Wprowadzenie – ciąg dalszy

Patryk Czarnik

Wydział Matematyki, Informatyki i Mechaniki
Uniwersytet Warszawski

Bezpieczeństwo sieci komputerowych – MSUI 2009/10

Szyfry asymetryczne

- Wymyślone w latach 70-tych
- Używają dwóch różnych (ale „pasujących do siebie”) kluczy do szyfrowania i deszyfrowania
- Klucza deszyfrującego nie można otrzymać w efektywny sposób z klucza szyfrującego
- Opierają się na solidnych podstawach matematycznych (teoria liczb, teoria złożoności obliczeniowej)
- Są nieefektywne:
  - szyfrowanie i deszyfrowanie przebiega wolno
  - klucze muszą być duże – co najmniej 1000 bitów

Szyfry asymetryczne

- Wymyślone w latach 70-tych
- Używają dwóch różnych (ale „pasujących do siebie”) kluczy do szyfrowania i deszyfrowania
- Klucza deszyfrującego nie można otrzymać w efektywny sposób z klucza szyfrującego
- Opierają się na solidnych podstawach matematycznych (teoria liczb, teoria złożoności obliczeniowej)
- Są nieefektywne:
  - szyfrowanie i deszyfrowanie przebiega wolno
  - klucze muszą być duże – co najmniej 1000 bitów

Szyfry asymetryczne

- Wymyślone w latach 70-tych
- Używają dwóch różnych (ale „pasujących do siebie”) kluczy do szyfrowania i deszyfrowania
- Klucza deszyfrującego nie można otrzymać w efektywny sposób z klucza szyfrującego
- Opierają się na solidnych podstawach matematycznych (teoria liczb, teoria złożoności obliczeniowej)
- Są nieefektywne:
  - szyfrowanie i deszyfrowanie przebiega wolno
  - klucze muszą być duże – co najmniej 1000 bitów

Szyfry asymetryczne

- Wymyślone w latach 70-tych
- Używają dwóch różnych (ale „pasujących do siebie”) kluczy do szyfrowania i deszyfrowania
- Klucza deszyfrującego nie można otrzymać w efektywny sposób z klucza szyfrującego
- Opierają się na solidnych podstawach matematycznych (teoria liczb, teoria złożoności obliczeniowej)
- Są nieefektywne:
  - szyfrowanie i deszyfrowanie przebiega wolno
  - klucze muszą być duże – co najmniej 1000 bitów

Szyfry asymetryczne

- Wymyślone w latach 70-tych
- Używają dwóch różnych (ale „pasujących do siebie”) kluczy do szyfrowania i deszyfrowania
- Klucza deszyfrującego nie można otrzymać w efektywny sposób z klucza szyfrującego
- Opierają się na solidnych podstawach matematycznych (teoria liczb, teoria złożoności obliczeniowej)
- Są nieefektywne:
  - szyfrowanie i deszyfrowanie przebiega wolno
  - klucze muszą być duże – co najmniej 1000 bitów

Matematyczne podstawy szyfrów asymetrycznych

Faktoryzacja liczb naturalnych

- Mnożenie liczb szybkie
- Faktoryzacja (dla dużych czynników pierwszych) trudna
- Algorytm RSA, szyfr Rabina

Problem logarytmu dyskretnego

- Potęgowanie modulo szybkie
- Logarytmowanie dyskretne („modulo”) trudne
- Algorytmy ElGamal, DSA

Dyskretny problem plecakowy

- Problem NP-trudny
- Przy odpowiednim doborze wag (znowu teoria liczb) daje szyfr asymetryczny
- Szyfr plecakowy Chora-Rivesta

Log. d. na krzywych eliptycznych

- Potęgowanie i logarytmowanie w grupach dyskretnych opartych o krzywe eliptyczne
- Umożliwia stosowanie krótszych kluczy
- Algorytm ECC

Matematyczne podstawy szyfrów asymetrycznych

Faktoryzacja liczb naturalnych

- Mnożenie liczb szybkie
- Faktoryzacja (dla dużych czynników pierwszych) trudna
- Algorytm RSA, szyfr Rabina

Problem logarytmu dyskretnego

- Potęgowanie modulo szybkie
- Logarytmowanie dyskretne („modulo”) trudne
- Algorytmy ElGamal, DSA

Dyskretny problem plecakowy

- Problem NP-trudny
- Przy odpowiednim doborze wag (znowu teoria liczb) daje szyfr asymetryczny
- Szyfr plecakowy Chora-Rivesta

Log. d. na krzywych eliptycznych

- Potęgowanie i logarytmowanie w grupach dyskretnych opartych o krzywe eliptyczne
- Umożliwia stosowanie krótszych kluczy
- Algorytm ECC

Matematyczne podstawy szyfrów asymetrycznych

Faktoryzacja liczb naturalnych

- Mnożenie liczb szybkie
- Faktoryzacja (dla dużych czynników pierwszych) trudna
- Algorytm RSA, szyfr Rabina

Problem logarytmu dyskretnego

- Potęgowanie modulo szybkie
- Logarytmowanie dyskretne („modulo”) trudne
- Algorytmy ElGamal, DSA

Dyskretny problem plecakowy

- Problem NP-trudny
- Przy odpowiednim doborze wag (znowu teoria liczb) daje szyfr asymetryczny
- Szyfr plecakowy Chora-Rivesta

Log. d. na krzywych eliptycznych

- Potęgowanie i logarytmowanie w grupach dyskretnych opartych o krzywe eliptyczne
- Umożliwia stosowanie krótszych kluczy
- Algorytm ECC

Matematyczne podstawy szyfrów asymetrycznych

Faktoryzacja liczb naturalnych

- Mnożenie liczb szybkie
- Faktoryzacja (dla dużych czynników pierwszych) trudna
- Algorytm RSA, szyfr Rabina

Problem logarytmu dyskretnego

- Potęgowanie modulo szybkie
- Logarytmowanie dyskretne („modulo”) trudne
- Algorytmy ElGamal, DSA

Dyskretny problem plecakowy

- Problem NP-trudny
- Przy odpowiednim doborze wag (znowu teoria liczb) daje szyfr asymetryczny
- Szyfr plecakowy Chora-Rivesta

Log. d. na krzywych eliptycznych

- Potęgowanie i logarytmowanie w grupach dyskretnych opartych o krzywe eliptyczne
- Umożliwia stosowanie krótszych kluczy
- Algorytm ECC

Kryptografia klucza publicznego

Założenie o wykorzystywaniu pary kluczy (prywatnego i publicznego) nie jest specyficzne tylko dla systemów szyfrujących. Także inne podstawowe mechanizmy kryptograficzne mogą zyskać opierając swoje działanie na takim założeniu.
Kryptografię wykorzystującą dualność kluczy nazywamy **kryptografią klucza publicznego** w odróżnieniu od **kryptografii klucza sekretnego**.

Uwierzytelnianie

Uwierzytelnianie to tak naprawdę dwa zagadnienia:

- **protokół identyfikacji** – sprawdzenie, że uczestnicy protokołu są tymi, za których się podają
- **weryfikacja integralności danych** – zapewnienie, że dane pochodzą od właściwego uczestnika protokołu i nie zostały przez nikogo zmodyfikowane „po drodze”

W pierwszym przypadku wymagamy, aby uczestnik protokołu był dostępny w momencie uwierzytelniania, natomiast w drugim przypadku to wymaganie jest zbędne.

Uwierzytelnianie

Uwierzytelnianie to tak naprawdę dwa zagadnienia:

- **protokół identyfikacji** – sprawdzenie, że uczestnicy protokołu są tymi, za których się podają
- **weryfikacja integralności danych** – zapewnienie, że dane pochodzą od właściwego uczestnika protokołu i nie zostały przez nikogo zmodyfikowane „po drodze”

W pierwszym przypadku wymagamy, aby uczestnik protokołu był dostępny w momencie uwierzytelniania, natomiast w drugim przypadku to wymaganie jest zbędne.

Protokoły identyfikacji

Protokół identyfikacji to protokół między dwoma uczestnikami: **dowodzącym** P i **weryfikującym** V. Celem protokołu jest sprawdzenie przez V czy P jest rzeczywiście tym, za kogo się podaje.

Oczekiwane własności

- **poprawność** – jeśli P jest rzeczywiście tym, za kogo się podaje, to protokół zakończy działanie z wynikiem *prawda*
- **transferowalność** – V nie może wykorzystać informacji zdobytej podczas wykonania protokołu do podszycia się pod P
- **nie podszywanie się** – prawdopodobieństwo tego, że przy wykonaniu protokołu P uda się podszyć za kogoś innego jest zaniedbywalne

Protokoły identyfikacji

Protokół identyfikacji to protokół między dwoma uczestnikami: **dowodzącym** P i **weryfikującym** V. Celem protokołu jest sprawdzenie przez V czy P jest rzeczywiście tym, za kogo się podaje.

Oczekiwane własności

- **poprawność** – jeśli P jest rzeczywiście tym, za kogo się podaje, to protokół zakończy działanie z wynikiem *prawda*
- **transferowalność** – V nie może wykorzystać informacji zdobytej podczas wykonania protokołu do podszycia się pod P
- **nie podszywanie się** – prawdopodobieństwo tego, że przy wykonaniu protokołu P uda się podszyć za kogoś innego jest zaniedbywalne

Warunek nie podszywania się

Warunek nie podszywania się powinien być spełniony nawet jeśli podszywający się uczestniczył w wielu protokołach identyfikacji zarówno z P jak i V oraz wiele sesji protokołu działa w tym samym czasie.

Kategorie protokołów identyfikacji

Obecnie istniejące protokoły identyfikacji można podzielić na następujące kategorie:

- słabe uwierzytelnianie (hasła, PIN itp.),
  - raz podsłuchane hasło może być później użyte,
  - nawet jeśli wysyłane jest w postaci zaszyfrowanej!
- protokoły typu wyzwanie – odpowiedź (*challenge—response*):
  - pytanie o losową informację z dużego zbioru (małe prawdopodobieństwo powtórzeń),
  - hasła jednorazowe,
- protokoły z wiedzą zerową.

Kategorie protokołów identyfikacji

Obecnie istniejące protokoły identyfikacji można podzielić na następujące kategorie:

- słabe uwierzytelnianie (hasła, PIN itp.),
  - raz podsłuchane hasło może być później użyte,
  - nawet jeśli wysyłane jest w postaci zaszyfrowanej!
- protokoły typu wyzwanie – odpowiedź (*challenge—response*):
  - pytanie o losową informację z dużego zbioru (małe prawdopodobieństwo powtórzeń),
  - hasła jednorazowe,
- protokoły z wiedzą zerową.

Kategorie protokołów identyfikacji

Obecnie istniejące protokoły identyfikacji można podzielić na następujące kategorie:

- słabe uwierzytelnianie (hasła, PIN itp.),
  - raz podsłuchane hasło może być później użyte,
  - nawet jeśli wysyłane jest w postaci zaszyfrowanej!
- protokoły typu wyzwanie – odpowiedź (*challenge—response*):
  - pytanie o losową informację z dużego zbioru (małe prawdopodobieństwo powtórzeń),
  - hasła jednorazowe,
- protokoły z wiedzą zerową.

Protokoły z wiedzą zerową

Protokół z wiedzą zerową to protokół pomiędzy parą graczy P i V.
P próbuje przekonać V, że posiada pewien sekret *s*, ale w taki sposób, aby nie ujawnić żadnej informacji o *s*.

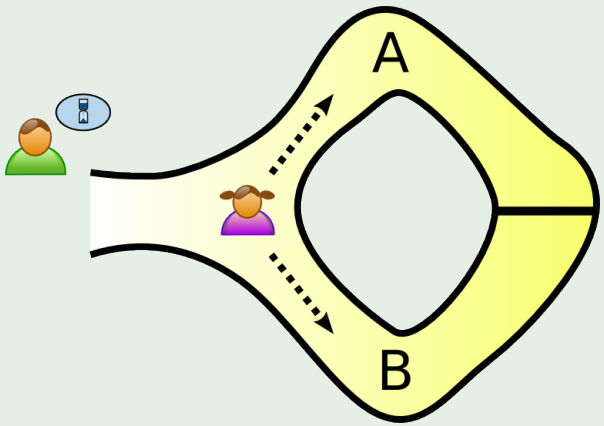
Przykład przedszkolny

Źródło: Wikipedia

Protokoły z wiedzą zerową

Protokół z wiedzą zerową to protokół pomiędzy parą graczy P i V.
P próbuje przekonać V, że posiada pewien sekret *s*, ale w taki sposób, aby nie ujawnić żadnej informacji o *s*.

Przykład przedszkolny

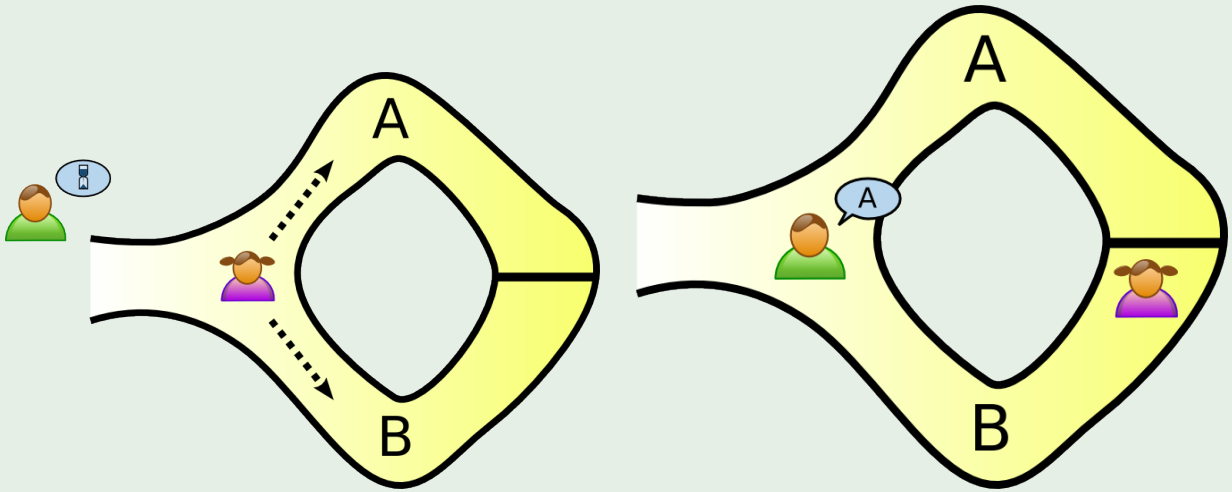


Źródło: Wikipedia

Protokoły z wiedzą zerową

Protokół z wiedzą zerową to protokół pomiędzy parą graczy P i V.
P próbuje przekonać V, że posiada pewien sekret *s*, ale w taki sposób, aby nie ujawnić żadnej informacji o *s*.

Przykład przedszkolny

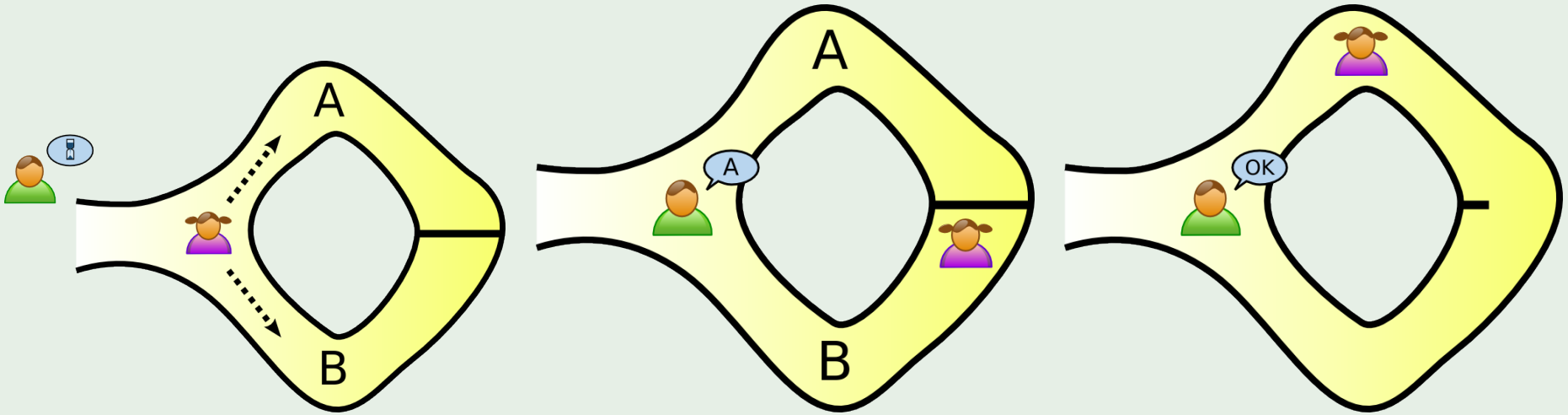


Źródło: Wikipedia

Protokoły z wiedzą zerową

Protokół z wiedzą zerową to protokół pomiędzy parą graczy P i V.
P próbuje przekonać V, że posiada pewien sekret *s*, ale w taki sposób, aby nie ujawnić żadnej informacji o *s*.

Przykład przedszkolny



Źródło: Wikipedia

Protokoły z wiedzą zerową

- Podział teoretyczny:
  - idealny protokół z wiedzą zerową,
  - statystyczny protokół z wiedzą zerową,
  - obliczeniowy protokół z wiedzą zerową.
- Najbardziej znane protokoły z wiedzą zerową:
  - Fiata-Shamira i Feige-Fiata-Shamira (oparty o faktoryzację liczb złożonych i dyskretny pierwiastek kwadratowy),
  - Schnorra (oparty o logarytm dyskretny),
  - GQ (Guillou-Quisquater).
- W (prawie?) wszystkich protokołach tego typu wykorzystywane są losowe liczby, generowane na potrzeby jednej sesji.
- W omawianych na przyszłych wykładach protokołach bezpiecznej komunikacji zwykle występuje unikalna/losowa wartość wysyłana na początku komunikacji.

Protokoły z wiedzą zerową

- Podział teoretyczny:
  - idealny protokół z wiedzą zerową,
  - statystyczny protokół z wiedzą zerową,
  - obliczeniowy protokół z wiedzą zerową.
- Najbardziej znane protokoły z wiedzą zerową:
  - Fiata-Shamira i Feige-Fiata-Shamira (oparty o faktoryzację liczb złożonych i dyskretny pierwiastek kwadratowy),
  - Schnorra (oparty o logarytm dyskretny),
  - GQ (Guillou-Quisquater).
- W (prawie?) wszystkich protokołach tego typu wykorzystywane są losowe liczby, generowane na potrzeby jednej sesji.
- W omawianych na przyszłych wykładach protokołach bezpiecznej komunikacji zwykle występuje unikalna/losowa wartość wysyłana na początku komunikacji.

Protokoły z wiedzą zerową

- Podział teoretyczny:
  - idealny protokół z wiedzą zerową,
  - statystyczny protokół z wiedzą zerową,
  - obliczeniowy protokół z wiedzą zerową.
- Najbardziej znane protokoły z wiedzą zerową:
  - Fiata-Shamira i Feige-Fiata-Shamira (oparty o faktoryzację liczb złożonych i dyskretny pierwiastek kwadratowy),
  - Schnorra (oparty o logarytm dyskretny),
  - GQ (Guillou-Quisquater).
- W (prawie?) wszystkich protokołach tego typu wykorzystywane są losowe liczby, generowane na potrzeby jednej sesji.
- W omawianych na przyszłych wykładach protokołach bezpiecznej komunikacji zwykle występuje unikalna/losowa wartość wysyłana na początku komunikacji.

Protokoły z wiedzą zerową

- Podział teoretyczny:
  - idealny protokół z wiedzą zerową,
  - statystyczny protokół z wiedzą zerową,
  - obliczeniowy protokół z wiedzą zerową.
- Najbardziej znane protokoły z wiedzą zerową:
  - Fiata-Shamira i Feige-Fiata-Shamira (oparty o faktoryzację liczb złożonych i dyskretny pierwiastek kwadratowy),
  - Schnorra (oparty o logarytm dyskretny),
  - GQ (Guillou-Quisquater).
- W (prawie?) wszystkich protokołach tego typu wykorzystywane są losowe liczby, generowane na potrzeby jednej sesji.
- W omawianych na przyszłych wykładach protokołach bezpiecznej komunikacji zwykle występuje unikalna/losowa wartość wysyłana na początku komunikacji.

Bezpieczne funkcje haszujące – definicja

Bezpieczna funkcja haszująca to funkcja

$$h : \{0,1\}^* \to \{0,1\}^n$$

taka, że trudne („obliczeniowo niewykonalne”) jest:

- dla ustalonego $c \in \{0,1\}^n$ znalezienie takiego m, że $h(m) = c$ („funkcja jednokierunkowa”),
- znalezienie dwóch ciągów m_1 i m_2 takich, że $h(m_1) = h(m_2)$.

Bezpieczne funkcje haszujące – zastosowania

Pozwalają one dokonać „kompresji” danych w taki sposób, aby nie popsuć bezpieczeństwa algorytmów, w których uczestniczą.

Zastosowania związane z bezpieczeństwem

- Podpisy cyfrowe
- Zapewnienie integralności danych
- Złożone protokoły kryptograficzne

Bezpieczne funkcje haszujące – zastosowania

Pozwalają one dokonać „kompresji” danych w taki sposób, aby nie popsuć bezpieczeństwa algorytmów, w których uczestniczą.

Zastosowania związane z bezpieczeństwem

- Podpisy cyfrowe
- Zapewnienie integralności danych
- Złożone protokoły kryptograficzne

MAC i MDC

- **MAC –** *Message Authentication Code*
  - Do generowania oraz weryfikacji kodu potrzebny jest (ten sam) tajny klucz.
- **MDC –** *Modification Detection Code*,
  MIC – *Message Integrity Code*
  - Nie wymaga tajnego klucza.
  - Stosowane do kontroli integralności danych oraz w schematach podpisów cyfrowych
    (wiadomość $\rightarrow$ MDC $\rightarrow$ podpis).

MAC i MDC

- **MAC –** *Message Authentication Code*
  - Do generowania oraz weryfikacji kodu potrzebny jest (ten sam) tajny klucz.
- **MDC –** *Modification Detection Code*, **MIC –** *Message Integrity Code*
  - Nie wymaga tajnego klucza.
  - Stosowane do kontroli integralności danych oraz w schematach podpisów cyfrowych (wiadomość $\rightarrow$ MDC $\rightarrow$ podpis).

Pokrewne algorytmy

Oba rodzaje bezpiecznych funkcji haszujących należy odróżnić od:

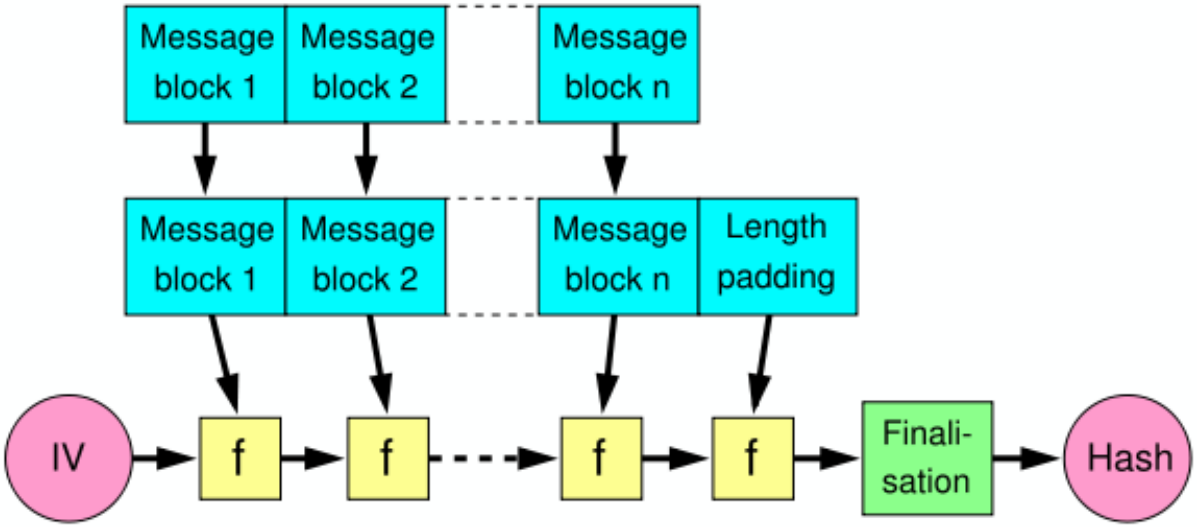
- sum kontrolnych – które nie muszą spełniać warunków bezpieczeństwa (MDC jest sumą kontrolną, ale nie każda suma kontrolna jest MDC),
- podpisów cyfrowych opartych o klucz publiczny – w MAC jest jeden tajny klucz.

Budowa funkcji haszującej z szyfru blokowego

- Szyfr blokowy może zostać wykorzystany do stworzenia bezpiecznej funkcji haszującej.
- Warunek: każdy ciąg bitów długości K jest dobrym kluczem.
- Popularny schemat na poziomie ciągu bloków:
  - Merkle-Damgård
- Popularne schematy na poziomie pojedynczych bloków:
  - Davies-Meyer
  - Matyas-Meyer-Oseas
  - Miyaguchi-Preneel

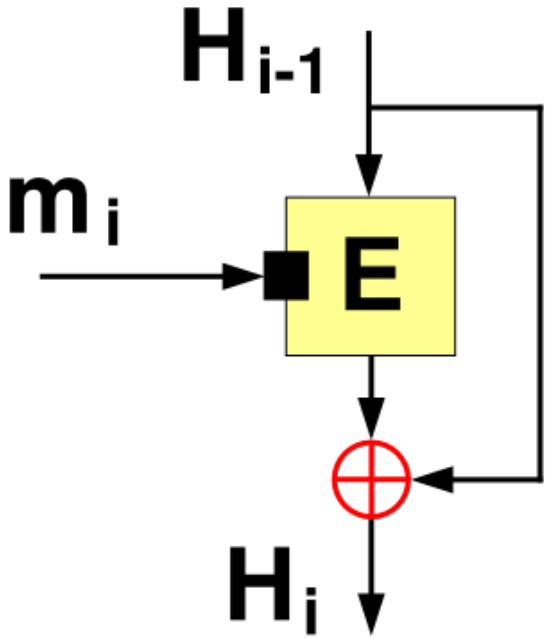
Schemat Merkle-Damgård (cała wiadomość)

- Dla bezpieczeństwa w ostatnim bloku uzupełnienie zawierające długość oryginalnej wiadomości.



Źródło obrazka: Wikipedia

Schemat Davies-Meyer (pojedyncze bloki)



Źródło obrazka: Wikipedia

Najpopularniejsze bezpieczne funkcje haszujące

- Algorytmy MDC:
  - MDC-2 (hasz 128 bitów, IBM 1987),
  - MDC-4 (hasz 128 bitów),
  - MD4 (hasz 128 bitów, MIT 1990, obecnie bardzo słaba),
  - MD5 (hasz 128 bitów, 1991, słaba),
  - RIPEMD-160 (hasz 160 bitów, europejski),
  - SHA-1 (hasz 160 bitów, pokazano schemat ataku o czasochłonności 2^{63}),
  - SHA-2 (hasz 224/256/384/512 bitów, na razie uznawana za bezpieczną).
- Schematy budowy MAC:
  - HMAC – z funkcji haszującej, np. HMAC-MD5, HMAC-SHA1
  - CMAC – z szyfru blokowego,
  - CBC-MAC – z szyfru blokowego,
    np. Data Authentication Algorithm (oparty o DES, standard ANSI),
  - UMAC – z wielu szyfrów blokowych.

Inne zagadnienia bezpieczeństwa w sieci

W sieciach komputerowych pojawiają się zagadnienia, dla których trudno jest znaleźć satysfakcjonujące z praktycznego punktu widzenia rozwiązanie kryptograficzne. Należą do nich:

- ataki typu DOS (uniemożliwienie świadczenia usług),
- niechciana poczta (spam),
- wirusy i robaki komputerowe.

Inne zagadnienia bezpieczeństwa w sieci

Zaproponowane w ich przypadku rozwiązania wymagają zbyt dużych zmian w istniejącej architekturze sieciowej.
Często w takim przypadku bardziej efektywne okazuje się zdroworozsądkowe rozwiązanie praktyczne. Jako przykład takiego podejścia można podać kwestię odróżniania człowieka od symulującego jego zachowanie programu komputerowego.

Złożone protokoły kryptograficzne

Protokoły tego typu rozwiązują problemy, które muszą spełniać wiele własności bezpieczeństwa jednocześnie. Przykłady:

- protokoły głosowania elektronicznego,
- protokoły dzielenia sekretu i bezpiecznych obliczeń rozproszonych,
- protokoły bezpiecznego rozsyłania grupowego,
- protokoły wspomagające przestrzeganie praw autorskich (cyfrowe znaki wodne),
- protokoły elektronicznej gotówki.

Przykład: Protokół elektronicznej gotówki

Podstawowe pożądane własności protokołu elektronicznej gotówki:

- pieniądz może być emitowany tylko przez upoważnioną instytucję,
- pieniędzy nie można powielać,
- pieniądz może być przekazywany pomiędzy stronami protokołu,
- pieniądz można rozmieniać,
- płatność pieniądzem elektronicznym zapewnia anonimowość stronom transakcji.

Inne zagadnienia dotyczące bezpieczeństwa w sieci

Przykładowe dodatkowe własności protokołu elektronicznej gotówki. Niektóre z nich mogą być ze sobą, lub z podstawowymi wymaganiami, sprzeczne.

- Utrudnianie „prania brudnych pieniędzy”.
- Możliwość opodatkowania transakcji dokonywanych elektroniczną gotówką.
- Możliwość dokonywania płatności nawet w przypadku, gdy strony nie mają dostępu do sieci.
- Możliwość identyfikacji i ewentualnie unieważnienia pieniędzy pochodzących z przestępstwa.