Kryptografia - lista 1

Oskar Makowski 236554

23 Marzec 2020

1 Zadanie 1

Celem zadania jest skonstruowanie ataku na *linear congruential generator* w ogólnej formie. Jest on zdefiniowany następująco:

$$X_{n+1} = (X_n * a + c)\ mod\ m$$

gdzie względem m wykonuje się operację modulo, a jest mnożnikiem, c jest stałą dodawania, X_0 jest tzw. "ziarnem", rozpoczynającym sekwencję.

Nie znając dobranych wartości X_0, a, c, m przygotowany algorytm powinien dokonywać predykcji kolejnych liczb zwracanych przez generator, tym samym będąc w stanie odróżnić go od źródła prawdziwie losowego. Przyjmuje się założenie że generator zwraca wszystkie uzyskane bity, a także że można zadać mu wielomianową liczbę pytań, uzyskując wielomianowo wiele próbek.

Wychodząc od prostszych przypadków, przedstawione zostaną metody odzyskiwania parametrów, które złożą się na ostateczny atak.

1.1 Nieznane c

Korzystając z definicji:

$$X_1 = (X_0 * a + c)\ mod\ m$$

toteż znając resztę parametrów można dokonać przekształcenia do postaci

$$c = X_1 - X_0 * a\ (mod\ m)$$

gdzie wszystkie wartości po prawej stronie są już znane.

1.2 Nieznane c, a

Korzystając z definicji:

$$X_1 = (X_0 * a + c)\ mod\ m$$

$$X_2 = (X_1 * a + c)\ mod\ m$$

Odejmując równanie górne od dolnego:

$$X_2 - X_1 = (X_1 * a + c) - (X_0 * a + c)\ mod\ m$$

$$X_1 - X_0 = (X_1 - X_0)a\ mod\ m$$

$$a = (X_2 - X_1)/(X_1 - X_0)\ mod\ m$$

Znając a problem został zredukowany do tego z części 1.1. Wyznaczenie a wymaga dzielenia modulo, tym samym potrzebny jest algorytm wyznaczania odwrotności modulo. Przykładową implementację można znaleźć np. tutaj [1]

Zauważmy, że nie potrzeba znać X_0, bowiem dzięki definicji można uzyskać równość dla wyrazu X_3 i dokonać analogicznych przekształceń.

1.3 Brak znanych parametrów

Posiadając m problem redukuje się do poprzedniego przypadku. Aby je odzyskać należy skorzystać z faktu teorii liczb: *gcd* losowych wielokrotności liczby m wyznacza samo m, gdzie funkcja *gcd* wyznacza największy wspólny dzielnik. Jak wielu liczb potrzeba, by uzyskać m z niepomijalnie małą przewagą? Zakładając możliwość uzyskania wielomianowej ilości próbek z generatora można przeprowadzić test zliczania, jaka wartość pojawia się najczęściej i ją wypróbować jako m, następnie wartość drugą co do częstości występowania itd.

Powyższy fakt jest pożyteczny, bowiem jeśli $x \neq 0$ i $x = 0\ mod\ m$ to wtedy z definicji x ma postać $x = km$ dla jakiegoś k. Obliczając gcd dla kolejnych par wyrazów z ciagu takich x-ów dąży się do odzyskania m.

Wprowadźmy ciąg

$$T_n = X_{n+1} - X_n$$

$$T_0 = X_1 - X_0$$

$$T_1 = X_2 - X_1 = (X_1 * a + c) - (X_0 * a + c) = a(X_1 - X_0) = a * T_0\ mod\ m$$

$$T_2 = X_3 - X_2 = (X_2 * a + c) - (X_1 * a + c) = a(X_2 - X_1) = a * T_1\ mod\ m$$

Stąd

$$T_2 * T_0 - T_1 * T_1 = m^2 T_0^2 - m^2 T_0^2 = 0\ mod\ m$$

Wartości T_i można poznać, pobierając kolejne próbki z generatora. Następnie obliczając *gcd* dla kolejnych wyrazów postaci $T_2 * T_0 - T_1 * T_1$ odzyskujemy m. Wtedy problem wyznaczenia pozostałych parametrów redukuje się do poprzednich przypadków.

[1] https://cs.pwr.edu.pl/cichon/2016_17_a/Algebra01.php - wykład z 19.10.2016

2 Zadanie 2

Przykładową implementację *glibc's random()* można zobaczyć tutaj [2]. Skupmy się na pierwszej części, działającej jak zdefiniowany powyżej generator z parametrami $X_0 = 1, a = 16807, m = 2147483647, c = 0$. Jedyną różnicą w zachowaniu generatora jest obcięcie najmniej znaczącego bitu za pomocą operacji przesunięcia bitowego w prawo o 1. Oznacza to, ze otrzymywane próbki są 2 razy mniejsze lub 2 razy mniejsze i zmniejszone o 1(obcięcie najmniej znaczącego bitu to $\lfloor \frac{x}{2} \rfloor$. Dla takiej implementacji nie komplikuje to znacząco ataku na tę część, która działa tak samo jak generator z zadania pierwszego. Otrzymywane liczby należy pomnożyć przez 2 lub pomnożyć i jeszcze dodać 1, tzn. wiemy, że otrzymując y, przed obcięciem było to $2y$ lub $2y + 1$, toteż dla każdej liczby są dwa przypadki. Wystarczy "wypróbować" jeden z możliwych ciągów wartości do otrzymania ukrytych parametrów, a następnie dokonać predykcji - jeśli była poprawna, wartości w ciągu zostały odpowiednio dobrane, jeśli nie - należy wybrać kolejny ciąg. Ciąg długości 10 powinien wystarczyć do przeprowadzenia ataku.

[2]https://www.mathstat.dal.ca/ selinger/random/