Autoreferat do rozprawy „Security against space-restricted physical attacks”

Tomasz Kazana

Uniwersytet Warszawski

20 grudnia 2012

Streszczenie

Rozprawa doktorska wprowadza nowy, zdefiniowany przez autora model obliczeń kryptograficznych, nazwany SBA–modelem. Charakterystyczne cechy tego modelu to ograniczona pamięć, wycieki oraz użycie losowej wyroczni. W rozprawie badane są trzy schematy kryptograficzne: *Jednorazowe funkcje pseudolosowe*, *Schemat ewolucji klucza* oraz *Funkcje jednorazowe* (ang. *One-time computable pseudorandom function*, *Key-evolution schemes* oraz *One-time programs*). Pokazujemy istnienie ww. schematów w SBA–modelu. Wyniki z roprawy zostały zaprezentowane w następujących pracach: [11, 14, 15].

1 Wprowadzenie

Praca dotyczy problemów kryptograficznych w modelu z wyciekami informacji oraz aktywnym przeciwnikiem. Zaczniemy od wyjaśnień intuicyjnych.

Wszystkie wyniki w tej pracy dotykają ogólnego schematu: istnieje długi ciąg bitów R (w zamiarze tajny), z którego potrafimy coś obliczyć (powiedzmy $f(R)$), a chcemy aby pozostało to sekretem. W trakcie eksperymentu pojawi się przeciwnik, który *czegoś* o R się dowie, ale prawie zawsze okaże się ta wiedza *zbyt mała*, aby wnioskować coś na temat $f(R)$.

We współczesnej kryptografii istnieje trend konstruowania protokołów odpornych na podobnych przeciwników. Zwykle albo zakładamy iż przeciwnik jest pasywny, co oznacza, że nie wpływa na R w trakcie wykonania obliczeń przez uczciwego użytkownika, a jedynie wybiera funkcję g i poznaje $g(R)$[1]. Oczywiście g nie może być dowolne, bo wówczas gdy $g = f$, to przeciwnik poznaje cały sekret od razu. Ta uwaga sugeruje, że rozsądne jest przyjęcie założenia, że g musi być wybrane z jakiejś (możliwie szerokiej) klasy, do której nie należy f. Przykładem jest założenie, że zbiór wartości g jest istotnie mniejszy niż f, tzn. $|g(R)| \ll |f(R)|$. Przykłady prac o atakach pasywnych: [1, 4, 5, 8–10, 17–19, 21, 22, 24, 25, 27, 28]. Inne założenie to tak zwany przeciwnik aktywny[2], który może złośliwie podmieniać R na wybrane R' czy wręcz zmieniać algorytm liczenia funkcji z f na wybrane f'. Przykłady prac z tej dziedziny: [2, 3, 6, 7, 12, 13, 16].

W tej pracy podjęta jest próba połączenia tych dwóch paradygmatów. To znaczy opisany niżej model zakłada, że istnieje aktywny wirus ($\mathcal{A}_{small}$), który dodatkowo może spowodować wybrany *wyciek*. Przy pewnych ograniczeniach pokazujemy, że skonstruowane schematy wciąż pozostają bezpieczne. Ten model nazywamy SBA–modelem. [3] Jest to pojęcie nowe, wprowadzone przez autora rozprawy.

[1] Popularnie mówiąc: przeciwnik powoduje wyciek g.
[2] Popularnie mówiąc: przeciwnik jest wirusem.
[3] Skrót SBA pochodzi od *Small and Big Adversary*.

1.1 Motywacja

Praca próbuje wypełnić lukę między światem praktyków i teoretyków. Z jednej strony bezpieczeństwo jest w pełni udowodnione, ale jak zwykle w kryptografii, przyjmuje się przy tym pewne założenia postulowane przez praktyków, którzy *wierzą*, że pewne konstrukcje są bezpieczne. Konkretnie, korzystamy z założenia o istnieniu losowej wyroczni, próbującym uchwycić ideę funkcji haszujących.

1.2 Losowa wyrocznia

Losowa wyrocznia (ang. *random oracle*) to program, który na dowolne zapytanie odpowiada losowo (a więc prawdopodobieństwo wyniku obliczeń jest jednostajnie rozłożone na przeciwdziedzinie), chyba że dane zapytanie pojawiło się już wcześniej. Wówczas losowa wyrocznia odpowiada tak samo, jak wcześniej.

1.3 Model obliczeń: SBA–model

Przez *przeciwnika* będziemy rozumieć parę algorytmów $\mathcal{A} = (\mathcal{A}_{small}, \mathcal{A}_{big})$, które uruchamiane są jednocześnie oraz mogą się komunikować. Oba algorytmy mają dostęp do wspólnej losowej wyroczni H. Zakładamy, iż tylko $\mathcal{A}_{small}$ ma bezpośredni dostęp do tajnego ciągu bitów R. Wynikiem obliczeń przeciwnika jest wynik obliczeń $\mathcal{A}_{big}$. A więc celem przeciwnika jest, aby algorytm $\mathcal{A}_{big}$ obliczył jakiś sekret zależny od R. [4]

Bedziemy oznaczać $\mathcal{A}^{H(\cdot)}(R) = \left(\mathcal{A}_{big}^{H(\cdot)} \leftrightarrows \mathcal{A}_{small}^{H(\cdot)}(R)\right)$ jednoczesne wykonanie $\mathcal{A}_{big}$ oraz $\mathcal{A}_{small}$, gdzie $\mathcal{A}_{small}$ na wejściu dostaje R i oba algorytmy mają dostęp do losowej wyroczni $H(\cdot)$. Jak wspomniano wyżej, wyjście tak opisanego $\mathcal{A}$ jest definiowane jako wyjście samego $\mathcal{A}_{big}$.

W większości twierdzeń będziemy twierdzić, że $\mathcal{A}$ nie jest w stanie *czegoś* policzyć, o ile spełnione są następujące założenia (dla konkretnych s, c oraz q podawanych w twierdzeniach):

- $\mathcal{A}_{small}$ ma ograniczoną pamięć przez s.
- Komunikacja od $\mathcal{A}_{small}$ do $\mathcal{A}_{big}$ jest ograniczona przez c. [5] W drugą stronę jest nieograniczona.
- Liczba pytań jakie $\mathcal{A}_{small}$ i $\mathcal{A}_{big}$ mogą łącznie zadać losowej wyroczni jest ograniczona przez q.

W wyżej wymionym przypadku będziemy pisać, że $\mathcal{A}$ jest (s, c, q) ograniczony.

Czasem, poza R, przeciwnik $\mathcal{A}$ może mieć dodatkowe wejście x. Wówczas zakładamy, że dane x początkowo znajduje się na wejściu $\mathcal{A}_{big}$.

2 Protokoły kryptograficzne w SBA–modelu

W pracy omawiane są trzy różne problemy kryptograficzne oraz ich rozwiązania w SBA–modelu. Podobne problemy były znane wcześniej, ale przy słabszych założeniach niż SBA–model.

[4] Zwykle $\mathcal{A}_{small}$ może łatwo obliczyć sekret ponieważ ma dostęp do R. Wracając do intuicji: należy myśleć, że $\mathcal{A}_{small}$ to mały wirus zainstalowany na urządzeniu zawierającym R, a dopiero $\mathcal{A}_{big}$ to *prawdziwy* przeciwnik, który chce poznać sekret.

[5] Intuicyjnie założenia dotyczące $\mathcal{A}_{small}$ wydają się rozsądne, gdyż $\mathcal{A}_{small}$ to wirus, a ten jest ograniczony przez zewnętrzne urządzenie na którym jest zainstalowany. Innymi słowy, aby je spełnić, wystarczy odpowiednio przygotować urządzenie, na którym przechowywany jest R.

2.1 Jednorazowe funkcje pseudolosowe (PRF)

Idea Tajny ciąg bitów R definiuje n deterministycznych [6] funkcji $F_{i,R}$, $i = 1, \ldots, n$. Idea jest taka, że przeciwnik nie może poznać $F_{i,R}(x)$ oraz $F_{i,R}(x')$ dla pewnych i oraz $x \neq x'$.

Definicja. Każdy ciąg bitów R (odpowiedniej długości, zależnej od parametrów) definiuje (w ustalony, zależny od H, sposób) funkcje $F_{i,R} : \{0,1\}^k \to \{0,1\}^k$. Powiemy, że algorytm W jest $(c, s, s', q, n, \epsilon)$–PRFem, gdy:

- Algorytm W [7] dla danego R na wejściu, potrafi obliczyć $F_{i,R}(x_i)$ dla dowolnych wejściowych $\{x_i\}_{i=1,\ldots n}$ w czasie wielomianowym od $(|R| + |x|)$, nawet gdy jest (s', c, q) ograniczony.
- Dla każdego algorytmu $\mathcal{A}$ który dostaje R na wejściu i jest (s,c,q) ograniczony, prawdopodobieństwo obliczenia $F_{i,R}(x)$ oraz $F_{i,R}(x')$ dla dowolnego i i dowolnych $x \neq x'$ jest nie większe niż ϵ.

Wynik. Jeśli:

$$n < \frac{U+1}{2(U-M+1)}$$

oraz

$$\frac{s+c+k}{k-\log q} < U$$

to w SBA–modelu istnieje $(c, s, Mk, q, n, (q+1)\cdot 2^{-k})$–PRF.

Wnioski Asymptotycznie (gdy ϵ jest zaniedbywalny, a q zależy wielomianowo od k), największe n jakie możemy uzyskać (a więc takie, dla których istnieje PRF) przy ustalonych s, s' oraz c to:

$$n \approx \frac{s+c}{2(s+c-s')}$$

Wynik na tle dziedziny Samo pojęcie PRF zostało wprowadzone przez autora rozprawy w [15]. Za jego pomocą łatwo (Rozdział 3.7 w rozprawie) pokazano istnienie *proof-of-erasure* w SBA-modelu. Podobne wyniki (przy innych założeniach) można znaleźć w [26].

2.2 Schemat ewolucji klucza (KES)

Idea W tym rozdziale nieco zmieniamy model. Będziemy mówić o *rundzie*. Runda charakteryzuje się tym, że za każdym razem *odświeża się* ograniczenie na komunikację po skończeniu rundy. Innymi słowy limit c na komunikację dotyczy każdej z rund osobno.

W tym schemacie z tajnego R można obliczyć wiele kluczy $K_{0,R}$, $K_{1,R}$, ..., $K_{T,R}$. Obliczenie konkretnego $K_{i,R}$ zmienia rundę (obliczenie klucza definiujemy jako moment, gdy cały klucz jest wpisany do ustalonej taśmy maszyny Turinga).

Intuicje Możemy mysleć o urządzeniu, które w jednostce czasu (np. każdego kolejnego dnia) liczy kolejny klucz. Intuicyjnie, pokazujemy, że jeśli przez cały dzień o numerze i znamy właściwy klucz $K_{i,R}$, to na pewno nie znamy tego dnia klucza $K_{i+1,R}$ ani dalszych.

[6] Przy ustalonym $H(\cdot)$. Funkcje te nie liczą nic istotnego, stąd określenie – znane z literatury – pseudolosowe. Novum to *jednorazowe* funkcje pseudolosowe.

[7] Intuicyjnie: uczciwy gracz.

Definicja Każdy ciąg bitów R definiuje (w określony, zależny od H, sposób) klucze $K_{0,R}$, $K_{1,R}$, ..., $K_{T,R}$. Algorytm W jest (c,s,s',q,T,ϵ)–KESem, gdy:

- Algorytm W [8], dla danego na wejściu R, potrafi obliczyć wszytkie $K_{i,R}$, nawet gdy jest (s',c,q) ograniczony, przy czym klucz $K_{i,R}$ jest obliczany w rundzie i.
- Dla każdego algorytmu $\mathcal{A}$ który dostaje R na wejściu oraz jest (s,c,q) ograniczony, prawdopodobieństwo obliczenia klucza $K_{i-1,R}$ w rundzie $(i+1)$ jest nie większe niż ϵ.

Wynik. Jeśli

$$\frac{4c+s+\lambda}{w-\log(q)} \leq N + \frac{N}{2}$$

to w SBA–modelu istnieje $(c,s,Nk,q,T,q\cdot 2^{-k}+T\cdot 2^{1-\lambda})$–KES.

Wynik na tle dziedziny Idea ewolucji klucza była podejmowana przez innych autorów, przy innych założeniach. Szczegóły można znaleźć w [17, 23, 29] oraz w samej rozprawie w rozdziale 4.2.

2.3 Funkcje jednorazowe (OTP)

Idea Tajny ciąg R zawiera opis pewnego programu C, który może zostać wykonany tylko raz, dla wybranego wejścia. Innymi słowy, dowolny użytkownik (również złośliwy) dostaje urządzenie z programem, ale nie wie, co to za program. Pokazujemy, że jedyne czego się dowie to wartość $C(x)$ dla dokładnie jednego x.

Definicja Niech $C : \{0,1\}^n \to \{0,1\}^m$ będzie programem (rozumianym jako obwód logiczny). Ciąg bitów R_C to pewien ciąg bitów, generowany z C przez efektywny niedeterministyczny algorytm używający H. Algorytm D jest (c,s,ϵ)–one-time programem dla klasy wszystkich funkcji $\{0,1\}^n \to \{0,1\}^m$, gdy:

- Algorytm D dla danego R_C oraz $x \in \{0,1\}^n$ oblicza $C(x)$, nawet gdy jest (s,c,q) ograniczony.
- Istnieje symulator S z dostępem do wyroczni jednokrotnego dostępu obliczającej C (ale bez dostępu do R_C) taki, że dla dowolnego przeciwnika $\mathcal{A}$ mającego dostęp do R_C i (s,c,q)–ograniczonego nie da się odróżnić wyniku obliczeń S od wyniku obliczeń $\mathcal{A}$ z prawdopodobieństem większym niż ϵ.[9]

Wynik Dla dowolnych (n,m) istnieje (c,s,ϵ)–one-time program dla klasy wszystkich funkcji $\{0,1\}^n \to \{0,1\}^m$ w SBA–modelu z paramterami c,s,ϵ opisanymi w Twierdzeniu 5.2 w rozprawie.

Wynik na tle dziedziny Pojęcie *One-time program* zostało wprowadzone przez Goldwasser et al. w [20]. Autorzy dowodzą tam istnienia OTP w modelu z założeniami o tzw. OTM (one-time memory), szczegóły w [20].

[8] Intuicyjnie: uczciwy gracz.

[9] Bardziej precyzyjnie: nie istnieje żaden algorytm (*odróżniacz*), który odróżnia wyżej opisane wyniki obliczeń z prawdopodobieństwem większym niż $\frac{1}{2}+\epsilon$, jeśli liczba jego pytań do wyroczni jest ograniczona przez q.

Literatura

[1] A. Akavia, S. Goldwasser, and V. Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In *TCC*, 2009.

[2] J. Alwen, Y. Dodis, M. Naor, G. Segev, S. Walfish, and D. Wichs. Public-key encryption in the bounded-retrieval model. In *EUROCRYPT*, 2010.

[3] J. Alwen, Y. Dodis, and D. Wichs. Leakage-resilient public-key cryptography in the bounded-retrieval model. In *CRYPTO*, 2009.

[4] Z. Brakerski and S. Goldwasser. Circular and leakage resilient public-key encryption under subgroup indistinguishability (or: Quadratic residuosity strikes back). CRYPTO, 2010.

[5] Z. Brakerski, Y. T. Kalai, J. Katz, and V. Vaikuntanathan. Cryptography resilient to continual memory leakage. FOCS, 2010.

[6] D. Cash, Y. Z. Ding, Y. Dodis, W. Lee, R. J. Lipton, and S. Walfish. Intrusion-resilient key exchange in the bounded retrieval model. In *TCC*, 2007.

[7] G. D. Crescenzo, R. J. Lipton, and S. Walfish. Perfectly secure password protocols in the bounded retrieval model. In *TCC*, 2006.

[8] F. Davì, S. Dziembowski, and D. Venturi. Leakage-resilient storage. SCN, 2010.

[9] Y. Dodis, S. Goldwasser, Y. T. Kalai, C. Peikert, and V. Vaikuntanathan. Public-key encryption schemes with auxiliary inputs. In *TCC*, 2010.

[10] Y. Dodis, K. Haralambiev, A. Lopez-Alt, and D. Wichs. Cryptography against continuous memory attacks. FOCS, 2010.

[11] K. Durnoga, S. Dziembowski, T. Kazana, and M. Zajac. One-time programs with limited memory. In *(submitted to Financial Crypto)*, 2012.

[12] S. Dziembowski. Intrusion-resilience via the bounded-storage model. In *TCC*, 2006.

[13] S. Dziembowski. On forward-secure storage. In *CRYPTO*, 2006.

[14] S. Dziembowski, T. Kazana, and D. Wichs. Key-evolution schemes resilient to space-bounded leakage. In *CRYPTO*, pages 335–353, 2011.

[15] S. Dziembowski, T. Kazana, and D. Wichs. One-time computable self-erasing functions. In *TCC*, pages 125–143, 2011.

[16] S. Dziembowski and K. Pietrzak. Intrusion-resilient secret sharing. In *FOCS*, pages 227–237, 2007.

[17] S. Dziembowski and K. Pietrzak. Leakage-resilient cryptography. In *FOCS*, 2008.

[18] ECRYPT. *The Side Channel Cryptanalysis Lounge* http://www.crypto.rub.de/en_sclounge.html.

[19] S. Faust, E. Kiltz, K. Pietrzak, and G. N. Rothblum. Leakage-resilient signatures. In *TCC*, 2010.

[20] S. Goldwasser, Y. T. Kalai, and G. N. Rothblum. One-time programs. In D. Wagner, editor, *CRYPTO*, volume 5157 of *LNCS*, pages 39–56, 2008.

[21] Y. Ishai, A. Sahai, and D. Wagner. Private Circuits: Securing Hardware against Probing Attacks. In *CRYPTO*, 2003.

[22] J. Katz and V. Vaikuntanathan. Signature schemes with bounded leakage resilience. In *ASIACRYPT*, pages 703–720, 2009.

[23] P. Kocher. Design and validation strategies for obtaining assurance in countermeasures to power analysis and related attacks. *NIST Physical Security Testing Workshop*, 2005. Available at www.smartcard.co.uk/DPAValidation.pdf.

[24] S. Micali and L. Reyzin. Physically observable cryptography (extended abstract). In *TCC*, 2004.

[25] M. Naor and G. Segev. Public-key cryptosystems resilient to key leakage. In *Advances in Cryptology - CRYPTO*, August 2009.

[26] D. Perito and G. Tsudik. Secure code update for embedded devices via proofs of secure erasure. In *ESORICS*, 2010.

[27] K. Pietrzak. A leakage-resilient mode of operation. In *EUROCRYPT*, 2009.

[28] F.-X. Standaert, T. Malkin, and M. Yung. A unified framework for the analysis of side-channel key recovery attacks. In *EUROCRYPT*, 2009.

[29] Y. Yu, F.-X. Standaert, O. Pereira, and M. Yung. Practical leakage-resilient pseudorandom generators. In *CCS: ACM Conference on Computer and Communications Security.*, 2010. To Appear.