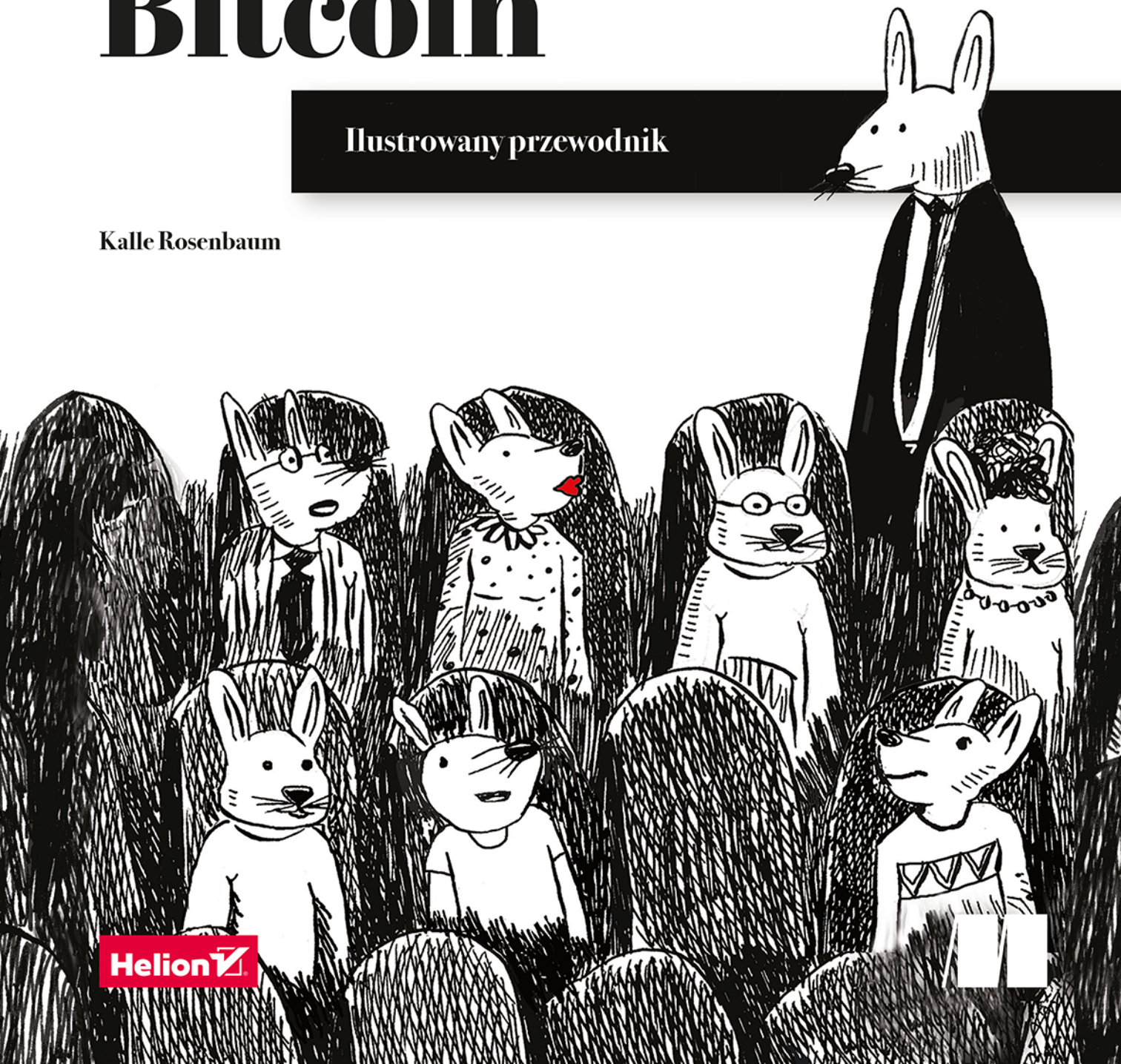
Bitcoin

Ilustrowany przewodnik

Kalle Rosenbaum



Helion

Tytuł oryginału: Grokking Bitcoin

Tłumaczenie: Krzysztof Konatowicz

ISBN: 978-83-283-9303-5

Helion S.A.
ul. Kościuszki 1c, 44-100 Gliwice
tel. 32 231 22 19, 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *https://helion.pl* (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
https://helion.pl/user/opinie/bitilv
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

Spis treści

Kryptograficzne funkcje skrótu i podpisy cyfrowe

2

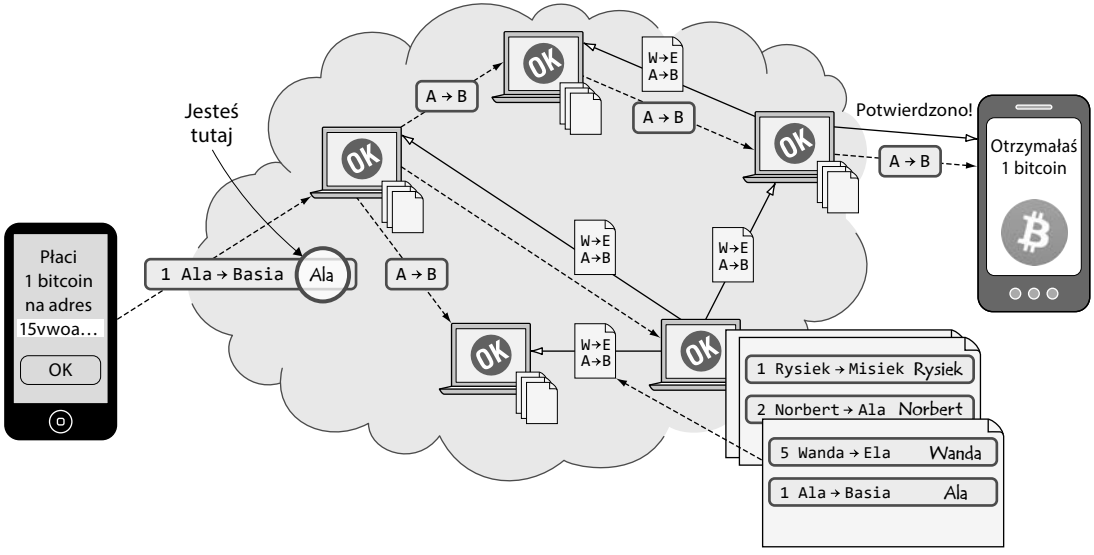
W tym rozdziale:

- Tworzenie prostego systemu płatności: tokenów ciasteczkowych.
- Kryptograficzne funkcje skrótu.
- Uwierzytelnianie płatności za pomocą podpisów cyfrowych.
- Tajność rzeczy tajnych.

Ten rozdział zacznę od opisania sceny, na której rozgrywać się będą dalsze wydarzenia. Zajmiemy się prostym systemem płatności, który będziemy stopniowo rozbudowywać i ulepszać, stosując techniki wykorzystywane przez Bitcoin. Gdy dojdziemy do rozdziału 8., nasz prosty system rozwinie się już na tyle, że stanie się tym, czym właśnie jest Bitcoin.

W dalszej części tego rozdziału przedstawię wszystko, co musisz wiedzieć o kryptograficznych funkcjach skrótu. Ponieważ są one bardzo ważne dla protokołu Bitcoin, musisz zrozumieć je, zanim zaczniemy omawiać kolejne tematy. Dowiesz się, w jaki sposób można wykorzystać kryptograficzną funkcję skrótu do sprawdzenia, czy plik nie zmienił się względem swojej poprzedniej wersji.

W kolejnej części rozdziału będziemy rozważać problem oszusta, czyli kogoś, kto podając się za kogoś innego, chciałby wypłacić sobie cudze środki. Problem ten rozwiążemy, wprowadzając do naszego prostego systemu podpisy cyfrowe (rysunek 2.1).



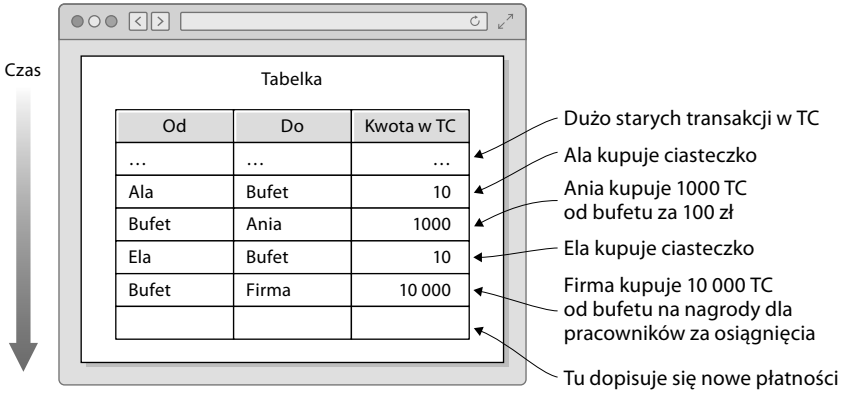
Rysunek 2.1. Podpisy cyfrowe w protokole Bitcoin

Tabelka tokenów ciasteczkowych

Załóżmy, że w Twojej pracy jest bufet. Wraz z innymi pracownikami rozliczacie się w bufecie, korzystając z tabelki zapisanej jako arkusz kalkulacyjny w formacie Excel. Jednostką rozliczeń są *tokeny ciasteczkowe* (rysunek 2.2), które oznaczymy symbolem TC. *Token* w języku angielskim oznacza „żeton", a przymiotnik „ciasteczkowy" wziął się stąd, że w bufecie możesz taki „żeton" wymienić na ciasteczko do kawy.

Waluta Bitcoin

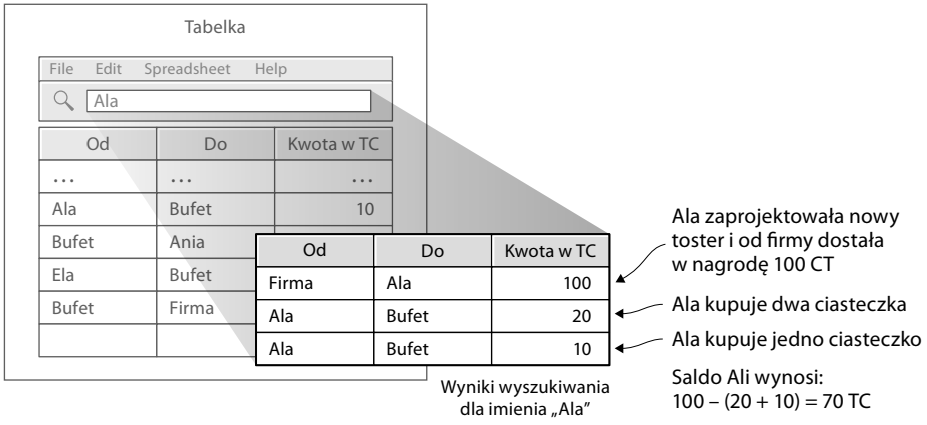
Token ciasteczkowy odpowiada bitcoinowi, jednostce walutowej sieci Bitcoin. Bitcoin po raz pierwszy został wyceniony w 2010 roku, gdy ktoś kupił dwie pizze za 10 000 BTC. W listopadzie 2018 roku za tę kwotę mógłbyś kupić sześć milionów pizz.



Rysunek 2.2. Tabelka tokenów ciasteczkowych ma kolumnę źródła (Od), odbiorcy (Do) oraz kolumnę z liczbą przekazanych tokenów (Kwota w TC). Nowe transakcje w tokenach ciasteczkowych dopisywane są na końcu tabelki

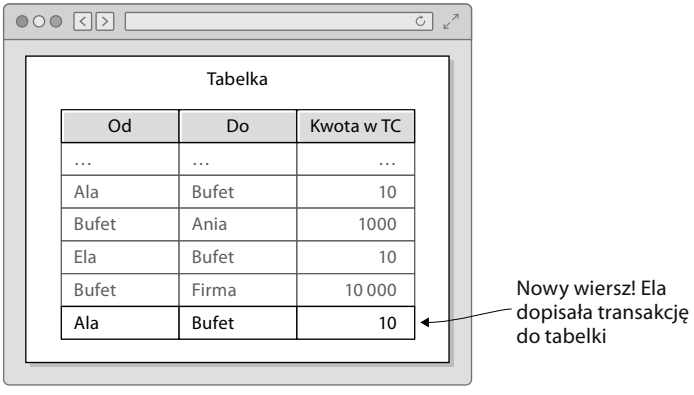
Plik z tabelką jest zapisany na komputerze Eli. Tabelka ta dostępna jest dla wszystkich użytkowników sieci firmowej w trybie tylko do odczytu. Każdy może ją tylko otworzyć i przeczytać jej zawartość. Ela może jednak nieco więcej. Ponieważ Eli ufają wszyscy pracownicy, ma ona nieograniczony dostęp do tabelki. Ty i inni pracownicy możecie jedynie czytać tabelkę, otwierając ją w trybie tylko do odczytu.

Gdy Ala ma ochotę na ciasteczko z bufetu, prosi Elę, która siedzi przy biurku najbliżej bufetu, o przekazanie 10 TC Ali na konto bufetu. Ela zna Alę i może szybko sprawdzić w tabelce, czy Ala ma wystarczająco dużo tokenów na swoim koncie. W tym celu filtruje tabelkę kluczem „Ala" i sumuje wszystkie kwoty wyświetlone dla imienia „Ala" w kolumnie „Do", a od tej sumy odejmuje wszystkie kwoty dla imienia „Ala" w kolumnie „Od". Rysunek 2.3 przedstawia pełny wynik wyszukiwania; trzy transakcje dotyczą Ali.



Rysunek 2.3. Ela oblicza saldo Ali. Suma otrzymanych tokenów ciasteczkowych jest równa 100, a suma wydanych tokenów ciasteczkowych wynosi 30. Saldo Ali to 70 TC

Ela oblicza, że Ala ma 70 TC. To wystarczająco dużo, aby zapłacić 10 TC bufetowi. Ela dopisuje wiersz na końcu tabelki (rysunek 2.4).



Rysunek 2.4. Ela dodaje płatność Ali za ciasteczko. Płatność ta jest dopisywana jako ostatnia w tabelce tokenów ciasteczkowych

Bufetowa widzi nowy wiersz dopisany do tabelki i wręcza Ali ciasteczko.

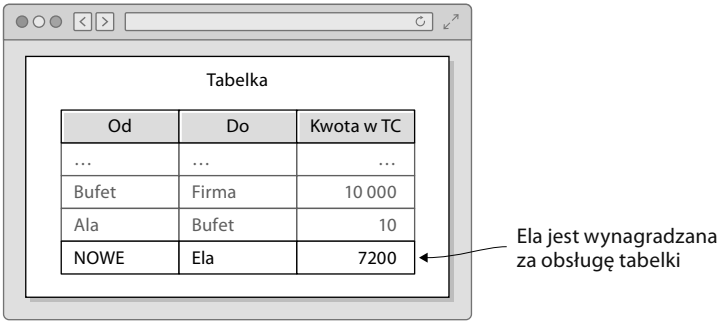
Gdy komuś zabraknie tokenów ciasteczkowych, może dokupić je za złotówki od kogoś, kto ma ich trochę na sprzedaż. Może to być Ania albo bufetowa. Tokeny sprzedawane są za cenę odpowiadającą obu stronom transakcji. Na potwierdzenie zmiany właściciela tokenów Ela dopisuje odpowiedni wiersz do tabelki.

Zarób tokeny

Tokeny możesz również otrzymywać jako wynagrodzenie

Ela obiecała, że nigdy niczego nie usunie ani nie zmieni w tabelce i że będzie wyłącznie dopisywać do niej kolejne wiersze. Wszystko, co działo się w tabelce, będzie zawsze w niej widoczne!

Ela, która dba o ten system rozliczeń, wykonuje pracę, którą cenią wszyscy. Otrzymuje za to codziennie wynagrodzenie w wysokości 7200 świeżo wybitych tokenów ciasteczkowych (rysunek 2.5). Dlatego codziennie dopisuje nowy wiersz do tabelki, w którym 7200 nowych tokenów ciasteczkowych jest przekazywanych „Eli".



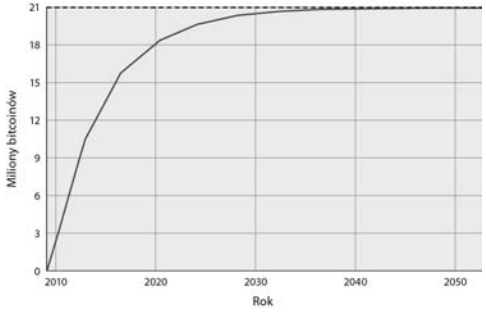
Rysunek 2.5. Ela jest wynagradzana tokenami ciasteczkowymi

W ten sposób w tabelce zapisywane jest utworzenie każdego nowego tokena ciasteczkowego. Pierwszy wiersz w tabelce to wiersz z pierwszym wynagrodzeniem dla Eli — jak w tabelce powyżej. Czyli wiersz ten potwierdza wyemitowanie pierwszych 7200 TC w historii. Wynagrodzenie Eli będzie wynosić 7200 TC dziennie przez pierwsze cztery lata, a następnie zostanie zmniejszone o połowę, do 3600 TC na kolejne cztery lata. Później zostanie ponownie zmniejszone o połowę na następne lata i tak dalej do momentu, aż spadnie do 0 TC dziennie.

Na razie nie martw się o to, co będzie, gdy wynagrodzenie Eli zbliży się do zera. To odległa przyszłość. Zajmiemy się tym w rozdziale 7. Dzięki stopniowemu zmniejszaniu wynagrodzenia całkowita liczba tokenów ciasteczkowych w obiegu będzie zbiegać do 21 milionów TC, ale nigdy nie przekroczy tej liczby.

Krzywa podaży tokenów

Bitcoin używa tego samego schematu emisji tokenów, jakiego użyliśmy w tabelce naszych tokenów ciasteczkowych. Wszystkie nowe bitcoiny są tworzone jako wynagrodzenie dla węzłów zabezpieczających łańcuch bloków sieci Bitcoin, w którym dokumentowane są wszystkie transakcje. To tak samo jak w naszym systemie, gdzie Ela wynagradzana jest za dbanie o bezpieczeństwo tabelki tokenów ciasteczkowych.



Co zrobi Ela z nowymi tokenami ciasteczkowymi, które otrzymuje za prowadzenie tabelki, zależy wyłącznie od niej. Może kupić sobie za nie ciasteczek. Może też nadmiar tokenów komuś sprzedać. Może również odkładać je sobie na przyszłość. System tabelkowy dobrze się spisuje i wszyscy jedzą ciasteczka z umiarem.

Ela tak naprawdę wykonuje tę samą pracę, którą wykonują górnicy w sieci Bitcoin. Weryfikuje płatności i aktualizuje zapisy księgowe, czyli tabelkę tokenów ciasteczkowych. Tabela 2.1 wyjaśnia, jak pojęcia z systemu tabelki tokenów ciasteczkowych mają się do pojęć opisujących sieć Bitcoin.

Tabela 2.1. Powiązanie głównych elementów systemu tokenów ciasteczkowych i sieci Bitcoin

Tokeny ciasteczkowe	Bitcoin	Omówiono w
1 token ciasteczkowy	1 bitcoin	Rozdział 2.
Tabelka	Łańcuch bloków	Rozdział 6.
Wiersz w tabelce	Transakcja	Rozdział 5.
Ela	Górnik	Rozdział 7.

Powyższa tabela będzie pojawiać się często w tej książce. Przedstawia ona różnice pomiędzy systemem tokenów ciasteczkowych a siecią Bitcoin. Różnic i wierszy będzie ubywać w miarę wprowadzania przeze mnie kolejnych koncepcji pochodzących z sieci Bitcoin. Na przykład w rozdziale 6., gdy do zapisu transakcji zacznę używać łańcucha bloków, usunę wiersz „Tabelka". Analogicznie, gdy do naszego systemu tokenów ciasteczkowych będę wprowadzał nowe koncepcje, różniące się od rozwiązań stosowanych przez sieć Bitcoin, będę dopisywał do tej tabeli nowe wiersze.

Na koniec rozdziału 8. tabela ta będzie zawierać już tylko pierwszy wiersz przyporządkowujący jeden token ciasteczkowy jednemu bitcoinowi. Ta chwila wyznaczy koniec ery tokenów ciasteczkowych. Od tego momentu będziemy mówić już wyłącznie o bitcoinach.

Tabela 2.2 jest punktem wyjścia umożliwiającym poznanie zasad działania Bitcoina. Możemy traktować ją jako wersję 1.0 naszego systemu tabelki tokenów ciasteczkowych.

Tabela 2.2. Informacje o zmianach, token ciasteczkowy 1.0

Wersja	Właściwość	Jak
 1.0	Prosty system płatności	Wykorzystuje to, że Eli ufają wszyscy pracownicy, oraz to, że Ela wie, jak każdy z nich wygląda.
	Ograniczona podaż tokenów	Ela otrzymuje codziennie wynagrodzenie w wysokości 7200 nowych TC. Jest ono zmniejszane o połowę co cztery lata.

W kolejnych rozdziałach będziemy rozbudowywać ten system rozliczeń o kolejne zaawansowane rozwiązania, tworząc jego kolejne wersje. Na przykład na koniec tego rozdziału opublikuję wersję 2.0, która aby poradzić sobie z problemem oszustów, będzie wykorzystywać podpisy cyfrowe. Każdy rozdział będzie nas przybliżać do wyniku końcowego, czyli do bitcoina.

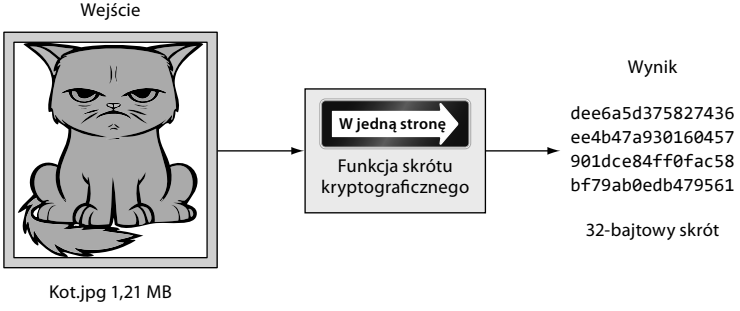
Oczywiście bitcoin nie ewoluował w ten sposób. Wymyślony przeze mnie system rozliczeń w tokenach ciasteczkowych pomaga mi jedynie w wyjaśnianiu kolejnych istotnych zagadnień.

Skróty kryptograficzne

Skróty kryptograficzne są używane w wielu miejscach w protokole Bitcoin. Próba zrozumienia Bitcoina bez znajomości podstaw skrótów kryptograficznych przypominałaby próbę nauki chemii bez wiedzy o budowie atomu.

Skrót kryptograficzny można interpretować jako pewnego rodzaju odcisk palca. Każda osoba, używając powiedzmy lewego kciuka, za każdym razem pozostawi ten sam odcisk palca. Bardzo trudno jest znaleźć kogoś z identycznym odciskiem lewego kciuka. Odcisk palca nie ujawnia też żadnych informacji na temat osoby, która go złożyła, poza samym odciskiem. Chodzi o to, że patrząc na odcisk palca, nikt nie będzie w stanie stwierdzić, jak dobrze osoba składająca go zna matematykę albo jaki ma kolor oczu.

Każda informacja w postaci cyfrowej może również mieć swój „odcisk palca". Nazywamy go **skrótem kryptograficznym**. Aby utworzyć skrót kryptograficzny jakiegoś pliku z danymi, należy przekazać ten plik do programu komputerowego zawierającego funkcję obliczającą wartość skrótu kryptograficznego. Załóżmy, że chcesz utworzyć skrót kryptograficzny, czyli „odcisk palca", swojej ulubionej podobizny kota. Rysunek 2.6 ilustruje ten proces.

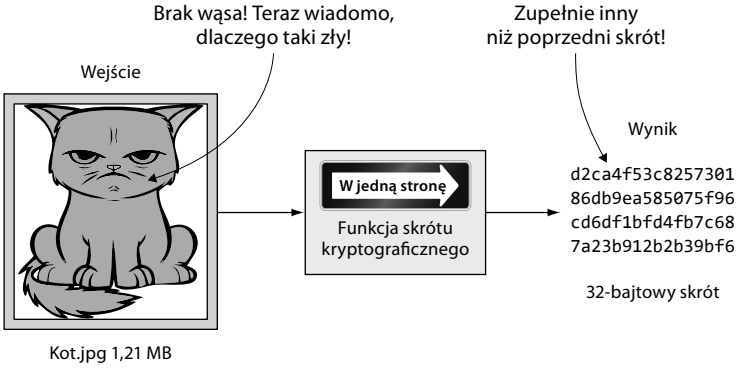


Rysunek 2.6. Tworzenie skrótu kryptograficznego pliku z rysunkiem kota. Dane wejściowe to plik z rysunkiem kota, a dane wynikowe to duża, 32-bajtowa liczba

Wynik — skrót — jest liczbą 256-bitową. 256 bitów to 32 bajty, ponieważ 1 bajt to 8 bitów. Tak więc jeśli zapiszemy tą liczbę w pliku, plik taki będzie miał tylko 32 bajty długości. To niewiele

w porównaniu z rozmiarem pliku z rysunkiem kota wynoszącego aż 1,21 MB. Funkcja skrótu kryptograficznego, której użyjemy w tym przykładzie, nosi nazwę SHA256 (ang. *Secure Hash Algorithm with 256-bit output* — bezpieczny algorytm skrótu z wynikiem 256-bitowym) i jest najczęściej używaną funkcją skrótu w protokole Bitcoin.

Samo słowo „skrót" — w języku angielskim *hash* — oznacza, że coś zostało pocięte na drobne kawałki lub wymieszane. To dość dobry opis działania kryptograficznej funkcji skrótu. Funkcja ta wykonuje obliczenia matematyczne na danych opisujących rysunek kota. Wynikiem jest duża liczba — skrót kryptograficzny, który w żaden sposób nie przypomina kota. Nie jest też możliwe „zrekonstruowanie" obrazu kota na podstawie wartości skrótu. Funkcja skrótu kryptograficznego jest funkcją jednokierunkową. Rysunek 2.7 pokazuje skutek drobnej modyfikacji rysunku kota i obliczenia wyniku tej samej funkcji skrótu kryptograficznego.



Rysunek 2.7. Obliczanie skrótu dla zmodyfikowanego rysunku. Czy widzisz różnicę? Funkcja skrótu ją dostrzegła

Okazuje się, że ten skrót całkowicie różni się od poprzedniego. Porównajmy je:

- stary skrót:

  `dee6a5d375827436ee4b47a930160457901dce84ff0fac58bf79ab0edb479561`

- nowy skrót:

  `d2ca4f53c825730186db9ea585075f96cd6df1bfd4fb7c687a23b912b2b39bf6`

Widać, jak bardzo nawet niewielka zmiana obrazu wpływa na wartość skrótu. Wartość skrótu jest zupełnie inna, ale długość jest zawsze taka sama, niezależnie od danych wejściowych. Jeśli wejściem dla funkcji skrótu będzie słowo „Witaj", to również spowoduje wygenerowanie 256-bitowej wartości skrótu.

Dlaczego kryptograficzne funkcje skrótu są przydatne?

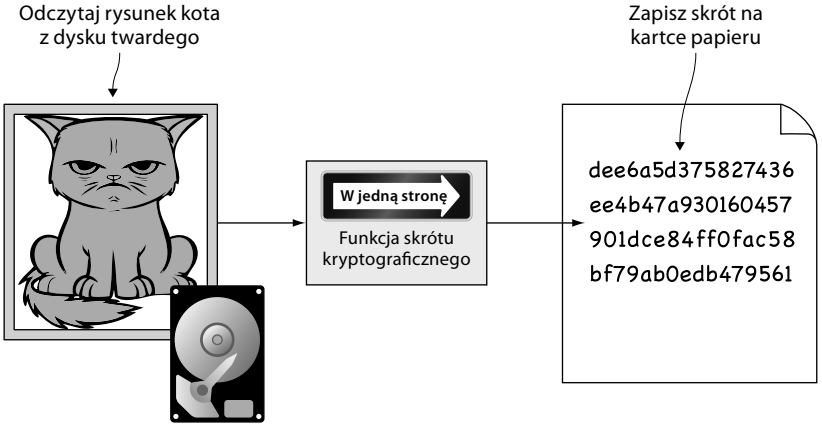
Kryptograficzne funkcje skrótu można wykorzystać do sprawdzenia, czy ktoś nieuprawniony nie zmodyfikował danych. Załóżmy, że chcesz zapisać swój ulubiony rysunek kota na dysku twardym

Bity? Bajty? Hex?

Bit to najmniejsza jednostka informacji w informatyce. Bit może mieć jedną z dwóch wartości: 0 lub 1. To jak żarówka, która może być albo włączona, albo wyłączona. **Bajt** to 8 bitów, które mogą wyrazić 256 różnych wartości. W tej książce często wartości bajtowe przedstawiam w postaci heksadecymalnej lub szesnastkowej (ang. *hexadecimal* lub w skrócie *hex*). Każdy bajt może być opisany przez dwie cyfry szesnastkowe. Każda cyfra szesnastkowa reprezentuje wartość od 0 do 15 — litery, które pojawiają się w tym zapisie, oznaczają wartości wyższe od dziewięciu, np. *a* = 10, a *f* = 15.

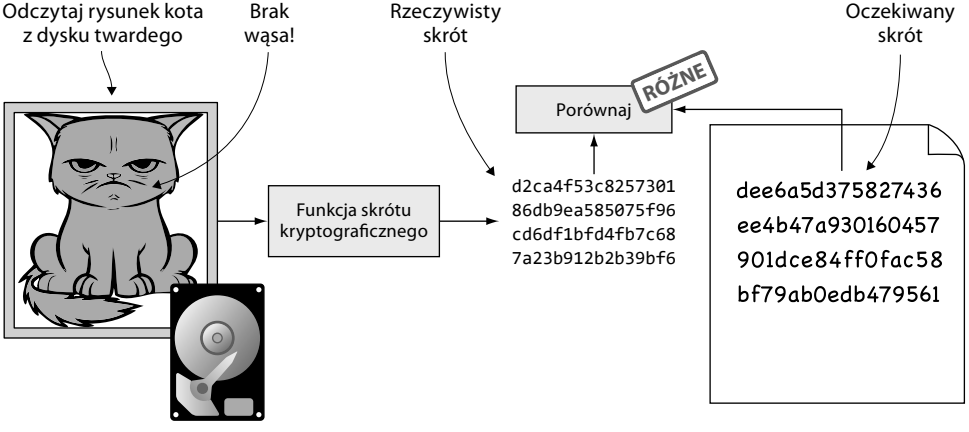
laptopa, ale podejrzewasz, że zapisany rysunek może zostać uszkodzony. Mogłoby do tego dojść na przykład z powodu uszkodzenia napędu dyskowego lub ataku hakerów. Jak jednoznacznie wykryć, czy plik uległ uszkodzeniu?

Trzeba obliczyć kryptograficzny skrót rysunku kota na twardym dysku, a wynik obliczeń zapisać na kartce papieru (rysunek 2.8).



Rysunek 2.8. Zapisz skrót rysunku kota na kartce papieru

Później, gdy będziesz chciał obejrzeć rysunek, będziesz mógł wcześniej sprawdzić, czy uległ on zmianie od czasu, gdy zapisałeś jego skrót na kartce. W tym celu ponownie obliczasz skrót kryptograficzny rysunku kota i porównujesz go ze starym skrótem z kartki (rysunek 2.9).



Rysunek 2.9. Sprawdzanie, czy rysunek kota się nie zmienił. Porównanie wykrywa zmianę

Jeśli nowy skrót jest taki sam jak zapisany na kartce, możesz być pewien, że obraz się nie zmienił. Z drugiej strony, jeśli skróty są różne, to bez wątpienia rysunek kota został zmodyfikowany.

Protokół Bitcoin wykorzystuje kryptograficzne funkcje skrótu do sprawdzania, czy dane nie zostały zmodyfikowane. Na przykład co pewien czas, przeciętnie co 10 minut, tworzony jest nowy skrót całej historii płatności. Próba zmiany tych danych zostanie więc zauważona przez każdego, kto porówna skrót zmodyfikowanych danych ze skrótem sprzed modyfikacji.

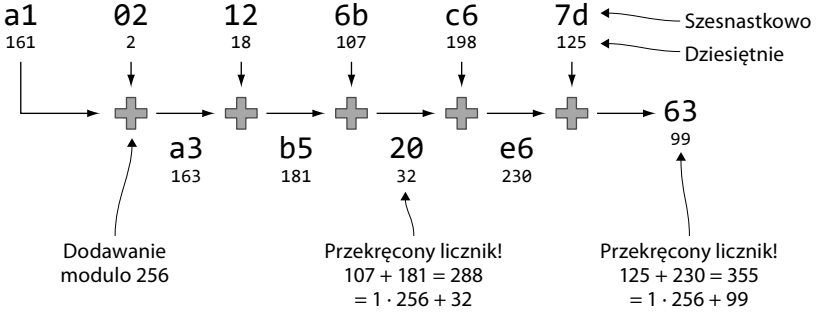
Skąd ta pewność?

Istnieje bardzo małe prawdopodobieństwo, że gdy oba skróty są identyczne, rysunek kota mimo wszystko się zmienił. Ale jak zobaczysz później, to prawdopodobieństwo jest tak małe, że możesz je zignorować.

Jak działa funkcja skrótu kryptograficznego?

Odpowiedź na to pytanie nie jest prosta, dlatego nie będę zagłębiał się w szczegóły. Jednak aby pomóc Ci zrozumieć działanie tej funkcji, przedstawię jej bardzo uproszczoną wersję. Prawdę mówiąc, nie będzie to funkcja w pełni kryptograficzna, ale wyjaśnię to później. Nazwijmy ją na razie funkcją skrótu.

Załóżmy, że chcesz zaszyfrować plik zawierający sześć bajtów o wartościach: `a1 02 12 6b c6 7d`. Chcesz, aby skrót był liczbą 1-bajtową (8 bitów). Możesz skonstruować funkcję skrótu, wykorzystując działanie dodawania modulo 256. W tym działaniu, gdy wynik dodawania przekroczy 255, następuje „przekręcenie licznika" i dalej wartości narastają od 0 (rysunek 2.10).



Rysunek 2.10. Uproszczona funkcja skrótu wykorzystująca bajtowe dodawanie modulo 256

Wynikiem jest liczba dziesiętna 99. Co liczba 99 mówi nam o pierwotnych danych, czyli o `a1 02 12 6b c6 7d`? Niewiele. 99 wygląda dość przypadkowo, jak każda inna liczba jednobajtowa.

Jeśli zmienisz dane wejściowe, zmieni się też skrót, chociaż istnieje pewne prawdopodobieństwo, że nadal będzie miał on wartość 99. W końcu ta prosta funkcja skrótu ma tylko 256 różnych możliwych wartości. W przypadku prawdziwych kryptograficznych funkcji skrótu, takich jak te, których używaliśmy do obliczenia skrótu rysunku kota, to prawdopodobieństwo jest niewyobrażalnie niskie. Niedługo przekonasz się, jak bardzo.

Modulo

Modulo to tak naprawdę reszta z dzielenia, ale działanie tej operacji możemy zilustrować jako przekręcanie licznika po osiągnięciu określonej wartości. Na przykład:

- 0 mod 256 = 0,
- 255 mod 256 = 255,
- 256 mod 256 = 0,
- 257 mod 256 = 1,
- 258 mod 256 = 2.

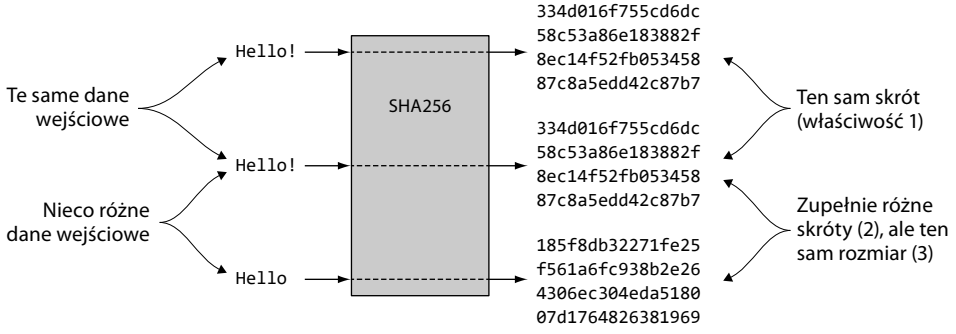
258 mod 256 to reszta z dzielenia całkowitego 258/256: 258 = 1 · 256 + 2. Resztą jest 2.

Właściwości kryptograficznej funkcji skrótu

Kryptograficzna funkcja skrótu dla dowolnych danych wejściowych, zwanych też **obrazem pierwotnym** (ang. *pre-image*), generuje dane wynikowe o stałej długości, zwane **skrótem**. W naszym przykładzie z rysunkiem kota na dysku twardym obrazem pierwotnym jest plik z tym rysunkiem o rozmiarze 1,21 MB, a skrótem jest 256-bitowa liczba. Funkcja generuje dokładnie ten sam skrót dla tego samego obrazu pierwotnego. Ale z bardzo dużym prawdopodobieństwem wygeneruje zupełnie inny skrót, gdy obraz pierwotny zmieni się nawet w najdrobniejszym szczególe.

Zastanówmy się, jakich właściwości można oczekiwać od funkcji skrótu kryptograficznego. Zilustruję to, używając funkcji skrótu SHA256, która najczęściej jest wykorzystywana w protokole Bitcoin. Dostępnych jest kilka funkcji skrótu kryptograficznego, ale wszystkie mają te same podstawowe właściwości:

1. **Te same dane wejściowe zawsze wygenerują ten sam skrót.**
2. **Nawet niewiele różniące się dane wejściowe będą generować skrajnie odmienne wartości skrótu.**
3. **Skrót ma zawsze ten sam, stały rozmiar. W przypadku SHA256 jest to 256 bitów.**
4. **Ataki *brute-force*, czyli metodą prób i błędów, to jedyny znany sposób odtworzenia danych wejściowych na podstawie skrótu.**

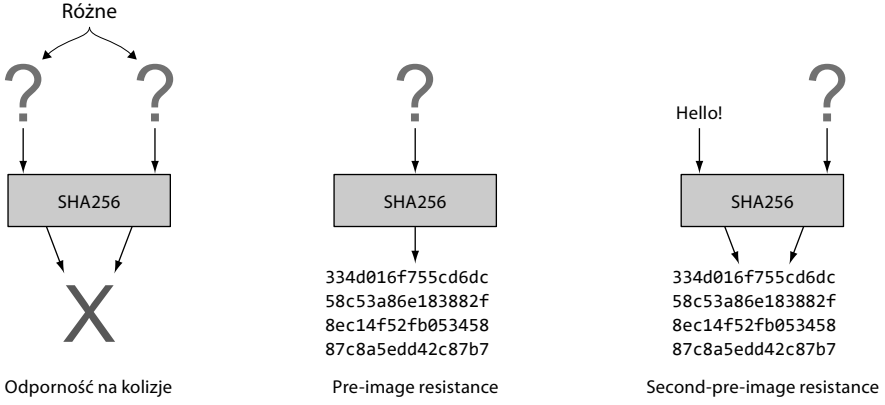


Rysunek 2.11. Kryptograficzna funkcja skrótu SHA256 w działaniu. Dla danych wejściowych „Hello!" daje za każdym razem ten sam skrót, ale nawet niewielka zmiana (na „Hello") daje zupełnie inny skrót

Rysunek 2.11 ilustruje pierwsze trzy właściwości. Czwarta właściwość kryptograficznej funkcji skrótu decyduje o tym, że jest to właśnie funkcja kryptograficzna. Właściwość ta wymaga nieco więcej wyjaśnień. Istnieją pewne warianty czwartej właściwości, z których wszystkie są pożądane dla kryptograficznych funkcji skrótu. Są to (rysunek 2.12):

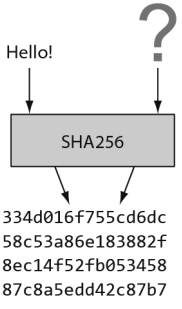
- **Odporność na kolizje.** Znalezienie dwóch *różnych* danych wejściowych, które *generują ten sam skrót*, mając do dyspozycji jedynie funkcję skrótu kryptograficznego, jest trudne.

- ***Pre-image resistance.*** *Odtworzenie obrazu pierwotnego* (*pre-image*) na podstawie posiadanego skrótu i funkcji skrótu jest trudne.
- ***Second-pre-image resistance.*** Przy założeniu, że masz funkcję skrótu i obraz pierwotny (a więc też skrót obrazu pierwotnego), znalezienie *innego obrazu pierwotnego dającego taki sam skrót* jest trudne.



Rysunek 2.12. Pożądane właściwości kryptograficznych funkcji skrótu. W przypadku odporności na kolizje X może być dowolną wartością. Istotne jest to, że dwa różne zbiory danych wejściowych dają ten san wynik X

Co znaczy „trudne”?



Słowo „trudne” w tym kontekście oznacza trudność wręcz niewyobrażalną. Nie ma sensu nawet podejmować prób jej ogarnięcia. Znaczenie słowa „trudne” wyjaśnię na przykładzie właściwości *second-pre-image resistance*, ale podobny przykład można by oprzeć na każdej z trzech.

Załóżmy, że chcesz znaleźć dane wejściowe, które po przekazaniu do funkcji SHA256 spowodują wygenerowanie takiego samego skrótu, jaki generowany jest dla ciągu `Hello!`, czyli:

```
334d016f755cd6dc58c53a86e183882f8ec14f52fb05345887c8a5edd42c87b7
```

Nie możesz zmienić danych wejściowych `Hello!` nawet minimalnie, tak żeby funkcja skrótu tego „nie zauważyła”. *Zawsze* to zauważy i jej wynikiem będzie zupełnie inny skrót. Jedynym sposobem znalezienia wejścia innego niż `Hello!`, dającego taki sam skrót, czyli `334d016f...d42c87b7`, jest sprawdzenie, czy kolejne ciągi danych wejściowych generują pożądany skrót.

Spróbujmy to zrobić, korzystając z tabeli 2.3.

Tabela 2.3. Znalezienie danych wejściowych generujących ten sam skrót co łańcuch Hello! jest praktycznie niemożliwe

Wejście	Skrót	Sukces?
`Hello1!`	`82642dd9...2e366e64`	Nie
`Hello2!`	`493cb8b9...83ba14f8`	Nie
`Hello3!`	`90488e86...64530bae`	Nie
`...`	`...`	Nie, nie… jeszcze raz nie
`Hello9998!`	`cf0bc6de...e6b0caa4`	Nie
`Hello9999!`	`df82680f...ef9bc235`	Nie
`Hello10000!`	`466a7662...ce77859c`	Nie
	`dee6a5d3...db479561`	Nie
Cała moja kolekcja muzyki	`a5bcb2d9...9c143f7a`	Nie

Jak widać, nie udało się nam. Zastanówmy się, ile czasu zajęłoby typowemu komputerowi biurowemu odnalezienie prawidłowych danych wejściowych. Komputer taki może obliczyć około 60 milionów skrótów na sekundę, ale przewidywana liczba prób, zanim takie dane zostaną odnalezione, wynosi 2^{255}. Wynik to $2^{255}/(60 \cdot 10^6)$ s $\approx 10^{68}$ s $\approx 3 \cdot 10^{61}$ lat, czyli około 30 000 000 000 000 000 00 0 000 000 000 000 000 000 000 000 000 000 000 000 000 lat.

Myślę, że możemy na tym zakończyć nasze próby… Nie sądzę też, aby skorzystanie z szybszego komputera wiele tu pomogło. Nawet gdybyśmy mieli do dyspozycji bilion komputerów, które pracowałyby nad tym problemem jednocześnie, to i tak zajęłoby to około $3 \cdot 10^{49}$ lat.

Właściwości *pre-image resistance*, *second-pre-image resistance* oraz odporność na kolizje są niezwykle ważne w protokole Bitcoin. Opiera się na nich większość jego zabezpieczeń.

Czy 2^{256} to dużo?

2^{256} to około 10^{77}, czyli prawie tyle, ile atomów we wszechświecie. Znalezienie obrazu pierwotnego skrótu SHA256 to jak wybranie jednego atomu z całego wszechświata z nadzieją, że będzie to dobry wynik.

Inne dobrze znane funkcje skrótu

Tabela 2.4 przedstawia kilka różnych kryptograficznych funkcji skrótu. Niektóre z nich nie są jednak uznawane za kryptograficznie bezpieczne.

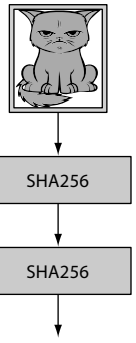
Tabela 2.4. Kilka przykładów kryptograficznych funkcji skrótu. Kilka starszych zostało uznanych za niepewne

Nazwa	Liczba bitów	Nadal bezpieczna?	Wykorzystywana przez Bitcoin?
SHA256	256	Tak	Tak
SHA512	512	Tak	Tak, w niektórych portfelach
RIPEMD160	160	Tak	Tak
SHA-1	160	Nie. Znaleziono kolizję	Nie
MD5	128	Nie. Tworzenie kolizji jest trywialne. Algorytm jest również podatny na ataki pre-image, choć nie w sposób trywialny	Nie

Zasadniczo po znalezieniu choćby jednej kolizji dla kryptograficznej funkcji skrótu większość kryptografów zaczyna uznawać taką funkcję za niebezpieczną.

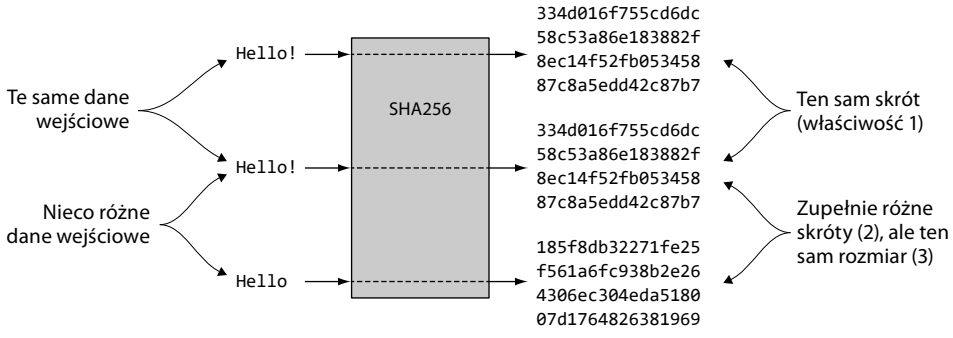
Podwójny skrót SHA256

W protokole Bitcoin najczęściej używamy podwójnego skrótu SHA256.



Podsumowanie informacji o skrótach kryptograficznych

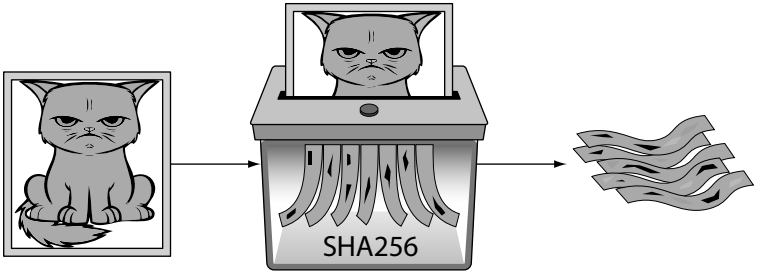
Kryptograficzna funkcja skrótu to algorytm, który na podstawie danych wejściowych oblicza dużą liczbę — skrót kryptograficzny.

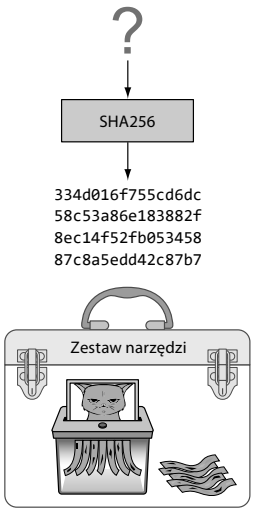


Znalezienie danych wejściowych, których skrót miałby konkretną wartość, jest niewyobrażalnie trudne. Dlatego te funkcje nazywamy funkcjami jednokierunkowymi. Żeby znaleźć takie dane, trzeba wielokrotnie obliczać skróty dla różnych danych wejściowych.

W tej książce będziemy omawiali wiele ważnych kwestii. Po opanowaniu każdego tematu, jak choćby kryptograficznych funkcji skrótu, będziesz mógł dołączyć do swojego zestawu narzędzi

kolejne narzędzia, które wykorzystasz w przyszłości. Twoim pierwszym narzędziem jest właśnie kryptograficzna funkcja skrótu, którą graficznie przedstawia niszczarka dokumentów; skrót kryptograficzny będziemy oznaczać stosem pociętych pasków.





Od teraz będziemy tymi symbolami oznaczać na rysunkach i schematach kryptograficzną funkcję skrótu i skróty kryptograficzne, z niewielkimi wyjątkami.

Ćwiczenia

Rozgrzewka

2.1. Ile bitów ma wynik funkcji SHA256?

2.2. Ile bajtów generuje w wyniku funkcja SHA256?

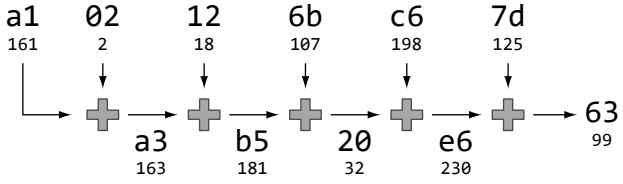
2.3. Co jest potrzebne do obliczenia kryptograficznego skrótu tekstu „skróć mnie”?

2.4. Podaj dziesiętną i binarną reprezentację danych szesnastkowych `061a`.

2.5. Czy da się zmodyfikować słowo „kot” w taki sposób, aby zmodyfikowane słowo generowało taki sam skrót kryptograficzny?

Wyższy poziom

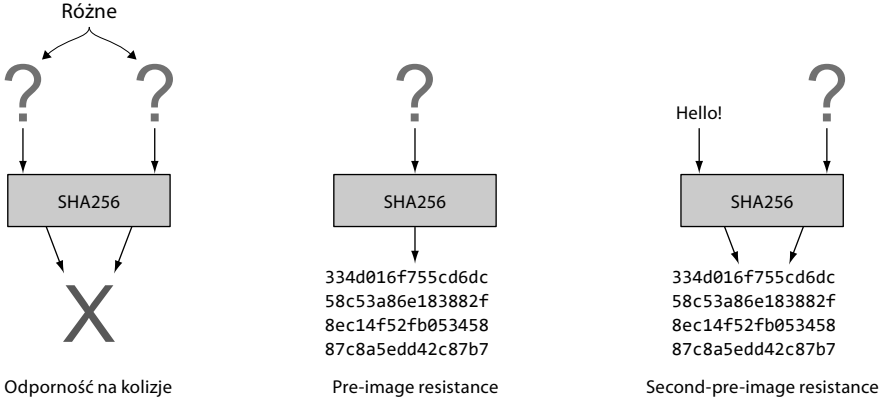
2.6. Uproszczona funkcja skrótu z punktu „Jak działa funkcja skrótu kryptograficznego?”, którą przypominamy poniżej, tak naprawdę nie jest funkcją skrótu kryptograficznego. Której z czterech właściwości kryptograficznej funkcji skrótu jej brak?



Poniżej przypominam też te cztery właściwości:

1. Te same dane wejściowe zawsze generują ten sam skrót.
2. Nawet niewiele różniące się dane wejściowe będą generować skrajnie odmienne wartości skrótu.
3. Skrót ma zawsze ten sam, stały rozmiar. W przypadku SHA256 jest to 256 bitów.
4. Ataki *brute-force*, czyli metodą prób i błędów, to jedyny znany sposób odtworzenia danych wejściowych na podstawie skrótu.

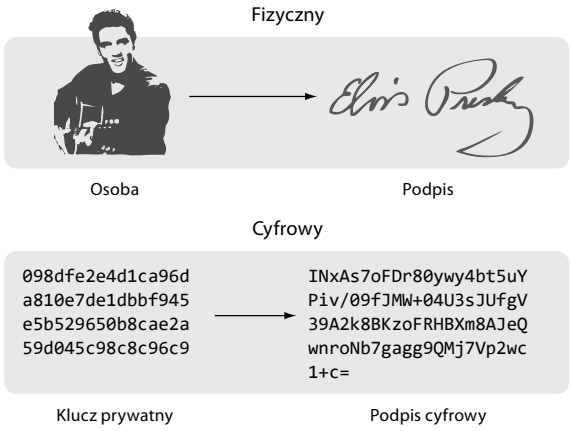
2.7. Powróćmy do przykładu z rysunkiem kota na twardym dysku i jego skrótem zapisanym na kartce papieru. Załóżmy, że ktoś chciał zmodyfikować rysunek kota na Twoim dysku twardym w sposób niezauważalny dla Ciebie. Który wariant czwartej właściwości pokrzyżuje atakującemu plany?



Podpisy cyfrowe

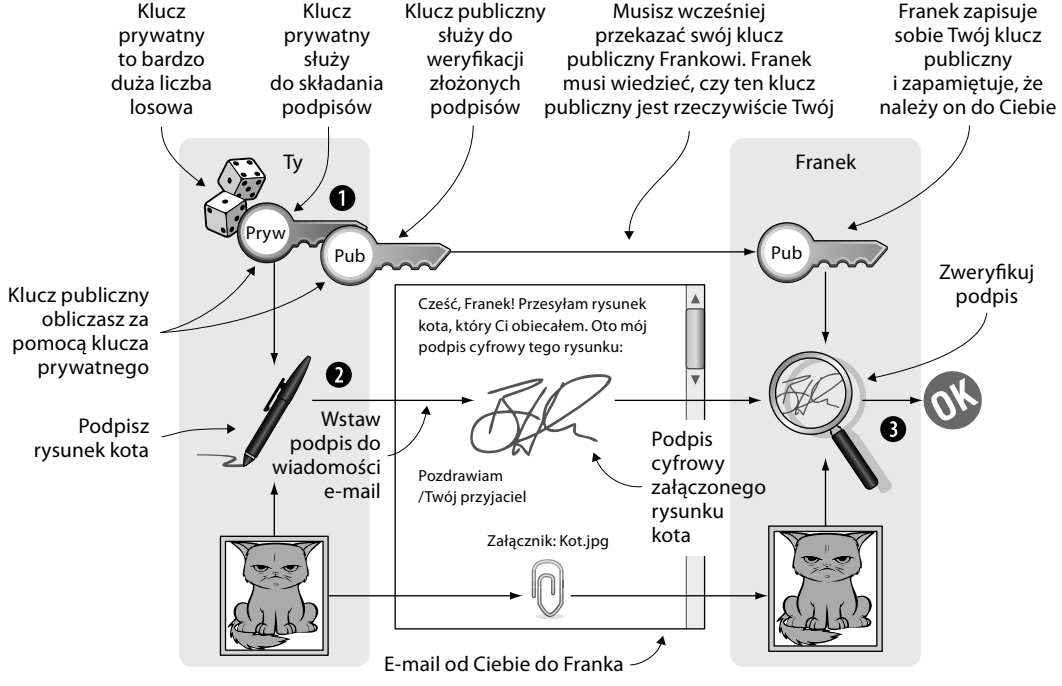
W tym podrozdziale wyjaśnię, w jaki sposób możesz udowodnić komuś, że faktycznie zatwierdzasz daną płatność.

W tym celu wykorzystamy podpisy cyfrowe. **Podpis cyfrowy** to cyfrowy odpowiednik podpisu złożonego odręcznie. Różnica polega na tym, że podpis odręczny jest powiązany z osobą, natomiast podpis cyfrowy jest powiązany z losową liczbą zwaną **kluczem prywatnym**. Podpis cyfrowy jest znacznie trudniejszy do podrobienia niż podpis odręczny.



Typowe zastosowanie podpisów cyfrowych

Załóżmy, że chcesz wysłać swój ulubiony rysunek kota swojemu przyjacielowi, Frankowi, pocztą elektroniczną. Podejrzewasz, że podczas przesyłania rysunek może zostać celowo lub przypadkowo uszkodzony. W jaki sposób Ty i Franek możecie sprawdzić, czy rysunek, który Franek otrzyma, jest dokładnie takim samym rysunkiem jak rysunek, który wysłałeś?



Rysunek 2.13. Wysyłasz Frankowi rysunek kota z podpisem cyfrowym. Franek weryfikuje podpis, aby sprawdzić, czy dostał tego samego kota co podpisany przez Ciebie

Do wiadomości e-mail możesz dołączyć podpis cyfrowy rysunku kota. Franek może następnie zweryfikować ten podpis cyfrowy, aby sprawdzić, czy rysunek kota jest autentyczny. Robisz to w trzech różnych fazach, jak przedstawiono na rysunku 2.13.

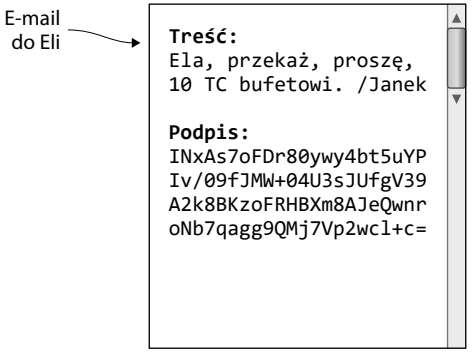
Krok ❶ to *przygotowanie*. Tworzysz dużą liczbę losową, czyli klucz prywatny. Będziesz mógł używać go do składania podpisów cyfrowych. Następnie tworzysz *klucz publiczny*, który będzie służył do weryfikacji podpisów składanych przy użyciu tego klucza prywatnego. Klucz publiczny jest *obliczany* na podstawie klucza prywatnego. Klucz publiczny przekazujesz Frankowi osobiście, dzięki czemu Franek na sto procent wie, że należy on do Ciebie.

Krok ❷ to *podpisywanie*. Piszesz do Franka wiadomość e-mail i dołączasz do niej rysunek kota. Za pomocą klucza prywatnego i rysunku generujesz podpis cyfrowy rysunku kota. Wynikiem jest podpis cyfrowy, który dołączasz do wiadomości e-mail. Następnie wysyłasz wiadomość e-mail Frankowi.

Krok ❸ to *weryfikacja*. Franek otrzymuje wiadomość e-mail, ale ma wątpliwości, czy rysunek kota nie został uszkodzony, dlatego chce zweryfikować podpis. Używa klucza publicznego, który dostał od Ciebie w kroku ❶, podpisu cyfrowego z wiadomości e-mail i załączonego rysunku kota. Jeśli podpis lub rysunek kota zmieniły się od czasu złożenia podpisu, weryfikacja zakończy się niepowodzeniem.

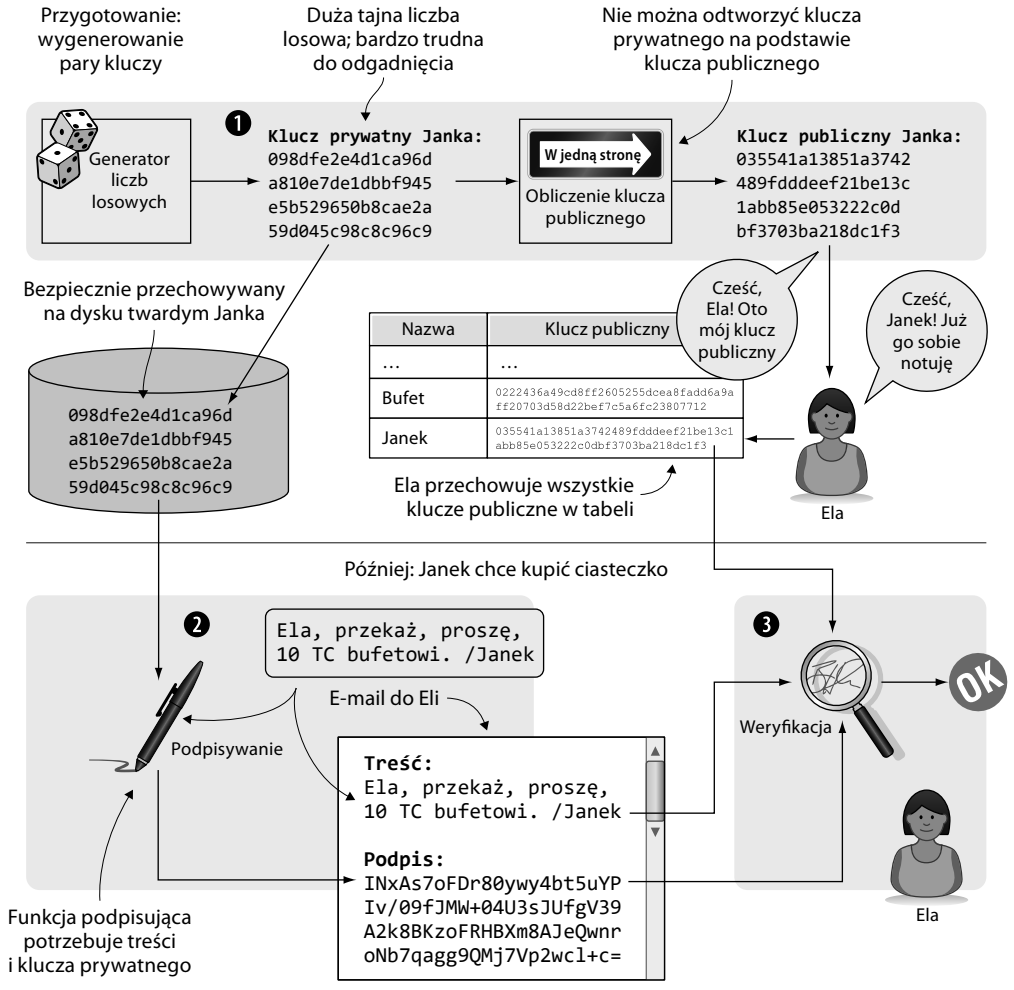
Poprawa bezpieczeństwa tokenów ciasteczkowych

Pora powrócić do naszej tabelki tokenów ciasteczkowych. Firma się rozwija, pracowników przybywa i Ela zaczyna mieć trudności z rozpoznawaniem wszystkich nowych twarzy. Zauważa też, że niektórzy ludzie nie są uczciwi. Na przykład Zdzichu oszukuje Elę, podszywając się pod Annę i prosząc ją o przekazanie tokenów Anny (zamiast swoich) na konto bufetu. Ela zastanawia się, czy nie poprosić wszystkich o potwierdzanie dyspozycji przekazania tokenów ciasteczkowych e-mailem z podpisem cyfrowym, jak pokazano na rysunku 2.14.



Rysunek 2.14. Janek musi podpisać cyfrowo swoją prośbę o przekazanie tokenów i dołączyć podpis do wiadomości e-mail

Załóżmy, że Janek jest nowym pracownikiem. Na początek dostał od firmy kilka tokenów ciasteczkowych jako prezent powitalny. Teraz Janek chce kupić ciastko w bufecie za 10 TC. Musi podpisać cyfrowo dyspozycję przekazania tokenów. Rysunek 2.15 pokazuje, co musi zrobić.



Rysunek 2.15. Proces składania podpisu cyfrowego. ❶ Janek tworzy parę kluczy i przekazuje klucz publiczny Eli. ❷ Janek podpisuje wiadomość za pomocą swojego klucza prywatnego. ❸ Ela sprawdza, czy wiadomość została podpisana kluczem prywatnym powiązanym z kluczem publicznym, który otrzymała od Janka

Podobnie jak w przypadku wiadomości e-mail wysłanej do Franka w poprzednim punkcie, proces ten składa się z trzech faz (znajdź podobieństwa do kroków z rysunku 2.13):

❶ Janek przygotowuje się, generując parę kluczy. Swój klucz prywatny przechowuje w bezpiecznym miejscu i przekazuje klucz publiczny Eli. Ten krok przygotowań Janek musi wykonać tylko jeden raz.

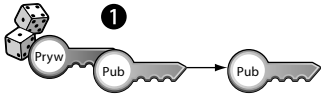
❷ Janek chce kupić ciasteczko. Pisze e-mail i podpisuje go swoim kluczem prywatnym. Wysyła e-mail z załączonym podpisem cyfrowym do Eli.

❸ Ela weryfikuje podpis wiadomości e-mail za pomocą klucza publicznego Janka i aktualizuje tabelkę.

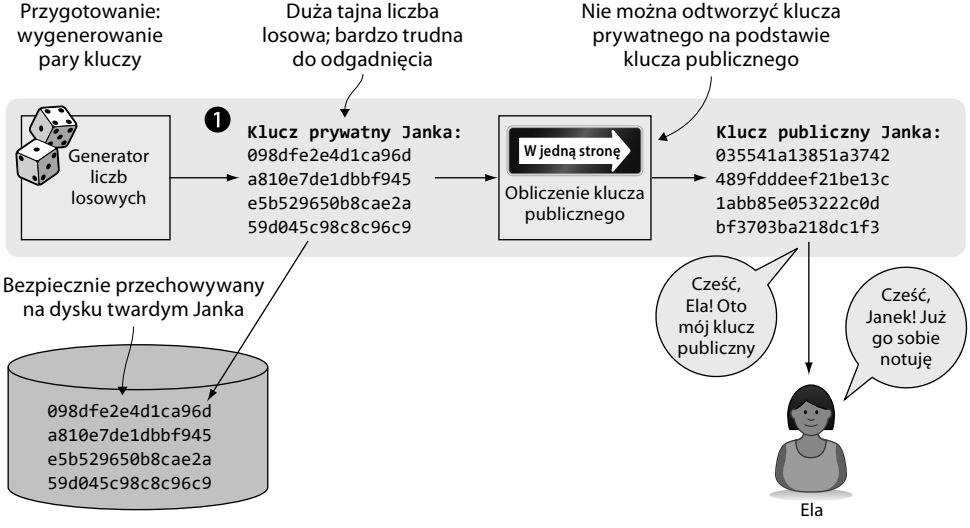
Ponowne użycie pary kluczy

Para kluczy tworzona jest tylko raz. Ten sam klucz prywatny można wykorzystać wiele razy do podpisywania cyfrowego różnych rzeczy.

Przygotowanie: Janek generuje parę kluczy



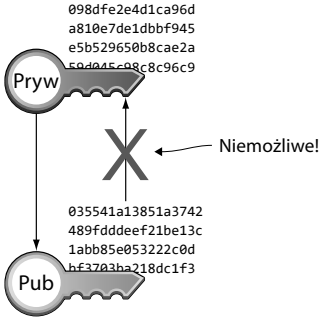
Procesy podpisywania i weryfikacji wykorzystują parę kluczy. Janek potrzebuje klucza prywatnego do podpisywania płatności, a Ela będzie potrzebować klucza publicznego Janka do weryfikacji składanych przez niego podpisów. Janek musi się przygotować i wygenerować najpierw parę kluczy. Robi to, generując klucz prywatny, a następnie generując na jego podstawie klucz publiczny, jak pokazano na rysunku 2.16.



Rysunek 2.16. Janek tworzy parę kluczy. Klucz prywatny jest dużą liczbą losową, a klucz publiczny jest generowany na jego podstawie. Janek przechowuje swój klucz prywatny na dysku twardym, a klucz publiczny przekazuje Eli

Do wygenerowania dużej (256-bitowej) liczby losowej Janek używa generatora liczb losowych. Generator liczb losowych dostępny jest w prawie każdym systemie operacyjnym. Wylosowana liczba staje się kluczem prywatnym Janka. Następnie za pomocą odpowiedniej funkcji z klucza prywatnego otrzymywany jest klucz publiczny.

Funkcja generująca klucz publiczny jest funkcją jednokierunkową, podobnie jak kryptograficzne funkcje skrótu, czyli nie ma możliwości odtworzenia klucza prywatnego na podstawie klucza publicznego. Bezpieczeństwo podpisów cyfrowych opiera się w dużej mierze na bezpieczeństwie tej funkcji. Również kolejne obliczenie klucza publicznego z klucza prywatnego za pomocą tej funkcji zawsze da w wyniku ten sam klucz publiczny.

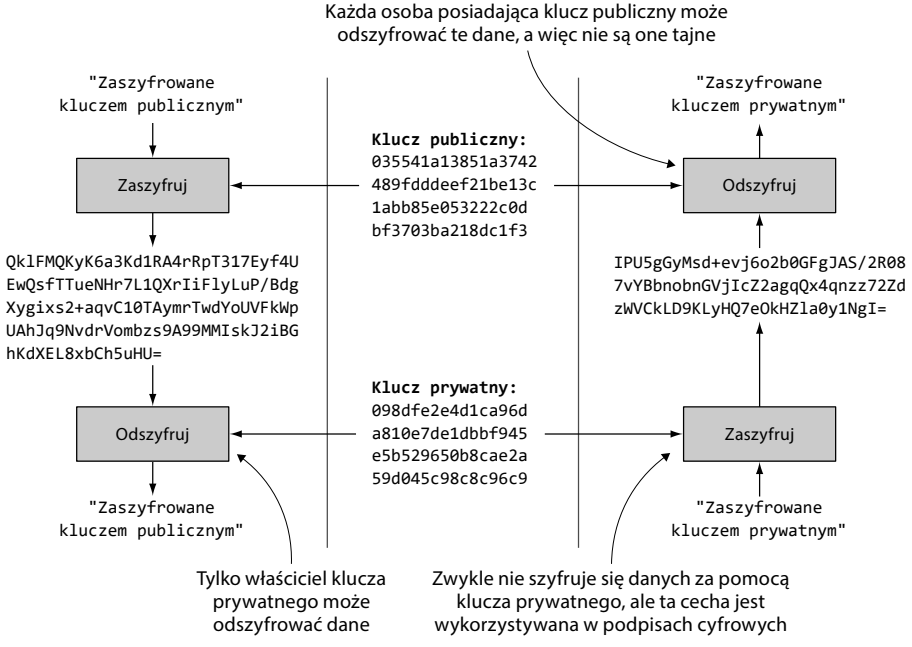


Klucz publiczny ma długość 33 bajtów (66 cyfr szesnastkowych). Jest on dłuższy niż klucz prywatny, który ma 32 bajty (64 znaki w kodzie szesnastkowym). Skąd wziął się ten „dodatkowy" bajt i w jaki sposób działa funkcja obliczająca klucz publiczny, opiszę w rozdziale 4. Na szczęście aby zrozumieć, jak działają podpisy z perspektywy użytkownika, nie trzeba być ekspertem w dziedzinie kryptografii.

Dwa sposoby użycia pary kluczy

Klucze służą do szyfrowania i deszyfrowania danych. Szyfrowanie przekształca dane do postaci niezrozumiałej dla wszystkich, z wyjątkiem posiadaczy odpowiedniego klucza umożliwiającego ich odszyfrowanie.

Klucze prywatny i publiczny powinniśmy traktować, jako parę, ponieważ są ze sobą silnie powiązane. Klucz publiczny może zostać użyty do zaszyfrowania danych, które można odszyfrować tylko kluczem prywatnym. Natomiast kluczem prywatnym można zaszyfrować dane, które można odszyfrować tylko kluczem publicznym (rysunek 2.17).



Rysunek 2.17. Szyfrowanie i deszyfrowanie za pomocą kluczy publicznego i prywatnego. Po lewej: szyfrowanie za pomocą klucza publicznego i odszyfrowywanie za pomocą klucza prywatnego. Po prawej: szyfrowanie za pomocą klucza prywatnego i odszyfrowywanie za pomocą klucza publicznego

Wskazówka

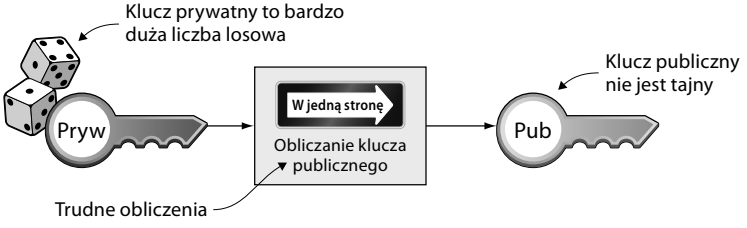
Do tworzenia podpisów cyfrowych używać będziemy prawej strony schematu z rysunku 2.17. Lewej strony nie używamy nigdzie w tej książce.

Przeanalizujmy schemat z rysunku 2.17 od lewej strony — tylko Janek może odczytać zaszyfrowane dane, ponieważ tylko on ma dostęp do swojego klucza prywatnego. Bitcoin w ogóle nie wykorzystuje tej właściwości kluczy publicznych i prywatnych. Jest ona używana, gdy dwie strony chcą komunikować się ze sobą poufnie, jak na przykład w bankowości internetowej. Gdy widzisz symbol kłódki na pasku adresu przeglądarki, oznacza to, że do zabezpieczenia komunikacji użyto procesu przedstawionego po lewej stronie schematu.

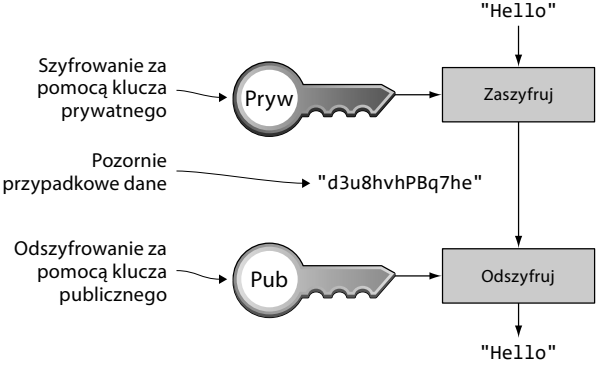
Zgodnie z prawą stronę schematu Ela może odszyfrować dane, ponieważ ma klucz publiczny powiązany z kluczem prywatnym Janka. Ta właściwość jest wykorzystywana w podpisach cyfrowych. Użycie klucza prywatnego do szyfrowania tajnych danych nie jest dobrym pomysłem, ponieważ klucz publiczny jest, no cóż, dostępny publicznie. Każdy więc, kto ma do niego dostęp, może odszyfrować dane, a więc dane takie nie będą tajne. Z drugiej strony w podpisach cyfrowych nie szyfrujemy żadnych tajnych danych. Podpisom cyfrowym przyjrzymy się bliżej już niedługo. Ale najpierw podsumujmy to, czego dowiedziałeś się do tej pory.

Pary kluczy — podsumowanie

Podsumujmy to, czego dowiedziałeś się o kluczach publicznych i prywatnych. Tworzysz parę kluczy — najpierw klucz prywatny. Klucz prywatny to bardzo duża liczba losowa. Klucz publiczny jest obliczany na podstawie klucza prywatnego.



Za pomocą klucza prywatnego możesz zaszyfrować dane, które można odszyfrować tylko przy użyciu klucza publicznego.

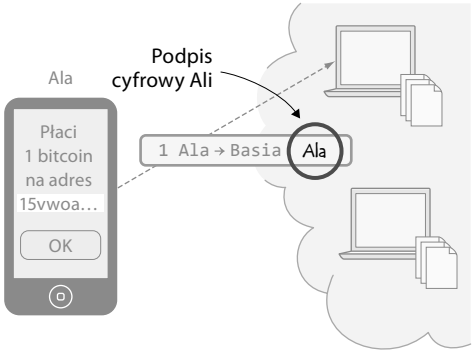


Procesy szyfrowania i odszyfrowywania przedstawione na tym rysunku są fundamentem podpisów cyfrowych. Procesy te nie nadają się do przesyłania tajnych danych, ponieważ klucz publiczny jest zwykle powszechnie znany.

Proces odwrotny jest również powszechnie stosowany. Do szyfrowania wykorzystuje się w nim klucz publiczny, a klucz prywatny do odszyfrowywania. Proces ten służy do przesyłania tajnych danych. Ale w sieci Bitcoin nie jest on wykorzystywany.

Gdzie to byliśmy?

Podpisy cyfrowe wspomniałem krótko w rozdziale 1., w którym Ala podpisała za pomocą swojego klucza prywatnego swoją transakcję przekazania w sieci Bitcoin 1 BTC Basi (rysunek 2.18).

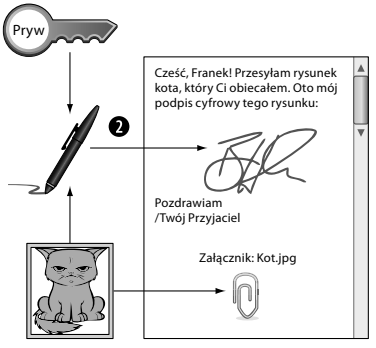


Rysunek 2.18. Podpisy cyfrowe w protokole Bitcoin

Janek utworzył parę kluczy i ma zamiar podpisać transakcję przekazującą jego tokeny bufetowi swoim kluczem prywatnym, aby Ela wiedziała, że to właśnie on przekazał te środki. Ela zweryfikuje to, używając klucza publicznego Janka.

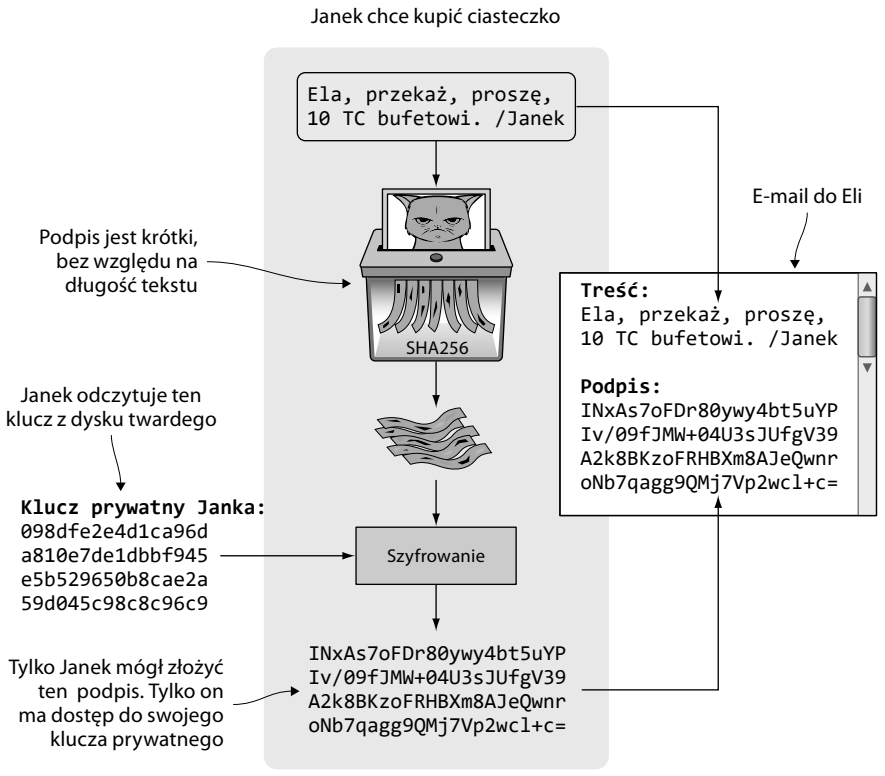
Janek podpisuje swoją płatność

Przyjrzyjmy się dokładnie, jak naprawdę przebiega składanie podpisu (rysunek 2.19).



Danymi, które Janek chce podpisać, jest ciąg znaków: „Ela, przekaż, proszę, 10 TC bufetowi. /Janek". Funkcja podpisywania obliczy skrót SHA256 tego tekstu, który będzie miał postać liczby 256-bitowej. Wartość skrótu zostanie następnie zaszyfrowana kluczem prywatnym Janka. Wynikiem będzie podpis cyfrowy mający postać:

```
INxAs7oFDr80ywy4bt5uYPIv/09fJMW+04U3sJUfgV39A2k8BKzo
FRHBXm8AJeQwnroNb7qagg9QMj7Vp2wcl+c=
```



Rysunek 2.19. Janek podpisuje cyfrowo dyspozycję przekazania 10 TC bufetowi. Wiadomość do Eli zostaje najpierw skrócona, a następnie zaszyfrowana za pomocą klucza prywatnego Janka. Wiadomość e-mail do Eli zawiera zarówno treść w postaci czystego tekstu, jak i jej podpis

Podpis jest zaszyfrowanym skrótem wiadomości. Gdyby Janek użył innego klucza prywatnego do złożenia podpisu lub nieco innej treści wiadomości, jej podpis wyglądałby zupełnie inaczej.

Na przykład zaszyfrowanie treści „Ela, przekaż, proszę, 10 TC Zdziśkowi. /Janek" wygenerowałoby taki podpis:

```
ILDtL+AVMmOrcrvCRwnsJUJUtzedNkSoLb7OLRoH2iaDG1f2WX1dAOTYkszR1z
0TfTVIVwdAlD0W7B2hBTAzFkk=
```

Ten podpis zupełnie nie przypomina poprzedniego. To dobra wiadomość dla Janka, ponieważ wie on, że jego podpisu nie da się użyć do innej treści wiadomości poza tą, którą sam utworzył.

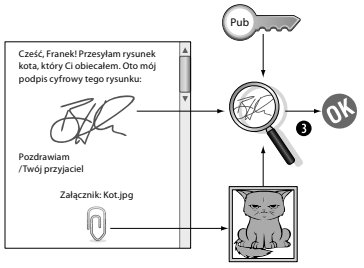
Janek przygotował więc e-mail dla Eli. Składa się on treści oraz jej podpisu. Na koniec Janek wysyła e-mail do Eli.

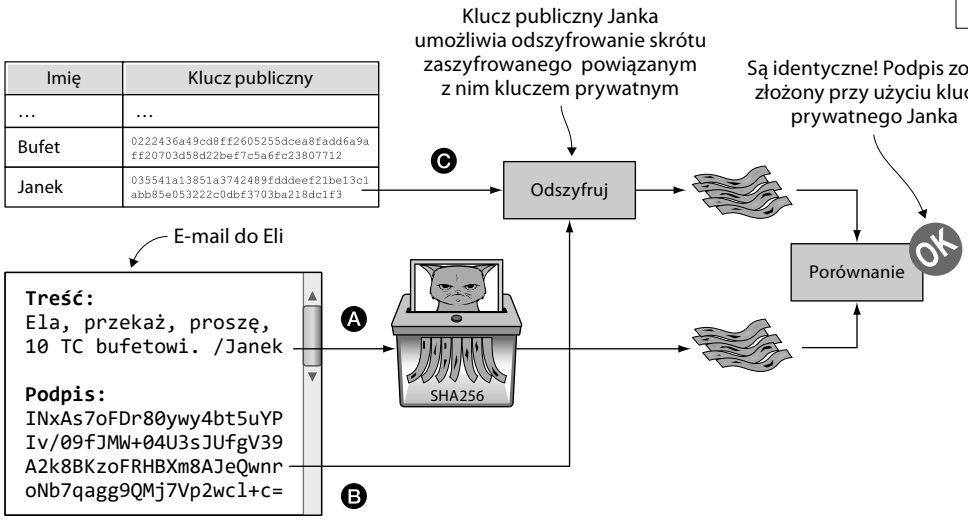
Podpisy w sieci Bitcoin

Sieć Bitcoin wykorzystuje opisany rodzaj podpisu w większości dzisiejszych transakcji, ale nie jest to jedyna metoda uwierzytelnienia płatności.

Ela sprawdza podpis

Ela widzi e-mail, który rzekomo został nadesłany jej przez Janka. Wyszukuje więc Janka w swojej tabeli kluczy publicznych (rysunek 2.20).





Rysunek 2.20. Za pomocą podpisu (B) i klucza publicznego (C) Janka Ela sprawdza, czy treść wiadomości (A) została podpisana kluczem prywatnym Janka

Działania Eli przedstawione na tym rysunku mają na celu ustalenie, czy dyspozycja przekazania tokenów ciasteczkowych została podpisana kluczem prywatnym osoby, która twierdzi, że ten podpis złożyła. Z nagłówka e-maila *wynika*, że pochodzi on od Janka. Ela otrzymała kiedyś klucz publiczny Janka i wpisała go do swojej tabeli kluczy publicznych. A więc ma ona do dyspozycji:

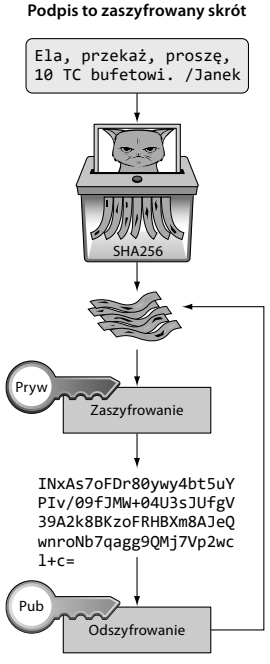
Ⓐ Treść wiadomości: „Ela, przekaż, proszę, 10 TC bufetowi. /Janek”.

Ⓑ Podpis: `INxAs7oFDr8....`

Ⓒ Klucz publiczny Janka, który właśnie odszukała w swojej tabelce.

Janek zaszyfrował skrót treści wiadomości, używając swojego *klucza prywatnego*. Ten zaszyfrowany skrót jest podpisem. Jeśli Ela odszyfruje ten podpis Ⓑ *kluczem publicznym* Janka Ⓒ, wynikiem powinien być skrót, który powinien być taki sam jak skrót treści Ⓐ z e-maila.

Ela bierze podpis Ⓑ i odszyfrowuje go, używając klucza publicznego Ⓒ, który znalazła w swojej tabeli kluczy publicznych. Po odszyfrowaniu otrzymuje dużą liczbę. Jeśli liczba ta jest taka sama jak skrót treści wiadomości Ⓐ, potwierdza to, że do podpisania wiadomości użyto klucza prywatnego Janka. Ela bierze więc dokładną treść e-maila Ⓐ i oblicza jej skrót



dokładnie tak, jak zrobił to Janek, kiedy tworzył podpis. Ten skrót treści wiadomości porównuje następnie z odszyfrowanym podpisem. Skrót wiadomości i odszyfrowany podpis są takie same, co oznacza, że podpis jest poprawny.

Zwróć uwagę, że proces ten zadziała tylko wtedy, gdy Janek i Ela użyją dokładnie tego samego schematu podpisu cyfrowego. Muszą to wcześniej uzgodnić między sobą, ale z reguły jest to w jakiś sposób unormowane. W sieci Bitcoin każdy dokładnie wie, jakiego schematu podpisu cyfrowego ma użyć.

Ela może być teraz pewna, że nikt nie próbował jej oszukać. Aktualizuje tabelkę o dyspozycję przekazania tokenów Janka, jak pokazano na rysunku 2.21.

Od	Do	Kwota w TC
…	…	…
Firma	Janek	100
Janek	Bufet	10

Nowy wiersz! Janek podpisał dyspozycję przekazania środków, a Ela zweryfikowała ją. Ela sprawdza saldo Janka i dopisuje dyspozycję do tabelki

Rysunek 2.21. Po sprawdzeniu podpisu wiadomości otrzymanej od Janka Ela dodała wiersz z dyspozycją przekazania jego tokenów

Bezpieczeństwo klucza prywatnego

Janek ma kontrolę nad swoimi tokenami ciasteczkowymi, ponieważ posiada swój klucz prywatny. Nikt oprócz Janka nie może użyć jego tokenów, ponieważ jest on jedyną osobą mającą dostęp do tego klucza. Jeśli klucz prywatny Janka zostanie wykradziony, Janek może stracić wszystkie swoje tokeny.

Następnego ranka po zleceniu dyspozycji przelewu Janek przychodzi do biura, bierze laptop z biurka i idzie prosto do bufetu, aby kupić dwa ciasteczka do kawki na śniadanie. Otwiera laptop, aby napisać e-mail do Eli:

> Dzień dobry, Elu! Przekaż, proszę, 20 TC bufetowi. /Janek
>
> Podpis:
>
> `H1CdE34cRuJDsHo5VnpvKqllC5JrMJ1jWcUjL2VjPbsjX6pi/up07q/gWx`
> `Stb1biGU2fjcKpT4DIxlNd2da9x0o=`

Wysyła e-mail z tą treścią i z tym podpisem do Eli. Ale bufetowa nie chce mu wydać ciasteczek! Mówi, że nie dotarło do niej jeszcze tych 20 tokenów. Ela zwykle szybko reaguje na e-maile i wykonuje przelewy.

Janek otwiera tabelkę Eli — pamiętaj, że może to zrobić w trybie tylko do odczytu — i wyszukuje swoje imię. Rysunek 2.22 przedstawia, co widzi Janek.

Od	Do	Kwota w TC
…	…	…
Firma	Janek	100
Janek	Bufet	10
Janek	Zdzisiek	90

Saldo 100 TC

To za wczorajsze ciasteczko. Saldo wynosi 90 TC

Co? Janek NIE przesłał 90 TC Zdziśkowi!

Rysunek 2.22. Ktoś ukradł Jankowi środki. Kim w ogóle jest Zdzisiek i jak to się mogło stać? Janek nie podpisał takiej dyspozycji!

Janek idzie do pokoju Eli i prosi ją o wyjaśnienia. Ela mówi mu, że dostała wiadomość podpisaną prywatnym kluczem Janka z prośbą o przekazanie środków nowemu pracownikowi, Zdzisławowi. Ela pokazuje mu nawet e-mail z tą prośbą i podpis. Oczywiście w biurze nie pracuje żaden Zdzisiek, chociaż w firmie podjęło pracę kilku nowych pracowników. Eli nie interesują już imiona, tylko klucze publiczne i podpisy. Imienia potrzebuje tylko po to, by wyszukać klucz publiczny w tabelce kluczy.

Czyli sytuację tę można wyjaśnić następująco:

1. Zdzisiek zdołał w jakiś sposób skopiować klucz prywatny Janka. Laptop Janka leżał na biurku całą noc. Ktoś mógł wyjąć z niego dysk twardy i wyszukać na nim zapisany klucz prywatny Janka.
2. Ktoś mógł utworzyć nową parę kluczy i wysłać nowy klucz publiczny do Eli w e-mailu o następującej treści:

   Witaj, Ela! Nazywam się Zdzisław i właśnie zacząłem u was pracę. Mój klucz publiczny to:

   ```
   02c5d2dd24ad71f89bfd99b9c2132f796fa746596a06f5a33c53c9d762e37d9008
   ```
3. Następnie mógł wysłać następującą fałszywą wiadomość do Eli, podpisaną skradzionym kluczem prywatnym:

   Cześć, Ela! Przekaż 90 TC Zdziśkowi. Dzięki, Janek

   Podpis:

   ```
   IPSq8z0IyCVZNZNMIgrOz5CNRRtRO+A8Tc3j9og4pWbAH/zT22dQEhSaFSwOXNp0lOy
   E34d1+4e30R86qzEbJIw=
   ```

Ela sprawdziła dyspozycję z kroku 3., stwierdziła, że była poprawna i ważna, i przekazała tokeny Zdziśkowi. Janek prosi Elę o odwołanie tego — jego zdaniem — oszukańczego przelewu. Ale Ela odmawia. Uważa, że przelew był poprawny. Jeśli Janek pozwolił komuś podejrzeć swój klucz prywatny, to jego problem, a nie Eli. Ela robi to, do czego się zobowiązała, i tylko dlatego, że zawsze dotrzymuje obietnic, wszyscy w firmie jej ufają.

Janek tworzy nową parę kluczy i prosi Elę o dodanie jego nowego klucza publicznego pod nazwą Janek2. W jaki sposób Janek może zabezpieczyć swój nowy klucz prywatny, wciąż mając do niego łatwy dostęp, gdy przyjdzie mu ochota na ciasteczka? Janek zakłada, że nie będzie przechowywał więcej niż 1000 tokenów ciasteczkowych na tym kluczu.

Twoja odpowiedzialność

To Ty ponosisz pełną odpowiedzialność za bezpieczeństwo kluczy prywatnych.

Bezpieczeństwo tabelki zmieniło się. Nie jest już systemem, w którym Ela wie, jak wygląda każdy pracownik. Teraz jest to system, w którym Ela zna klucz publiczny każdego pracownika. W pewnym sensie bezpieczeństwo może być w tym momencie niższe, ponieważ Zdziśkowi łatwiej ukraść klucz prywatny Janka, niż wyprowadzić Elę w pole, podszywając się pod Janka. Bezpieczeństwo systemu zależy teraz to od tego, jak dobrze Janek chroni swój klucz prywatny. Ważną kwestią, na którą należy zwrócić uwagę, jest teraz to, że bezpieczeństwo klucza prywatnego Janka zależy wyłącznie od niego samego. Nikt inny nie będzie w stanie odtworzyć klucza prywatnego, jeśli Janek go zgubi. A Ela z pewnością nie wycofa żadnej rzekomo nieuczciwej dyspozycji tylko dlatego, że Janek nie dbał o bezpieczeństwo.

Jeśli Janek przechowywał swój klucz prywatny w postaci niezaszyfrowanego tekstu w folderze udostępnionym w intranecie firmy, to każdy mógł go łatwo skopiować i użyć do kradzieży jego tokenów. Ale jeżeli Janek będzie przechowywał swój klucz prywatny w zaszyfrowanym pliku, chronionym silnym hasłem na dysku twardym swojego laptopa, to zdobycie kopii jego klucza prywatnego będzie znacznie trudniejsze. Atakujący musiałby bowiem:

- dostać się do dysku twardego Janka,
- znać hasło Janka.

Jeśli Janek nigdy nie będzie miał więcej niż 50 TC na swoim kluczu prywatnym, może nie zawracać sobie głowy bezpieczeństwem. Ale bufet, przez który przepływa około 10 000 TC dziennie, ma już powody do zmartwień. Najwyraźniej Janek i bufet potrzebują różnych strategii przechowywania swoich kluczy prywatnych.

Istnieje kompromis pomiędzy bezpieczeństwem a wygodą. Możesz na przykład przechowywać swój klucz prywatny w postaci zaszyfrowanej na laptopie bez dostępu do sieci, zdeponowanym w skrytce depozytowej w banku. Gdy będziesz chciał kupić ciasteczko, musisz pójść do banku, wyjąć laptop ze skrytki, odszyfrować klucz prywatny za pomocą hasła i użyć go do cyfrowego podpisania zapisanej wiadomości dla Eli, którą to wiadomość zgrasz na pendrive. Potem musisz odłożyć laptop do skrytki, przynieść pamięć USB z powrotem do biura i wysłać wiadomość e-mail do Eli. Klucz prywatny nigdy nie opuszcza laptopa spoczywającego bezpiecznie w skrytce depozytowej. To bardzo bezpieczne, ale też bardzo niewygodne.

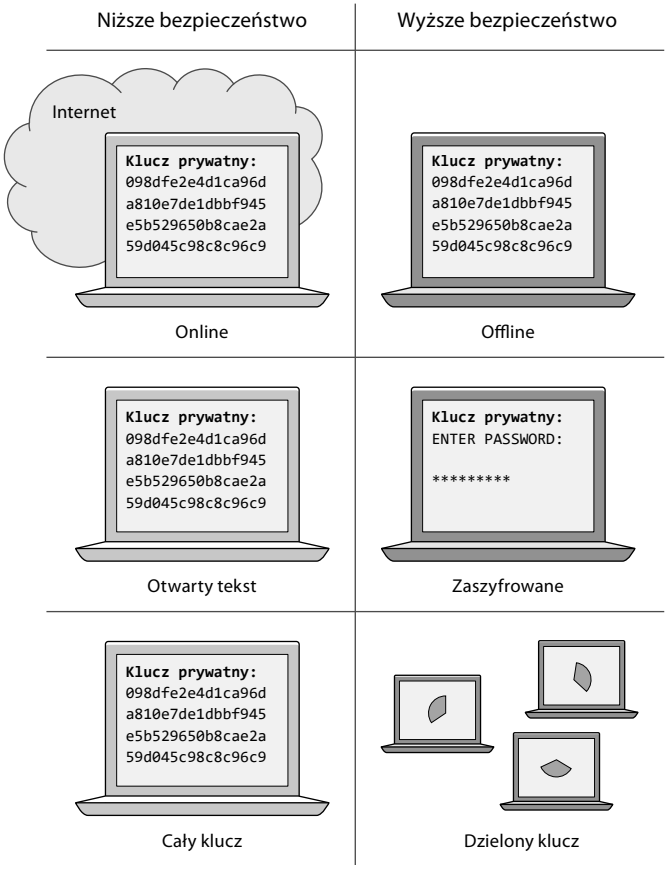
Z drugiej strony możesz przechowywać swój klucz prywatny w postaci niezaszyfrowanego tekstu na telefonie komórkowym. Klucz masz wtedy do dyspozycji cały czas i możesz podpisać wiadomości w ciągu kilku sekund od momentu, gdy tylko pomyślisz o chrupnięciu ciasteczka. To z kolei bardzo niepewne, ale bardzo wygodne.

Na rysunku 2.23 dostrzeżemy kilka kompromisów:

- **Online czy offline.** Online oznacza, że klucz prywatny jest przechowywany na urządzeniu z dostępem do sieci, takim jak telefon komórkowy lub laptop. Offline oznacza, że klucz prywatny jest przechowywany na kartce papieru lub na komputerze bez dostępu do sieci. Przechowywanie

online jest ryzykowne, ponieważ luki w zabezpieczeniach dostępu zdalnego do Twojego komputera lub zainstalowane na nim złośliwe oprogramowanie, takie jak wirusy komputerowe, mogą umożliwić zdobycie komuś Twojego klucza prywatnego zupełnie bez Twojej wiedzy. Jeśli urządzenie nie ma połączenia z siecią, nikt nie będzie w stanie zdobyć klucza prywatnego, o ile nie uzyska fizycznego dostępu do urządzenia.

- **Niezaszyfrowane czy zaszyfrowane.** Jeśli klucz prywatny jest przechowywany w postaci niezaszyfrowanej, w pliku na dysku twardym komputera, każdy, kto uzyska dostęp do komputera zdalnie, przez sieć komputerową lub bezpośrednio, będzie mógł skopiować Twój klucz prywatny. Obejmuje to też wszelkie wirusy, których ofiarą może paść Twój komputer. Możesz uniknąć wielu ataków tego typu, szyfrując swój klucz prywatny hasłem, które będziesz znać tylko Ty. Atakujący, aby zdobyć klucz prywatny, musiałby wtedy zarówno uzyskać dostęp do dysku twardego, jak też znać tajne hasło.



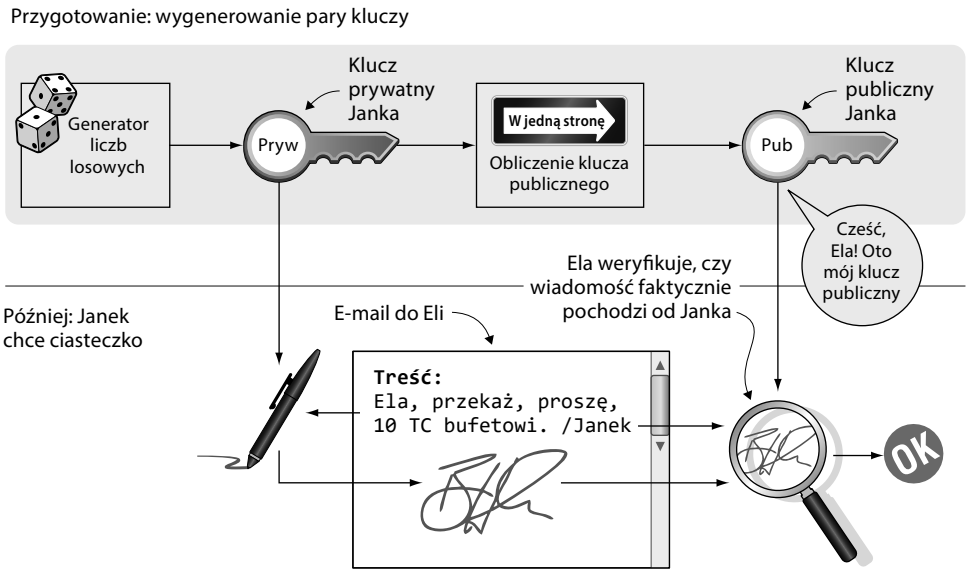
Rysunek 2.23. Zabezpieczenia przed atakami. Zwróć uwagę, że bardziej bezpieczne opcje są również bardziej niewygodne

- **Cały klucz czy klucz podzielony.** Ludzie zwykle przechowują cały klucz prywatny na jednym komputerze. Jest to wygodne, bo aby operować swoimi tokenami, potrzeba tylko jednego komputera. Z kolei wystarczy, że atakujący uzyska dostęp do dysku twardego, aby wykraść klucz prywatny. Ale jeśli Twój klucz prywatny jest podzielony na trzy części (istnieją w tym zakresie dobre i złe praktyki, więc radzę zachować ostrożność) i przechowujesz te trzy części osobno, na trzech różnych komputerach, to atakujący będzie musiał uzyskać dostęp do dysków twardych trzech komputerów. Jest to znacznie trudniejsze, ponieważ atakujący musi skądś wiedzieć, które trzy komputery ma zaatakować, a także te trzy ataki muszą się jednocześnie powieść. Realizacja płatności tokenami w tej konfiguracji będzie jednak bardzo kłopotliwa, ale zarazem bardzo bezpieczna.

Do przechowywania kluczy możesz użyć dowolnej kombinacji tych metod. Zasada jest taka, że im większe bezpieczeństwo przed atakami, tym większe ryzyko przypadkowej utraty dostępu do klucza. Na przykład jeśli przechowujesz klucz prywatny zaszyfrowany na dysku twardym, ryzykujesz utratę klucza nie tylko z powodu awarii komputera, ale też w przypadku, gdybyś zapomniał hasła. A więc, paradoksalnie, im więcej środków bezpieczeństwa zastosujesz, tym mniej bezpieczny będzie klucz!

Podsumowanie

Ela rozwiązała problem z ludźmi podającymi się za kogoś innego podczas płatności. Wymaga to od wszystkich uczestników systemu cyfrowego podpisywania dyspozycji przekazania tokenów ciasteczkowych. Każdy użytkownik tabelki musi mieć klucz prywatny i klucz publiczny. Ela tylko notuje, kto ma jaki klucz publiczny. Teraz dyspozycje muszą być przesyłane e-mailem do Eli, a ich treść musi zostać podpisana cyfrowo za prywatnym kluczem nadawcy. Ela może wtedy zweryfikować podpis, sprawdzając, czy nie doszło do próby oszustwa. Istotne jest to, że dopóki Janek nie udostępni nikomu swojego klucza prywatnego, nikt nie będzie w stanie dysponować jego środkami.



Musimy dodać „E-mail do Eli" do naszej tabeli pojęć (tabela 2.5).

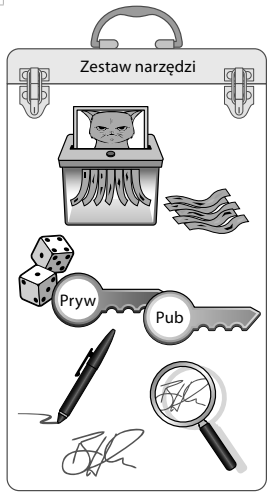
Tabela 2.5. Dodawanie „E-mail do Eli" jako kluczowego pojęcia

Tokeny ciasteczkowe	Bitcoin	Omówiono w
1 token ciasteczkowy	1 bitcoin	Rozdział 2.
Tabelka	Łańcuch bloków	Rozdział 6.
E-mail do Eli	**Transakcja**	**Rozdział 5.**
Wiersz w tabelce	Transakcja	Rozdział 5.
Ela	Górnik	Rozdział 7.

W rozdziale 5. „e-mail do Eli" zastąpię transakcją. Transakcja zastąpi zarówno e-mail do Eli, jak i „wiersz w tabelce". Czas opublikować wersję 2.0 naszego systemu tabelki tokenów ciasteczkowych (tabela 2.6).

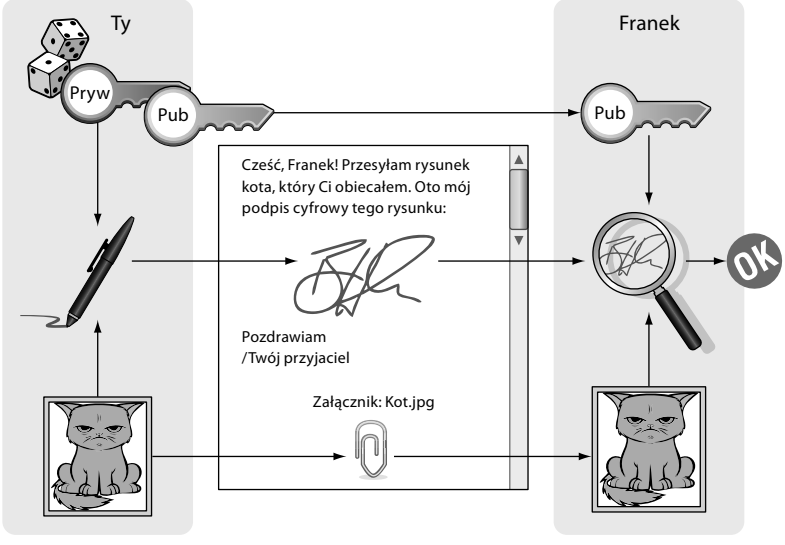
Tabela 2.6. Informacje o zmianach, token ciasteczkowy 2.0

Wersja	Właściwość	Jak
NOWA 2.0	Bezpieczne płatności	Podpisy cyfrowe rozwiązują problem podszywania się pod kogoś.
1.0	Prosty system płatności	Wykorzystuje to, że Eli ufają wszyscy, oraz to, że Ela zna z widzenia każdego pracownika.
	Ograniczona podaż pieniądza	Ela otrzymuje codziennie wynagrodzenie w wysokości 7200 nowych TC. Wynagrodzenie to zmniejszane jest o połowę co cztery lata.



Wszyscy nadal ufają Eli, wierząc, że nigdy nie zmieni zawartości tabelki, chyba że przez dopisanie na koniec dyspozycji przekazania tokenów ciasteczkowych. Gdyby Ela chciała, mogłaby ukraść tokeny, po prostu dopisując swoją dyspozycję do tabelki. Ale przecież tego nie zrobi, prawda?

Masz teraz wiele nowych narzędzi, które możesz dołączyć do swojej palety: generowanie par kluczy, podpis cyfrowy, podpisywanie i weryfikacja.



Ćwiczenia

Rozgrzewka

2.8. Ela otrzymuje obecnie 7200 TC dziennie za swoją pracę. Dlaczego podaż tokenów nie może wzrastać w nieskończoność? Dlaczego po 10 000 dniach nie będzie ich 7200 · 10 000 = 72 miliony TC?

2.9. Jak inni pracownicy mogą sprawdzić, czy Ela nie przekazuje sobie wynagrodzenia za dużo lub za często?

2.10. Jak tworzony jest klucz prywatny z pary kluczy?

2.11. Jakiego klucza używa się do cyfrowego podpisywania wiadomości?

2.12. Proces podpisywania oblicza skrót podpisywanych danych. Dlaczego?

2.13. Czego potrzebowałby Zdzisiek, aby ukraść Jankowi jego tokeny?

Wyższy poziom

2.14. Załóżmy, że masz swój klucz prywatny i udostępniłeś go swojemu przyjacielowi, Frankowi. Zaproponuj, w jaki sposób Franek mógłby przesłać Ci tajną wiadomość, którą tylko Ty mógłbyś poprawnie odczytać.

2.15. Załóżmy, że Ty (powiedzmy, że masz na imię Olek) i Franek nadal macie klucze z poprzedniego ćwiczenia. Chcesz teraz wysłać Frankowi wiadomość o treści:

Cześć, Franek! Czy możemy spotkać się jutro wieczorem w restauracji? /Olek

Wyjaśnij, jak chcesz podpisać wiadomość, aby Franek miał pewność, że faktycznie pochodzi ona od Ciebie. Wyjaśnij, jakie kroki podejmiecie Ty i Franek.

Najważniejsze kwestie w skrócie

- Bitcoiny są tworzone jako wynagrodzenie dla węzłów dbających o bezpieczeństwo łańcucha bloków.
- Wynagrodzenie zmniejszane jest o połowę co cztery lata, aby liczba generowanych tokenów malała.
- Do wykrywania zmian w pliku lub w wiadomości możesz użyć kryptograficznych funkcji skrótu.
- Odtworzenie pierwotnego obrazu na podstawie skrótu kryptograficznego jest niemożliwe. Obraz pierwotny to dane wejściowe generujące pewne znane dane wyjściowe.
- Podpisy cyfrowe są przydatne do potwierdzenia autentyczności płatności. Tylko prawowity właściciel bitcoinów może je wydawać.
- Ktoś, kto sprawdza podpis cyfrowy, nie musi wiedzieć, kto go złożył. Musi tylko wiedzieć, że podpis został faktycznie złożony przy użyciu klucza prywatnego należącego do osoby twierdzącej, że podpis ten złożyła.
- Aby otrzymywać bitcoiny lub tokeny ciasteczkowe, potrzebujesz klucza publicznego. Najpierw dla siebie tworzysz klucz prywatny, którego nikomu nie będziesz ujawniać. Następnie, na jego podstawie, generujesz swój klucz publiczny.
- Dostępnych jest kilka strategii przechowywania kluczy prywatnych, począwszy od przechowywania w postaci niezaszyfrowanej w telefonie komórkowym, po postać dzieloną i szyfrowaną oraz przechowywanie na kilku urządzeniach offline.
- Ogólna zasada mówi, że im lepiej zabezpieczono klucz prywatny przed kradzieżą, tym łatwiej jest go przypadkowo utracić i odwrotnie.

Bitcoin: zrozum, czym jest ta przełomowa technologia!

Inflacja, sterowane kryzysy gospodarcze, pękające bańki, afery związane z udzielaniem kredytów i zadłużaniem się — to tylko kilka problemów, których doświadczamy w efekcie scentralizowanej bankowości powiązanej z polityką. Z drugiej strony bitcoin i łańcuch bloków kojarzą się z mrocznym półświatkiem i łatwymi pieniędzmi, przez co rozbudzają ludzkie lęki i chciwość. Jeśli jednak odsunie się na bok wątki sensacyjne, łatwo zauważyć, że zdecentralizowana cyfrowa waluta ma potencjał wyrwania wielu ważnych aspektów naszego życia spod kontroli polityków i bankowców. Aby jednak trafnie ocenić przydatność bitcoina w niepowtarzalnej, osobistej sytuacji, trzeba dobrze pojąć tę koncepcję i zasady, na których się opiera.

Dzięki tej książce zrozumiesz mechanizmy rządzące technologią bitcoina oraz łańcucha bloków. Poznasz też wiele koncepcji leżących u podstaw bitcoina, takich jak podpisy cyfrowe, dowód pracy oraz sieci peer-to-peer. Dowiesz się, jak zainstalować i korzystać z portfela bitcoina, jak przechowywać klucze prywatne oraz jak uruchomić pełny węzeł bitcoina, a także — jak rozpoznawać fałszywe deklaracje oszustów liczących na łatwy zysk. Aby w pełni skorzystać z zawartych tu informacji, nie musisz dysponować umiejętnością programowania ani zaawansowaną wiedzą techniczną i matematyczną. Dzięki przystępnemu językowi i łatwym do przyswojenia schematom zrozumiesz, czym w rzeczywistości jest bitcoin i czym może się stać w niedalekiej przyszłości.

Najważniejsze zagadnienia:

- solidne podstawy bitcoina
- anatomia transakcji bitcoinowych
- adresy odbiorców i portfele w sieci bitcoina
- łańcuch bloków i kopanie bitcoina
- nowości w zaktualizowanym protokole bitcoina

Kalle Rosenbaum — programista z dwudziestoletnim doświadczeniem. W 2013 roku zafascynował się bitcoinem, a w 2015 roku założył firmę doradczą specjalizującą się w tej tematyce i w pełni się jej poświęcił. W wolnych chwilach prowadzi bloga, na którym wyjaśnia różne techniczne kwestie dotyczące kryptowalut.

Helion
helion.pl
HELION SA
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

ISBN 978-83-283-9303-5
9 788328 393035

Cena: 99,00 zł