Ataki na RSA

Andrzej Chmielowiec

andrzej.chmielowiec@cmmsigma.eu

Centrum Modelowania Matematycznego Sigma

Plan prezentacji

- Wprowadzenie
- Ataki algebraiczne
- Ataki z kanałem pobocznym
- Podsumowanie

Algorytm RSA

Bezpieczeństwo algorytmu RSA zostało oparte na problemie faktoryzacji liczb całkowitych.

- Klucze: generujemy liczby pierwsze p, q, wyznaczamy $N = pq$ i szukamy takich liczb d (klucz prywatny) oraz e (klucz publiczny), że $ed \equiv 1 \bmod (p-1)(q-1)$.
- Operacja prywatna: $C \equiv M^d \bmod N$.
- Operacja publiczna: $M \equiv C^e \equiv M^{ed} \bmod N$.

Optymalizacja operacji prywatnej

Wykonanie operacji prywatnej może być w bardzo prosty sposób przyspieszone. Wystarczy wykorzystać twierdzenie chińskie o resztach:

- $C_p \equiv M^{d \bmod (p-1)} \bmod p$,
- $C_q \equiv M^{d \bmod (q-1)} \bmod q$,
- $$\begin{cases} C \equiv C_p \bmod p, \\ C \equiv C_q \bmod q. \end{cases}$$

Optymalizacja operacji publicznej

Aby przyspieszyć wykonywanie operacji publicznej postanowiono zastosować *krótkie* wykładniki publiczne. W powszechnym użyciu są takie wartości, jak:

- 3,
- 5,
- 7,
- 65537 (najpopularniejszy).

Faktoryzacja modułu

Faktoryzacja modułu jest podstawową metodą algebraiczną, która zapewnia uzyskanie wszystkich informacji na temat klucza RSA.

Problem: Mając daną liczbę całkowitą N należy wyznaczyć jej czynniki pierwsze.

Algorytmy faktoryzacji

Do najpopularniejszych algorytmów faktoryzacji należą

- MPQS (Multiple Polynomial Quadratic Sieve) - modyfikacja sita kwadratowego,
- ECM (Elliptic Curve Method) - metoda szczególnie skuteczna przy niewielkich czynnikach,
- GNFS (General Number Field Sieve) - najszybszy znany algorytm faktoryzacji.

Jakie liczby można faktoryzować

Oficjalnie największym rozłożonym do tej pory modułem RSA jest RSA-200 (moduł mający 200 cyfr dziesiętnych – około 660 bitów).

Liczby 500 bitowe mogą być w tej chwili rozkładane na czynniki w przeciągu kilku miesięcy przy użyciu niewielkiej liczby komputerów (kilka, kilkanaście), a odpowiednie darmowe oprogramowanie można ściągnąć z internetu.

Dodatkowe warunki na czynniki modułu RSA

Aby uniemożliwić zastosowanie specjalnych metod faktoryzacji (dużo szybszych niż metody ogólne) nakłada się na czynniki RSA dodatkowe warunki:

- $p-1$, $q-1$, $p+1$ i $q+1$ mają mieć duży czynnik pierwszy,
- $|p-q| > 2^{\log_2 p - 100}$.

Ataki na krótki wykładnik prywatny

Zmniejszenie długości klucza prywatnego ma korzystny wpływ na czas wykonywania operacji prywatnej. Takie podejście mogłoby być bardzo atrakcyjne zwłaszcza w przypadku urządzeń o ograniczonych zasobach obliczeniowych. Niestety dla wszystkich kluczy prywatnych

$$d < N^{0.3}$$

istnieje efektywna metoda złamania RSA.

Ataki na krótki wykładnik publiczny

(Coppersmith) Niech N będzie liczbą całkowitą, a $f(X) \in \mathbb{Z}_N[X]$ wielomianem stopnia d, którego współczynnik przy wyrazie X^d wynosi 1. Ustalmy $x = N^{\frac{1}{d}-\epsilon}$ dla pewnego $\epsilon \geq 0$. Wtedy istnieje efektywny sposób wyznaczenia wszystkich liczb $x_0 < x$ takich, że $f(x_0) = 0 \bmod N$. Czas znalezienia pierwiastków zależy od wydajności algorytmu LLL dla kraty o wymiarze $O(w)$, gdzie $w = \min(1/\epsilon, \log_2 N)$.

Atak Hastada

Załóżmy, że ta sama wiadomość M wysyłana jest do trzech różnych osób, z których każda posługuje się kluczem publicznym postaci $\langle N_i, 3 \rangle$. W takiej sytuacji wysyłane są 3 szyfrogramy:

$$\begin{cases} C_1 = M^3 \bmod N_1, \\ C_2 = M^3 \bmod N_2, \\ C_3 = M^3 \bmod N_3. \end{cases}$$

Powyższy układ równań pozwala na odtworzenie wartości $M^3 = M^3 \bmod (N_1 N_2 N_3)$.

Atak Franklina-Reitera

Metoda ta działa tylko w przypadku wykładnika publicznego o wartości 3. Pozwala na złamanie dwóch szyfrogramów $C_1 = M_1^3$ i $C_2 = M_2^3$, które zostały wysłane do tej samej osoby. Warunkiem koniecznym przeprowadzenia skutecznej kryptoanalizy jest znajomość liniowej zależności pomiędzy wiadomościami M_1 i M_2.

Znając funkcję $f(X) = aX + b \in \mathbb{Z}_N[X]$ ($b \neq 0$), dla której prawdziwa jest zależność $M_2 = f(M_1)$ możemy znaleźć M_1 i M_2.

Atak Coppersmitha

Gdy atakujący przechwyci dwa szyfrogramy postaci:

1. $C_1 = (2^m M + r_1)^e$, gdzie $0 \leq r_1 < 2^m$,
2. $C_2 = (2^m M + r_2)^e$, gdzie $0 \leq r_2 < 2^m$,

to jest w stanie odzyskać wiadomość M.

W przypadku 1024-bitowego klucza i wykładnika publicznego równego 3 możliwe jest odzyskanie pierwszych 911 bitów M, o ile losowe dopełnienie jest nie dłuższe niż 113 bitów.

Atak na częściowo jawny klucz prywatny

Jeżeli $N = pq$ jest modułem RSA o długości n-bitów, a wykładnik publiczny e spełnia zależność $e < \sqrt{N}$, to można:

1. znaleźć wszystkie bity d w czasie $e \log_2 e$, o ile znamy $\lfloor n/4 \rfloor$ najmłodszych bitów wykładnika prywatnego d,
2. znaleźć rozkład modułu RSA na czynniki, jeżeli znamy $\lfloor n/4 \rfloor$ najmłodszych lub najstarszych bitów czynnika p.

Ataki czasowe

Czas wykonania poszczególnych instrukcji procesora może być zależny od jej argumentów wejściowych. Sytuacja taka ma miejsce w większości współczesnych procesorów, które wyposażone są w całą gamę mechanizmów optymalizujących wykonywanie kodu. Pomiar czasu wykonania danej operacji kryptograficznej jest więc w pewnym stopniu skorelowany z przetwarzanymi danymi (w szczególności kluczami kryptograficznymi).

Atak czasowy Kochera (1)

Atak pozwala na wyznaczenie wykładnika prywatnego x dla operacji $y^x \bmod N$ pod warunkiem, że znane są y i N.

$s_0 = 1$
for $i = 0$ to $n - 1$ do
    if $x_i = 1$ then $R_i = s_i y \bmod N$
    else $R_i = s_i$
    $s_{i+1} = R_i^2 \bmod N$
return R_{n-1}

Atak czasowy Kochera (2)

Podstawową techniką zabezpieczającą przed tym atakiem jest tak zwane *maskowanie*. Polega ono na pomnożeniu argumentu wejściowego x przez pewną losową wartość r^e, gdzie e jest wykładnikiem publicznym. Wykonując operację prywatną na xr^e otrzymujemy

$$x^d r^{ed} = x^d r \bmod N.$$

Na końcu usuwamy maskę poprzez pomnożenie uzyskanego wyniku przez $r^{-1} \bmod N$.

Atak czasowy na pamięć CACHE

Implementacja mnożenia Montgomery'ego bez instrukcji warunkowych daje wynik modulo $2N$. Aby uzyskać rezultat modulo N należy zatem dokonać redukcji warunkowej (jeżeli wynik jest w przedziale $[N..2N-1]$).

Obecność warunkowej instrukcji redukcji umożliwia złamanie RSA. Podstawą ataku są różnice czasu wykonania operacji prywatnej, które zależą od aktualnego stanu pamięci CACHE instrukcji.

Atak z generacją błędu podczas CRT (1)

Jeżeli wykładnik prywatny ma wartość d i $d_p = d \bmod (p-1)$, $d_q = d \bmod (q-1)$ to operacja prywatna przebiega następująco:

1. $C_p = M^{d_p} \bmod p$,
2. $C_q = M^{d_q} \bmod q$,
3. $C = T_1 C_p + T_2 C_q \bmod N$, gdzie

$$T_1 = \begin{cases} 1 \bmod p \\ 0 \bmod q \end{cases}, \qquad T_2 = \begin{cases} 0 \bmod p \\ 1 \bmod q \end{cases}.$$

Atak z generacją błędu podczas CRT (2)

Celem ataku jest spowodowanie błędu obliczeń, który zmieni na przykład wartość C_q. Otrzymamy zatem podpis $\hat{C}$, który będzie spełniał zależności:

$$\hat{C}^e = M \bmod p \qquad \text{i} \qquad \hat{C}^e \neq M \bmod q.$$

Oznacza to, że gcd $\left(\hat{C}^e - M, N\right)$ jest nietrywialnym czynnikiem N.

Podsumowanie

- Istnieje wiele potencjalnych ataków na algorytm RSA.
- Przygotowanie bezpiecznej implementacji wymaga dużej wiedzy i doświadczenia.
- Zła optymalizacja może znacznie pogorszyć bezpieczeństwo implementacji.