બીટકોઇનઃ પીઅર ટુ પીઅર ઇલેક્ટ્રોનિક કેશ સિસ્ટમ

સતોશી નાકામોતો

satoshin@gmx.com

www.bitcoin.org

સારઃ ઈલેક્ટ્રોનિક રોકડનું સંપૂર્ણ પીઅર-ટુ-પીઅર વર્ઝન જેનાથી નાણાંકીય સંસ્થાનમાંથી પસાર થયા વિના એક પક્ષકારથી બીજા પક્ષકારને સીધું ઓનલાઇન પેમેન્ટ થઈ શકશે. ડિજિટલ સિગ્નેચર્સ સોલ્યુશનનો એક ભાગ છે, પરંતુ જો વિશ્વસનીય થર્ડ પાર્ટીને ડબલ ખર્ચ અટકાવવા માટેની જરૂર પડે તો મુખ્ય લાભો નહીં મળે. અમે પીઅર-ટુ-પીઅર નેટવર્કનો ઉપયોગ કરીને ડબલ ખર્ચની સમસ્યા ઉકેલવાની રજૂઆત કરીએ છીએ. નેટવર્ક ટ્રાન્ઝેક્શનને હેશ-આધારિત પ્રૂફ-ઓફ-વર્કની ચાલુ ચેઇનમાં હેશ કરીને તેના પર ટાઇમસ્ટેમ્પ લગાવે છે, જે એક એવો રેકોર્ડ બનાવે છે જે પ્રૂફ-ઓફ-વર્કને ફરીથી કર્યા વિના બદલી શકાતો નથી. સૌથી લાંબી ચેઇન માત્ર ઘટનાઓના ક્રમના પુરાવા તરીકે જ કામ કરતી નથી, પરંતુ સાબિતી આપે છે કે તે સીપીયુ પાવરના સૌથી મોટા પૂલમાંથી આવી છે. જ્યાં સુધી મોટાભાગના સીપીયુ પાવર નોડ્સ દ્વારા નિયંત્રિત થાય છે જે નેટવર્ક પર હુમલો કરવા માટે કોઓપરેટ કરતા નથી, તેઓ સૌથી લાંબી ચેઇન જનરેટ કરશે અને હુમલાખોરોને માત કરશે. નેટવર્કને પોતે ન્યૂનતમ માળખું હોવું જરૂરી છે. મેસેજીસ શ્રેષ્ઠ પ્રયાસના આધારે પ્રસારિત કરવામાં આવે છે અને નોડ્સ તેની ઇચ્છા મુજબ નેટવર્ક છોડીને ફરીથી જોડાઈ શકે છે, જે તેઓ ગયા ત્યારે શું થયું તેના પુરાવા તરીકે સૌથી લાંબી પ્રૂફ-ઓફ-વર્ક ચેઇનને સ્વીકારે છે.

1. પ્રસ્તાવના

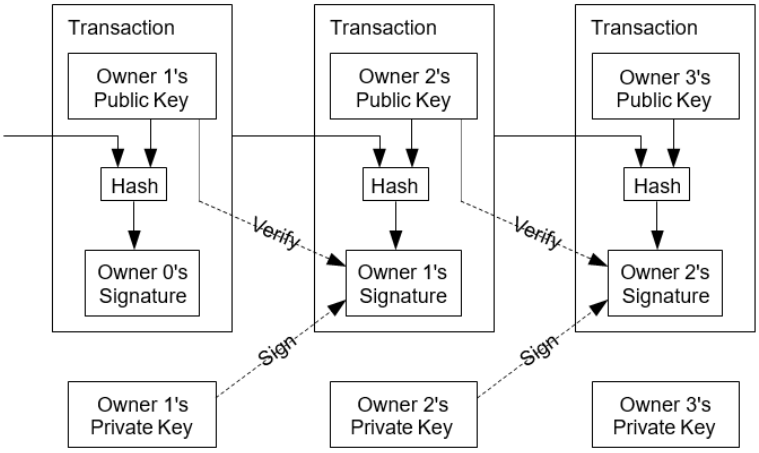
ઇન્ટરનેટ પર વ્યાપારી કામગીરીઓ મોટાભાગે માત્ર નાણાંકીય સંસ્થાનો પર જ નભે છે જે ઇલેક્ટ્રોનિક પેમેન્ટની પ્રોસેસ કરવા માટે વિશ્વસનીય થર્ડ પાર્ટી તરીકે કામ કરે છે. સિસ્ટમ મોટાભાગના વ્યવહારો માટે સારી રીતે કામ કરે છે, પરંતુ તે હજુ પણ ટ્રસ્ટ આધારિત મોડેલની અંતર્ગત નબળાઈઓથી પીડાય છે. સંપૂર્ણપણે નોન-રિવર્સેબલ શકાય તેવા વ્યવહારો ખરેખર શક્ય નથી, કારણ કે નાણાંકીય સંસ્થાઓ મધ્યસ્થી વિવાદોને ટાળી શકતી નથી. મધ્યસ્થીનો ખર્ચ ટ્રાન્ઝેક્શન ખર્ચમાં વધારો કરે છે, મિનિમમ પ્રેક્ટિકલ ટ્રાન્ઝેક્શન સાઇઝને મર્યાદિત કરે

છે અને નાના કેઝ્યુઅલ વ્યવહારોની શક્યતાને કાપી નાખે છે અને નોન-રિવર્સેબલ સર્વિસીઝ માટે નોન-રિવર્સેબલ પેમેન્ટ્સ કરવાની ક્ષમતા ગુમાવવાથી મોટાપાયે ખર્ચ થાય છે. રિવર્સલની શક્યતા સાથે, ટ્રસ્ટની જરૂરિયાત વધે છે. મર્ચન્ટ્સે તેમના ગ્રાહકોથી સાવચેત રહેવું જોઈએ, તેઓને અન્યથા જરૂર હોય તેના કરતાં વધુ માહિતી માટે તેમની પાછળ પડવું જોઈએ. છેતરપિંડીની ચોક્કસ ટકાવારી અનિવાર્ય તરીકે સ્વીકારવામાં આવે છે. ફિઝિકલ કરન્સીનો ઉપયોગ કરીને આ ખર્ચ અને પેમેન્ટની અનિશ્ચિતતાઓ વ્યક્તિગત રીતે ટાળી શકાય છે, પરંતુ વિશ્વસનીય પક્ષ વિના કમ્યૂનિકેશન ચેનલ પર પેમેન્ટ્સ કરવા માટે કોઈ પદ્ધતિ અસ્તિત્વમાં નથી.

વિશ્વાસને બદલે ક્રિપ્ટોગ્રાફિક પ્રૂફ પર આધારિત ઇલેક્ટ્રોનિક પેમેન્ટ સિસ્ટમની જરૂર છે, જે કોઈપણ બે ઇચ્છુક પક્ષકારોને વિશ્વાસપાત્ર થર્ડ પાર્ટીની જરૂરિયાત વિના એકબીજા સાથે સીધો વ્યવહાર કરવાની મંજૂરી આપે છે. રિવર્સ કરવા માટે કમ્પ્યૂટેશનલી પ્રેક્ટિકલ ન હોય તેવા ટ્રાન્ઝેક્શન્સ સેલર્સને છેતરપિંડીથી બચાવશે અને ખરીદદારોને સુરક્ષિત રાખવા માટે રૂટિન એસ્ક્રો મિકેનિઝમ સરળતાથી લાગુ કરી શકાય છે. આ પેપરમાં અમે વ્યવહારોના ક્રોનોલોજીકલ ક્રમના કમ્પ્યૂટેશનલ પુરાવા જનરેટ કરવા માટે પીઅર-ટુ-પીઅર ડિસ્ટ્રીબ્યુટેડ ટાઇમસ્ટેમ્પ સર્વરનો ઉપયોગ કરીને ડબલ ખર્ચની સમસ્યાના સોલ્યુશનની રજૂઆત કરીએ છીએ. જ્યાં સુધી પ્રમાણિક નોડ્સ સામૂહિક રીતે હુમલાખોર નોડ્સના કોઈપણ સહકારી જૂથ કરતાં વધુ સીપીયુ પાવરને નિયંત્રિત કરે ત્યાં સુધી સિસ્ટમ સુરક્ષિત છે.

2. વ્યવહારોઃ

અમે ઇલેક્ટ્રોનિક કોઇનને ડિજિટલ સિગ્નેચરની ચેઇન તરીકે ઓળખીએ છીએ. દરેક માલિક અગાઉના ટ્રાન્ઝેક્શનના હેશ અને પછીના માલિકની પબ્લિક કી પર ડિજિટલ સિગ્નેચર કરીને અને કોઇનના અંતમાં આને ઉમેરીને કોઇનને બીજામાં સ્થાનાંતરિત કરે છે. માલિકીની ચેઇનને ચકાસવા માટે નાણાં લેનાર સિગ્નેચર ચકાસી શકે છે.

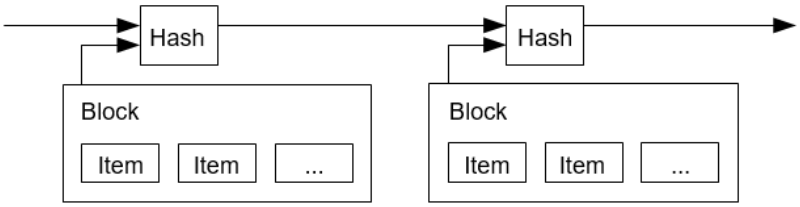


અલબત્ત સમસ્યા એ છે કે ચૂકવનાર એ ચકાસી શકતો નથી કે માલિકોમાંથી એકે સિક્કાનો ડબલ-સ્પેન્ડ કર્યો નથી. એક સામાન્ય સોલ્યુશન એ છે કે વિશ્વાસપાત્ર કેન્દ્રીય સત્તામંડળ અથવા ટંકશાળ (મિન્ટ) રજૂ કરવી, જે દરેક વ્યવહારને ડબલ ખર્ચ માટે તપાસે છે. દરેક ટ્રાન્ઝેક્શન પછી, નવો સિક્કો જારી કરવા માટે સિક્કો ટંકશાળમાં પાછો ફરવો આવશ્યક છે અને માત્ર ટંકશાળમાંથી સીધા જ જારી કરાયેલા સિક્કાઓ ડબલ-ખર્ચ ન થાય તે માટે વિશ્વાસ કરવામાં આવે છે. આ સોલ્યુશનની સમસ્યા એ છે કે સમગ્ર મની સિસ્ટમનું ભાવિ ટંકશાળ ચલાવતી કંપની પર આધારિત છે અને દરેક વ્યવહારો બેંકની જેમ જ તેમાંથી પસાર થવાના હોય છે.

ચૂકવણી કરનારને જાણવું જરૂરી છે કે અગાઉના માલિકોએ અગાઉના કોઈપણ વ્યવહારો પર સિગ્નેચર કર્યા નથી. અમારા હેતુઓ માટે, સૌથી પહેલો થયેલો વ્યવહાર ગણવામાં આવે છે, તેથી અમે પછીથી બમણો ખર્ચ કરવાના પ્રયત્નોની પરવા કરતા નથી. વ્યવહારની ગેરહાજરીની પુષ્ટિ કરવાનો એકમાત્ર રસ્તો એ છે કે તમામ વ્યવહારો વિશે જાગૃત રહેવું. ટંકશાળ આધારિત મોડેલમાં, ટંકશાળ તમામ વ્યવહારોથી વાકેફ હતી અને નક્કી કરે છે કે કયો વ્યવહાર પ્રથમ આવ્યો છે. વિશ્વાસુ પક્ષ વિના આને પરિપૂર્ણ કરવા માટે, વ્યવહારોએ જાહેરમાં એવી જાહેરાત કરવી જરૂરી છે કે [1] અમને એક સિસ્ટમની જરૂર છે કે સહભાગીઓ જે ક્રમમાં તેઓ પ્રાપ્ત થયા હતા તેના એક ઇતિહાસ પર સંમત થાય. ચૂકવણી કરનારને પુરાવાની જરૂર છે કે દરેક ટ્રાન્ઝેક્શન સમયે, મોટાભાગના નોડ્સ સંમત થયા હતા કે તે પ્રથમ પ્રાપ્ત થયું હતું.

3. ટાઇમસ્ટેમ્પ સર્વર

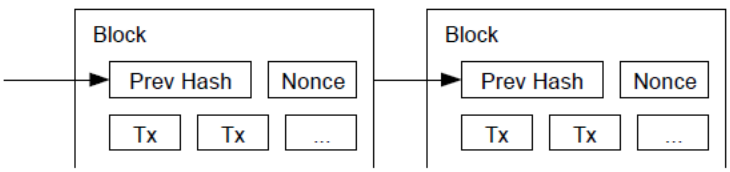
અમે જે સોલ્યુશન રજૂ કરીએ છીએ તે ટાઇમસ્ટેમ્પ સર્વરથી શરૂ થાય છે. ટાઇમસ્ટેમ્પ સર્વર ટાઇમસ્ટેમ્પ કરવા માટે આઈટમ્સના બ્લોકની હેશ લઈને અને હેશને વ્યાપકપણે પબ્લિશ કરીને કામ કરે છે, જેમ કે અખબારમાં અથવા યુઝનેટ પોસ્ટ [2-5]. ટાઇમસ્ટેમ્પ સાબિત કરે છે કે સ્વાભાવિક રીતે હેશમાં જવા માટે તે સમયે ડેટા અસ્તિત્વમાં હોવો જોઈએ. દરેક ટાઈમસ્ટેમ્પમાં તેના હેશમાં અગાઉનો ટાઈમસ્ટેમ્પ સમાવિષ્ટ હોય છે, જે એક સાંકળ બનાવે છે. દરેક વધારાનો ટાઈમસ્ટેમ્પ તેની પહેલાના ટાઇમસ્ટેમ્પને મજબૂત બનાવે છે.



4. પ્રૂફ-ઓફ-વર્ક

પીઅર-ટુ-પીઅર આધારે ડિસ્ટ્રીબ્યુટેડ ટાઇમસ્ટેમ્પ સર્વરને લાગુ કરવા માટે, આપણે અખબાર અથવા યુઝનેટ પોસ્ટને બદલે એડમ બેકના હેશકેશ (6) જેવી પ્રૂફ-ઓફ-વર્ક સિસ્ટમનો ઉપયોગ કરવાની જરૂર પડશે. પ્રૂફ-ઓફ-વર્કમાં એવી વેલ્યુ માટે સ્કેનિંગનો સમાવેશ થાય છે જેને હેશ કરવામાં આવે, જેમ કે SHA-256 સાથે, ત્યારે હેશ અનેક ઝીરો બિટ્સથી શરૂ થાય છે. સરેરાશ કાર્ય જરૂરી ઝીરો બિટ્સની સંખ્યામાં ઘાતાંકીય છે અને સિંગલ હેશ ચલાવીને વેરિફાઇ કરી શકાય છે.

અમારા ટાઈમસ્ટેમ્પ નેટવર્ક માટે, અમે બ્લોકના હેશને જરૂરી ઝીરો બિટ્સ આપે છે તે વેલ્યુ ન મળે ત્યાં સુધી બ્લોકમાં નોન્સ વધારીને પ્રૂફ-ઓફ-વર્કનો અમલ કરીએ છીએ. એકવાર સીપીયુ પ્રયત્નો તેને પ્રૂફ-ઓફ-વર્કને સંતોષવા માટે વિસ્તારવામાં આવે, પછી કામને ફરીથી કર્યા વિના બ્લોક બદલી શકાતો નથી. પછીના બ્લોક્સ તેના પછી ચેઇન કરવામાં આવે છે ત્યારે બ્લોકને બદલવાના કામમાં તેના પછીના તમામ બ્લોક્સને ફરીથી કરવાનો સમાવેશ થાય છે.



પ્રૂફ-ઓફ-વર્ક બહુમતીના નિર્ણય લેવામાં પ્રતિનિધિત્વ નક્કી કરવાની સમસ્યાને પણ હલ કરે છે. જો બહુમતી એક-આઈપી-એડ્રેસ-એક-વોટ પર આધારિત હોય, તો ઘણા બધા આઈપી ફાળવવામાં સક્ષમ હોય તેવા કોઈના પણ દ્વારા તેને બદલી શકાય છે. પ્રૂફ-ઓફ-વર્ક આવશ્યકપણે એક-સીપીયુ-એક-વોટ છે. બહુમતી નિર્ણયને સૌથી લાંબી ચેઇન દ્વારા રજૂ કરવામાં આવે છે, જેમાં તેમાં સૌથી વધુ પ્રૂફ-ઓફ-વર્ક પ્રયાસો કરવામાં આવેલા છે. જો મોટાભાગના સીપીયુ પાવરને પ્રમાણિક નોડ્સ દ્વારા નિયંત્રિત કરવામાં આવે તો પ્રમાણિક ચેઇન સૌથી ઝડપી વૃદ્ધિ કરશે અને કોઈપણ સ્પર્ધા કરતી ચેઇન્સને પાછળ છોડી દેશે. ભૂતકાળના બ્લોકને મોડિફાઇ કરવા માટે, હુમલાખોરે બ્લોક અને તેના પછીના તમામ બ્લોકના પ્રૂફ-ઓફ-વર્કને ફરીથી કરવું પડશે અને પછી પ્રામાણિક નોડ્સના કામની સાથે આગળ વધવું પડશે. અમે પછીથી બતાવીશું કે જેવા પછીના બ્લોક્સ ઉમેરવામાં આવે ત્યારે ધીમા હુમલાખોરને પકડવાની સંભાવના ઝડપથી ઘટી જાય છે.

વધતી હાર્ડવેરની ઝડપ અને સમયની સાથે નોડ્સ ચલાવવામાં વિવિધ રસની ભરપાઈ કરવા માટે, દર કલાકે બ્લોક્સની સરેરાશ સંખ્યાને લક્ષ્યાંકિત કરતી મૂવિંગ એવરેજ દ્વારા પ્રૂફ-ઓફ-વર્કની મુશ્કેલી નક્કી કરવામાં આવે છે. જો તેઓ ખૂબ ઝડપથી જનરેટ થાય છે, તો મુશ્કેલી વધે છે.

5. નેટવર્ક

નેટવર્ક ચલાવવા માટેના પગલાં નીચે મુજબ છે:

1) નવા વ્યવહારો તમામ નોડ્સ પર બ્રોડકાસ્ટ થાય છે.

2) દરેક નોડ બ્લોકમાં નવા વ્યવહારો એકત્રિત કરે છે.

3) દરેક નોડ તેના બ્લોક માટે મુશ્કેલ પ્રૂફ-ઓફ-વર્ક શોધવાનું કામ કરે છે.

4) જ્યારે નોડને પ્રૂફ-ઓફ-વર્ક મળે ત્યારે તે બ્લોકને તમામ નોડ્સ પર બ્રોડકાસ્ટ કરે છે.

5) નોડ્સ બ્લોકને ત્યારે જ સ્વીકારે છે જો તેમાં થયેલા તમામ વ્યવહારો માન્ય હોય અને પહેલાથી જ ખર્ચવામાં ન આવ્યા હોય.

6) નોડ્સ અગાઉના હેશ તરીકે સ્વીકારાયેલા બ્લોકના હેશનો ઉપયોગ કરીને ચેઇનમાં આગામી બ્લોક બનાવવા પર કામ કરીને બ્લોકની તેમની સ્વીકૃતિ વ્યક્ત કરે છે.

નોડ્સ હંમેશા સૌથી લાંબી ચેઇનને સાચી માને છે અને તેને લંબાવવાનું કામ ચાલુ રાખશે. જો

બે નોડ્સ એકસાથે આગામી બ્લોકના વિવિધ વર્ઝનને બ્રોડકાસ્ટ કરે તો કેટલાક નોડ્સ એક અથવા બીજા પ્રથમ પ્રાપ્ત કરી શકે છે. આવા કિસ્સામાં, તેઓ તેમને મળેલી પ્રથમ બ્રાન્ચ પર કામ કરે છે, પરંતુ જો તે લાંબી થઈ જાય તો બીજી બ્રાન્ચને સેવ કરશે. જ્યારે આગળનું પ્રૂફ-ઓફ-વર્ક મળી જાય અને એક બ્રાન્ચ લાંબી થાય ત્યારે ટાઈ તૂટી જશે. નોડ્સ કે જે બીજી બ્રાન્ચ પર કામ કરતા હતા તે પછી લાંબી તરફ વળશે.

નવા ટ્રાન્ઝેક્શન બ્રોડકાસ્ટે તમામ નોડ્સ સુધી પહોંચવું જરૂરી નથી. જ્યાં સુધી તે ઘણા નોડ્સ સુધી પહોંચે ત્યાં સુધી તેઓ વહેલીતકે પહેલા બ્લોકમાં પ્રવેશી જશે. બ્લોક બ્રોડકાસ્ટ ડ્રોપ થયેલા મેસેજીસ પ્રત્યે પણ સહનશીલ હોય છે. જો નોડને બ્લોક પ્રાપ્ત ન થાય, તો તે આગલો બ્લોક મેળવે ત્યારે તેના માટે વિનંતી કરશે અને તેને ખબર પડશે કે તે એક બ્લોક ચૂકી ગયો છે.

6. પ્રોત્સાહન

પરંપરા મુજબ બ્લોકમાં પ્રથમ વ્યવહાર એ એક વિશિષ્ટ વ્યવહાર છે જે બ્લોક બનાવનારની માલિકીનો નવો કોઇન શરૂ કરે છે. આ નેટવર્કને ટેકો આપવા માટે નોડ્સ માટે પ્રોત્સાહન ઉમેરે છે અને શરૂઆતમાં કોઇન્સને સર્ક્યુલેશનમાં ડિસ્ટ્રીબ્યુટ કરવાનો માર્ગ પૂરો પાડે છે, કારણ કે તેમને ઇશ્યૂ કરવા માટે કોઈ કેન્દ્રીય સત્તા નથી. નવા કોઇન્સના સતત જથ્થામાં થતો સ્થિર ગતિએ ઉમેરો એ સોનાની ખાણિયોને સોનાને સર્ક્યુલેશનમાં ઉમેરવા માટે સંસાધનો ખર્ચવા બરાબર છે. અમારા કિસ્સામાં, તે સીપીયુ સમય અને વીજળીનો ખર્ચ કરવામાં આવે છે.

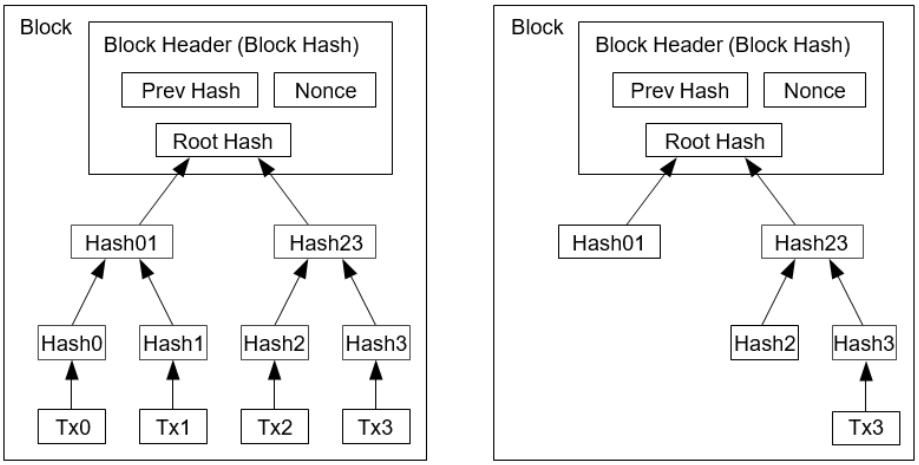
પ્રોત્સાહનને ટ્રાન્ઝેક્શન ફી સાથે પણ ફંડ કરી શકાય છે. જો ટ્રાન્ઝેક્શનની આઉટપુટ વેલ્યુ તેના ઇનપુટ મૂલ્ય કરતાં ઓછી હોય, તો તફાવત એ ટ્રાન્ઝેક્શન ફી છે જે ટ્રાન્ઝેક્શન ધરાવતા બ્લોકની ઇન્સેન્ટિવ વેલ્યુમાં ઉમેરવામાં આવે છે. એકવાર કોઇન્સની પૂર્વનિર્ધારિત સંખ્યા સર્ક્યુલેશનમાં આવે પછી, પ્રોત્સાહન સંપૂર્ણપણે ટ્રાન્ઝેક્શન ફીમાં સંક્રમિત થઈ શકે છે અને સંપૂર્ણપણે ફુગાવા મુક્ત થઈ શકે છે.

પ્રોત્સાહન નોડ્સને પ્રમાણિક રહેવા માટે પ્રોત્સાહિત કરવામાં મદદ કરી શકે છે. જો કોઈ લોભી હુમલાખોર તમામ પ્રામાણિક નોડ્સ કરતાં વધુ સીપીયુ પાવર એસેમ્બલ કરવામાં સક્ષમ હોય, તો તેણે તેનો ઉપયોગ કરીને તેના પેમેન્ટની ચોરી કરીને લોકોને છેતરવા અથવા નવા કોઇન

બનાવવા માટે તેનો ઉપયોગ કરવા વચ્ચે ગમે તે એકની પસંદગી કરવી પડશે. તેણે નિયમો પ્રમાણે રમવાનું વધુ નફાકારક સમજવું જોઈએ, એવા નિયમો કે જે તેને બીજા બધા કરતા વધુ કોઇન પૂરા પાડે છે અને સિસ્ટમ તથા તેની પોતાની સંપત્તિની માન્યતાને નબળી પાડતા નથી.

7. ડિસ્ક સ્પેસ પાછી મેળવવી

એકવાર કોઇનમાં લેટેસ્ટ ટ્રાન્ઝેક્શનને પૂરતા બ્લોક્સ હેઠળ મૂકી દેવામાં આવે તે પછી, ડિસ્ક સ્પેસ બચાવવા માટે તેના પહેલાં ખર્ચવામાં આવેલા વ્યવહારોને છોડી શકાય છે. બ્લોકના હેશને તોડ્યા વિના તેને સરળ બનાવવા માટે, મર્કલ ટ્રી [7][2][5]માં વ્યવહારોને હેશ કરવામાં આવે છે, જેમાં બ્લોકની હેશમાં માત્ર રૂટનો સમાવેશ થાય છે. જૂના બ્લોક્સને પછી ઝાડની ડાળીઓને કાપીને નાના કરી શકાય છે. આંતરિક હેશને સ્ટોર કરવાની જરૂર નથી.



Transactions Hashed in a Merkle Tree

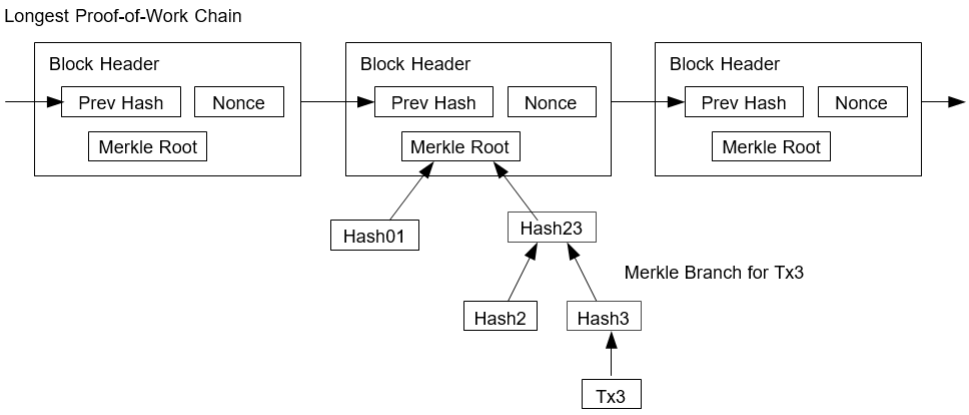
After Pruning Tx0-2 from the Block

કોઈ વ્યવહારો વિનાનું બ્લોક હેડર લગભગ 80 બાઇટ્સનું હશે. જો આપણે ધારીએ કે દર 10 મિનિટે બ્લોક્સ જનરેટ થાય છે, તો દર વર્ષે 80 બાઇટ્સ * 6 * 24 * 365 = 4.2 એમબી થશે. સામાન્ય રીતે 2008 સુધીમાં 2 જીબી રેમ સાથેની કમ્પ્યૂટર સિસ્ટમ્સ વેચાય છે અને મૂર્સ લૉ દર વર્ષે 1.2 જીબીની વર્તમાન વૃદ્ધિની આગાહી કરે છે ત્યારે જો બ્લોક હેડર્સને મેમરીમાં રાખવાની જરૂર હોય તો પણ સ્ટોરેજમાં કોઈ સમસ્યા ન હોવી જોઈએ.

8. સરળ પેમેન્ટ વેરિફિકેશન

સંપૂર્ણ નેટવર્ક નોડ ચલાવ્યા વિના પેમેન્ટ્સને વેરિફાઇ કરવા શક્ય છે. યુઝરને માત્ર સૌથી

લાંબી પ્રૂફ-ઓફ-વર્ક ચેઇનના બ્લોક હેડર્સની એક કોપી રાખવાની જ જરૂર છે, જે તે નેટવર્ક નોડ્સની ક્વેરી કરીને મેળવી શકે છે જ્યાં સુધી તેને ખાતરી ન થાય કે તેની પાસે સૌથી લાંબી ચેઇન છે અને તેણે ટાઇમસ્ટેમ્પ કરેલો હોય તે બ્લોક સાથે ટ્રાન્ઝેક્શનને લિંક કરીને મર્કલ બ્રાન્ચ મેળવી શકે છે. તે પોતે વ્યવહારને જાતે ચેક કરી શકતો નથી પરંતુ ચેઇનમાં એક જગ્યાએ તેને લિંક કરીને તે જોઈ શકે છે કે નેટવર્ક નોડે તેનો સ્વીકાર કર્યો છે અને તેના પછી ઉમેરાયેલા બ્લોક્સ વધુમાં પુષ્ટિ કરે છે કે નેટવર્કે તેને સ્વીકારી લીધા છે.

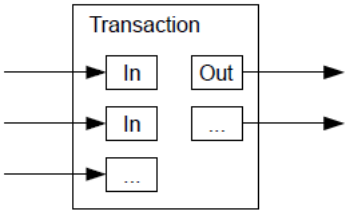


જ્યાં સુધી પ્રમાણિક નોડ્સ નેટવર્કને નિયંત્રિત કરે ત્યાં સુધી વેરિફિકેશન વિશ્વસનીય બની રહે છે, પરંતુ જો કોઈ હુમલાખોર નેટવર્કને ખોરવી નાખે તો તે વધુ અસુરક્ષિત બની જાય છે. નેટવર્ક નોડ્સ પોતાને માટે વ્યવહારો ચકાસી શકે છે, પરંતુ આ સરળ મેથડને હુમલાખોરના બનાવટી વ્યવહારો દ્વારા ત્યાં સુધી મૂર્ખ બનાવી શકાય છે જ્યાં સુધી હુમલાખોર નેટવર્ક પર કબ્જો જમાવવાનું ચાલુ રાખી શકે છે. આની સામે રક્ષણ મેળવવા માટેની એક વ્યૂહરચના એવી હોઈ શકે કે જ્યારે તેઓ અમાન્ય બ્લોક શોધે ત્યારે નેટવર્ક નોડ્સ તરફથી ચેતવણીઓ સ્વીકારવામાં આવે અને યુઝરના સોફ્ટવેરને સંપૂર્ણ બ્લોક અને ચેતવણી અપાયેલા વ્યવહારો ડાઉનલોડ કરવા માટે પ્રેરિત કરવામાં આવે જેથી અસંગતતાની પુષ્ટિ થઈ શકે. વારંવાર પેમેન્ટ્સ મેળવતા વ્યવસાયો કદાચ હજુ વધુ સ્વતંત્ર સુરક્ષા અને ઝડપી ચકાસણી માટે તેમના પોતાના નોડ્સ ચલાવવા માંગશે.

9. વેલ્યુને ભેગી કરવી અને છૂટી પાડવી

કોઇનને વ્યક્તિગત રીતે હેન્ડલ કરવા શક્ય હોવા છતાં, ટ્રાન્સફર થતા દરેક સેન્ટ માટે

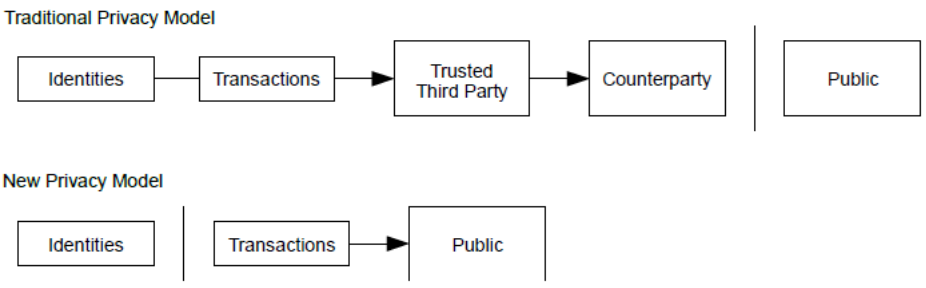
અલગ વ્યવહાર કરવો અનિચ્છનીય હશે. વેલ્યુને વિભાજિત અને સંયુક્ત કરવાની મંજૂરી આપવા માટે, વ્યવહારોમાં બહુવિધ ઇનપુટ્સ અને આઉટપુટ હોય છે. સામાન્ય રીતે મોટા પાછલા ટ્રાન્ઝેક્શનમાંથી કાં તો એક ઇનપુટ અથવા વિવિધ ઇનપુટ હશે જેમાં નાની રકમને ભેગી કરશે અને વધુમાં વધુ બે આઉટપુટ હશે: એક પેમેન્ટ માટે અને એક ફેરફાર, જો કોઈ હોય તો, સેન્ડરને પાછો મોકલવા માટે.



એ નોંધનીય છે કે ફેન-આઉટ, જેમાં એક વ્યવહાર ઘણા વ્યવહારો પર આધાર રાખે છે અને તે વ્યવહારો બીજા ઘણા પર આધાર રાખે છે, એ કોઈ સમસ્યા નથી. વ્યવહારના ઇતિહાસની સંપૂર્ણ એકલી નકલ કાઢવાની ક્યારેય જરૂર નથી.

10. પ્રાઇવસી

પરંપરાગત બેંકિંગ મોડલ સામેલ પક્ષકારો અને વિશ્વસનીય થર્ડ પાર્ટીની માહિતીની એક્સેસને મર્યાદિત કરીને એક લેવલની પ્રાઇવસી હાંસલ કરે છે. તમામ વ્યવહારોને જાહેરમાં જાહેર કરવાની જરૂરિયાત આ પદ્ધતિને બાકાત રાખે છે, પરંતુ અન્ય જગ્યાએ માહિતીના પ્રવાહને તોડીને ગોપનીયતા જાળવી શકાય છે: પબ્લિક કીને અનામી રાખીને. લોકો જોઈ શકે છે કે કોઈ વ્યક્તિ બીજી વ્યક્તિને રકમ મોકલી રહી છે, પરંતુ કોઈની સાથે વ્યવહાર લિંક કરતી માહિતી વિના. આ સ્ટોક એક્સચેન્જો દ્વારા બહાર પાડવામાં આવતી માહિતી જેવું જ છે, જ્યાં વ્યક્તિગત સોદાનો સમય અને કદ, એટલે કે "ટેપ" જાહેર કરવામાં આવે છે, પરંતુ પક્ષકારો કોણ હતા તે જણાવાતું નથી.



વધારાની ફાયરવોલ તરીકે, દરેક વ્યવહાર માટે એક નવી કી પેરનો ઉપયોગ થવો જોઈએ

જેથી કરીને તેને સામાન્ય માલિક સાથે જોડવામાં ન આવે. મલ્ટી-ઇનપુટ વ્યવહારો સાથે કેટલાક લિંકિંગ કરવા હજુ પણ અનિવાર્ય છે, જે આવશ્યકપણે જણાવે છે કે તેમના ઇનપુટ્સ એક જ માલિકની માલિકીના હતા. જોખમ એ છે કે જો કીનો માલિક જાહેર થાય તો લિંક કરવાથી તે જ માલિકના અન્ય વ્યવહારો જાહેર થઈ શકે છે.

11. ગણતરીઓ

અમે એક એવી સ્થિતિની કલ્પના કરીએ છીએ જેમાં કોઈ હુમલાખોર પ્રામાણિક ચેઇન કરતાં વધુ ઝડપથી વૈકલ્પિક ચેઇન બનાવવાનો પ્રયાસ કરે છે. જો આ કામ પૂરું થાય તો પણ, તે સિસ્ટમને મનસ્વી ફેરફારો માટે ખુલ્લું મૂકતી નથી, જેમ કે હવામાંથી પૈસા બનાવવા અથવા હુમલાખોરના ક્યારેય નહોતા તેવા પૈસા લેવા. નોડ્સ અમાન્ય ટ્રાન્ઝેક્શનને પેમેન્ટ તરીકે સ્વીકારશે નહીં અને પ્રમાણિક નોડ્સ ક્યારેય પણ તે ધરાવતા બ્લોકને સ્વીકારશે નહીં. હુમલાખોર તેણે તાજેતરમાં ખર્ચેલા નાણાં પરત લેવા માટે માત્ર તેના પોતાના વ્યવહારોમાંથી એક બદલવાનો પ્રયાસ કરી શકે છે.

પ્રામાણિક ચેઇન અને હુમલાખોર ચેઇન વચ્ચેની રેસને બાઇનોમિઅલ રેન્ડમ વૉક તરીકે દર્શાવી શકાય છે. એક બ્લોક દ્વારા પ્રમાણિક ચેઇનને વિસ્તારવામાં આવે અને તેની લીડ +1 વધારવામાં આવે તે સફળ કામગીરી છે અને નિષ્ફળતા કામગીરી એ છે જેમાં હુમલાખોરની ચેઇનને એક બ્લોક દ્વારા લંબાવવામાં આવે છે, જે અંતરને -1 દ્વારા ઘટાડે છે.

કોઈ હુમલાખોર માટે તેને થયેલા નુકસાનમાંથી બહાર નીકળવાની સંભાવના ગેમ્બલર્સ રૂઇન પ્રોબ્લેમ સમાન છે. ધારો કે અમર્યાદિત ક્રેડિટ ધરાવતો જુગારી ખોટથી શરૂઆત કરે છે અને બ્રેકઇવન સુધી પહોંચવાનો પ્રયાસ કરવા માટે સંભવિતપણે અસંખ્ય ટ્રાયલ લે છે. અમે તે સંભવિતતાની ગણતરી કરી શકીએ છીએ કે તે ક્યારે બ્રેકવેન સુધી પહોંચી શકશે અથવા હુમલાખોર ક્યારેય પ્રામાણિક ચેઇનને પકડી લેશે. આ ગણતરી નીચે પ્રમાણે [8]:

p = probability an honest node finds the next block
q = probability the attacker finds the next block
q_z = probability the attacker will ever catch up from z blocks behind

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

અમારી ધારણાને જોતાં કે p > q, સંભાવના ઝડપથી ઘટી જાય છે કારણ કે હુમલાખોરને પકડવા માટેના બ્લોક્સની સંખ્યા વધી જાય છે. પરિસ્થિતિ તેની વિરુદ્ધની હોવાથી જો તે શરૂઆતમાં જ આગળ વધવા માટે નસીબદાર ન હોય તો તે જેમ જેમ પાછળ પડતો જાય તેમ તેની તકો ખૂબ ઓછી થઈ જાય છે.

અમે હવે ધ્યાનમાં લઈએ છીએ કે નવો વ્યવહાર મેળવનારે પૂરતા પ્રમાણમાં નિશ્ચિત થતા પહેલા કેટલી રાહ જોવી પડશે કે મોકલનાર વ્યવહારને બદલી શકતો નથી. અમે ધારીએ છીએ કે મોકલનાર એક હુમલાખોર છે જે મેળવનારને વિશ્વાસ કરાવવા માંગે છે કે તેણે તેને થોડા સમય માટે પેમેન્ટ કર્યું છે અને પછી થોડો સમય પસાર થયા પછી તેને પોતાને પાછું પેમેન્ટ માટે સ્વિચ કરશે. જ્યારે આમ થશે ત્યારે રિસિવરને ચેતવણી આપવામાં આવશે, પરંતુ મોકલનારને આશા છે કે તે ખૂબ મોડું થઈ ગયું હશે.

રિસિવર એક નવી કી પેર બનાવે છે અને સાઇન કરતા પહેલા મોકલનારને પબ્લિક કી આપે છે. આ મોકલનારને તેના પર સતત કામ કરીને સમય પહેલાં બ્લોક્સની ચેઇન તૈયાર કરતા અટકાવે છે જ્યાં સુધી તે એટલો નસીબદાર ન બને કે તે ખાસ્સો આગળ વધી જાય અને પછી તે જ ક્ષણે ટ્રાન્ઝેક્શનનો અમલ કરી શકે. એકવાર ટ્રાન્ઝેક્શન મોકલ્યા પછી, અપ્રમાણિક મોકલનાર તેના વ્યવહારનું વૈકલ્પિક વર્ઝન ધરાવતી સમાંતર ચેઇન પર ગુપ્ત રીતે કામ કરવાનું શરૂ કરે છે.

પ્રાપ્તકર્તા ત્યાં સુધી રાહ જુએ છે જ્યાં સુધી ટ્રાન્ઝેક્શન બ્લોકમાં ઉમેરવામાં ન આવે અને તે પછી z બ્લોક્સને લિંક કરવામાં ન આવે. તે જાણતો નથી કે હુમલાખોરે કેટલી પ્રગતિ કરી છે, પરંતુ એમ માનતા કે પ્રમાણિક બ્લોક્સે બ્લોક દીઠ સરેરાશ અપેક્ષિત સમય લીધો છે, હુમલાખોરની સંભવિત પ્રગતિ અપેક્ષિત મૂલ્ય સાથે પોઈસન ડિસ્ટ્રીબ્યુશન હશે:

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

To get the probability the attacker could still catch up now, we multiply the Poisson density for each amount of progress he could have made by the probability he could catch up from that point:

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

Rearranging to avoid summing the infinite tail of the distribution...

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

Converting to C code...

```
#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;
}
```

Running some results, we can see the probability drop off exponentially with z.

```
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
```

Solving for P less than 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. નિષ્કર્ષ

અમે વિશ્વાસ પર આધાર રાખ્યા વિના ઈલેક્ટ્રોનિક વ્યવહારો માટે એક સિસ્ટમનો પ્રસ્તાવ મૂક્યો છે. અમે ડિજિટલ સિગ્નેચરથી બનેલા કોઇન્સના સામાન્ય માળખાથી શરૂઆત કરી છે, જે માલિકીનું મજબૂત નિયંત્રણ પૂરું પાડે છે, પરંતુ ડબલ-ખર્ચ અટકાવવાના માર્ગ વિના તે અધૂરી છે. આના ઉકેલ માટે, અમે વ્યવહારોના સાર્વજનિક ઇતિહાસને રેકોર્ડ કરવા માટે પ્રૂફ-ઓફ-વર્કનો ઉપયોગ કરીને પીઅર-ટુ-પીઅર નેટવર્કનો પ્રસ્તાવ મૂક્યો છે જે કોઈ હુમલાખોર માટે જલ્દીથી ગાણિતિક રીતે અવ્યવહારુ બની જાય છે જો પ્રમાણિક નોડ્સ મોટાભાગના સીપીયુ પાવરને નિયંત્રિત કરે. નેટવર્ક તેની અસંરચિત સરળતામાં મજબૂત છે. નોડ્સ થોડા સંકલન સાથે એક જ સમયે કામ કરે છે. તેમને ઓળખવાની જરૂર નથી, કારણ કે સંદેશાઓ કોઈ ચોક્કસ

જગ્યાએ મોકલવામાં આવતા નથી અને માત્ર શ્રેષ્ઠ પ્રયાસના આધારે ડિસ્ટ્રીબ્યુશન કરવાની જરૂર છે. નોડ્સ તેમની ઇચ્છા મુજબ નેટવર્ક છોડી શકે છે, ફરીથી જોડાઈ શકે છે અને પ્રૂફ-ઓફ-વર્ક ચેઇનને પુરાવા તરીકે સ્વીકારી શકે છે કે જ્યારે તેઓ ગયા હતા ત્યારે શું થયું હતું. તેઓ તેમની સીપીયુ શક્તિ સાથે વોટ આપે છે, માન્ય બ્લોક્સને લંબાવવા પર કામ કરીને તેમની સ્વીકૃતિ વ્યક્ત કરે છે અને તેમના પર કામ કરવાનો ઇનકાર કરીને અમાન્ય બ્લોક્સનો અસ્વીકાર કરે છે. કોઈપણ જરૂરી નિયમો અને પ્રોત્સાહનોને આ સર્વસંમતિ ધરાવતી મિકેનિઝમ દ્વારા લાગુ કરી શકાય છે.

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.