UNIWERSYTET WARSZAWSKI
WYDZIAŁ MATEMATYKI, INFORMATYKI I MECHANIKI

Ciekawe problemy kryptografii asymetrycznej

Piotr Gaweł
nr indeksu 174718

REFERAT DO WYKŁADU MONOGRAFICZNEGO
Algorytmiczne aspekty kryptografii

WARSZAWA, MAJ 2002

Spis treści

1 Wprowadzenie

1.1 Wstęp

Wynalezienie technik kryptografii asymetrycznej stworzyło nowe, nieznane dotąd możliwości zabezpieczania informacji, zwłaszcza tych przekazywanych publicznymi kanałami komunikacyjnymi. Początki kryptografii z wykorzystaniem klucza publicznego sięgają oficjalnie roku 1976, jednak jak podaje [4], pierwsze dokumenty na ten temat pojawiły się w zamkniętych kręgach wojskowych już w styczniu roku 1970.

Istotą szyfrów asymetrycznych jest wykorzystywanie pary kluczy - innego do szyfrowania, innego do dekodowania informacji. W zależności od tego, czemu operacja szyfrowania ma służyć, jeden z kluczy (tzw. klucz publiczny) jest, jak sama nazwa wskazuje, ogólnie dostępny, drugi (tzw. klucz prywatny) powinna znać tylko jedna strona. Bezpieczeństwo szyfrów asymetrycznych opiera się zatem na trudności odwracania pewnej funkcji jednokierunkowej sprowadzając się do problemu odzyskania klucza prywatnego na podstawie znajomości klucza publicznego i egzemplarzy zaszyfrowanych informacji.

Obecnie znanych jest już wiele implementacji szyfrowania asymetrycznego, ze względu na wybór wykorzystywanych w nich funkcji jednokierunkowych wymienić należy przede wszystkim :

1. Szyfry opierające swoje bezpieczeństwo na problemie faktoryzacji liczb, do nich należy m.in. najbardziej znany i najczęściej wykorzystywany obecnie szyfr **RSA**,

2. Szyfry wykorzystujące problem logarytmu dyskretnego, np. pochodzący z roku 1985 szyfr **El-Gamala**, czy też **DSA**,

3. Coraz popularniejsze szyfry korzystające z problemu logarytmu dyskretnego na krzywych eliptycznych, np. **ECDSA**,

4. Szyfry oparte na znanych problemach NP-trudnych, np. szyfr **Merkle-Hellman'a** wykorzystujacy tzw. problem pakowania plecaka [1],

5. Algebraiczne kryptosystemy wielu zmiennych opierające się na problemie rozwiązywania równań liniowych wielu zmiennych nad ciałami skończonymi. Tworzą one dziedzinę tzw. kryptografii wielu zmiennych.

[1] Oryginalny szyfr Merkle-Hellman'a w wersji z roku 1978 został złamany już 6 lat później przez jednego z autorów RSA, Shamira, który udowodnił możliwość rozwiązania problemu plecakowego w czasie wielomianowym dla pewnej podklasy danych wejściowych.

1.2 Ogólny opis algorytmów szyfrowania i deszyfrowania

W praktyce operacje kodowania/dekodowania informacji dla poszczególnych technik wyglądają następująco :

- **RSA** - p, q - duże liczby pierwsze, podobnej długości. Niech :

  - $n = pq$,
  - $\phi(n) = (p-1)(q-1)$,
  - $e \in Z_{\phi(n)}, d = e^{-1}\, mod\, \phi(n)$

  Kluczem publicznym jest para (e, n), kluczem prywatnym d. Kodowanie wiadomości $m \in Z_n$ polega na obliczeniu $m^e\, mod\, n$, dekodowanie zaszyfrowanej wiadomości c sprowadza się do obliczenia $c^d\, mod\, n$.

- **El-Gamal** - $(G, *)$ niech będzie grupą, oraz :

  - α - generator grupy,
  - $a \in Z_{|G|}, \beta = \alpha^a$

  Kluczem publicznym jest (G, α, β), kluczem prywatnym a. Operacja kodowania polega na wygenerowaniu losowego $k \in Z_{|G|}$, następnie obliczeniu szyfrogramu : $(y_1, y_2) = (\alpha^k, m * \beta^k)$. Dekodowanie wiadomości (y_1, y_2) polega na wykonaniu operacji $y_2 * (y_1^a)^{-1}$, co z kolei równa się $(m * \beta^k) * (\alpha^{ka})^{-1} = m * \beta^k * (\beta^k)^{-1} = m$.

- Szyfry oparte na **krzywych eliptycznych** działają, ze względu na podobne podstawy teoretyczne, analogicznie do opisanego wyżej szyfru El-Gamala z tą różnicą, że wykorzystywaną grupą są punkty krzywej o równaniu $y^2 = x^3 + ax + b$ z określoną operacją dodawania.

- Szyfr **Merkle-Hellman'a** - dany ciąg bardzo rosnący[2] W. Następnie tworzymy nowy ciąg W' : $w'_i = a * w_i\, mod\, p$, gdzie a oraz p są stałymi, dodatkowo p jest pierwsza.

  Klucz publiczny stanowi ciąg W', klucz prywatny - a oraz p. Wiadomość M w postaci zakodowanej jest sumą $\sum_{i=0}^{i \leq n} w'_i m_i$. Natomiast mając a oraz p możemy z powyższej sumy odzyskać wiadomość M.

- Spośród szyfrów stanowiących tzw. kryptografię wielu zmiennych do bardziej znanych należy występujący w wielu różnych odmianach **HFE** *(Hidden Field Equation)*. Ze względu na budowę znacznie bardziej złożoną w porównaniu do wymienionych wyżej kryptosystemów, opisowi HFE poświęcony został w całości rozdział 4.

[2]Def. Ciąg a jest bardzo rosnący (ang. *superincreasing*), jeśli $\forall a_i\ a_i \geq \sum_{j=0}^{i-1} a_j$.

2 Łamanie systemów kryptograficznych

2.1 Wstęp

Fakt, iż nie istnieją systemy absolutnie bezpieczne, jest jednym z częściej powtarzanych stwierdzeń na temat kryptografii. Istotnie, wszystkie istniejące kryptosystemy swoje bezpieczeństwo opierają na braku znanych metod które umożliwiałyby ich łamanie w możliwym do praktycznego zastosowania czasie oraz nadziei, że metody takie nie istnieją. Powoduje to jednak, że w przypadku odkrycia algorytmu rozwiązującego szybko np. problem faktoryzacji, wszelkie dane zaszyfrowane RSA stałyby się jawne. Podobnie z innymi otwartymi problemami matematycznymi.

Poza tym, fakt iż nie jesteśmy w stanie obecnie szybko rozkładać liczb na czynniki, nie powoduje wcale, iż systemy opierające się o faktoryzację są w ogólnym przypadku bezpieczne. Przede wszystkim, przy korzystaniu z każdego rodzaju szysfrowania, uwzględnić należy wiele przypadków szczególnych, w których deszyfracja wiadomości może okazać się znacznie łatwiejsza niż w przypadku ogólnym. I tak w przypadku RSA, żadna z losowanych liczby pierwszych nie powinna być zbyt mała - powinny być one mniej więcej podobnej długości. Z drugiej strony, nie mogą być zbyt bliskie sobie, gdyż algorytm „siłowy" rozpoczynający proces zgadywania od $\sqrt{n}$ mógłby takie liczby szybko odnaleźć.

Po drugie, w przypadku odpowiednio krótkiego klucza szyfrującego oraz odpowiedniej ilości dostępnej mocy obliczeniowej, można próbować taki szyfr łamać jeśli nie metodą „siłową", to dużo szybszymi różnego rodzaju heurystykami. Choć są to algorytmy wielokrotnie szybsze od badania wszystkich możliwych przypadków, to jak sama nazwa wskazuje, są probabilistyczne. Odnoszą one jednak sukces wystarczająco często, aby uznać je za najbardziej zaawansowane narzędzia w walce z otwartymi problemami matematycznymi.

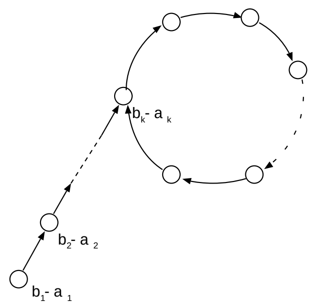
2.2 Heurystyka „rho" Pollard'a

W celu rozłożenia liczby na czynniki pierwsze możemy zastosować heurystykę znaną pod nazwą „rho" pochodzącą od greckiej litery ρ. Istnieje jej wiele odmian, z których część jest używana w praktyce do łamania kodów. W wersji oryginalnej jej działanie wygląda następujaco :

Niech n będzie rozkładaną liczbą. Definiujemy sobie ciąg a_i, b_i następująco :

- $a_1 = b_1 = 2,\ f(x) = x^2 + 1$
- dla $i > 1$ $a_i = f(a_{i-1}),\ b_i = f^2(b_{i-1})$

- dla kolejnych wartości i sprawdzamy wartość $NWD(b_i - a_i, n)$. Jeśli jest ona większa od 1, otrzymaliśmy nietrywialny dzielnik n. W przeciwnym przypadku przechodzimy do następnej iteracji. Ponieważ kiedyś, po skończonej liczbie iteracji, wartości a_i oraz b_i się powtórzą, nie sprawdzimy wszystkich dzielników n, lecz wpadniemy w pętlę, jak obrazuje to poniższy rysunek :



Rys.1. Heurystyka „rho"[3].

Jest zatem możliwe, że rozkładana liczba oprze się naszej heurystyce. Jednakże czas działania heurystyki jest znacznie mniejszy od n - jeśli $n = pq$, to czynnik q powinien zostać znaleziony średnio po $\sqrt{q}$ iteracjach. Zatem w interesującym nas przypadku, gdy p i q są podobnej długości, oczekiwany czas działania heurystyki wynosi $\sqrt[4]{n}$. W praktyce oznacza to czas wystarczająco długi, aby RSA uznać za bezpieczny, jednak tylko i wyłącznie w przypadku korzystania z klucza dłuższego niż 512 bitów[4].

2.3 Heurystyka „rho" dla problemu logarytmu dyskretnego

Problem logarytmu dyskretnego w ogólności wygląda następująco :
- Niech G - grupa, P - generator, $d \in G$ - losowa wartość, $1 \leq d \leq n-1$
- Kluczem publicznym jest $Q = d * P$, G, P, prywatnym - d.
- W celu złamania szyfru trzeba policzyć d znając P, Q, G.

Heurystyka działa w następujący sposób :
- Wybieramy dwie losowe liczby $a_0, b_0 \in G$ takie, że $1 \leq a_0, b_0 \leq n-1$;
- Dzielimy G na 3 równe podgrupy S_1, S_2, S_3;
- Niech $X_0 = a_0 * P + b_0 * Q$;
- Dla $i > 0$ mamy

$$X_i = \begin{cases} P + X_{i-1} & \text{jeśli } X_{i-1} \in S_1 \\ 2 * X_{i-1} & \text{jeśli } X_{i-1} \in S_2 \\ Q + X_{i-1} & \text{jeśli } X_{i-1} \in S_3 \end{cases}$$

[3]Nazwa pochodzi od cyklu, w jaki wpadają liczby tworząc kształt greckiej litery ρ.
[4]Stan na maj 2002 r.

- Przyjmując $X_i = a_i * P + b_i * Q$, z X_i otrzymujemy iteracyjnie nowe wartości a_i, b_i. Jeśli teraz zdarzy się dla pewnych $i \neq j$, że $X_i = X_j$, otrzymujemy :

$$a_i * P + b_i * Q = a_j * P + b_j * Q,$$

a ponieważ $Q = d * P$, to :

$$d = \frac{a_i - a_j}{b_i - b_j} \, mod \, n$$

Oczekiwany czas działania powyższej heurystyki wynosi $\sqrt{\frac{\pi n}{2}} * t$, gdzie t jest kosztem obliczenia kolejnej wartości X_i.

2.4 Przypadki heurystycznego złamania kryptosystemów

Używając wyżej opisanej heurystycznej metody obliczania logarytmu dyskretnego, w marcu 1998 roku złamano problem logarytmu dyskretnego na krzywych eliptycznych dla długości klucza 97 bitów. Operacji złamania szyfru pochłonęła $2.0 * 10^{14}$ operacji dodawania na grupach. Użyto w tym celu ponad 1200 komputerów spośród 16 krajów. Każdy z klientów uruchomił niewielki (40 kilobajtów) program liczący, który co pewien stały czas przesyłał wyniki swoje pracy do centralnego serwera. Program ten wykonywał zmodyfikowaną wersję heurystyki „rho", korzystającej w celu zwiększenia wydajności z 16 różnych iteratorów. Cała operacja zakończyła się sukcesem po 53 dniach obliczeń - było to dosyć szybko, wartość oczekiwana czasu niezbędnego do złamania szyfru wynosiła 85-90 dni. Wniosek z całego przedsięwzięcia potwierdził oczekiwany czas pracy użytej heurystyki wynikający z teorii algorytmicznej : praktyka pokazała, iż istotnie złożoność heurystyki „rho" dla rozwiązania problemu logarytmu dyskretnego na krzywych eliptycznych oscyluje wokół $\sqrt{\frac{\pi * n}{4}}$ [1].

Kolejnym wyzwaniem ekipy BT Laboratories, która w największym stopniu przyczyniła się do powyższego przedsięwzięcia, miało być złamanie kolejnego krzywoeliptycznego kryptosystemu o długości klucza 109-bitów. Według wstępnych obliczeń 12000 maszyn typu Pentium 200 MHz powinno podołać temu zadaniu w niecały rok.

Do łamania RSA używane są podobnie różnego heurystyki. I tak, w roku 1994 użyto metody znanej pod nazwą *Quadratic Sieve* do sfaktoryzowania 428-bitowej liczby. Metoda ta działa w czasie $T(n) = e^{(1+o(n))\sqrt{lnn * ln(lnn)}}$, co w praktyce oznaczało użycie 1600 komputerów pracujących przez 8 miesięcy.

W 1999 roku używając algorytmu *Number field Sieve*, działającego w oczekiwanym czasie $T(n) = e^{(1.923+o(1))\sqrt[3]{lnn}*(ln(lnn))^{\frac{2}{3}}}$, rozłożono 512-bitową liczbę, co zajęło 35 CPU-lat. Metoda okazała się w praktyce ponad czterokrotnie szybsza od QS.

2.5 Praktyczne stosowanie rozwiązań „siłowych"

2.5.1 Wstęp

Okazuje się, iż pomimo złożoności wykładniczej czasem nawet rozwiązania okreslane mianem *brutal force* są możliwe do realnego stosowania. Mało tego, dla odpowiednich danych może to być nawet operacja o czasie niezależnym od długości klucza! Otóż, jak podają [2], jeśli kodowana informacja jest odpowiednia krótka (rzędu wielkości kilkudziesięciu bitów[5]), to przy pewnych dodatkowych założeniach możemy taką wiadomość rozłożyć w całkiem rozsądnym czasie (kilkanaście minut - kilka godzin na pojedynczym komputerze). Jeśli wiadomość jest długości m bitów, to na przykład :

- dla klucza publicznego RSA $< N, e >$, $C = M^e \, mod \, N$ można odtworzyć oryginalną wiadomość M w czasie $2*2^{\frac{m}{2}}$ przy użyciu $2^{\frac{m}{2}}m$ bitów pamięci. Algorytm kończy się sukcesem z prawdopodobieństwem $\frac{18}{100}$ (ze względu na wiadomość M).

- jeśli $< p, g, y >$ jest publicznym kluczem El-Gamal'a, a rząd g jest conajmniej $\frac{p}{2^m}$, to sytuacja jest analogiczna - odzyskujemy M z prawdopodobieństwem również $\frac{18}{100}$ przy tym samym asymptotycznym czasie i pamięci.

Powyższe wartości prawdopodobieństwa sukcesu można zwiększyć, ceną jednak jest znaczny wzrost kosztu zarówno czasowego, jak i pamięciowego.

2.5.2 Opis algorytmu

Opisywana metoda ponosi sukces przy założeniu, że wiadomość M można rozłożyć na czynniki :

$$M = M_1 * M_2, \text{ przy czym } M_1 \leq 2^{m_1},\ M_2 \leq 2^{m_2}$$

Właśnie stałe m_1 oraz m_2 ograniczają czas pracy algorytmu, jego wymagania pamięciowe oraz prawdopodobieństwo sukcesu, które jest tożsame z prawdopodobieństwem, że wiadomość M można w ten sposób rozłożyć.

[5]Tak krótkie wiadomości występują np. jako klucze sesyjne w protokole HTTPS

I tak, na przykład, dla RSA algorytm działa następująco :

$$c = M^e \, mod \, N = M_1^e * M_2^e \, mod \, N$$

innymi słowy :

$$\frac{c}{M_2^e} = M_1^e \, mod \, N$$

Teraz budujemy tablicę rozmiaru $2^{m_1} + 1$ zawierającą wartości M_1^e dla kolejno $M_1 = 0, \ldots, 2^{m_1}$. Dla każdego $M_2 = 1, \ldots, 2^{m_2}$ sprawdzamy, czy wartość $\frac{c}{M_2^e} \, mod \, N$ jest obecna w tabeli.

W ten oto prosty sposób jesteśmy w stanie odczytać oryginalną wiadomość. Dla szyfru El-Gamal'a algorytm działa analogicznie.

Jak podają autorzy [2], przeprowadzone testy wykazały, iż na pojedynczym dosyć silnym komputerze (500 MHz 64-bit DEC/Alpha) dla długości M równej 40 bitów oraz wartości m_1, m_2 równych 20 rozkodowywanie pojedynczej wiadomości trwa ok. 10 minut. Natomiast wiadomość 64-bitową zaszyfrowaną El-Gamal'em odkodowano używając PII-400 MHz w czasie 2,5 godziny[6].

Wniosek z powyższych badań jest jednoznaczny : przy kodowaniu krótkich wiadomości (tzn. krótszych od używanych powszechnie kluczy) należy koniecznie stosować pewną obróbkę wstępną kodowanej wiadomości. Może to być jej wydłużenie poprzez dodanie entropii lub też inna operacja, która uniemożliwi łatwą jej faktoryzację.

3 Porównanie wydajności RSA-DSA-ECDSA

Przy podjęciu decyzji, którą metodę zastosować w konkretnym przypadku, powstaje naturalne pytanie o ogólną wydajność poszczególnych metod. Obiektywne porównanie sprowadza się do zbadania czasu „legalnych" operacji przy takich długościach kluczy, które oferują podobny poziom bezpieczeństwa. Oczywiście założyć należy, że problemy matematyczne, na których poszczególne techniki się opierają, nie zostaną prędko rozwiązane - takie wydarzenie spowodowałoby, że tego typu porównanie traci wszelki sens.

W przypadku RSA, używając najlepszej dotychczas metody znanej pod nazwą *General Number Field Sieve*[7], rozłożenie 1024-bitowego szyfru wymaga $3 * 10^{11}$ MIPS-lat[8]. Dla 300 MHz peceta oznacza to czas pracy równy okrągłemu milardowi lat.

[6] Dla tych parametrów maksymalny czas pracy wynosi ok. 14h, więc było to dosyć krótko

[7] Informacja z drugiej połowy 1998 roku

[8] 1 MIPS-rok = rok pracy komputera wykonującego milion operacji/sekundę

Szyfrowanie DSA jest pod tym względem podobne. Ilość obliczeń potrzebnych na sfaktoryzowanie lub obliczenie logarytmu dyskretnego 1024-bitowej liczby jest podobny. Można więc uznać, że poziom bezpieczeństwa odpowiadający 1024-bitowemu RSA zapewnia DSA o długości klucza również 1024 bity.

Nieco inna sytuacja jest z ECDSA. Otóż ten kryptosystem ma tę zaletę, że nie są wciąż znane algorytmy obliczania logarytmu dyskretnego na krzywych eliptycznych działające w czasie podwykładniczym, co nie dotyczy DSA. Można więc pozwolić sobie na stosowanie znacznie krótszych kluczy. I rzeczywiście, biorąc pod uwagę koszt operacji dodawania punktów na krzywych, otrzymujemy łączny czas obliczenia logarytmu analogiczny jak czas złamania 1024-bitowego RSA przy długości klucza 138 bity. Jednakże tu pojawia się dodatkowy problem : atak nie musi być wykonywany programowo przez komputer. Metoda, której można tu użyć, *Parallel Collision Search*, ma bardzo małe wymagania pamięciowe, można ją zatem zaimplementować również na niewielkich chipach. A trzeba wziąć pod uwagę, iż potencjalny włamywacz nie będzie koniecznie używał do łamania szyfru peceta, lecz sprzęt, który przy danym koszcie zapewni najwyższą możliwą wydajność. Stan na rok 1998 był następujący : przyjmując koszt użycia PC do obliczeń na 250 $, za tą samą kwotę możemy nabyć 16 sztuk 64MHz specjalizowanych chipów, które są razem od takiego peceta znacznie szybsze[9]. Tak więc w świetle tego, poziom bezpieczeństwa podobny do 1024-bitowego RSA zapewni nie klucz 138, lecz ok.170-bitowy [3].

Tak więc, mamy podstawy, by porównać ze sobą sprawność wykonywania podstawowych operacji : podpisu, weryfikacji i generacji kluczy dla 1024-bitowego RSA, 1024-bitowego DSA oraz 168-bitowego ECDSA. Wyniki podane w milisekundach na maszynie 200MHz Pentium Pro obrazuje poniższa tabela :

	RSA-1024	**DSA-1024**	**ECDSA-168**
podpis	43	7	5
weryfikacja	0.6	27	19
generacja kluczy	1100	7	7

Tabela 1. Koszty podstawowych operacji kryptograficznych

[9] 200MHz pecet jest w stanie wykonać ok. 2^{39} operacji dodawania na krzywych w przeciągu roku, opisywany chip - 2^{51}

Powyższe dane mogą być pewną wskazówką, do czego jakie kryptosystemy lepiej się nadają :

1. **Systemy certyfikujące**. Niezbędna szybka weryfikacja ewidentnie wskazuje na RSA jako zwycięzcę.

2. **Szyfrowanie poczty**. Dowolny. Jednak w przypadku wysyłania zbiorowej poczty, gdy jedna maszyna koduje wszystkie wiadomości, a rozkodowywanie ma miejsce na wielu, przewagę ma znowu RSA, które jest znacznie szybsze w kodowaniu, a wolniejsze w rozkodowywaniu od pozostałych dwóch.

3. **Karty chipowe**. Systemy oparte na krzywych eliptycznych zdecydowanie lepiej się nadają do tego celu ze względu na brak konieczności implementacji operacji mnożenia, co pozwala zmniejszyć rozmiary układu scalonego.

4. **Komunikacja bezprzewodowa**. Wysokie koszty takiej komunikacji po raz kolejny wskazują na krzywe eliptyczne ze względu na korzystny rozmiar klucza.

Należy jednak zauważyć, iż jest to klasyfikacja bardzo ogólna. W każdej konkretnej sytuacji, gdy dochodzi do wyboru metody szyfrowania, należy po raz kolejny dokładnie rozważyć koszty użycia rozmaitych dostępnych technik.

4 Kryptografia wielu zmiennych

Jak to określił sam Adi Shamir, *RSA opiera się na jednym równaniu modularnym z jedną zmienną. Naturalnym uogólnieniem tego podejścia jest rozważanie systemów wielu równań z wieloma zmiennymi.*

Tzw. kryptografia wielu zmiennych stanowi osobną jakość w dziedzinie technik kryptograficznych. Nie rozważamy tu kluczy w postaci pojedynczych liczb, lecz operujemy na równaniach określonego stopnia. Ma to swoje konsekwencje - klucze są duże, lecz istniejące od końca lat 70-tych kryptosystemy wielu zmiennych skutecznie oparły się próbom kryptoanalizy, wyróżniając problemy takie jak SD czy MinRank, dla których wszystkie znane do tej pory ataki są wykładnicze [4].

Jedną z bardziej znanych tego typu technik jest HFE oparte o problem rozkładu wielomianów na ciałach skończonych. Matematyczne podstawy tego kryptosystemu są następujące :

Niech P będzie wielomianem nad ciałem $GF(q^n)$ postaci :

$$P(a) \ = \ \sum_i \alpha_i * a^{q^{s_i}+q^{t_i}}$$

Można wtedy P przedstawić w postaci n wielomianów drugiego stopnia n zmiennych. Następnie, niech S oraz T będą losowymi przekształceniami afinicznymi (czyli przemianowaniami zmiennych). Definiujemy sobie teraz następną funkcję :

$$F \ = \ T \circ P \circ S$$

Mając dany wielomian P możemy go odwrócić w czasie $d^2(ln\, d)^{O(1)}n^2$ operacji na ciele $GF(q)$, gdzie d jest stopniem wielomianu P. Natomiast trudno jest rozszyfrować funkcję F, dopóki nie znamy S, T ani P^{-1}. F bowiem wygląda dla osoby nie znającej klucza prywatnego jak losowy układ równań drugiego stopnia.

Tak więc funkcję F (która de facto jest układem równań drugiego stopnia) możemy traktować jako klucz publiczny, a trójkę T, P, S jako klucz prywatny. Może w tym przypadku zadziwić wielkość kluczy. Rozmiar klucza publicznego wynosi $n(1 + n + \frac{n(n-1)}{2} + n(q > 2))$. W praktyce oznacza to dla $n = 80$ objętość 31,6 Kbitów, dla $n = 128$ jest to 129 Kbitów, a dla $n = 256$ 1 Mbit.

W stosunku do klucza publicznego, rozmiar klucza prywatnego jest znikomy, bo rzędu $2n^2$. Pomimo jednak ogromnego klucza publicznego, można ten kryptosystem stosować praktycznie i w rozsądnym czasie zarówno kodować, jak i odkodowywać dane - o ile jednak zna się odpowiedni klucz. Praktyka bowiem potwierdza, że łamanie HFE jest niezwykle trudne. Nagroda w wysokości 1000 $ wciąż jest do zdobycia - należy jedynie rozwiązać podany układ równań o parametrach $n = 36$, $d = 4352$ nad ciałem $K = GF(16)$. Więcej informacji na ten temat można znaleźć w [5]. W każdym razie do tej pory znanych ataków nie było, co najlepiej świadczy o poziomie tej techniki.

5 Wykorzystanie technik asymetrycznych

Poza ogólnie znanymi i stosowanymi powszechnie zastosowaniami kryptografii asymetrycznej takimi, jak uwierzytelnianie czy kodowanie informacji, istnieje wiele możliwości wykorzystania faktu, iż niemalże niemożliwe jest otrzymanie klucza szyfrującego z deszyfrującego i odwrotnie.

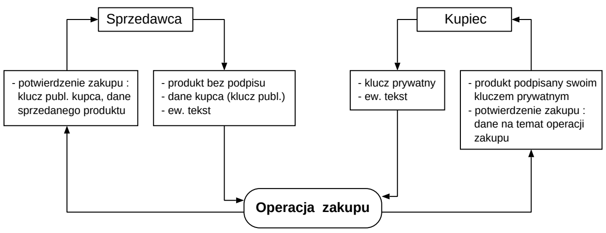
Ciekawym problemem, w rozwiązaniu którego może pomóc kryptografia asymetryczna, jest zabezpieczanie danych elektronicznych przed kopiowaniem. Istnieją bowiem dwa możliwe podejścia do tego problemu. Można próbować uniemożliwić fizyczne nielegalne skopiowanie danych (stosując np. klucze sprzętowe). Praktyka pokazuje jednak, że jest to praktycznie niewykonalne, o czym można się przekonać odwiedzając Stadion Dziesięciolecia. Pirackie wersje nawet reklamowanego jako niemożliwe do złamania Windows XP pojawiły się już tydzień po premierze.

Interesujące więc wydaje się być drugie podejście do problemu praw autorskich - kopiowanie fizycznie jest możliwe, jednak dzięki ukrytym znakom i podpisom producent może na podstawie posiadanej nielegalnej kopii sprawdzić i udowodnić przed niezależnym arbitrem (np. sądem), kto tę kopię rozpowszechnił. Jak można się domyślać, w opracowaniu protokołów służących operacjom sprzedaży, sprawdzania i dowodu pochodzenia danej kopii wykorzystane zostaną techniki szyfrowania asymetrycznego. Nie będziemy tu zajmować się technikami ukrywania informacji - to jest osobną kwestią odpowiedniej steganografii.

Według protokołu opisanego w [6], kolejne etapy operacji sprzedaży i identyfikacji nielegalnej kopii wyglądają następująco :

1. Generacja kluczy. Klient generuje parę kluczy : (K_{pub}, K_{pr}), następnie przekazuje klucz publiczny sprzedawcy i arbitrowi.

2. Sprzedaż. Reprezentant sprzedawcy wprowadza klucz publiczny klienta, sprzedawany produkt oraz ewentualnie dodatkowe dane (np. treść umowy sprzedaży). Kupiec wprowadza swók klucz prywatny oraz ewentualnie dodatkowe dane. Do transakcji dochodzi w pewnej przestrzeni niedostępnej dla obu stron, w wyniku czego sprzedawca otrzymuje raport z operacji, a kupiec produkt podpisany swoim kluczem prywatnym oraz również raport. Co jest istotne, to że sprzedawca nie ma dostępu do produktu podpisanego, a kupiec do oryginalnego.



Rys.2. Schemat operacji sprzedaży

3. Identyfikacja klienta. Sprzedawca, znalazłszy nielegalny egzemplarz produktu, na podstawie znalezionej kopii oraz własnego egzemplarza oryginalnego, uzyskuje klucz publiczny klienta oraz raport z operacji sprzedaży, ewentualnie wynik typu „błąd” w przypadku gdy operacja identyfikacji się nie powiodła.

4. Dowód winy. Mając powyższe dane, sprzedawca prosi arbitra o rozstrzygnięcie sprawy przekazując mu uzyskane dowody. Kupiec, jeśli nie przyznaje się do winy, może to udowodnić wprowadzając swój raport ze sprzedaży wraz ze swoim kluczem publicznym. Arbiter na podstawie tych danych rozstrzyga, czy dany znaleziony nielegalny egzemplarz został zakupiony przez pozwanego klienta, czy też nie.

Podany tu opis jest dosyć ogólny. Więcej informacji na temat dziedziny zwanej potocznie „asymmetric fingerprinting” można znaleźć w [6] oraz [7].

Bibliografia

[1] Adrian E. Escott, John C. Sager, Alexander P. L. Selrirk, Dimitrios Tsapakidis *Attacking Elliptic Curve Cryptosystems using the Parallel Pollard rho method*, magazyn „*CryptoBytes*”, RSA Laboratories, nr 2/1999.

[2] Dan Boneh, Antoine Joux, Phong Q. Nguyen *Why textbook ElGamal and RSA encryption are insecure.*

[3] Michael J. Wiener *Performance comparison of public-key cryptosystems*, magazyn „*CryptoBytes*”, RSA Laboratories, nr 1/1998.

[4] Nicolas Tadeusz Courtois *Kryptografia wielu zmiennych*, konferencja Enigma 2000. Warszawa, maj 2000 r.

[5] Strona domowa HFE. HTTP://HFE.MINRANK.ORG

[6] Birgit Pfitzmann, Matthias Schunter *Asymmetric fingerprinting*, konferencja Eurocrypt 1996. Zaragoza, maj 1996 r.

[7] Karou Kurosawa, Yvo Desmedt *Optimum traitor tracing and asymmetric schemes*