Jak szyfrować i podpisywać

Formaty podpisywanych i szyfrowanych wiadomości na przykładzie PKCS #1

Stefan Dziembowski

Podstawowy problem

Dla matematyka RSA jest funkcją $Z_n^* \rightarrow Z_n^*$.

Podstawowy problem

Dla matematyka RSA jest funkcją $Z_n^* \to Z_n^*$.

Wady:

- abstrahuje od rzeczywistości (w praktyce zawsze szyfrujemy/podpisujemy ciągi bitów, a nie elementy Z_n^*).

Podstawowy problem

Dla matematyka RSA jest funkcją $Z_n^* \to Z_n^*$.

Wady:

- abstrahuje od rzeczywistości (w praktyce zawsze szyfrujemy/podpisujemy ciągi bitów, a nie elementy Z_n^*).
- Szyfrogram daje przeciwnikowi pewne informacje na temat wiadomości (wycieka symbol Jacobiego).

Podstawowy problem

Dla matematyka RSA jest funkcją $Z_n^* \to Z_n^*$.

Wady:

- abstrahuje od rzeczywistości (w praktyce zawsze szyfrujemy/podpisujemy ciągi bitów, a nie elementy Z_n^*).
- Szyfrogram daje przeciwnikowi pewne informacje na temat wiadomości (wycieka symbol Jacobiego).
- Przeciwik widzi kiedy zaszyfrowano wielokrotnię tę samą wiadomość;

Podstawowy problem

Dla matematyka RSA jest funkcją $Z_n^* \to Z_n^*$.

Wady:

- abstrahuje od rzeczywistości (w praktyce zawsze szyfrujemy/podpisujemy ciągi bitów, a nie elementy Z_n^*).
- Szyfrogram daje przeciwnikowi pewne informacje na temat wiadomości (wycieka symbol Jacobiego).
- Przeciwik widzi kiedy zaszyfrowano wielokrotnię tę samą wiadomość; szyfrowanie nie jest bezpieczne jeśli zbiór potencjalnych wiadomości jest mały (to jest cech wszystkich szyfrów deterministycznych).

Problemy - c.d.

- W przypadku podpisu przeciwnki może zawsze uzyskać podpis na losowej wiadomości.
- Przeciwnik może wykryć algebraiczne związki między szyfrowanymi wiadomościami:

$$E_K(M_1) \cdot E_K(M_2) = E_K(M_1 \cdot M_2)$$

Problemy - c.d.

- W przypadku podpisu przeciwnki może zawsze uzyskać podpis na losowej wiadomości.
- Przeciwnik może wykryć algebraiczne związki między szyfrowanymi wiadomościami:

$$E_K(M_1) \cdot E_K(M_2) = E_K(M_1 \cdot M_2)$$

  Podobnie, przeciwnik może podrobić podpis na wiadomości $M_1 \cdot M_2$ jeśli dysponuje podpisami na M_1 i M_2.
- W przypadku podpisu: jeśli wiadomość jest długa to bez sensu jest dzielić ją na bloki i podpisywać blok po bloku.

Rozwiązanie

Trzeba ustalić sposób zakodowania wiadomości przed zaszyfrowaniem/podpisaniem.

Rozwiązanie

Trzeba ustalić sposób zakodowania wiadomości przed zaszyfrowaniem/podpisaniem.

wiadomość M

Rozwiązanie

Trzeba ustalić sposób zakodowania wiadomości przed zaszyfrowaniem/podpisaniem.

wiadomość $M \xrightarrow{\text{kodowanie}}$

Rozwiązanie

Trzeba ustalić sposób zakodowania wiadomości przed zaszyfrowaniem/podpisaniem.

$$\text{wiadomość } M \xrightarrow{\text{kodowanie}} \text{zakodowane } M$$

Rozwiązanie

Trzeba ustalić sposób zakodowania wiadomości przed zaszyfrowaniem/podpisaniem.

$$\text{wiadomość } M \xrightarrow{\text{kodowanie}} \text{zakodowane } M \xrightarrow{\text{szyfrowanie}} C$$

Rozwiązanie

Trzeba ustalić sposób zakodowania wiadomości przed zaszyfrowaniem/podpisaniem.

$$\text{wiadomość } M \xrightarrow{\text{kodowanie}} \text{zakodowane } M \xrightarrow{\text{szyfrowanie}} C$$

Kodowanie powinno ochronić nas przed atakami wykorzystującymi algebraiczne własności RSA.

Rozwiązanie

Trzeba ustalić sposób zakodowania wiadomości przed zaszyfrowaniem/podpisaniem.

$$\text{wiadomość } M \xrightarrow{\text{kodowanie}} \text{zakodowane } M \xrightarrow{\text{szyfrowanie}} C$$

Kodowanie powinno ochronić nas przed atakami wykorzystującymi algebraiczne własności RSA.

Dodatkowo (przynajmniej w przypadku szyfrowania) to kodowanie powinno być „zrandomizowane” (albo algorytm powinien posiadać stan).

Jak ustalić to kodowanie?

Można próbować rozwiązań ad-hoc.

Jak ustalić to kodowanie?

Można próbować rozwiązań ad-hoc.

Historia pokazuje, że to zły pomysł [Ble98].

Jak ustalić to kodowanie?

Można próbować rozwiązań ad-hoc.

Historia pokazuje, że to zły pomysł [Ble98].

Lepiej zrobić to za pomocą metod „naukowych”.

Jak zdefiniować bezpieczne szyfrowanie

Aby móc oceniać bezpieczeństwo kodowania trzeba ustalić definicję bezpieczeństwa.

Jak zdefiniować bezpieczne szyfrowanie

Aby móc oceniać bezpieczeństwo kodowania trzeba ustalić definicję bezpieczeństwa. (Poniższe rozważania mają sens zarówno dla modelu asymetrycznego jak i symetrycznego.)

Jak zdefiniować bezpieczne szyfrowanie

Aby móc oceniać bezpieczeństwo kodowania trzeba ustalić definicję bezpieczeństwa. (Poniższe rozważania mają sens zarówno dla modelu asymetrycznego jak i symetrycznego.)

Naiwne podejście

Szyfrowanie jest bezpieczne, jeśli na podstawie szyfrogramu $C = E_K(M)$ trudno obliczyć wiadomość M.

Jak zdefiniować bezpieczne szyfrowanie

Aby móc oceniać bezpieczeństwo kodowania trzeba ustalić definicję bezpieczeństwa. (Poniższe rozważania mają sens zarówno dla modelu asymetrycznego jak i symetrycznego.)

Naiwne podejście

Szyfrowanie jest bezpieczne, jeśli na podstawie szyfrogramu $C = E_K(M)$ trudno obliczyć wiadomość M.

Wątpliwość 1

Jeśli trudno obliczyć M ale łatwo obliczyć pierwszą połowę M, to co?

Jak zdefiniować bezpieczne szyfrowanie

Aby móc oceniać bezpieczeństwo kodowania trzeba ustalić definicję bezpieczeństwa. (Poniższe rozważania mają sens zarówno dla modelu asymetrycznego jak i symetrycznego.)

Naiwne podejście

Szyfrowanie jest bezpieczne, jeśli na podstawie szyfrogramu $C = E_K(M)$ trudno obliczyć wiadomość M.

Wątpliwość 1

Jeśli trudno obliczyć M ale łatwo obliczyć pierwszą połowę M, to co?
Lepiej: trudno obliczyć dowolną funkcję M.

Jak zdefiniować bezpieczne szyfrowanie

Aby móc oceniać bezpieczeństwo kodowania trzeba ustalić definicję bezpieczeństwa. (Poniższe rozważania mają sens zarówno dla modelu asymetrycznego jak i symetrycznego.)

Naiwne podejście

Szyfrowanie jest bezpieczne, jeśli na podstawie szyfrogramu $C = E_K(M)$ trudno obliczyć wiadomość M.

Wątpliwość 1

Jeśli trudno obliczyć M ale łatwo obliczyć pierwszą połowę M, to co?
Lepiej: trudno obliczyć dowolną funkcję M.
(zauważmy, że to eliminuje od razy wszystkie deterministyczne schematy).

Definicja bezpieczeństwa - c.d.

Wątpliwość 2

Z reguły tego samego klucza używa się wielokrotnie, więc trzeba to jakoś uwzględnić w definicji.

Definicja bezpieczeństwa - c.d.

Wątpliwość 2

Z reguły tego samego klucza używa się wielokrotnie, więc trzeba to jakoś uwzględnić w definicji.

Uwaga: definiowanie bezpieczeństwa przez tajność klucza ma mały sens.

Krok w kierunku definicja bezpieczeństwa

Model zawiera dwie fazy:

uczenie się Przeciwnik będzie ma dostęp do wyroczni (mającej tajny klucz K) umieszczonej w czarnej skrzynce.

Krok w kierunku definicja bezpieczeństwa

Model zawiera dwie fazy:

uczenie się Przeciwnik będzie ma dostęp do wyroczni (mającej tajny klucz K) umieszczonej w czarnej skrzynce.
Przeciwnik ma prawo zadawać pytania wyroczni (rodzaj pytań zależy od rodzaju modelu).

Krok w kierunku definicja bezpieczeństwa

Model zawiera dwie fazy:

uczenie się Przeciwnik będzie ma dostęp do wyroczni (mającej tajny klucz K) umieszczonej w czarnej skrzynce.
Przeciwnik ma prawo zadawać pytania wyroczni (rodzaj pytań zależy od rodzaju modelu).

zgadywanie Przeciwnik ma za zadanie rozwiązać jakąś zagadkę (rodzaj też zależy od modelu).

Krok w kierunku definicja bezpieczeństwa

Model zawiera dwie fazy:

uczenie się Przeciwnik będzie ma dostęp do wyroczni (mającej tajny klucz K) umieszczonej w czarnej skrzynce.
Przeciwnik ma prawo zadawać pytania wyroczni (rodzaj pytań zależy od rodzaju modelu).

zgadywanie Przeciwnik ma za zadanie rozwiązać jakąś zagadkę (rodzaj też zależy od modelu).

Z reguły zagadka jest zadawana po zakończeniu fazy uczenia się.

Uczenie się (CPA)

Przypomnijmy, że ta faza modeluje fakt, że przecinik zyskuje informacje na temat klucza na podstawie dostępnych szyfrogramów

$$C_1 = E_K(M_1), \ldots, C_n = E_K(M_n).$$

Uczenie się (CPA)

Przypomnijmy, że ta faza modeluje fakt, że przecinik zyskuje informacje na temat klucza na podstawie dostępnych szyfrogramów

$$C_1 = E_K(M_1), \ldots, C_n = E_K(M_n).$$

Staramy się przyjąć jak najbardziej pesymistyczne założenia.

Uczenie się (CPA)

Przypomnijmy, że ta faza modeluje fakt, że przecinik zyskuje informacje na temat klucza na podstawie dostępnych szyfrogramów

$$C_1 = E_K(M_1), \ldots, C_n = E_K(M_n).$$

Staramy się przyjąć jak najbardziej pesymistyczne założenia.

Przeciwnik może mieć pewien wpływ na zawartość szyfrowanych wiadomości, zatem:

Uczenie się (CPA)

Przypomnijmy, że ta faza modeluje fakt, że przecinik zyskuje informacje na temat klucza na podstawie dostępnych szyfrogramów

$$C_1 = E_K(M_1), \ldots, C_n = E_K(M_n).$$

Staramy się przyjąć jak najbardziej pesymistyczne założenia.

Przeciwnik może mieć pewien wpływ na zawartość szyfrowanych wiadomości, zatem:

Chosen Plaintext Attack (CPA)

Przeciwnik ma prawo (adaptywnego) wyboru wiadomości $M_1, \ldots, M_n$ i poznania ich kryptogramów $C_1, \ldots, C_n$. (W przypadku kryptografii klucza publicznego ten atak jest zawsze

Uczenie się (CCA)

W niektórych przypadkach przeciwnik może też wybrać kryptogram C i poznać wiadomość $D_K(C)$.

Uczenie się (CCA)

W niektórych przypadkach przeciwnik może też wybrać kryptogram C i poznać wiadomość $D_K(C)$.

W praktyce np. przeciwnik może uzyskać informację, czy wiadomość jest zgodna z pewnym formatem.

Uczenie się (CCA)

W niektórych przypadkach przeciwnik może też wybrać kryptogram C i poznać wiadomość $D_K(C)$.

W praktyce np. przeciwnik może uzyskać informację, czy wiadomość jest zgodna z pewnym formatem.

Dlatego definiujemy

Chosen Ciphertext Attack (CCA)

Przeciwnik ma prawo (adaptywnego) wyboru kryptogramów $C_1, \ldots, C_n$ i poznania ich dekrypcji $M_1, \ldots, M_n$.

Uczenie się (CCA)

W niektórych przypadkach przeciwnik może też wybrać kryptogram C i poznać wiadomość $D_K(C)$.

W praktyce np. przeciwnik może uzyskać informację, czy wiadomość jest zgodna z pewnym formatem.

Dlatego definiujemy

Chosen Ciphertext Attack (CCA)

Przeciwnik ma prawo (adaptywnego) wyboru kryptogramów $C_1, \ldots, C_n$ i poznania ich dekrypcji $M_1, \ldots, M_n$.
+ to samo co w CPA.

„zagadka” - IND

Rozważamy dwa warianty (IND i NM).

„zagadka” - IND

Rozważamy dwa warianty (IND i NM).

„Indistinguishability” (IND)

„zagadka” - IND

Rozważamy dwa warianty (IND i NM).

„Indistinguishability” (IND)

1. Przeciwnik wybiera dowolną parę wiadomości (M_0, M_1) (tej samej długości) i podaje je wyroczni.

„zagadka” - IND

Rozważamy dwa warianty (IND i NM).

„Indistinguishability” (IND)

1. Przeciwnik wybiera dowolną parę wiadomości (M_0, M_1) (tej samej długości) i podaje je wyroczni.
2. Wyrocznia losuje $i \in \{0, 1\}$ i zwraca $C = C_K(M_i)$.

„zagadka” - IND

Rozważamy dwa warianty (IND i NM).

„Indistinguishability” (IND)

1. Przeciwnik wybiera dowolną parę wiadomości (M_0, M_1) (tej samej długości) i podaje je wyroczni.
2. Wyrocznia losuje $i \in \{0, 1\}$ i zwraca $C = C_K(M_i)$.
3. Zadaniem przeciwnika jest zgadnięcie i.

Szanse przeciwnika nie powinny być większe niż $\frac{1}{2} + \epsilon$.

„zagadka” - NM

„Non-malleability” (NM)

1. Przeciwnik otrzymuje szyfrogram losowej wiadomości $C = E(M_0)$.
2. Zadaniem przeciwnika jest wyprodukowanie kryptogramu C' i relacji R, takich, że prawdopodobieństwo, że

$$(D(C'), M_0) \in R,$$

jest o ϵ większe niż prawdopodobieństwo, że

$$(D(C'), M_1) \in R,$$

dla losowego M_1.

„zagadka” – PA

„Plaintext-awareness”

Wymagamy by przeciwnik nie był w stanie wyprodukować dowolnego szyfrogramu C bez znajomości $D_K(C)$.

„zagadka” – PA

„Plaintext-awareness”

Wymagamy by przeciwnik nie był w stanie wyprodukować dowolnego szyfrogramu C bez znajomości $D_K(C)$.

(Jak to sformalizować?).

„zagadka” – PA

„Plaintext-awareness”

Wymagamy by przeciwnik nie był w stanie wyprodukować dowolnego szyfrogramu C bez znajomości $D_K(C)$.

(Jak to sformalizować?).

Implikuje to oczywiście, że zbiór poprawnych szyfrogramów jest „rzadki” w $\{0,1\}^n$.

Możliwe kombinacje

W przypadku CCA możemy rozważyć 2 rodzaje ataków:

Możliwe kombinacje

W przypadku CCA możemy rozważyć 2 rodzaje ataków:

adaptywny CCA2 przeciwnik ma dostęp do wyroczni nawet w trakcie rozwiązywania zagadki (wykluczamy pytania które pozwoliłyby mu trywialnie wygrać).

Możliwe kombinacje

W przypadku CCA możemy rozważyć 2 rodzaje ataków:

adaptywny CCA2 przeciwnik ma dostęp do wyroczni nawet w trakcie rozwiązywania zagadki (wykluczamy pytania które pozwoliłyby mu trywialnie wygrać).

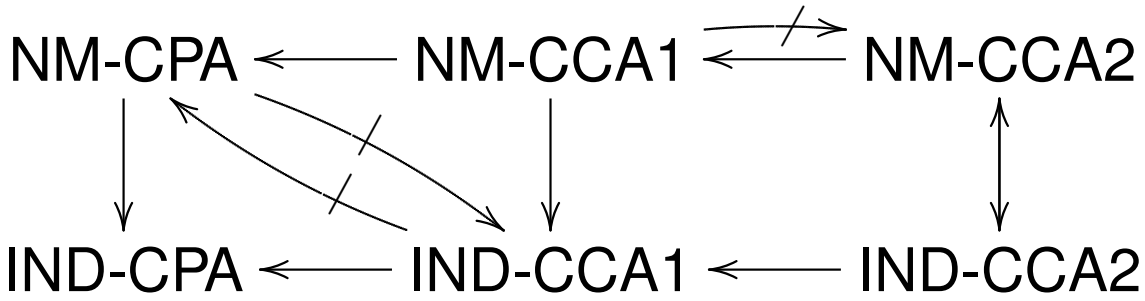
nieadaptywny CCA1 dostęp do wyroczni jest możliwy tylko przed przystąpieniem do rozwiązywania zagadki (to się też nazywa *lunchtime attack*).

To daje następujące możliwości:

IND-CPA	IND-CCA1	IND-CCA2
NM-CPA	NM-CCA1	NM-CCA1

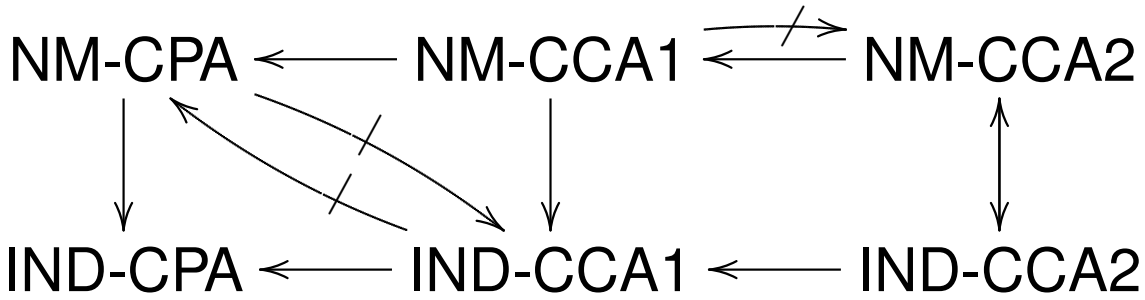
Jakie są związki między tymi pojęciami

Poniższe pochodzi z pracy [BDPR98]:



Jakie są związki między tymi pojęciami

Poniższe pochodzi z pracy [BDPR98]:



Konkluzja

Najlepiej konstruować systemy które są IND/NM-CCA2 bezpieczne.

Konstrukcje

W tej dziedzienie mamy dwa podejścia:

teoretyczne Interesują nas dowolne systemy byle działały w czasie wielomianowym.

Konstrukcje

W tej dziedzienie mamy dwa podejścia:

teoretyczne Interesują nas dowolne systemy byle działały w czasie wielomianowym.

praktyczne Interesują nas systemy wydajne w praktyce (tylko wtedy są szanse, że ktoś tego użyje).

Założenie

Na poczatek czynimy następujące (słabe założenie):

RSA assumption

Zakładamy, że RSA jest jednokierunkową permutacją z zapadką.

Założenie

Na poczatek czynimy następujące (słabe założenie):

RSA assumption

Zakładamy, że RSA jest jednokierunkową permutacją z zapadką.

To znaczy:

Na podstawie znajomości $f(x)$ (i klucza publicznego) trudno obliczyć x.

Założenie

Na poczatek czynimy następujące (słabe założenie):

RSA assumption

Zakładamy, że RSA jest jednokierunkową permutacją z zapadką.

To znaczy:

Na podstawie znajomości $f(x)$ (i klucza publicznego) trudno obliczyć x.

Zauważmy, że to nie wyklucza, że można policzyć część x. Jest to więc bardzo słabe założenie.

OAEP

Najbardziej popularnym systemem używanym w praktyce jest Optimal Assymetric Encryption Padding (OAEP).

OAEP

Najbardziej popularnym systemem używanym w praktyce jest Optimal Assymetric Encryption Padding (OAEP).

OAEP został zaproponowany w pracy
M. Bellare and P. Rogaway.
Optimal asymmetric encryption – How to encrypt with RSA.
Eurocrypt 94

OAEP

Najbardziej popularnym systemem używanym w praktyce jest Optimal Assymetric Encryption Padding (OAEP).

OAEP został zaproponowany w pracy
M. Bellare and P. Rogaway.
Optimal asymmetric encryption – How to encrypt with RSA.
Eurocrypt 94

Jest to część popularnego standardu PKCS #1.

OAEP

Najbardziej popularnym systemem używanym w praktyce jest Optimal Assymetric Encryption Padding (OAEP).

OAEP został zaproponowany w pracy
M. Bellare and P. Rogaway.
Optimal asymmetric encryption – How to encrypt with RSA.
Eurocrypt 94

Jest to część popularnego standardu PKCS #1. W tym standardzie jest też alternatywny system kodowania (typu „ad-hoc”), aby zapewnić kompatybilność wsteczną.

OAEP

Najbardziej popularnym systemem używanym w praktyce jest Optimal Assymetric Encryption Padding (OAEP).

OAEP został zaproponowany w pracy
M. Bellare and P. Rogaway.
Optimal asymmetric encryption – How to encrypt with RSA.
Eurocrypt 94

Jest to część popularnego standardu PKCS #1. W tym standardzie jest też alternatywny system kodowania (typu „ad-hoc”), aby zapewnić kompatybilność wsteczną. OAEP jest jednak preferowany.

Zalety OAEP

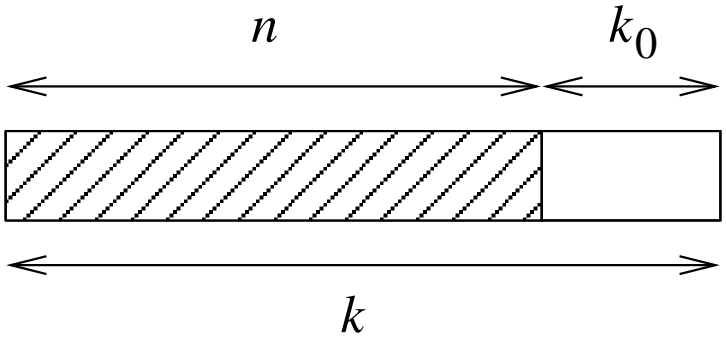
1. Posiada dowód bezpieczeństwa (z tym dowodem były pewne problemy — o tym za chwilę).
2. Jest bardzo wydajny:
   - operacja RSA jest wykonywana raz (jakby było więcej to nikt by tego nie używał).
   - czas potrzebny do wykonania reszty obliczeń jest (w porównianiu z operacją RSA) zaniedbywalny,
   - kryptogram ma optymalną długość ($|n|$), wiadomość zaszyfrowana musi być trochę mniejsza od $|n|$.

Zalety OAEP

1. Posiada dowód bezpieczeństwa (z tym dowodem były pewne problemy — o tym za chwilę).
2. Jest bardzo wydajny:
   - operacja RSA jest wykonywana raz (jakby było więcej to nikt by tego nie używał).
   - czas potrzebny do wykonania reszty obliczeń jest (w porównianiu z operacją RSA) zaniedbywalny,
   - kryptogram ma optymalną długość ($|n|$), wiadomość zaszyfrowana musi być trochę mniejsza od $|n|$. Zauważmy, że z reguły $|n|$ to znacznie więcej niż nam potrzeba...

Jak działa OAEP

f — szyfrowanie RSA
k — długość modułu
n — długość zaszyfrowanej wiadomości
$k_0 = k - n$



$G : \{0,1\}^{k_0} \to \{0,1\}^n$, $H : \{0,1\}^n \to \{0,1\}^{k_0}$

$$E(x) := f(\underbrace{x \oplus G(r)}_{n} || \underbrace{r \oplus H(x \oplus G(r))}_{k_0})$$

Co to są funkcje G i H?

„Mask Generating Functions”.
Idea: funkcje G i H są funkcjami „losowymi”

Co to są funkcje G i H?

„Mask Generating Functions”.
Idea: funkcje G i H są funkcjami „losowymi”

Matematycznie modelujemy je jako „losowe wyrocznie” (random oracles).

Co to są funkcje G i H?

„Mask Generating Functions”.
Idea: funkcje G i H są funkcjami „losowymi”

Matematycznie modelujemy je jako „losowe wyrocznie” (random oracles).

W standardzie PKCS konstruowane są mniej-więcej tak:

Co to są funkcje G i H?

„Mask Generating Functions”.
Idea: funkcje G i H są funkcjami „losowymi”

Matematycznie modelujemy je jako „losowe wyrocznie” (random oracles).

W standardzie PKCS konstruowane są mniej-więcej tak:

1. Na wejściu x obliczamy $H(1, x), H(2, x), \ldots,$ (H — funkcja hashująca)

Co to są funkcje G i H?

„Mask Generating Functions”.
Idea: funkcje G i H są funkcjami „losowymi”

Matematycznie modelujemy je jako „losowe wyrocznie” (random oracles).

W standardzie PKCS konstruowane są mniej-więcej tak:

1. Na wejściu x obliczamy $H(1, x), H(2, x), \ldots,$ (H — funkcja hashująca)
2. Bierzemy prefiks takiej długości jaką potrzebujemy.

Co wiadomo

RSA-OAEP jest CCA2-IND bezpieczne,

Co wiadomo

RSA-OAEP jest CCA2-IND bezpieczne, przy następujących założeniach:

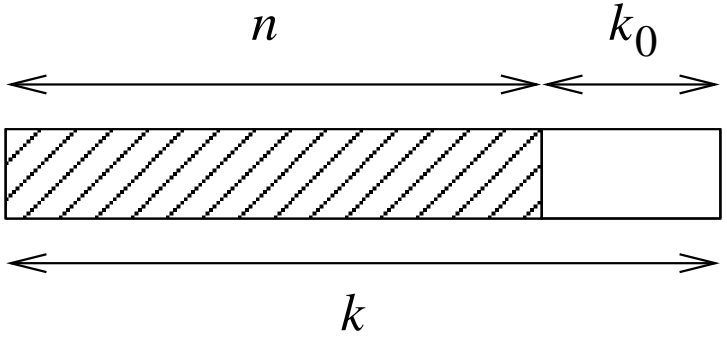
- Szyfrowanie RSA (f) jest jednokierunkową permutacją z zapadką.

Co wiadomo

RSA-OAEP jest CCA2-IND bezpieczne, przy następujących założeniach:

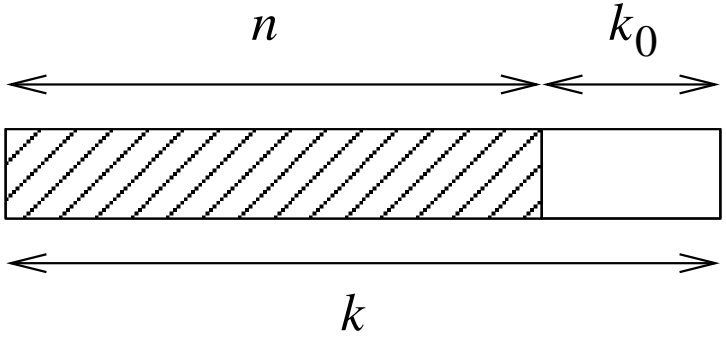
- Szyfrowanie RSA (f) jest jednokierunkową permutacją z zapadką.
- Funkcje G i H są „idealnymi funkcjami losowymi”. Inaczej mówiąc: dowód działa tylko w modelu z losową wyrocznią („random-oracle model”).

Intuicja: Dlaczego OAEP jest bezpieczne?



$$y := f(\underbrace{x \oplus G(r)}_{n} \,||\, \underbrace{r \oplus H(x \oplus G(r))}_{k_0})$$

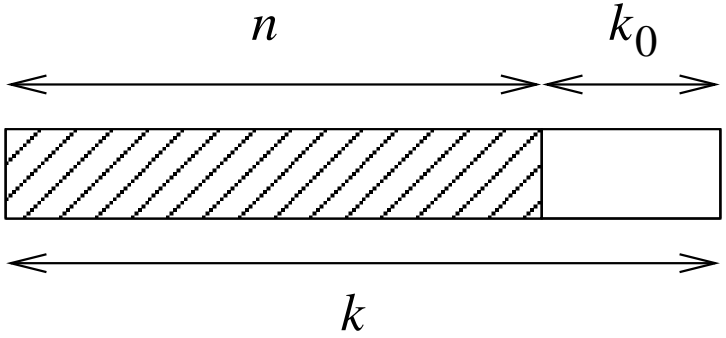
Intuicja: Dlaczego OAEP jest bezpieczne?



$$y := f(\underbrace{x \oplus G(r)}_{n} \,||\, \underbrace{r \oplus H(x \oplus G(r))}_{k_0})$$

1. Aby obliczyć cokolwiek na temat x trzeba obliczyć całe r (tu np. używamy założenia o losowej wyroczni).

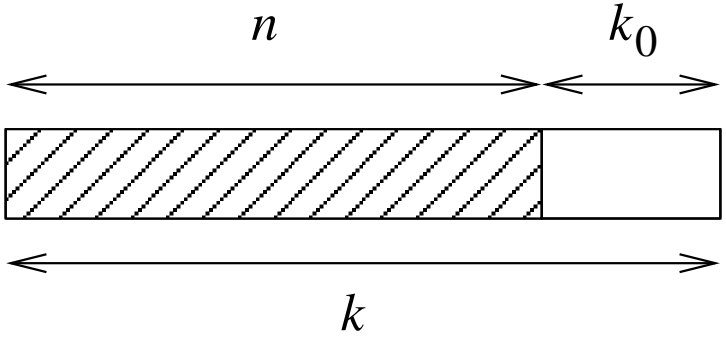
Intuicja: Dlaczego OAEP jest bezpieczne?



$$y := f(\underbrace{x \oplus G(r)}_{n} \,||\, \underbrace{r \oplus H(x \oplus G(r))}_{k_0})$$

1. Aby obliczyć cokolwiek na temat x trzeba obliczyć całe r (tu np. używamy założenia o losowej wyroczni).
2. Aby obliczyć całe r trzeba obliczyć

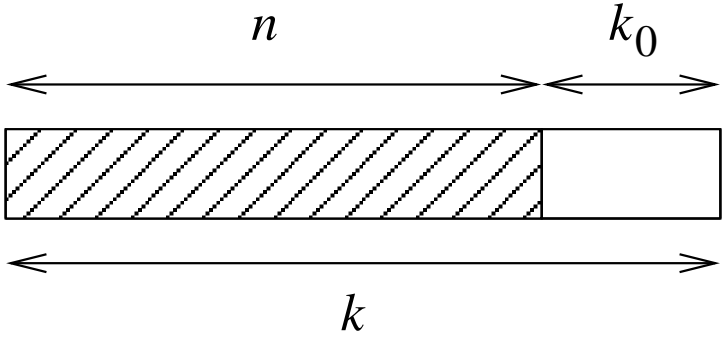
Intuicja: Dlaczego OAEP jest bezpieczne?



$$y := f(\underbrace{x \oplus G(r)}_{n} \,||\, \underbrace{r \oplus H(x \oplus G(r))}_{k_0})$$

1. Aby obliczyć cokolwiek na temat x trzeba obliczyć całe r (tu np. używamy założenia o losowej wyroczni).
2. Aby obliczyć całe r trzeba obliczyć
   - całą „drugą część” $f^{-1}(y)$,

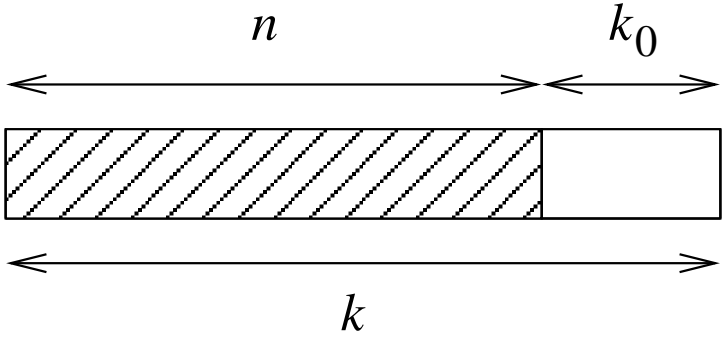
Intuicja: Dlaczego OAEP jest bezpieczne?



$$y := f(\underbrace{x \oplus G(r)}_{n} \,||\, \underbrace{r \oplus H(x \oplus G(r))}_{k_0})$$

1. Aby obliczyć cokolwiek na temat x trzeba obliczyć całe r (tu np. używamy założenia o losowej wyroczni).
2. Aby obliczyć całe r trzeba obliczyć
   - całą „drugą część” $f^{-1}(y)$,
   - całe $x \oplus G(r)$

Intuicja: Dlaczego OAEP jest bezpieczne?



$$y := f(\underbrace{x \oplus G(r)}_{n} \,||\, \underbrace{r \oplus H(x \oplus G(r))}_{k_0})$$

1. Aby obliczyć cokolwiek na temat x trzeba obliczyć całe r (tu np. używamy założenia o losowej wyroczni).
2. Aby obliczyć całe r trzeba obliczyć
   - całą „drugą część” $f^{-1}(y)$,
   - całe $x \oplus G(r)$ — czyli całą „pierwszą część” $f^{-1}(y)$.

Historia dowodu bezpieczeństwa OAEP

W pracy w któej OAEP zostało wprowadzone (M. Bellare and P. Rogaway. Optimal asymmetric encryption – How to encrypt with RSA., Eurocrypt 94) autorzy przedstawili dowód bezpieczeństwa OAEP

Historia dowodu bezpieczeństwa OAEP

W pracy w któej OAEP zostało wprowadzone (M. Bellare and P. Rogaway. Optimal asymmetric encryption – How to encrypt with RSA., Eurocrypt 94) autorzy przedstawili dowód bezpieczeństwa OAEP korzystając wyłącznie z założenia, że f jest permutacją jednokierunkową.

Historia dowodu bezpieczeństwa OAEP

W pracy w któej OAEP zostało wprowadzone (M. Bellare and P. Rogaway. Optimal asymmetric encryption – How to encrypt with RSA., Eurocrypt 94) autorzy przedstawili dowód bezpieczeństwa OAEP korzystając wyłącznie z założenia, że f jest permutacją jednokierunkową.

W roku 2001 Victor Shoup znalazł błąd w tym dowodzie. W pracy OAEP Reconsidered (Eurocrypt 2001) pokazał, że

Historia dowodu bezpieczeństwa OAEP

W pracy w któej OAEP zostało wprowadzone (M. Bellare and P. Rogaway. Optimal asymmetric encryption – How to encrypt with RSA., Eurocrypt 94) autorzy przedstawili dowód bezpieczeństwa OAEP korzystając wyłącznie z założenia, że f jest permutacją jednokierunkową.

W roku 2001 Victor Shoup znalazł błąd w tym dowodzie. W pracy OAEP Reconsidered (Eurocrypt 2001) pokazał, że

- OAEP wcale nie jest IND-CCA2 bezpieczne dla dowolnej permutacji jednokierunkowej (ale jest IND-CCA1 bezpieczne).

Historia dowodu bezpieczeństwa OAEP

W pracy w któej OAEP zostało wprowadzone (M. Bellare and P. Rogaway. Optimal asymmetric encryption – How to encrypt with RSA., Eurocrypt 94) autorzy przedstawili dowód bezpieczeństwa OAEP korzystając wyłącznie z założenia, że f jest permutacją jednokierunkową.

W roku 2001 Victor Shoup znalazł błąd w tym dowodzie. W pracy OAEP Reconsidered (Eurocrypt 2001) pokazał, że

- OAEP wcale nie jest IND-CCA2 bezpieczne dla dowolnej permutacji jednokierunkowej (ale jest IND-CCA1 bezpieczne).
- RSA-OAEP jest IND-CCA2 bezpieczne dla wykładnika $e = 3$.

Historia dowodu bezpieczeństwa OAEP

W pracy w któej OAEP zostało wprowadzone (M. Bellare and P. Rogaway. Optimal asymmetric encryption – How to encrypt with RSA., Eurocrypt 94) autorzy przedstawili dowód bezpieczeństwa OAEP korzystając wyłącznie z założenia, że f jest permutacją jednokierunkową.

W roku 2001 Victor Shoup znalazł błąd w tym dowodzie. W pracy OAEP Reconsidered (Eurocrypt 2001) pokazał, że

- OAEP wcale nie jest IND-CCA2 bezpieczne dla dowolnej permutacji jednokierunkowej (ale jest IND-CCA1 bezpieczne).
- RSA-OAEP jest IND-CCA2 bezpieczne dla wykładnika $e = 3$.
- Ponadto, praca zawiera alternatywną propozycję: OAEP+ (i podaje dowód jej IND-CCA2 bezpieczeństwa)

RSA-OAEP jeszcze żyje!

W 2000 roku ukazała się praca
E. Fujisaki, T. Okamoto, D. Pointcheval, J. Stern. RSA-OAEP is Still Alive

Znajduje się w niej dowód, że RSA-OAEP jest jednak IND-CCA2 bezpieczne (dowód opiera się na szczególnych własnościach szyfrowania RSA).

Jak definiujemy bezpieczeństwo schematu podpisu?

Jest trochę łatwiej niż z szyfrowaniem.

Jak definiujemy bezpieczeństwo schematu podpisu?

Jest trochę łatwiej niż z szyfrowaniem.

Zakładamy, że przeciwnik ma przez jakiś czas dostęp do „wyroczni podpisującej”, która podpisuje wybrane przez niego wiadomości.

Jak definiujemy bezpieczeństwo schematu podpisu?

Jest trochę łatwiej niż z szyfrowaniem.

Zakładamy, że przeciwnik ma przez jakiś czas dostęp do „wyroczni podpisującej”, która podpisuje wybrane przez niego wiadomości.

Następnie, zadaniem przeciwnika jest wyprodukowanie podpisu na jakiejś (wybranej przez niego) wiadomości.

Hash-and-sign

Najczęstszą metodą jest „hash-and-sign”: podpisem na M jest wartość $H(M)^d \bmod n$, gdzie H jest funkcją hashującą.

Hash-and-sign

Najczęstszą metodą jest „hash-and-sign”: podpisem na M jest wartość $H(M)^d \bmod n$, gdzie H jest funkcją hashującą.

Problem

Popularne funkcje hashujące mają output długości znacznie mniejszej niż długość popularnych modułów RSA.

Hash-and-sign

Najczęstszą metodą jest „hash-and-sign”: podpisem na M jest wartość $H(M)^d \bmod n$, gdzie H jest funkcją hashującą.

Problem

Popularne funkcje hashujące mają output długości znacznie mniejszej niż długość popularnych modułów RSA.

Rozwiązanie (ad-hoc)

Dopisujemy jakieś ustalone cyfry, żeby długość się zgodziła.

Hash-and-sign

Najczęstszą metodą jest „hash-and-sign”: podpisem na M jest wartość $H(M)^d \bmod n$, gdzie H jest funkcją hashującą.

Problem

Popularne funkcje hashujące mają output długości znacznie mniejszej niż długość popularnych modułów RSA.

Rozwiązanie (ad-hoc)

Dopisujemy jakieś ustalone cyfry, żeby długość się zgodziła.

To jest jedna z metod w PKCS #1 (nie polecana).

Ulepszenie

Problem z powyższą metodą

Nie można udowodnić bezpieczeństwa tylko na podstawie założenia, że RSA jest trudno odwracalne na losowym elemencie dziedziny, bo korzystamy z bardzo małego ustalonego podzbioru dziedziny.

Ulepszenie

Problem z powyższą metodą

Nie można udowodnić bezpieczeństwa tylko na podstawie założenia, że RSA jest trudno odwracalne na losowym elemencie dziedziny, bo korzystamy z bardzo małego ustalonego podzbioru dziedziny.

Pomysł: Full Domain Hash (Bellare i Rogaway, 1993)

Wymagamy, by output funkcji H obejmował całą dziedzinę.

Ulepszenie

Problem z powyższą metodą

Nie można udowodnić bezpieczeństwa tylko na podstawie założenia, że RSA jest trudno odwracalne na losowym elemencie dziedziny, bo korzystamy z bardzo małego ustalonego podzbioru dziedziny.

Pomysł: Full Domain Hash (Bellare i Rogaway, 1993)

Wymagamy, by output funkcji H obejmował całą dziedzinę.

Problem z FDH

Dowód bezpieczeństwa nie jest zbyt silny.

Ulepszenie

Problem z powyższą metodą

Nie można udowodnić bezpieczeństwa tylko na podstawie założenia, że RSA jest trudno odwracalne na losowym elemencie dziedziny, bo korzystamy z bardzo małego ustalonego podzbioru dziedziny.

Pomysł: Full Domain Hash (Bellare i Rogaway, 1993)

Wymagamy, by output funkcji H obejmował całą dziedzinę.

Problem z FDH

Dowód bezpieczeństwa nie jest zbyt silny. Nieformalnie: Jeśli przeciwnik ma zasoby do obliczenia q razy funkcję H, to

RSA-FDH jest q razy łatwiej złamać niż RSA

Inny pomysł PSS

Przedstawimy Probabilistic Signature Scheme (Bellare i Rogoway, 1996).

Inny pomysł PSS

Przedstawimy Probabilistic Signature Scheme (Bellare i Rogoway, 1996).

Wersja prostsza (PSS0)

Nie dopisujemy ustalonych cyfr, tylko losowe.

Inny pomysł PSS

Przedstawimy Probabilistic Signature Scheme (Bellare i Rogoway, 1996).

Wersja prostsza (PSS0)

Nie dopisujemy ustalonych cyfr, tylko losowe.

Potem trzeba je dołączyć do podpisu:

$$E_K(M) = (H(r||M)^d \bmod n, r).$$

(aby można było zweryfikować podpis).

Inny pomysł PSS

Przedstawimy Probabilistic Signature Scheme (Bellare i Rogoway, 1996).

Wersja prostsza (PSS0)

Nie dopisujemy ustalonych cyfr, tylko losowe.

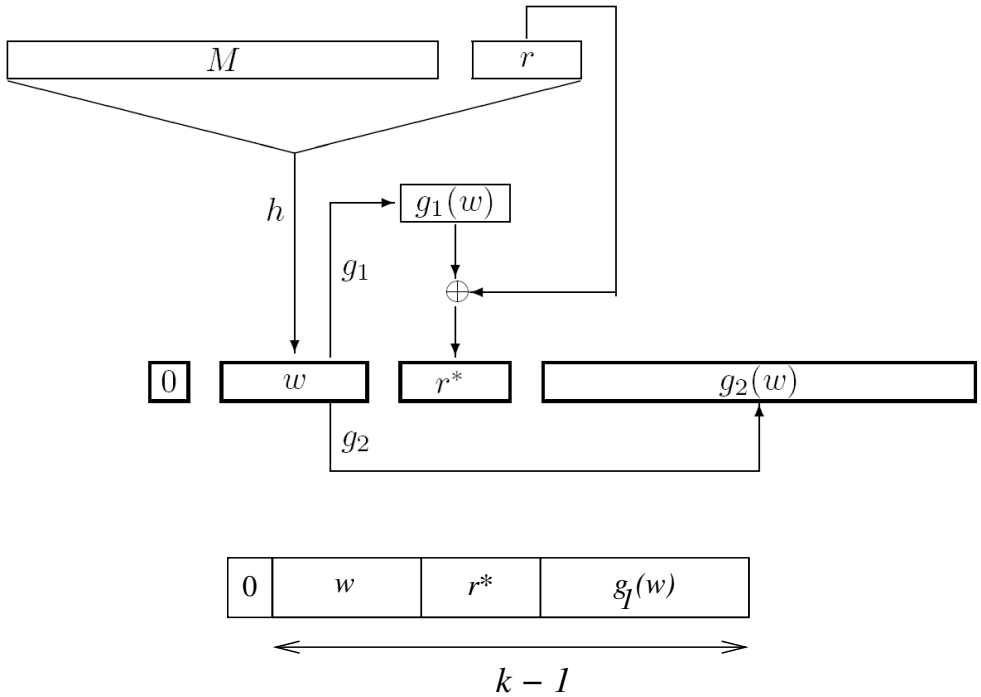
Potem trzeba je dołączyć do podpisu:

$$E_K(M) = (H(r||M)^d \bmod n, r).$$

(aby można było zweryfikować podpis).

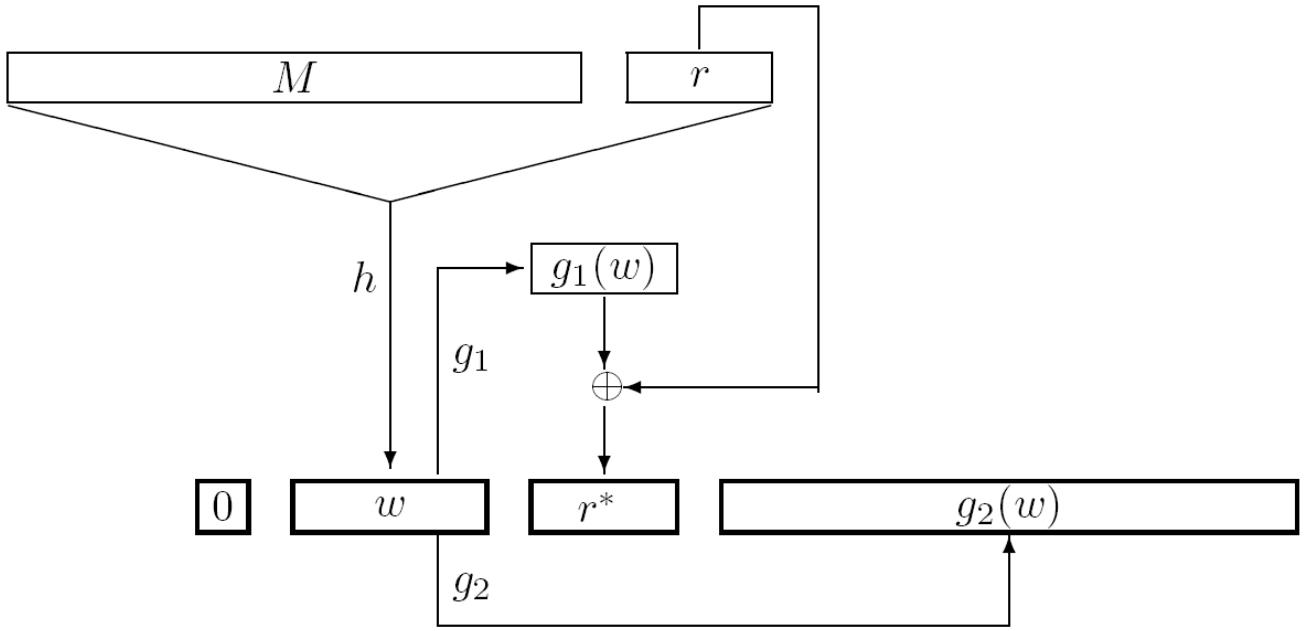
Problem: Wydłuża się podpis.

Wersja pełna PSS



Np. $k = 1024$, $k_0 = k_1 = 128$.
Za pomocą RSA podpisujemy $y = 0||w||r^*||g_2(w)$.

PSS - c.d.



h, g_1, g_2 — „losowe funkcje” takie jak w przypadku OAEP.
r — losowy ciąg.
0 jest po to, żeby nie mieć pewność, że jesteśmy w Z_n^*.

Co można udowodnić na temat PSS?

Nieformalnie: PSS jest tylko nieznacznie mniej bezpieczny niż odwrócenie RSA.

Co można udowodnić na temat PSS?

Nieformalnie: PSS jest tylko nieznacznie mniej bezpieczny niż odwrócenie RSA.

To oczywiście można udowodnić tylko w modelu z losową wyrocznią

Użycie PSS

Wariant PSS jest (rekomendowną) częścią standardu PKCS #1,

Podsumowanie

Konstrukcje przedstawione na dzisiejszym wykładzie są praktyczne, tzn. ich użycie nie wiąże się ze znaczącym spadkiem wydajności.

Podsumowanie

Konstrukcje przedstawione na dzisiejszym wykładzie są praktyczne, tzn. ich użycie nie wiąże się ze znaczącym spadkiem wydajności.

Ceną jest to, że musimy się zadowolić dowodami w modelu z losową wyrocznią.

Podsumowanie

Konstrukcje przedstawione na dzisiejszym wykładzie są praktyczne, tzn. ich użycie nie wiąże się ze znaczącym spadkiem wydajności.

Ceną jest to, że musimy się zadowolić dowodami w modelu z losową wyrocznią.

W literaturze jest szereg konstrukcji teoretycznych (o większej złożoności obliczeniowej), które nie wymagają tego typu założeń.
Schematy podpisu opisane są np. w: S. Goldwasser i M. Bellare, Lecture Notes on Cryptography, rozdział 9.4.

Podsumowanie

Konstrukcje przedstawione na dzisiejszym wykładzie są praktyczne, tzn. ich użycie nie wiąże się ze znaczącym spadkiem wydajności.

Ceną jest to, że musimy się zadowolić dowodami w modelu z losową wyrocznią.

W literaturze jest szereg konstrukcji teoretycznych (o większej złożoności obliczeniowej), które nie wymagają tego typu założeń.
Schematy podpisu opisane są np. w: S. Goldwasser i M. Bellare, Lecture Notes on Cryptography, rozdział 9.4.

Popularnym (wśród teoretyków) schematem szyfrowania jest system Cramera i Shoupa:
R. Cramer, V. Shoup A Practical Public Key Cryptosystem

Literatura I

M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway.

Relations among notions of security for public-key encryption schemes.

In *Advances in Cryptology- Crypto 98*, volume 1462 of *LNCS*, 1998.

D. Bleichenbacher.

Chosen ciphertext attacks against protocols based on the rsa encryption standard PKCS #1.

In *Advances in Cryptology, Crypto '98*, LNCS, pages 1 – 12, 1998.