Wykłady 7d-9d

Temat 7: Funkcje jednokierunkowe

Funkcja jednokierunkowa (**funkcja hashująca, funkcja *hash*, funkcja skrótu** lub **funkcja mieszająca**) to funkcja, która przyporządkowuje każdej liczbie wartość określaną jako *hash* (hasz) inaczej zwaną wartością skrótu funkcji czy skrótem. Jeżeli dwie liczby są różne, to ich hashe, z wysokim prawdopodobieństwem, też powinny być różne. Hash ma zwykle pewną z góry ustaloną długość (np. 128 bitów) i daje się bardzo łatwo obliczyć. Funkcja haszująca powinna uniemożliwiać łatwe obliczenie oryginalnej liczby na podstawie jej *hasha*.

Dla komputerów łańcuchy znaków to ciągi liczb. Zwykle hash oblicza się właśnie dla takich ciągów. Właściwości funkcji skrótu powodują, że zmiana choć jednego znaku w hashowanym tekście powinna powodować bardzo dużą zmianę jego skrótu. Jeżeli mamy hash jakiegoś tekstu, to odszukanie jego oryginalnej treści wymaga ogromnej mocy obliczeniowej i jest prawie niemożliwe.

Przykład funkcji

Bardzo prostą funkcją skrótu może być na przykład reszta z dzielenia danej liczby przez 19:

$$h(1200) = 3$$
$$h(3257) = 9$$
$$h(6890) = 12$$

Do zastosowań praktycznych podana funkcja jest zbyt prosta.

Zastosowania

Funkcje haszujące mają wiele zastosowań:

- buduje się za ich pomocą efektywne struktury danych;
- używa się ich do weryfikacji poprawności przesyłu danych;
- stosuje się je w kryptografii.

W kryptografii ważne jest, żeby mając wartość funkcji $\boldsymbol{x}$ nie można było zbudować wiadomości $\boldsymbol{m}$, takiej że $x = h(m)$.

Tematy 8-9: Jednokierunkowe funkcje hashujące: podstawowe własności i zastosowania, bezkonfliktowe funkcje hashujące, wybrane zastosowania, ataki przeciw funkcjom hashującym. Hashowanie o dowodliwych własnościach: hashowanie a dyskretny logarytm, rozszerzenie na teksty dowolnej długości. Praktyczne algorytmy hashujące.

Funkcje haszujące używane w kryptografii:

- historyczne: MD2, MD4, SHA
- współczesne: MD5, SHA-1, SHA2, RIPEMD-160

MD4 (*Message-Digest algorithm* 4 – Skrót Wiadomości wersja 5), to funkcja haszująca zaprojektowana do zastosowań kryptograficznych. Została jednak złamana (kolizje można generować na typowym PC-cie w czasie rzędu sekund) i jest obecnie bezużyteczna do celów wymagających bezpieczeństwa. **MD4** została więc zastąpiona do praktycznie wszystkich zastosowań przez MD5 oraz inne funkcje haszujące.

MD4 i **MD5** mają bardzo podobną budowę:

- Wiadomość jest uzupełniana do wielokrotności 512 bitów.
- Ustala się pewien 128-bitowy stan początkowy (cztery 32-bitowe rejestry).
- Dla każdego 512-bitowego bloku danych uruchamiana jest transformacja, która na podstawie aktualnego stanu oraz tego bloku wylicza nowy stan.
- Po przerobieniu wszystkich bloków końcowy stan jest zwracany jako wynik funkcji

Najważniejsze różnice wobec MD5:

- Transformacja ma tylko 3 rundy 48 kroków, zamiast 4 (64 kroków) jak MD5.
- Do wyniku każdego kroku nie jest dodawana zależna od numeru kroku stała, jedynie stała zależna od numeru rundy w rundach 2 i 3, ani też nie występuje dodawanie wartości jednego rejestru do drugiego.
- Używana jest inna funkcja w krokach 17-32, `G(x,y,z) = (x and y) or (x and z) or (y and z).`

Kolizja funkcji haszującej `H` to taka para różnych wiadomości `m1`, `m2`, że mają one taką samą wartość *hasza*, tj. `H(m1) = H(m2)`. Ponieważ funkcja haszująca zwraca skończenie wiele wartości, a przestrzeń argumentów jest nieskończona (w przypadku funkcji akceptujących dowolnie długie argumenty), lub przynajmniej znacznie większa od przestrzeni wyników, dla każdej funkcji haszującej jakieś kolizje istnieją.

W wielu zastosowaniach zależy nam na tym, żeby nie znana była żadna kolizja funkcji haszującej. Jest to jednak bardzo silne wymaganie, i często zależy nam na słabszej właściwości funkcji (od silniejszych do słabszych właściwości):

- niemożliwość łatwego generowania nowych kolizji
- niemożliwość znalezienia, dla danego `m1` takiego `m2`, że `H(m1) = H(m2)`, czyli second preimage resistance (*odporność na znalezienie drugiego przeciwobrazu*);
- niemożliwość znalezienia, dla danego `h` takiego `m` że `H(m) = h`, czyli preimage resistance.

Generowanie kolizji

Jeśli funkcja haszująca zwraca k bitów, to zgodnie z paradoksem dnia urodzin[1] sprawdzenie wśród zbioru losowo wybranych wiadomości rozmiaru rzędu $2^{k/2}$ prawdopodobnie istnieje jakaś kolizja. Jest to zasada działania ataku dnia narodzin.

Tak więc znalezienie kolizji 128-bitowej funkcji haszującej (takiej jak MD5) jest zadaniem o trudności porównywalnej ze znalezieniem klucza 64-bitowego szyfru symetrycznego. Nie jest to zadanie trywialne, aczkolwiek znajduje się w zasięgu możliwości współczesnego sprzętu i sieci rozproszonych. Znajdowanie kolizji funkcji 160-bitowych (SHA1, RIPEMD160) jest równie trudne jak łamanie 80-bitowego szyfru symetrycznego, i jest na dzień dzisiejszy uważane za zbyt trudne.

Liczby te dotyczą tylko sytuacji w której funkcją haszującą posługujemy się jako "czarną skrzynką", tzn. nie korzystamy z wiedzy o jej strukturze. Wykorzystując słabości struktury możemy często znajdować kolizje o wiele szybciej (np. dla MD4 kolizje można znaleźć w czasie rzeczywistym).

---

[1] Pytanie stawiane w **paradoksie dnia urodzin** brzmi: *Ile osób należy wybrać, żeby prawdopodobieństwo, że co najmniej dwie z nich mają urodziny tego samego dnia było większe od jednej drugiej*. Odpowiedzią jest zaskakująco niskie 23. Ogólniej - jeśli losowo przyporządkujemy każdemu obiektowi jedną z n etykietek, to żeby prawdopodobieństwo że dwa obiekty będą oznaczone taką samą etykietką było większe od jednej drugiej trzeba zbioru obiektów o liczności rzędu $\sqrt{n}$.

Użycie paradoksu dnia urodzin ma znaczenie w kryptografii - i jest zasadą działania tzw. ataku dnia narodzin; np. jeśli funkcje haszujące zwracają 2^k możliwych odpowiedzi, to znalezienie kolizji, czyli dwóch takich wiadomości m_1 i m_2, że $H(m_1) = H(m_2)$ wymaga sprawdzenia jedynie $2^{k/2}$ obiektów. Np. żeby znaleźć kolizję dla MD5 z prawdopodobieństwem większym od 50% trzeba by sprawdzić około 2^{64} losowych wiadomości. Znajdowanie przeciwobrazu czy drugiego przeciwobrazu jest równoważne atakowi na wszystkie bity funkcji haszującej, dlatego też 128-bitowe MD5 jest uważane za bezpieczne jeśli zależy nam tylko na tych właściwościach, choć nie chroni przed kolizjami.

SHA-1

SHA (**Secure Hash Algorithm**) - rodzina powiązanych ze sobą funkcji haszujących zaprojektowanych przez NSA (National Security Agency) i publikowanych przez NIST (National Institute of Standards and Technology). Pierwszy z nich opublikowany w 1993 oficjalnie nazwany SHA (nieoficjalnie, żeby nie pomylić z następcami określany jako SHA-0). SHA-1 opublikowany został w 1995 i całkowicie zastąpił wycofanego (ze względu na nieujawnione oficjalnie wady) z użytku SHA-0. Potem od 2001 powstały cztery następne warianty określane jako SHA-2 (SHA-224, SHA-256, SHA-384, SHA-512). W 2004 zgłoszono udane ataki na funkcje haszujące mające strukturę podobną do SHA-1 co podniosło kwestię długotrwałego bezpieczeństwa SHA-1. NIST ogłosił, że do 2010 zaprzestanie stosować SHA-1 na rzecz różnych wariantów SHA-2. SHA-0 i SHA-1 tworzą 160-bitowy skrót z wiadomości o maksymalnym rozmiarze 2^{64} bity i jest oparty o podobne zasady co MD5. Podstawowym celem publikacji SHA był Standard Podpisu Cyfrowego (Digital Signature Standard), którego SHA był częścią. SHA jest podstawą szyfru blokowego SHACAL. Cały mechanizm ochrony przed nielegalnym kopiowaniem microsoftowego Xboksa (konsola gier wideo) zależy od bezpieczeństwa SHA-1 jeśli zostanie przeprowadzony skuteczny atak wykorzystujący przeciwobraz będzie możliwe dowolne kopiowanie gier bez pomocy modchipa[2].

RIPEMD-160 jest zmodyfikowanym algorytmem RIPEMD, który w swojej pierwotnej postaci powstał w ramach projektu Unii Europejskiej o nazwie RIPE (RACE Integrity Primitives Evaluation) realizowanego w latach 1988-1992. Projektantami algorytmu są Hans Dobbertin, Antoon Bosselaers oraz Bart Preneel. Wersja przekształcająca wiadomość o dowolnej długości na stałej długości skrót 160-bitowy powstała w roku 1996.

---

[2] **Modchip** - zmodyfikowany chip tak aby móc na danej maszynie uruchomić nieautoryzowane programy:

- programy z CD-R;
- własnoręcznie napisane gry;
- zmodyfikowane gry;
- filmy DVD z regionem inny niż nasz odtwarzacz.

Powstały modchipy do:

- PSX (***PlayStation*** – StacjaGier) - 32-bitowa konsola do gier wideo, wyprodukowana w Japonii przez firmę Sony; **PSX2** - kombajn multimedialny, który składał się z konsoli PlayStation 2, czytnika DVD-ROM i zintegrowanego (opcjonalnie) dysku twardego - od tego czasu zaprzestano nazywania pierwszej konsoli firmy *PSX* na korzyść *PS1* lub *PSOne*. Następnym następcą zostało PlayStation 3, które ma o wiele bardziej rozbudowany interfejs, itp.;
- PS2 (***PlayStation* 2)** - konsola do gier telewizyjnych firmy Sony, następca konsoli PlayStation. Pozwala na uruchamianie gier przeznaczonych specjalnie na tę platformę, zapisanych na płytach CD-ROM i DVD-ROM;
- Xbox (konsola gier wideo) oraz niektórych odtwarzaczy DVD.