Rozwój i bezpieczeństwo technologii blockchain

Adam Sołtysik
11 października 2018

Wstęp

- blockchain to rozproszona baza transakcji

Wstęp

- blockchain to rozproszona baza transakcji
- transakcje są podpisywane kluczami prywatnymi użytkowników i rozsyłane w sieci

Wstęp

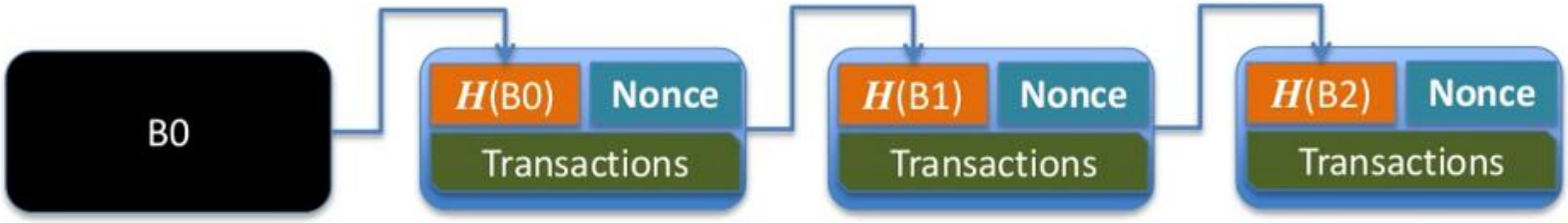
- blockchain to rozproszona baza transakcji
- transakcje są podpisywane kluczami prywatnymi użytkowników i rozsyłane w sieci
- następnie transakcje są weryfikowane, łączone w bloki i potwierdzane przez wyspecjalizowane węzły, zwane górnikami

Wstęp

- praca górnika polega na znalezieniu pewnej wartości, która dołączona do bloku sprawi, że jego hash będzie wystarczająco mały

Wstęp

- praca górnika polega na znalezieniu pewnej wartości, która dołączona do bloku sprawi, że jego hash będzie wystarczająco mały



Find a nonce x such that:

SHA-256(SHA-256($r + x$)) < T/d

https://www.slideshare.net/DavidEvansUVa/the-blockchain

Wstęp

- praca górnika polega na znalezieniu pewnej wartości, która dołączona do bloku sprawi, że jego hash będzie wystarczająco mały
- górnik, który pierwszy wykopie blok, dostaje nagrodę i prowizje z transakcji, a blok jest dołączany do łańcucha

Wstęp

- praca górnika polega na znalezieniu pewnej wartości, która dołączona do bloku sprawi, że jego hash będzie wystarczająco mały
- górnik, który pierwszy wykopie blok, dostaje nagrodę i prowizje z transakcji, a blok jest dołączany do łańcucha
- trudność (wartość d) jest dostosowywana, aby kopanie zawsze zajmowało stałą ilość czasu

Wstęp

- wszystkie węzły sieci muszą się zgadzać co do historii transakcji (tzw. konsensus)

Wstęp

- wszystkie węzły sieci muszą się zgadzać co do historii transakcji (tzw. konsensus)
- przyjmuje się, że najdłuższy poprawny łańcuch jest tym obowiązującym

Wstęp

- wszystkie węzły sieci muszą się zgadzać co do historii transakcji (tzw. konsensus)
- przyjmuje się, że najdłuższy poprawny łańcuch jest tym obowiązującym

https://bitsonblocks.net/2015/09/09/a-gentle-introduction-to-blockchain-technology/

Problemy z PoW

- Proof of Work w standardowej postaci wymaga ogromnej mocy obliczeniowej – tym większej, im większa sieć i liczba górników

Problemy z PoW

- Proof of Work w standardowej postaci wymaga ogromnej mocy obliczeniowej – tym większej, im większa sieć i liczba górników
- skutkuje to bardzo wysokim zużyciem energii – kilkaset kWh na transakcję (~60 TWh rocznie) w przypadku bitcoina

Problemy z PoW

- Proof of Work w standardowej postaci wymaga ogromnej mocy obliczeniowej – tym większej, im większa sieć i liczba górników
- skutkuje to bardzo wysokim zużyciem energii – kilkaset kWh na transakcję (~60 TWh rocznie) w przypadku bitcoina
- powstały zatem alternatywne algorytmy uzyskiwania konsensusu w sieci

Proof of Stake

- zamiast przeprowadzania kosztownych obliczeń, w algorytmie PoS zwycięzca wybierany jest pseudolosowo (ale deterministycznie) na podstawie kombinacji pewnych atrybutów, np. liczby posiadanych monet albo ich wieku

Proof of Stake

- zamiast przeprowadzania kosztownych obliczeń, w algorytmie PoS zwycięzca wybierany jest pseudolosowo (ale deterministycznie) na podstawie kombinacji pewnych atrybutów, np. liczby posiadanych monet albo ich wieku
- zwycięzca dołącza blok wg własnego uznania

Proof of Stake

- zamiast przeprowadzania kosztownych obliczeń, w algorytmie PoS zwycięzca wybierany jest pseudolosowo (ale deterministycznie) na podstawie kombinacji pewnych atrybutów, np. liczby posiadanych monet albo ich wieku
- zwycięzca dołącza blok wg własnego uznania
- założenie polega m.in. na tym, że im więcej ktoś posiada monet, tym większa szansa, że zależy mu na sieci i nie spróbuje jej zaszkodzić

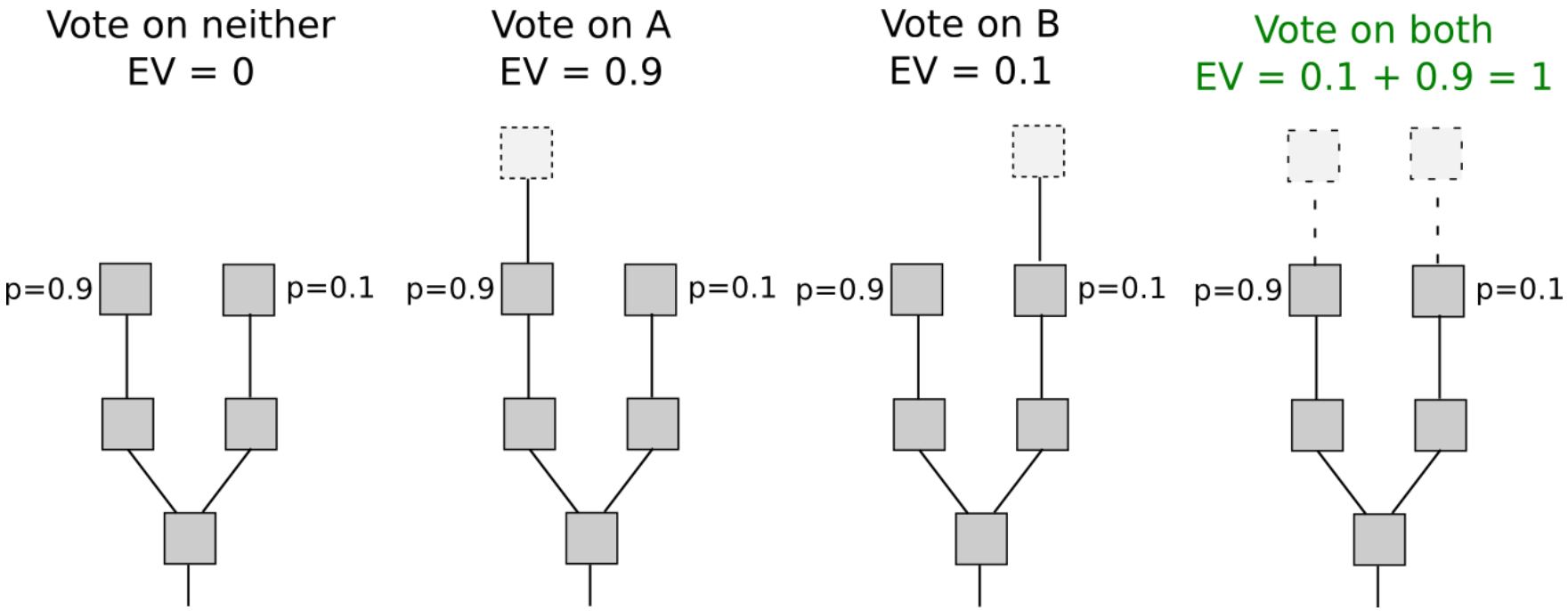
Proof of Stake

- niestety nawet w zupełnie uczciwej sieci istnieje możliwość, że konsensus nie zostanie osiągnięty

Proof of Stake

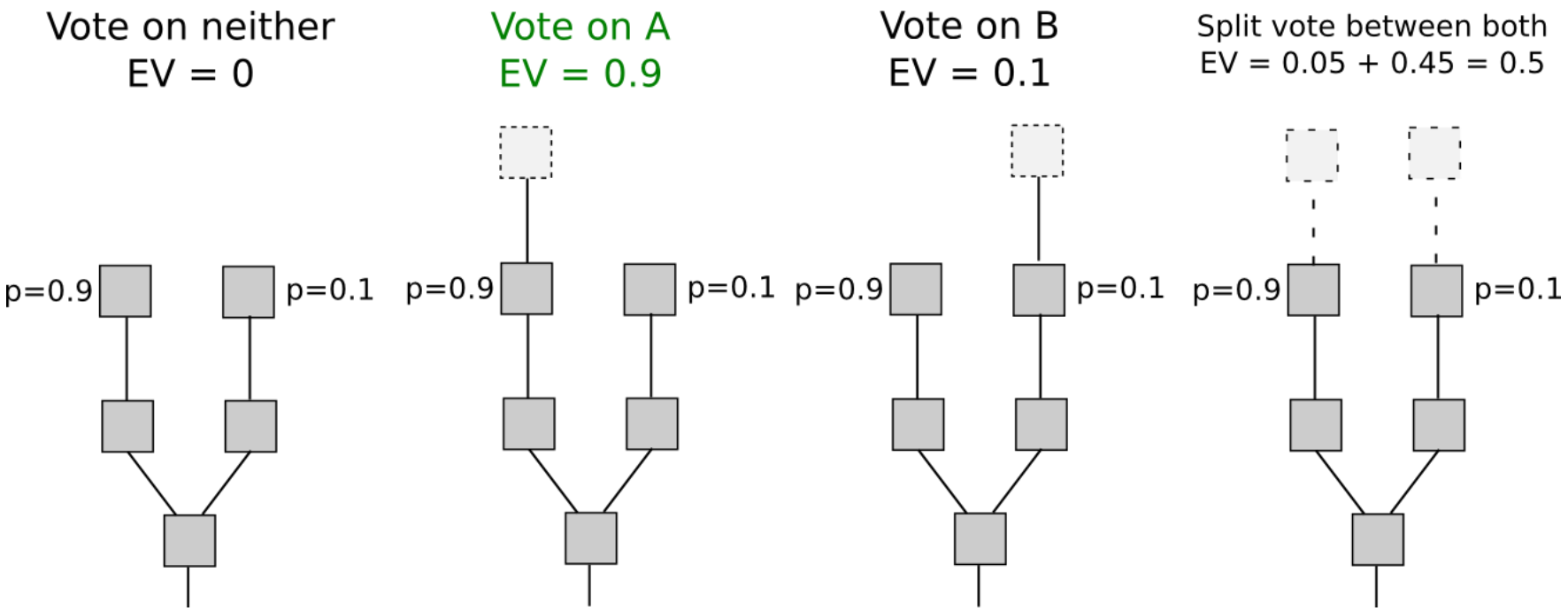
- niestety nawet w zupełnie uczciwej sieci istnieje możliwość, że konsensus nie zostanie osiągnięty
- w przypadku rozdziału łańcucha, w najlepszym interesie walidatora jest bowiem utworzyć bloki na obu łańcuchach – w przeciwieństwie do PoW, nic go to dodatkowo nie kosztuje

Proof of Stake



https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQs

Proof of Stake



https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQs

Proof of Stake

- niestety nawet w zupełnie uczciwej sieci istnieje możliwość, że konsensus nie zostanie osiągnięty
- w przypadku rozdziału łańcucha, w najlepszym interesie walidatora jest bowiem utworzyć bloki na obu łańcuchach – w przeciwieństwie do PoW, nic go to dodatkowo nie kosztuje
- jednym z rozwiązań jest wprowadzenie kar za tworzenie wielu bloków (na zasadzie “proof of misbehaviour”)

Delegated Proof of Stake

- jest to odmiana PoS, w której bloki są potwierdzane przez tzw. delegatów

Delegated Proof of Stake

- jest to odmiana PoS, w której bloki są potwierdzane przez tzw. delegatów
- delegaci są wybierani przez użytkowników

Delegated Proof of Stake

- jest to odmiana PoS, w której bloki są potwierdzane przez tzw. delegatów
- delegaci są wybierani przez użytkowników
- działania delegatów są publicznie dostępne

Delegated Proof of Stake

- jest to odmiana PoS, w której bloki są potwierdzane przez tzw. delegatów
- delegaci są wybierani przez użytkowników
- działania delegatów są publicznie dostępne
- moc głosu użytkownika na delegata zależy od ilości posiadanych przez niego środków

Delegated Proof of Stake

- jest to odmiana PoS, w której bloki są potwierdzane przez tzw. delegatów
- delegaci są wybierani przez użytkowników
- działania delegatów są publicznie dostępne
- moc głosu użytkownika na delegata zależy od ilości posiadanych przez niego środków
- przykład: Lisk, 101 delegatów

Proof of Capacity

- analogicznie do mocy obliczeniowej w PoW, algorytm PoC opiera się o pojemność dyskową

Proof of Capacity

- analogicznie do mocy obliczeniowej w PoW, algorytm PoC opiera się o pojemność dyskową
- zamiast obliczać blok na bieżąco, obliczenia przeprowadzane są z góry i zapisywane na dysku

Proof of Capacity

- analogicznie do mocy obliczeniowej w PoW, algorytm PoC opiera się o pojemność dyskową
- zamiast obliczać blok na bieżąco, obliczenia przeprowadzane są z góry i zapisywane na dysku
- do potwierdzenia bloku wystarczy przeczytać z dysku pasujące rozwiązanie

Proof of Capacity

- analogicznie do mocy obliczeniowej w PoW, algorytm PoC opiera się o pojemność dyskową
- zamiast obliczać blok na bieżąco, obliczenia przeprowadzane są z góry i zapisywane na dysku
- do potwierdzenia bloku wystarczy przeczytać z dysku pasujące rozwiązanie
- ta metoda nie jest aktualnie popularna i jest słabo przetestowana w praktyce

Proof of Capacity

- analogicznie do mocy obliczeniowej w PoW, algorytm PoC opiera się o pojemność dyskową
- zamiast obliczać blok na bieżąco, obliczenia przeprowadzane są z góry i zapisywane na dysku
- do potwierdzenia bloku wystarczy przeczytać z dysku pasujące rozwiązanie
- ta metoda nie jest aktualnie popularna i jest słabo przetestowana w praktyce
- przykład: Burstcoin

Problemy ze skalowalnością

- liczba transakcji w bloku jest ograniczona przez jego maksymalny rozmiar w bajtach

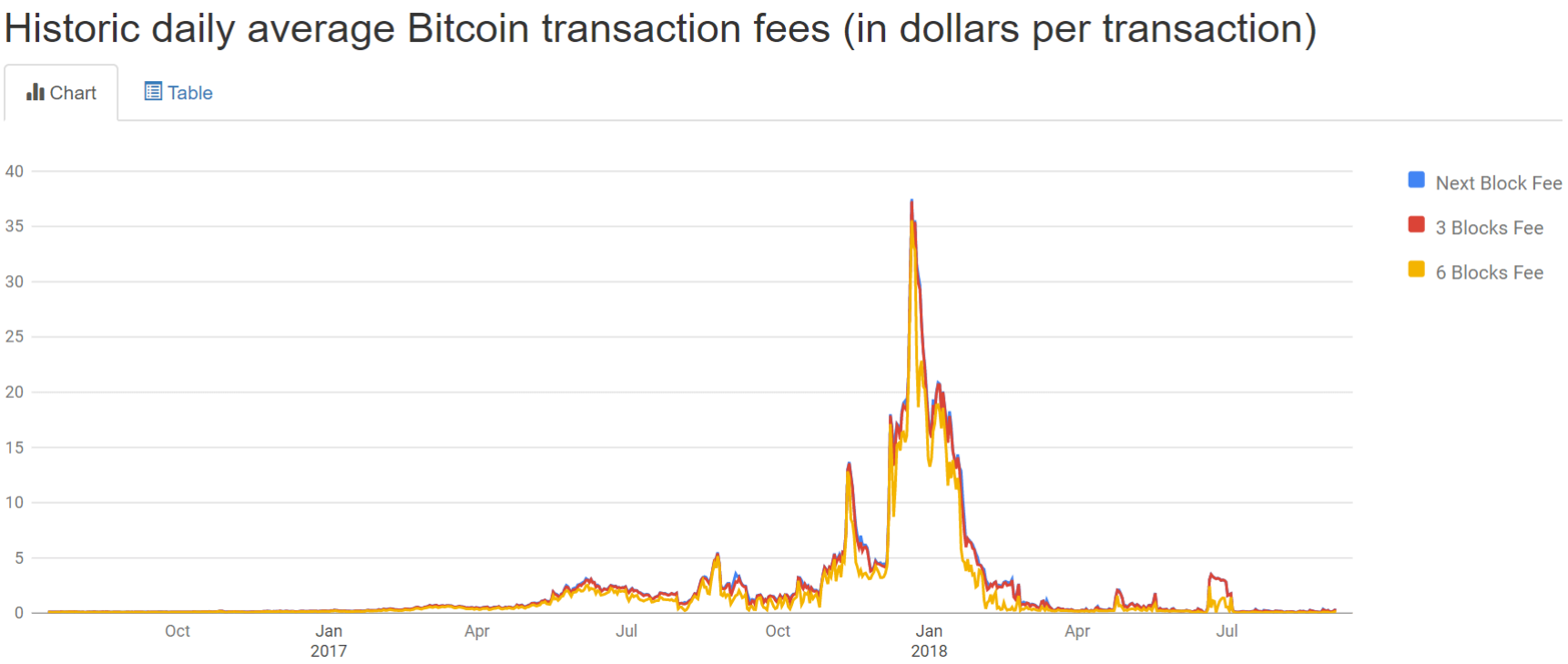
Problemy ze skalowalnością

- liczba transakcji w bloku jest ograniczona przez jego maksymalny rozmiar w bajtach
- ograniczenie zapobiega atakom spamerskim i DDOS oraz pomaga w decentralizacji sieci...

Problemy ze skalowalnością

- liczba transakcji w bloku jest ograniczona przez jego maksymalny rozmiar w bajtach
- ograniczenie zapobiega atakom spamerskim i DDOS oraz pomaga w decentralizacji sieci…
- … jednak istotnie zmniejsza jej przepustowość

Problemy ze skalowalnością



https://masterthecrypto.com/blockchain-scalability-bitcoin-scalability-problem-effects/

Problemy ze skalowalnością

- liczba transakcji w bloku jest ograniczona przez jego maksymalny rozmiar w bajtach
- ograniczenie zapobiega atakom spamerskim i DDOS oraz pomaga w decentralizacji sieci...
- ... jednak istotnie zmniejsza jej przepustowość
- jedno z rozwiązań to zwiększenie rozmiaru bloku (tak powstał np. fork Bitcoin Cash)

Problemy ze skalowalnością

- liczba transakcji w bloku jest ograniczona przez jego maksymalny rozmiar w bajtach
- ograniczenie zapobiega atakom spamerskim i DDOS oraz pomaga w decentralizacji sieci...
- ... jednak istotnie zmniejsza jej przepustowość
- jedno z rozwiązań to zwiększenie rozmiaru bloku (tak powstał np. fork Bitcoin Cash)
- inne rozwiązanie to przesyłanie mniejszych transakcji poza łańcuchem głównym

Lightning Network

- Lightning Network opiera się na otwieraniu prywatnych kanałów między węzłami sieci

Lightning Network

- Lightning Network opiera się na otwieraniu prywatnych kanałów między węzłami sieci
- podstawą kanału są transakcje, których wyjściem jest multi-signature address – adres, z którego wypłacić środki można tylko za podpisem kilku (w tym przypadku 2) stron

Lightning Network

- główny pomysł opiera się na takiej sytuacji: Alicja chce płacić Bobowi cyklicznie za pewną usługę

Lightning Network

- główny pomysł opiera się na takiej sytuacji: Alicja chce płacić Bobowi cyklicznie za pewną usługę
- Alicja otwiera kanał, do którego wpłaca *X* BTC, z zastrzeżeniem, że środki do niej wrócą, jeśli transakcja nie zostanie podpisana i rozesłana przez Boba np. w ciągu tygodnia

Lightning Network

- główny pomysł opiera się na takiej sytuacji: Alicja chce płacić Bobowi cyklicznie za pewną usługę
- Alicja otwiera kanał, do którego wpłaca X BTC, z zastrzeżeniem, że środki do niej wrócą, jeśli transakcja nie zostanie podpisana i rozesłana przez Boba np. w ciągu tygodnia
- w miarę wykonywania usługi przez Boba, Alicja podpisuje i przesyła Bobowi nowe, zastępcze transakcje, w których Bob otrzymuje Y_i BTC

Lightning Network

- Bob w każdym momencie zakończyć współpracę, rozsyłając transakcję i – otrzyma Y_i fizycznych środków, a Alicja dostanie z powrotem X-Y_i

Lightning Network

- Bob w każdym momencie zakończyć współpracę, rozsyłając transakcję i – otrzyma Y_i fizycznych środków, a Alicja dostanie z powrotem X-Y_i
- jeśli natomiast Bob nie podejmie żadnego działania w ciągu tygodnia od ostatniej transakcji, to straci wszystkie zarobione środki

Lightning Network

- Bob w każdym momencie zakończyć współpracę, rozsyłając transakcję i – otrzyma Y_i fizycznych środków, a Alicja dostanie z powrotem X-Y_i
- jeśli natomiast Bob nie podejmie żadnego działania w ciągu tygodnia od ostatniej transakcji, to straci wszystkie zarobione środki
- wadą całego systemu jest konieczność tworzenia kanałów dla każdego sklepu, dla którego chcemy za coś zapłacić, albo scentralizowanych “hubów”

Bitcoin overflow error (2010)

- bitcoin używa 64-bitowych unsigned intów

Bitcoin overflow error (2010)

- bitcoin używa 64-bitowych unsigned intów
- 2^64 = 18 446 744 073 709 551 615

Bitcoin overflow error (2010)

- bitcoin używa 64-bitowych unsigned intów
- 2^64 = 18 446 744 073 709 551 615
- 1 BTC = 100 000 000 sat

Bitcoin overflow error (2010)

- bitcoin używa 64-bitowych unsigned intów
- 2^64 = 18 446 744 073 709 551 615
- 1 BTC = 100 000 000 sat
- max ≈ 184 467 440 737 BTC

Bitcoin overflow error (2010)

- bitcoin używa 64-bitowych unsigned intów
- 2^64 = 18 446 744 073 709 551 615
- 1 BTC = 100 000 000 sat
- max ≈ 184 467 440 737 BTC
- (ale bitcoinów może być tylko 21 000 000)

Bitcoin overflow error (2010)

- bitcoin używa 64-bitowych unsigned intów
- 2^64 = 18 446 744 073 709 551 615
- 1 BTC = 100 000 000 sat
- max ≈ 184 467 440 737 BTC
- (ale bitcoinów może być tylko 21 000 000)
- warunkiem koniecznym poprawności transakcji jest to, aby $\sum$ input ≤ $\sum$ output

Bitcoin overflow error (2010)

- błąd w kliencie polegał na tym, że sprawdzana była tylko suma, ale nie poszczególne transakcje

Bitcoin overflow error (2010)

- błąd w kliencie polegał na tym, że sprawdzana była tylko suma, ale nie poszczególne transakcje
- atakujący wykonał ze swojego portfela, zawierającego 50.51 BTC, transakcję z trzema wyjściami, w tym dwoma po 92 mld BTC

Bitcoin overflow error (2010)

- błąd w kliencie polegał na tym, że sprawdzana była tylko suma, ale nie poszczególne transakcje
- atakujący wykonał ze swojego portfela, zawierającego 50.51 BTC, transakcję z trzema wyjściami, w tym dwoma po 92 mld BTC
- transakcja była “poprawna”, ponieważ suma po przepełnieniu wynosiła mniej niż 50.51 BTC

Bitcoin overflow error (2010)

- błąd w kliencie polegał na tym, że sprawdzana była tylko suma, ale nie poszczególne transakcje
- atakujący wykonał ze swojego portfela, zawierającego 50.51 BTC, transakcję z trzema wyjściami, w tym dwoma po 92 mld BTC
- transakcja była “poprawna”, ponieważ suma po przepełnieniu wynosiła mniej niż 50.51 BTC
- błąd został zauważony i naprawiony w ciągu kilku godzin, a transakcja wycofana (chain split)

Chain split vs Hard fork vs Soft fork

- rozdzielenie łańcucha może być celowe lub może wystąpić w wyniku aktualizacji oprogramowania

Chain split vs Hard fork vs Soft fork

- rozdzielenie łańcucha może być celowe lub może wystąpić w wyniku aktualizacji oprogramowania
- aktualizację nazywa się forkiem, nawet jeśli nie następuje rozdzielenie łańcucha (albo jeden łańcuch bezsprzecznie zdominuje drugi)

Chain split vs Hard fork vs Soft fork

- rozdzielenie łańcucha może być celowe lub może wystąpić w wyniku aktualizacji oprogramowania
- aktualizację nazywa się forkiem, nawet jeśli nie następuje rozdzielenie łańcucha (albo jeden łańcuch bezsprzecznie zdominuje drugi)
- aktualizacja może albo znieść pewne ograniczenia w protokole, albo wprowadzić nowe

Soft fork

- soft fork zawęża ograniczenia protokołu

Soft fork

- soft fork zawęża ograniczenia protokołu
- oznacza to, że nowe bloki są kompatybilne ze starą wersją oprogramowania, ale nie na odwrót

Soft fork

- soft fork zawęża ograniczenia protokołu
- oznacza to, że nowe bloki są kompatybilne ze starą wersją oprogramowania, ale nie na odwrót
- aby soft fork zadziałał, wystarczy żeby górnicy zaktualizowali swoje oprogramowanie i rozgłaszali bloki zgodne z nowymi zasadami

Soft fork

- soft fork zawęża ograniczenia protokołu
- oznacza to, że nowe bloki są kompatybilne ze starą wersją oprogramowania, ale nie na odwrót
- aby soft fork zadziałał, wystarczy żeby górnicy zaktualizowali swoje oprogramowanie i rozgłaszali bloki zgodne z nowymi zasadami
- małe ryzyko stałego rozdziału łańcucha

Soft fork

- soft fork zawęża ograniczenia protokołu
- oznacza to, że nowe bloki są kompatybilne ze starą wersją oprogramowania, ale nie na odwrót
- aby soft fork zadziałał, wystarczy żeby górnicy zaktualizowali swoje oprogramowanie i rozgłaszali bloki zgodne z nowymi zasadami
- małe ryzyko stałego rozdziału łańcucha
- przykład: zmniejszenie rozmiaru bloku, poprawki błędów

Hard fork

- hard fork rozluźnia ograniczenia protokołu

Hard fork

- hard fork rozluźnia ograniczenia protokołu
- nowe bloki nie są kompatybilne ze starą wersją oprogramowania

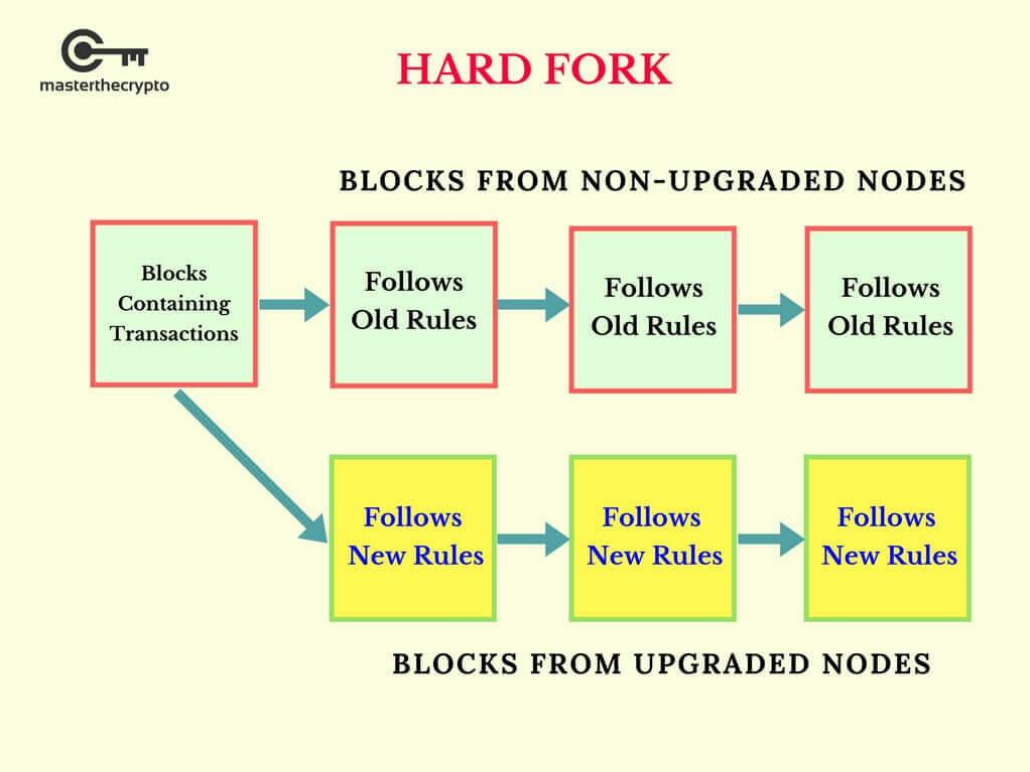
Hard fork

- hard fork rozluźnia ograniczenia protokołu
- nowe bloki nie są kompatybilne ze starą wersją oprogramowania
- wszystkie węzły muszą zostać zaktualizowane

Hard fork

- hard fork rozluźnia ograniczenia protokołu
- nowe bloki nie są kompatybilne ze starą wersją oprogramowania
- wszystkie węzły muszą zostać zaktualizowane
- wysokie ryzyko stałego rozdziału łańcucha i ataku double spend

Hard fork



https://masterthecrypto.com/guide-to-forks-hard-fork-soft-fork/

Hard fork

- hard fork rozluźnia ograniczenia protokołu
- nowe bloki nie są kompatybilne ze starą wersją oprogramowania
- wszystkie węzły muszą zostać zaktualizowane
- wysokie ryzyko stałego rozdziału łańcucha i ataku double spend
- przykłady: zwiększenie rozmiaru bloku (BCH), inny algorytm PoW (BTG)

Bitcoin Core bugs (2017-18)

- Bitcoin Core to pierwsza i najpopularniejsza implementacja pełnego klienta sieci (full node)

Bitcoin Core bugs (2017-18)

- Bitcoin Core to pierwsza i najpopularniejsza implementacja pełnego klienta sieci (full node)
- w wersji 0.14.0 (marzec 2017), w wyniku zbiegu różnych okoliczności, w kodzie walidacji bloku zostały wprowadzone błędy, które przetrwały aż do wersji 0.16.2 (wrzesień 2018)

Bitcoin Core bugs (2017-18)

- pierwszy błąd sprawiał, że próba dwukrotnego wydania tych samych środków w jednej transakcji powodowała crash klienta

Bitcoin Core bugs (2017-18)

```
-bool CTransaction::UpdateCoins(CCoinsViewCache &inputs, CTxUndo &txundo, int nHeight, const uint256 &txha
sh) const
+bool CTransaction::UpdateCoins(CValidationState &state, CCoinsViewCache &inputs, CTxUndo &txundo, int nHe
ight, const uint256 &txhash) const
 {
     // mark inputs spent
     if (!IsCoinBase()) {
         BOOST_FOREACH(const CTxIn &txin, vin) {
             CCoins &coins = inputs.GetCoins(txin.prevout.hash);
             CTxInUndo undo;
-            if (!coins.Spend(txin.prevout, undo))
-                return error("UpdateCoins() : cannot spend input");
+            assert(coins.Spend(txin.prevout, undo));
             txundo.vprevout.push_back(undo);
         }
     }

     // add outputs
-    if (!inputs.SetCoins(txhash, CCoins(*this, nHeight)))
-        return error("UpdateCoins() : cannot update output");
+    assert(inputs.SetCoins(txhash, CCoins(*this, nHeight)));

     return true;
 }
```

Bitcoin Core bugs (2017-18)

- pierwszy błąd sprawiał, że próba dwukrotnego wydania tych samych środków w jednej transakcji powodowała crash klienta
- do ataku teoretycznie wystarczyło wykopanie i rozesłanie jednego bloku ze złośliwą transakcją – koszt 12.5 BTC za (tymczasowe) zniszczenie sieci

Bitcoin Core bugs (2017-18)

- w wersji 0.15.0 (wrzesień 2017) została wprowadzona druga iteracja błędu

Bitcoin Core bugs (2017-18)

- w wersji 0.15.0 (wrzesień 2017) została wprowadzona druga iteracja błędu
- zamiast crashu, klient po prostu akceptował transakcję z podwójnie wydanymi środkami

Bitcoin Core bugs (2017-18)

- w wersji 0.15.0 (wrzesień 2017) została wprowadzona druga iteracja błędu
- zamiast crashu, klient po prostu akceptował transakcję z podwójnie wydanymi środkami
- wykorzystanie błędu spowodowałoby rozdzielenie łańcucha

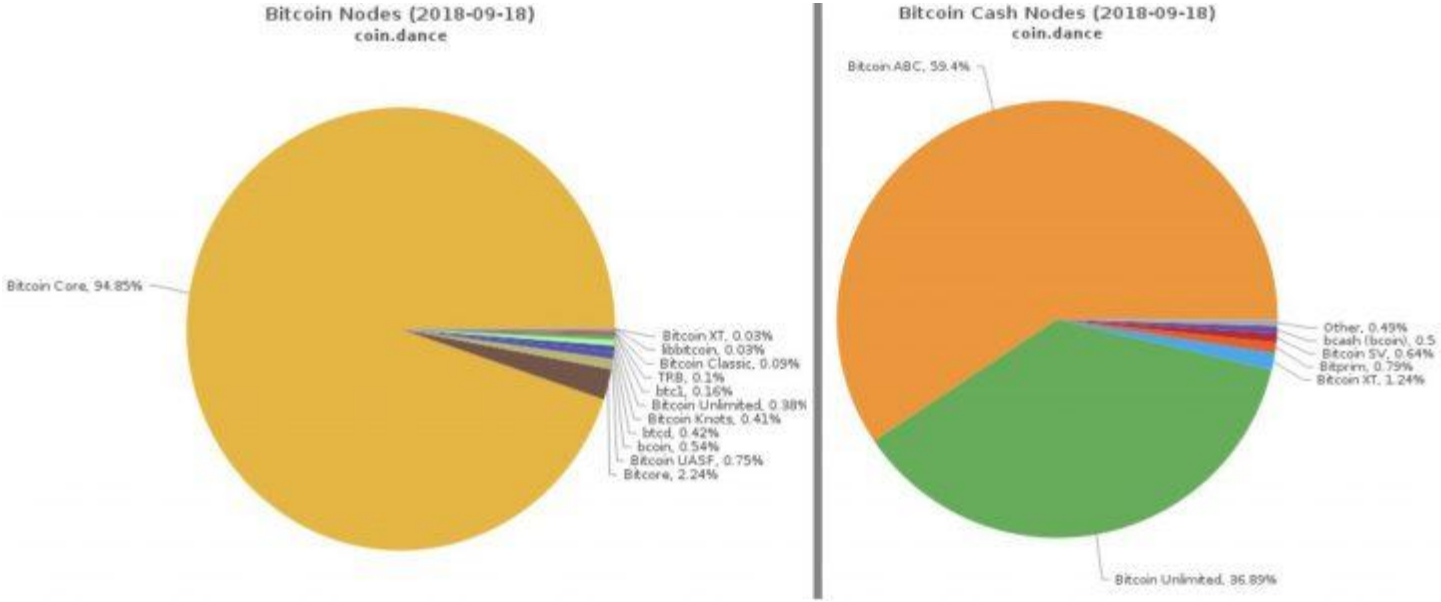
Bitcoin Core bugs (2017-18)

- błędy nie zostały jednak przez nikogo wykorzystane i zostały załatane w ciągu kilku godzin od zgłoszenia

Bitcoin Core bugs (2017-18)

- błędy nie zostały jednak przez nikogo wykorzystane i zostały załatane w ciągu kilku godzin od zgłoszenia
- podniosły się głosy o zbyt dużym “monopolu” implementacji protokołu bitcoina

Bitcoin Core bugs (2017-18)



https://news.bitcoin.com/critical-bug-found-in-bitcoin-core-invokes-the-multiple-client-argument/

Bitcoin Core bugs (2017-18)

- błędy nie zostały jednak przez nikogo wykorzystane i zostały załatane w ciągu kilku godzin od zgłoszenia
- podniosły się głosy o zbyt dużym “monopolu” implementacji protokołu bitcoina
- inne wnioski: krytyczne miejsca w kodzie powinny być napisane jak najbardziej przejrzyście, testy powinny sprawdzać wszelkie dziwne przypadki brzegowe

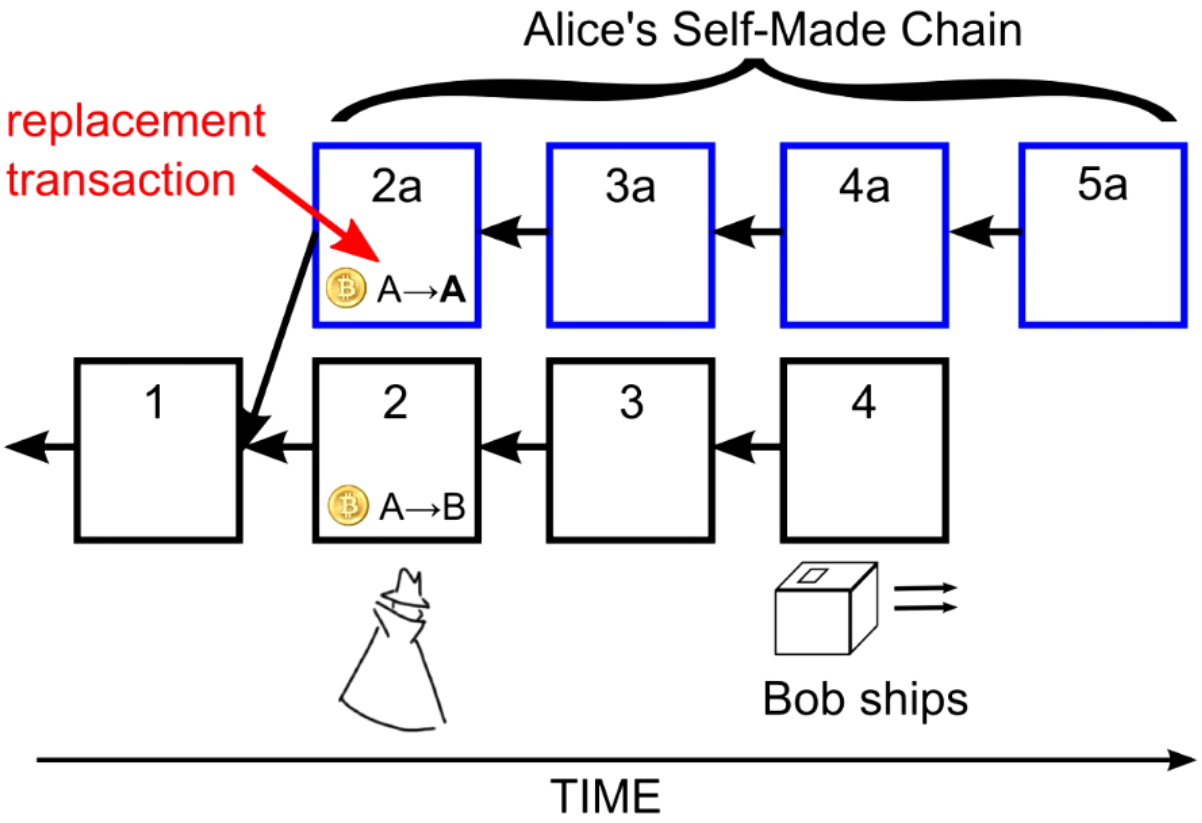
Bitcoin Core bugs (2017-18)

- na błędzie ucierpiała natomiast niszowa kryptowaluta Pigeoncoin, która jako jedna z wielu korzysta z części kodu bitcoina

Bitcoin Core bugs (2017-18)

- na błędzie ucierpiała natomiast niszowa kryptowaluta Pigeoncoin, która jako jedna z wielu korzysta z części kodu bitcoina
- zanim została załatana, atakujący był w stanie “dodrukować” ¼ monet znajdujących się na rynku, wartych $15000

51% double-spend attacks



http://www.imponderablethings.com/2013/07/how-bitcoin-works-under-hood.html

BTG 51% attack (2018)

- Bitcoin Gold (BTG) to fork bitcoina powstały w listopadzie 2017

BTG 51% attack (2018)

- Bitcoin Gold (BTG) to fork bitcoina powstały w listopadzie 2017
- różni się innym algorytmem PoW, wymagającym większej ilości pamięci

BTG 51% attack (2018)

- Bitcoin Gold (BTG) to fork bitcoina powstały w listopadzie 2017
- różni się innym algorytmem PoW, wymagającym większej ilości pamięci
- miało to na celu zwiększenie decentralizacji poprzez utrudnienie wykorzystania wyspecjalizowanych układów (ASIC)

BTG 51% attack (2018)

- mimo to, w maju 2018 grupa górników przejęła sieć i przeprowadziła udany atak double-spend na giełdy

BTG 51% attack (2018)

- mimo to, w maju 2018 grupa górników przejęła sieć i przeprowadziła udany atak double-spend na giełdy
- choć zespół rozesłał ostrzeżenia o koniecznym zwiększeniu liczby potwierdzeń, atakujący zdobyli monety warte 18 milionów dolarów

BTG 51% attack (2018)

- mimo to, w maju 2018 grupa górników przejęła sieć i przeprowadziła udany atak double-spend na giełdy
- choć zespół rozesłał ostrzeżenia o koniecznym zwiększeniu liczby potwierdzeń, atakujący zdobyli monety warte 18 milionów dolarów
- giełda Bittrex we wrześniu usunęła kryptowalutę po tym, jak deweloperzy odmówili zapłacenia za szkody

Verge (XVG) hack (2018)

- ustalenie dokładnego czasu w rozproszonej sieci nie jest łatwym zadaniem

Verge (XVG) hack (2018)

- ustalenie dokładnego czasu w rozproszonej sieci nie jest łatwym zadaniem
- protokół XVG dopuszczał niezgodność czasu w wysokości 2 godzin

Verge (XVG) hack (2018)

- ustalenie dokładnego czasu w rozproszonej sieci nie jest łatwym zadaniem
- protokół XVG dopuszczał niezgodność czasu w wysokości 2 godzin
- bloki powinny pojawiać się co 30 sekund

Verge (XVG) hack (2018)

- ustalenie dokładnego czasu w rozproszonej sieci nie jest łatwym zadaniem
- protokół XVG dopuszczał niezgodność czasu w wysokości 2 godzin
- bloki powinny pojawiać się co 30 sekund
- aby dostosować trudność kopania, algorytm PoW bierze pod uwagę, z jaką częstotliwością nowe bloki pojawiały się w ciągu ostatnich 30 minut

Verge (XVG) hack (2018)

- atakujący kopał bloki z czasem przesuniętym o godzinę do tyłu

Verge (XVG) hack (2018)

- atakujący kopał bloki z czasem przesuniętym o godzinę do tyłu
- spowodowało to spadek trudności prawie do zera

Verge (XVG) hack (2018)

- atakujący kopał bloki z czasem przesuniętym o godzinę do tyłu
- spowodowało to spadek trudności prawie do zera
- samo to jednak dałoby taką samą przewagę atakującemu, jak i wszystkim innym górnikom

Verge (XVG) hack (2018)

- aby uniknąć centralizacji w kopaniu kryptowaluty, protokół XVG zakłada wykorzystanie 5 różnych algorytmów PoW

Verge (XVG) hack (2018)

- aby uniknąć centralizacji w kopaniu kryptowaluty, protokół XVG zakłada wykorzystanie 5 różnych algorytmów PoW
- każdy z nich ma własny współczynnik trudności

Verge (XVG) hack (2018)

- aby uniknąć centralizacji w kopaniu kryptowaluty, protokół XVG zakłada wykorzystanie 5 różnych algorytmów PoW
- każdy z nich ma własny współczynnik trudności
- w rezultacie atakujący był w stanie przejąć sieć, mając znacznie mniej niż 50% mocy

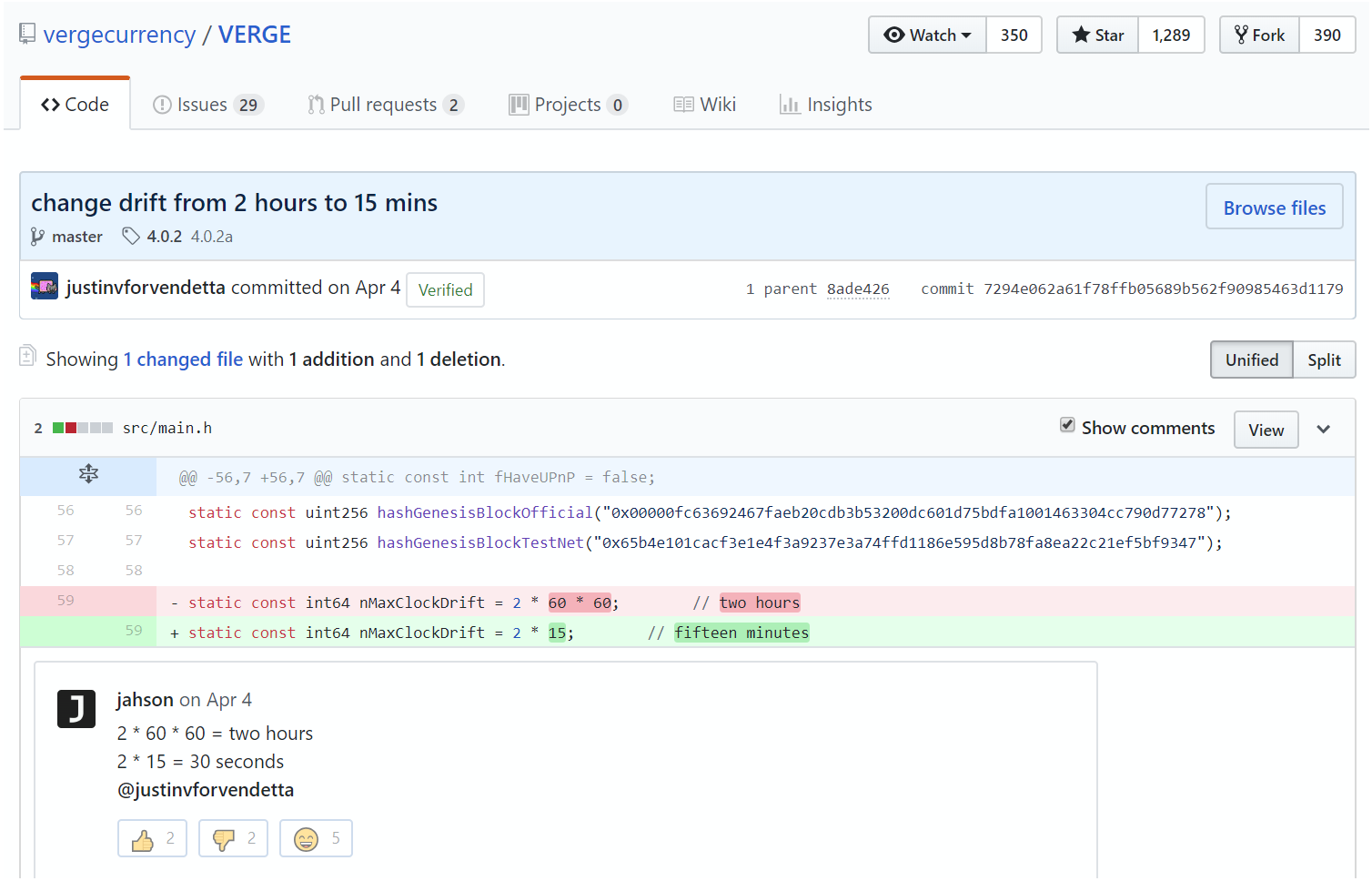
Verge (XVG) hack (2018)

- aby uniknąć centralizacji w kopaniu kryptowaluty, protokół XVG zakłada wykorzystanie 5 różnych algorytmów PoW
- każdy z nich ma własny współczynnik trudności
- w rezultacie atakujący był w stanie przejąć sieć, mając znacznie mniej niż 50% mocy
- sieć została całkiem zablokowana, a haker generował monety o wartości $80 na sekundę

Verge (XVG) hack (2018)

- deweloperzy w ciągu kilku dni wypuścili kilka średnio przemyślanych łat

Verge (XVG) hack (2018)



vergecurrency / VERGE

Watch 350 | Star 1,289 | Fork 390

Code | Issues 29 | Pull requests 2 | Projects 0 | Wiki | Insights

change drift from 2 hours to 15 mins

master 4.0.2 4.0.2a

Browse files

justinvforvendetta committed on Apr 4 Verified

1 parent 8ade426 commit 7294e062a61f78ffb05689b562f90985463d1179

Showing 1 changed file with 1 addition and 1 deletion.

Unified | Split

2 src/main.h

Show comments | View

```
@@ -56,7 +56,7 @@ static const int fHaveUPnP = false;
 56  56   static const uint256 hashGenesisBlockOfficial("0x00000fc63692467faeb20cdb3b53200dc601d75bdfa1001463304cc790d77278");
 57  57   static const uint256 hashGenesisBlockTestNet("0x65b4e101cacf3e1e4f3a9237e3a74ffd1186e595d8b78fa8ea22c21ef5bf9347");
 58  58
 59      - static const int64 nMaxClockDrift = 2 * 60 * 60;        // two hours
     59  + static const int64 nMaxClockDrift = 2 * 15;        // fifteen minutes
```

jahson on Apr 4

2 * 60 * 60 = two hours

2 * 15 = 30 seconds

@justinvforvendetta

2 | 2 | 5

Verge (XVG) hack (2018)

quick update

master 4.0.2 4.0.2a

Browse files

justinvforvendetta committed on Apr 4 Verified

1 parent 7294e06 commit b6c380727ebe285538b9e5ac330176d9e8983f87

Showing 1 changed file with 1 addition and 1 deletion.

Unified Split

2 src/main.h View

```
@@ -56,7 +56,7 @@ static const int fHaveUPnP = false;
 56  56   static const uint256 hashGenesisBlockOfficial("0x00000fc63692467faeb20cdb3b53200dc601d75bdfa1001463304cc790d77278");
 57  57   static const uint256 hashGenesisBlockTestNet("0x65b4e101cacf3e1e4f3a9237e3a74ffd1186e595d8b78fa8ea22c21ef5bf9347");
 58  58
 59      - static const int64 nMaxClockDrift = 2 * 15;        // fifteen minutes
     59  + static const int64 nMaxClockDrift = 2 * 15 * 15;
 60  60
 61  61   extern CScript COINBASE_FLAGS;
 62  62
```

7 comments on commit `b6c3807`

justinvforvendetta replied on Apr 4 Collaborator

waits for peanut gallery to get calculators out and make a comment...

1 1

Verge (XVG) hack (2018)

Update main.h

master 4.0.2 4.0.2a

Browse files

justinvforvendetta committed on Apr 4 Verified

1 parent b6c3807 commit 66673a508482d2eb7ccb2d32f144863cd48b43a6

Showing 1 changed file with 1 addition and 1 deletion.

Unified Split

2 src/main.h View

```
@@ -56,7 +56,7 @@ static const int fHaveUPnP = false;
 56  56   static const uint256 hashGenesisBlockOfficial("0x00000fc63692467faeb20cdb3b53200dc601d75bdfa1001463304cc790d77278");
 57  57   static const uint256 hashGenesisBlockTestNet("0x65b4e101cacf3e1e4f3a9237e3a74ffd1186e595d8b78fa8ea22c21ef5bf9347");
 58  58
 59      - static const int64 nMaxClockDrift = 2 * 15 * 15;
     59  + static const int64 nMaxClockDrift = 2 * 60 * 60;        // two hours - will mod into fork
 60  60
 61  61   extern CScript COINBASE_FLAGS;
 62  62
```

Verge (XVG) hack (2018)

3 src/main.h View

```
@@ -31,6 +31,7 @@ class CNode;

 static const int MULTI_ALGO_SWITCH_BLOCK = 340000;
 static const int STEALTH_TX_SWITCH_BLOCK = 1824150;
+static const int TIMESTAMP_RULES_SWITCH_BLOCK = 2040000;
 static const unsigned int MAX_BLOCK_SIZE = 1000000;
 static const unsigned int MAX_BLOCK_SIZE_GEN = MAX_BLOCK_SIZE/2;
 static const unsigned int MAX_BLOCK_SIGOPS = MAX_BLOCK_SIZE/50;
@@ -56,7 +57,7 @@ static const int fHaveUPnP = false;
 static const uint256 hashGenesisBlockOfficial("0x00000fc63692467faeb20cdb3b53200dc601d75bdfa1001463304cc790d77278");
 static const uint256 hashGenesisBlockTestNet("0x65b4e101cacf3e1e4f3a9237e3a74ffd1186e595d8b78fa8ea22c21ef5bf9347");

-static const int64 nMaxClockDrift = 2 * 60 * 60;        // two hours - will mod into fork
+static const int64 nMaxClockDrift = 60 * 20;        // 20 minutes

 extern CScript COINBASE_FLAGS;
```

Verge (XVG) hack (2018)

Change clock drift back to old value

master 4.0.2 4.0.2a

Browse files

justinvforvendetta committed on Apr 6 — 1 parent 80c81ae — commit 2b6faff66ecfd8b9361aa5db14ffa5019b784f4f

Showing 1 changed file with 1 addition and 1 deletion.

Unified | Split

2 src/main.h — View

```
@@ -58,7 +58,7 @@ static const int fHaveUPnP = false;
 58  58   static const uint256 hashGenesisBlockOfficial("0x00000fc63692467faeb20cdb3b53200dc601d75bdfa1001463304cc790d77278");
 59  59   static const uint256 hashGenesisBlockTestNet("0x65b4e101cacf3e1e4f3a9237e3a74ffd1186e595d8b78fa8ea22c21ef5bf9347");
 60  60
 61      - static const int64 nMaxClockDrift = 60 * 20;        // 20 minutes
     61  + static const int64 nMaxClockDrift = 2 * 60 * 60;        // 2 hours
 62  62
 63  63   extern CScript COINBASE_FLAGS;
 64  64
```

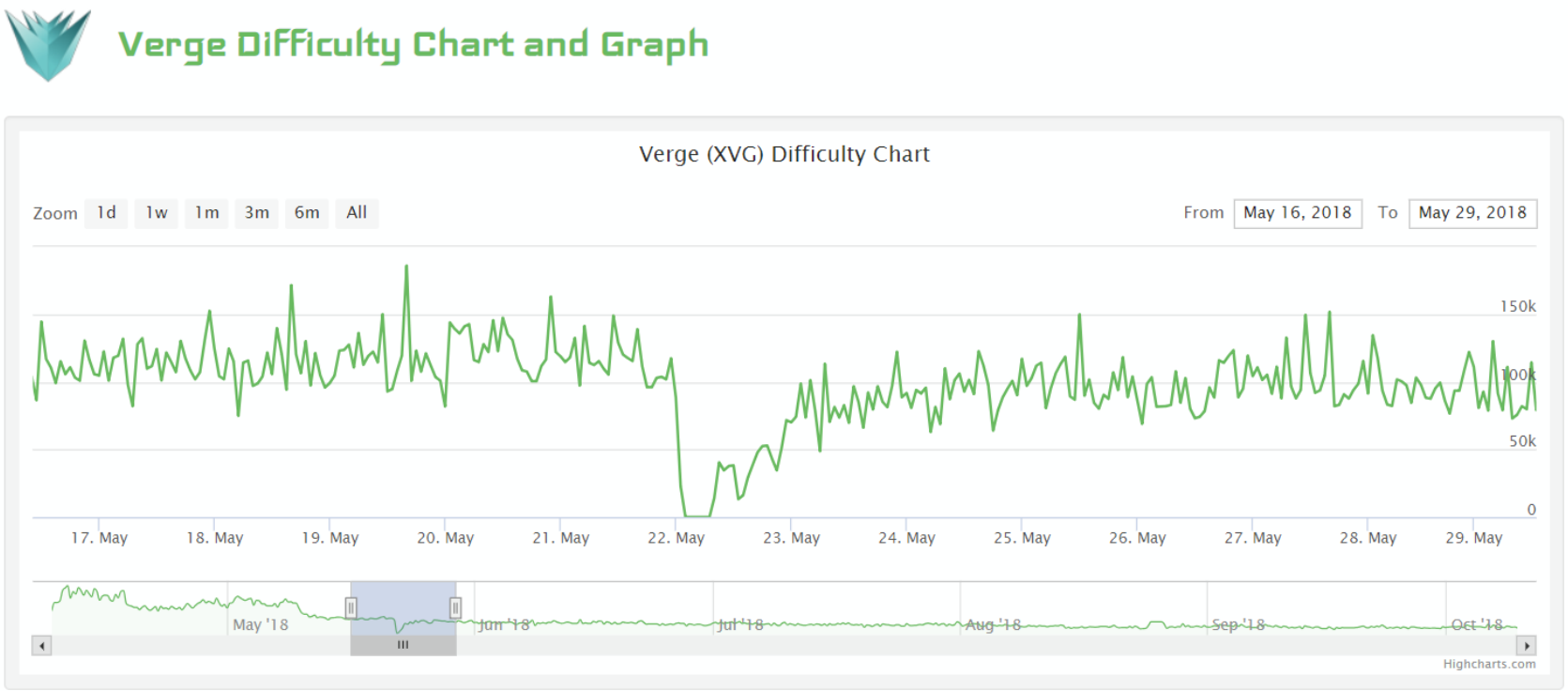
Verge (XVG) hack (2018)

- deweloperzy w ciągu kilku dni wypuścili kilka średnio przemyślanych łat
- a półtora miesiąca później atak został przeprowadzony ponownie, tym razem z użyciem bloków kopanych na zmianę dwoma algorytmami

Verge (XVG) hack (2018)

```
bool CBlock::CheckPrevAlgo(CBlockIndex* pIndex)
{
    unsigned checkedBlocks = 0;
    unsigned sameAlgoBlocks = 0;

    while (pIndex && checkedBlocks != 2*SAME_ALGO_MAX_COUNT)
    {
        if (::GetAlgo(pIndex->nVersion) == GetAlgo())
            ++sameAlgoBlocks;

        ++checkedBlocks;
        pIndex = pIndex->pprev;
    }

    if (sameAlgoBlocks > SAME_ALGO_MAX_COUNT)
        return false;

    return true;
}
```

Verge (XVG) hack (2018)



https://www.coinwarz.com/difficulty-charts/verge-difficulty-chart

Verge (XVG) hack (2018)

- deweloperzy w ciągu kilku dni wypuścili kilka średnio przemyślanych łat
- a półtora miesiąca później atak został przeprowadzony ponownie, tym razem z użyciem bloków kopanych na zmianę dwoma algorytmami
- dopiero po tym ataku kod został poprawiony na stałe i waluta została sforkowana kilka dni później

Verge (XVG) hack (2018)

7 src/main.h View

```
@@ -59,7 +59,12 @@ static const int fHaveUPnP = false;
 59  59   static const uint256 hashGenesisBlockOfficial("0x00000fc63692467faeb20cdb3b53200dc601d75bdfa1001463304cc790d77278");
 60  60   static const uint256 hashGenesisBlockTestNet("0x65b4e101cacf3e1e4f3a9237e3a74ffd1186e595d8b78fa8ea22c21ef5bf9347");
 61  61
 62      - static const int64 nMaxClockDrift = 2 * 60 * 60;        // 2 hours
     62  + inline int64 GetMaxClockDrift(int Height=nBestHeight){
     63  +     if (Height < 2218500)
     64  +         return 2 * 60 * 60; // old 2 hour drift
     65  +     else
     66  +         return 10 * 60; // new 10 min drift
     67  + }
 63  68
 64  69   extern CScript COINBASE_FLAGS;
 65  70
```

Dziękuję za uwagę!

- Źródła:

- https://medium.com/cybertrustbank/block-size-why-does-it-matter-15f76d04ace
- https://masterthecrypto.com/blockchain-scalability-bitcoin-scalability-problem-effects/
- https://en.wikipedia.org/wiki/Proof-of-stake
- https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQs
- https://bitcoin.stackexchange.com/questions/43700/how-does-the-lightning-network-work-in-simple-terms
- https://lightning.network/
- https://bitfalls.com/2018/01/14/curious-case-184-billion-bitcoin/
- https://news.bitcoin.com/critical-bug-found-in-bitcoin-core-invokes-the-multiple-client-argument/
- https://bitcoincore.org/en/2018/09/20/notice/
- https://medium.com/@jimmysong/bitcoin-core-bug-cve-2018-17144-an-analysis-f80d9d373362
- https://www.coindesk.com/bitcoin-bug-exploited-on-crypto-fork-as-attacker-prints-235-million-pigeoncoins/
- https://cointelegraph.com/news/why-bitcoin-gold-got-delisted-from-bittrex
- https://blog.theabacus.io/the-verge-hack-explained-7942f63a3017
- https://blog.theabacus.io/lets-do-the-time-warp-again-the-verge-hack-part-deux-c6396ab36ecb
- https://github.com/vergecurrency/VERGE