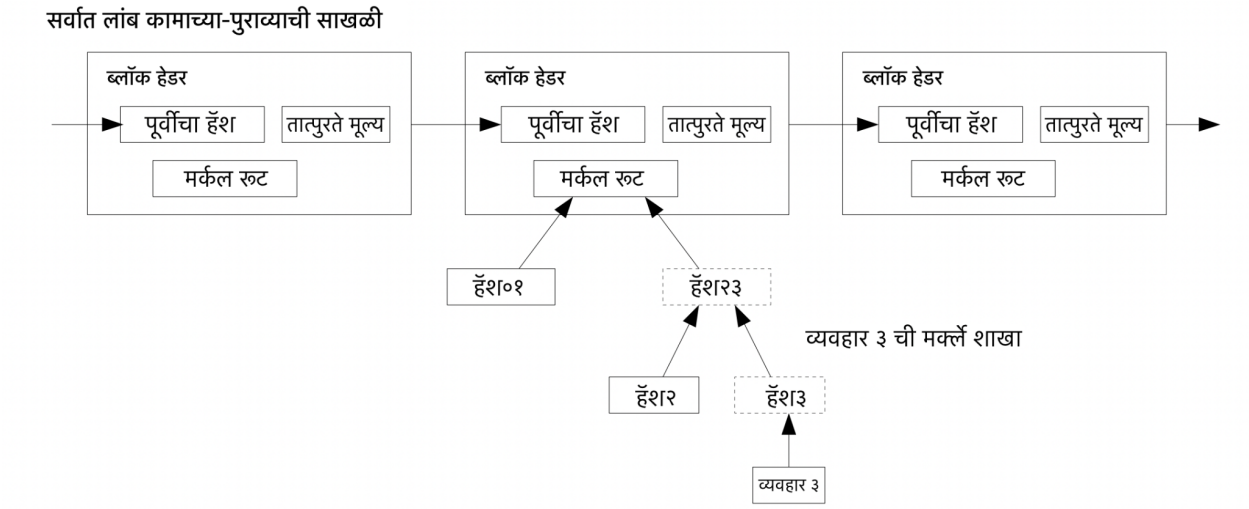


ब्लॉकमधून व्यवहार ० ते २ ची छाटणी केल्यावर

कोणतेही व्यवहार नसलेले ब्लॉक हेडर सुमारे 80 बाइट्सचे असेल. जर आपण समजू की दर १० मिनिटांनी ब्लॉक्स तयार होतात, तर ८० बाइट्स * ६ * २४ * ३६५ = प्रति वर्ष ४.२MB. २००८ मध्ये संगणक प्रणाली सामान्यत: २GB RAM सह विकली जातात आणि आणि मूरचा कायदा दर वर्षी १.२GB च्या वर्तमान वाढीचा अंदाज लावतो, जरी ब्लॉक हेडर मेमरीमध्ये ठेवले तरीही स्टोरेजमध्ये अडचण येऊ नये.

८. सरलीकृत पेमेंट पडताळणी

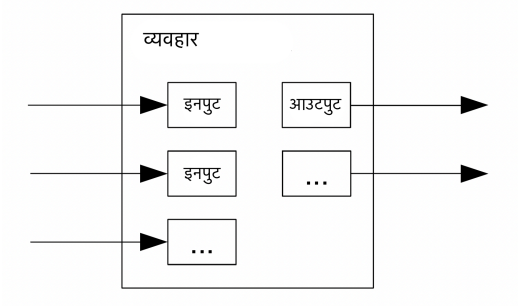
पूर्ण नेटवर्क नोड न चालवता देखील पेमेंट्सची पडताळणी करणे शक्य आहे. वापरकर्त्याला फक्त सर्वात लांब कामाच्या-पुराव्याच्या साखळीच्या ब्लॉक हेडरची एक प्रत ठेवणे आवश्यक आहे, जी त्याला त्याच्याकडे सर्वात लांब शृंखला असल्याची खात्री होईपर्यंत नेटवर्क नोड्सची चौकशी करून मिळेल, आणि ज्या ब्लॉक मध्ये तो टाइमस्टँप केलेले आहे त्या ब्लॉकशी त्या व्यवहाराला लिंक करणारी मर्कल ब्रांच मिळवणे आवश्यक आहे. तो स्वत:साठी व्यवहार तपासू शकत नाही, परंतु तो साखळीतील एका ठिकाणी लिंक करून, तो पाहू शकतो की नेटवर्क नोडने तो स्वीकारला आहे आणि त्या नंतर जोडलेले ब्लॉक्स नेटवर्कने तो स्वीकारला आहे याची आणखी पुष्टी देतात.



त्यामुळे, प्रामाणिक नोड्स नेटवर्कवर नियंत्रण ठेवत असेल तोपर्यंत सत्यापन विश्वसनीय असते, परंतु जर आक्रमणकर्त्याने नेटवर्कवर नियंत्रण ठेवले असेल तर ते अधिक असुरक्षित असते. नेटवर्क नोड्स स्वतःसाठी व्यवहार सत्यापित करू शकतात, परंतु हल्लेखोराने नेटवर्कवर मात करणे सुरू ठेवल्यास हल्लेखोराच्या बनावट व्यवहारांद्वारे सरलीकृत पद्धत फसवू शकते. यापासून संरक्षण करण्यासाठी एक धोरण म्हणजे नेटवर्क नोड्सना अवैध ब्लॉक आढळल्यावर त्यांच्याकडून सूचना स्वीकारून, वापरकर्त्याच्या सॉफ्टवेअरला पूर्ण ब्लॉक आणि अलर्ट केलेले व्यवहार डाउनलोड करून विसंगतीची पुष्टी करण्यास प्रवृत्त करावे. ज्या व्यवसायांना वारंवार देयके मिळतात ते कदाचित अधिक स्वतंत्र सुरक्षितता आणि जलद पडताळणीसाठी त्यांचे स्वतःचे नोड्स चालवू इच्छित असतील.

९. मूल्याचे संयोजन आणि विभाजन

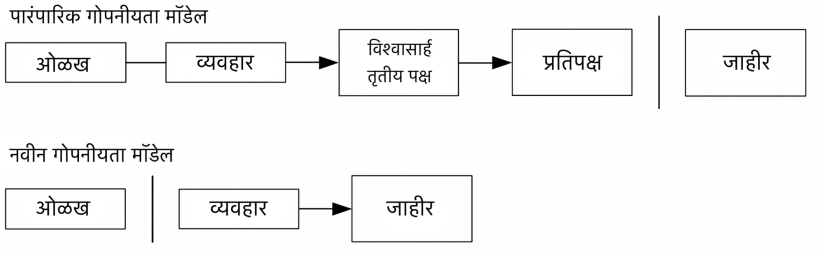
नाणी वैयक्तिकरित्या हाताळणे शक्य असले तरी, हस्तांतरणामध्ये प्रत्येक नाण्यासाठी स्वतंत्र व्यवहार करणे त्रासदायक होईल. मूल्य विभाजित आणि एकत्रित होण्यासाठी, व्यवहारांमध्ये एकाधिक इनपुट आणि आउटपुट असतात. सामान्यत: पूर्वीच्या मोठ्या व्यवहारातून एकतर एक इनपुट किंवा लहान रकमेचे एकत्रीकरण करणारे अनेक इनपुट्स आणि जास्तीत जास्त दोन ओउटपूटस असतील: एक पेमेंटसाठी, आणि जर उरलेली रकम असेल तर एक आउटपुट, प्रेषकाला परत करण्यासाठी.



हे लक्षात घेतले पाहिजे की फॅन-आउट, जेथे एक व्यवहार अनेक व्यवहारांवर अवलंबून असतो आणि ते व्यवहार अनेकांवर अवलंबून असतात, ती समस्या इथे नाही. एखाद्या व्यवहाराच्या इतिहासाची संपूर्ण स्वतंत्र प्रत काढण्याची गरज कधीच नसते.

१०. गोपनीयता

पारंपारिक बँकिंग मॉडेल गुंतलेल्या पक्ष आणि विश्वासार्ह तृतीय पक्षापर्यंत माहितीचा प्रवेश मर्यादित करून गोपनीयतेची पातळी गाठते. सर्व व्यवहार सार्वजनिकपणे जाहीर करण्याची गरज या पद्धतीला प्रतिबंधित करते, परंतु तरीही अन्य ठिकाणी माहितीचा प्रवाह खंडित करून गोपनीयता राखली जाऊ शकते: सार्वजनिक चावी निनावी ठेवून. लोक पाहू शकतात की कोणीतरी दुसऱ्याला रक्कम पाठवत आहे, परंतु कोणाशीही व्यवहार लिंक न करणाऱ्या माहितीशिवाय. हे स्टॉक एक्स्चेंजद्वारे जारी केलेल्या माहितीच्या पातळीसारखेच आहे, जेथे वैयक्तिक व्यापारांची वेळ आणि आकार, "टेप" सार्वजनिक केली जाते, परंतु पक्ष कोण होते हे न सांगता.



अतिरिक्त सुरक्षितते साठी, प्रत्येक व्यवहारासाठी एक नवीन चाव्यांची जोडी वापरण्यात यावी जेणेकरुन त्यांना एका सामाईक मालकाशी जोडले जाऊ नये. बहु-इनपुट व्यवहारांसह काही लिंकिंग अद्याप अपरिहार्य आहे, जे अनिवार्यपणे प्रकट करतात की त्यांचे इनपुट एकाच मालकाच्या मालकीचे होते. जोखीम अशी आहे की जर एखाद्या चावीचा मालक उघड झाला तर, लिंक केल्याने त्याच मालकाचे इतर व्यवहार उघड होऊ शकतात.

११. गणना

आम्ही प्रामाणिक साखळीपेक्षा अधिक वेगाने पर्यायी साखळी निर्माण करण्याचा प्रयत्न करणाऱ्या आक्रमणकर्त्याच्या परिस्थितीचा विचार करतो. जरी हे पूर्ण झाले असले तरी, ते प्रणालीला अनियंत्रित बदलांसाठी उघडत नाही, जसे की पातळ हवेतून मूल्य निर्माण करणे किंवा कधीही आक्रमणकर्त्याचे नसलेले पैसे घेणे. नोड्स अवैध व्यवहार पेमेंट म्हणून स्वीकारणार नाहीत, आणि प्रामाणिक नोड्स ते असलेला ब्लॉक कधीही स्वीकारणार नाहीत. हल्लेखोर स्वतःच्या व्यवहारांपैकी एकात बदल करून नुकतेच खर्च केलेले पैसे परत मिळवण्याचा प्रयत्न करू शकतो.

प्रामाणिक साखळी आणि हल्लेखोर साखळी यांच्यातील शर्यत द्विपदीय यादृच्छिक चाल म्हणून दर्शविली जाऊ शकते. यशाचा दर म्हणजे प्रामाणिक साखळी एका ब्लॉकने वाढवणे, त्याची आघाडी +१ ने वाढवणे आणि अपयशाचा दर म्हणजे आक्रमणकर्त्याची साखळी एका ब्लॉकने वाढवणे आणि अंतर -१ ने कमी करणे.

दिलेल्या तूटमधून आक्रमणकर्त्याला पकडण्याची संभाव्यता जुगाराच्या नाशाच्या समस्येसारखीच आहे. समजा अमर्यादित क्रेडिट असलेला जुगारी तोटयात सुरू होतो आणि तोट्यातून बाहेर निघण्याचा प्रयत्न करण्यासाठी संभाव्यपणे असंख्य चाल्या खेळतो. तो जुगारी कधी तोट्यातून बाहेर निघु शकतो का किंवा आक्रमणकर्ता कधी प्रामाणिक साखळीपर्यंत पोहोचू शकतो का, या संभाव्यतेची आम्ही खालीलप्रमाणे गणना करू शकतो [८]:

p = एक प्रामाणिक नोडला पुढील ब्लॉक सापडण्याची संभाव्यता
q = आक्रमणकर्त्याला पुढील ब्लॉक सापडण्याची शक्यता
q_z = z ब्लॉक मागे असलेल्या आक्रमणकर्ता बरोबर पोहोचण्याची संभाव्यता

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

$p > q$ हे आमचे गृहितक लक्षात घेता, आक्रमणकर्त्याला जितक्या ब्लॉक्स मागून पाठला चालू करतो तितकी प्रामाणिक साखळीपर्यंत पोहोचण्याची संभाव्यता झपाट्याने कमी होते. त्याच्या विरुद्ध चालत असलेल्या शक्यतांमुळे, जर त्याने सुरवातीलाच पटकन उडी घेतली नाही, तर त्याची शक्यता नगण्य होईल कारण तो आणखी मागे पडेल.

आम्ही आता प्रेषक व्यवहार बदलू शकत नाही याची पुरेशी खात्री होण्यापूर्वी नवीन व्यवहाराच्या प्राप्तकर्त्याला किती काळ प्रतीक्षा करावी लागेल याचा विचार करतो. आम्ही असे गृहीत धरतो की प्रेषक हा एक आक्रमणकर्ता आहे जो प्राप्तकर्त्याला काही काळ असा विश्वास द्यायला बघतोय की त्याने त्याला पैसे दिले आहेत, नंतर काही वेळ निघून गेल्यावर ते पैसे तो स्वतःलाच परत घेणार. असे झाल्यावर प्राप्तकर्त्याला सतर्क केले जाईल, परंतु प्रेषकाला आशा आहे की तोपर्यंत खूप उशीर झाला असेल.

प्राप्तकर्ता एक नवीन चाव्यांची जोडी व्युत्पन्न करतो आणि स्वाक्षरी करण्याच्या थोड्या वेळा आधी प्रेषकाला सार्वजनिक चावी देतो. हे देयकाला जोपर्यंत तो भाग्यवान होत नाही तोपर्यंत त्यावर सतत काम करून, वेळेपूर्वी ब्लॉक्सची एक साखळी तयार करून ठेऊन आणि मग त्याच वेळी व्यवहार पूर्ण करण्यापासून प्रतिबंधित करते. एकदा व्यवहार पाठवल्यानंतर, अप्रामाणिक प्रेषक त्याच्या व्यवहाराची पर्यायी आवृत्ती असलेल्या समांतर साखळीवर गुप्तपणे कार्य करण्यास सुरवात करतो.

प्राप्तकर्ता व्यवहार ब्लॉकमध्ये जोडले जाईपर्यंत आणि z ब्लॉक्स नंतर जोडले जाईपर्यंत प्रतीक्षा करतो. हल्लेखोराने किती प्रगती केली हे त्याला माहीत नाही, परंतु प्रामाणिक ब्लॉक्सने प्रति ब्लॉक सरासरी अपेक्षित वेळ घेतला असे गृहीत धरल्यास, आक्रमणकर्त्याची संभाव्य प्रगती अपेक्षित मूल्यासह पॉसॉन वितरण असेल:

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

हल्लेखोर अद्याप पकडू शकेल अशी संभाव्यता मिळविण्यासाठी, आम्ही पॉसॉन घनतेला त्याने केलेल्या प्रगतीच्या प्रत्येक रकमेला तो त्या बिंदूपासून पकडू शकलेल्या संभाव्यतेने गुणाकार करतो:

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

वितरणाच्या अनंत शेपटीची बेरीज टाळण्यासाठी पुनर्रचना करत आहे...

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

C कोडमध्ये रूपांतरित करत आहे...

```
#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
}
```

काही परिणाम चालवताना, आम्ही संभाव्यता z सह झपाट्याने कमी होणे पाहू शकतो.

```
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
```

०.१% पेक्षा कमी P साठी सोडवत आहे...

```
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
```

१२. निष्कर्ष

आम्ही विश्वासावर अवलंबून न राहता इलेक्ट्रॉनिक व्यवहारांसाठी एक प्रणाली प्रस्तावित केली आहे. आम्ही डिजिटल स्वाक्षरींपासून बनवलेल्या नाण्यांच्या नेहमीच्या चौकटीपासून सुरुवात केली, जी मालकीचं मजबूत नियंत्रण पुरवते, पण दुहेरी खर्च रोखण्याच्या मार्गाशिवाय ते अपूर्ण आहे. याचे निराकरण करण्यासाठी, आम्ही व्यवहारांच्या सार्वजनिक इतिहासची नोंद करण्यासाठी कामाचा-पुरावा वापरून परस्परांचा नेटवर्क प्रस्तावित केला आहे जो जोवर प्रामाणिक नोड्स बहुमत CPU पॉवर नियंत्रित करतात तोवर आक्रमणकर्त्याला व्यवहार बदलणे त्वरित संगणकीयदृष्ट्या अव्यवहार्य करतो. नेटवर्क त्याच्या असंरचित साधेपणामध्ये मजबूत आहे. नोड्स थोड्या समन्वयाने एकाच वेळी कार्य करतात. त्यांना ओळखण्याची गरज नाही, कारण संदेश कोणत्याही विशिष्ट ठिकाणी पाठवले जात नाहीत आणि केवळ सर्वोत्तम प्रयत्नांच्या आधारावर वितरित करणे आवश्यक आहे. नोड्स इच्छेनुसार नेटवर्क सोडू शकतात आणि पुन्हा सामील होऊ शकतात, ते गेले असताना काय घडले याचा पुरावा म्हणून कामाच्या-पुराव्याची साखळी स्वीकारतात. ते त्यांच्या CPUच्या सामर्थ्याने मतदान करतात, वैध ब्लॉक्सचा विस्तार करण्याचे काम करून त्यांची स्वीकृती व्यक्त करतात आणि अवैध ब्लॉक्सवर काम करण्यास नकार देऊन नाकारतात. या सर्वसंमतीच्या यंत्रणेद्वारे कोणतेही आवश्यक नियम आणि प्रोत्साहन लागू केले जाऊ शकतात.

संदर्भ

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

[२] 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.

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

[४] 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.

[५] 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.

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

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

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