Private-Key Encryption

Federico Pintore[1]

[1]Mathematical Institute

Outline

Outline

Computational Security

Definition (Concrete version)

An encryption scheme is (t, ϵ)-secure if any adversary running for time at most t succeeds in breaking the scheme with probability at most ϵ.

Definition (Asymptotic version)

An encryption scheme is secure if any *probabilistic polynomial-time algorithm in* n (PPT) succeeds in breaking the scheme with at most negligible probability (in n).

The running times of the encryption scheme and of the PPT algorithm, and the success probability of the latter are functions of n.

Notation

Polynomial time algorithm in n: its running time $f(n)$ belongs to $\mathcal{O}(p(n))$ for a given constant positive polynomial $p(n) \in \mathbb{N}[n]$ ($\exists N, c$ s.t $f(n) \leq cp(n)$ for all $n \geq N$).

Negligible function $g(n)$: for each positive polynomial $p(n) \in \mathbb{N}[n]$, there exists N s.t. $g(n) \leq 1/p(n)$ for all $n \geq N$.

Probabilistic Algorithm: the algorithm has access to a random source.

Perfect Indistinguishability

Perfect Indistinguishability Experiment $\text{PrivK}^{\text{perfect}-\text{ind}}_{\mathcal{A},E}$

Challenger Ch **Adversary** $\mathcal{A}$

$\xleftarrow{m_0, m_1, |m_0| = |m_1|}$

$b \leftarrow\$ \{0, 1\}$

$\xrightarrow{c = \text{Enc}(k, m_b)}$ Outputs their guess b'

Definition

An encryption scheme is perfectly indistinguishable if for every adversary $\mathcal{A}$ the following holds:

$$\Pr[\text{PrivK}^{\text{perfect}-\text{IND}}_{\mathcal{A},E} = 1] = 1/2$$

Where $\text{PrivK}^{\text{perfect}-\text{IND}}_{\mathcal{A},E} = 1$ if $b' = b$, and 0 otherwise.

Computational Indistinguishability

Adversarial Indistinguishability Experiment $\text{PrivK}^{\text{eav}}_{\mathcal{A},E}$

Challenger Ch **Adversary** $\mathcal{A}$

$\xleftarrow{m_0, m_1, |m_0| = |m_1|}$

$b \leftarrow_{\$} \{0, 1\}$

$\xrightarrow{c = \text{Enc}(k, m_b)}$ Outputs their guess b'

Definition

An encryption scheme is computationally indistinguishable if, for all PPT adversary $\mathcal{A}$, there exists a negligible function $\text{negl}(n)$ *s.t.*

$$\Pr[\text{PrivK}^{\text{eav}}_{\mathcal{A},E} = 1] \leq 1/2 + \text{negl}(n)$$

Where $\text{PrivK}^{\text{eav}}_{\mathcal{A},E} = 1$ if $b' = b$, and 0 otherwise.

- Does a computationally indistinguishable private-key encryption scheme exist?
- Does a computationally indistinguishable private-key encryption scheme with $|k| \leq |m|$ exist?

What if we use pseudo-random generators?

- Does a computationally indistinguishable private-key encryption scheme exist?
- Does a computationally indistinguishable private-key encryption scheme with $|k| \leq |m|$ exist?

What if we use pseudo-random generators?

- Does a computationally indistinguishable private-key encryption scheme exist?
- Does a computationally indistinguishable private-key encryption scheme with $|k| \leq |m|$ exist?

What if we use pseudo-random generators?

Outline

Pseudo-Random Generators (PRGs)

- *Pseudorandomness* is a property of a distribution on strings. Say you have a distribution X on ℓ-bit strings that assigns some probability to every string in $\{0,1\}^\ell$. Pseudorandomness means that sampling form X is indistinguishable from sampling a uniform string of length ℓ.

- Ideally, we want a PRG to efficiently produce, from short seeds, longer bit strings that appear uniform.

Pseudo-Random Generators (PRGs)

- *Pseudorandomness* is a property of a distribution on strings. Say you have a distribution X on ℓ-bit strings that assigns some probability to every string in $\{0,1\}^\ell$. Pseudorandomness means that sampling form X is indistinguishable from sampling a uniform string of length ℓ.

- Ideally, we want a PRG to efficiently produce, from short seeds, longer bit strings that appear uniform.

Pseudo-Random Generators PRGs

Definition

Let $G : \{0,1\}^n \to \{0,1\}^{\ell(n)}$ *a deterministic polynomial-time algorithm in* n*, where* $\ell(n) > n$*.* G *is a secure pseudorandom generator if* ***for all*** *PPT distinguisher* D *(also called statistical test), the advantage*

$$\mathsf{Adv}^{\mathrm{prg}}_{G,D}(n) = |\Pr[D(r) = 1] - \Pr[D(G(s)) = 1]| \leq \mathsf{negl}(n)$$

where the probabilities are taken over uniform choice of $s \in \{0,1\}^n$*,* $r \in \{0,1\}^{\ell(n)}$ *and the randomness of* D*.*

- D outputs 1 when their guess is *pseudorandom string*.
- $\ell(n)$ is called *the expansion factor* of G.

Pseudorandom Generators

- Do PRGs exist? Can we construct them?
- Not if $NP = P$
- What is the weakest assumption under which we can construct PRGs?
- It is the existence of *one-way functions* (i.e. easy to compute-hard to invert)
- Practical constructions using stream and block ciphers

Pseudorandom Generators

- Do PRGs exist? Can we construct them?
- Not if $NP = P$
- What is the weakest assumption under which we can construct PRGs?
- It is the existence of *one-way functions* (i.e. easy to compute-hard to invert)
- Practical constructions using stream and block ciphers

Pseudorandom Generators

- Do PRGs exist? Can we construct them?
- Not if $NP = P$
- What is the weakest assumption under which we can construct PRGs?
- It is the existence of *one-way functions* (i.e. easy to compute-hard to invert)
- Practical constructions using stream and block ciphers

Pseudorandom Generators

- Do PRGs exist? Can we construct them?
- Not if $NP = P$
- What is the weakest assumption under which we can construct PRGs?
- It is the existence of *one-way functions* (i.e. easy to compute-hard to invert)
- Practical constructions using stream and block ciphers

Pseudorandom Generators

- Do PRGs exist? Can we construct them?
- Not if $NP = P$
- What is the weakest assumption under which we can construct PRGs?
- It is the existence of *one-way functions* (i.e. easy to compute-hard to invert)
- Practical constructions using stream and block ciphers

Fixed-length Encryption scheme using a PRG

Let G be a pseudorandom generator with expansion factor $\ell(n)$. For messages of length $\ell(n)$, we define the following encryption scheme $E = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$:

- $\mathsf{KeyGen}(n)$: It picks a uniform bit string k of length n, i.e. $k \in \{0,1\}^n$.
- Enc : it takes as input a key $k \in \{0,1\}^n$ and a message $m \in \{0,1\}^{\ell(n)}$, it outputs

$$c \leftarrow \mathsf{Enc}(G(k), m) = G(k) \oplus m$$

- Dec : it takes as input a key $k \in \{0,1\}^n$ and a ciphertext $c \in \{0,1\}^{\ell(n)}$, it outputs

$$m \leftarrow \mathsf{Dec}(G(k), c) = G(k) \oplus c.$$

Theorem

If G is a secure PRG, then the encryption scheme E derived from G is computationally indistinguishable.

Proof.

(By reduction) Let $\mathcal{A}$ a PPT adversary against the scheme E. $\mathcal{A}$ is exploited as a subroutine to construct a distinguisher D.

- D receives a bit string $w \in \{0,1\}^{\ell(n)}$
- D runs $\mathcal{A}$ to obtain two messages $m_0, m_1 \in \{0,1\}^{\ell(n)}$
- D chooses a random $b \in \{0,1\}$ ad sends $c = w \oplus m_b$ to $\mathcal{A}$
- upon reception of b' from $\mathcal{A}$, D outputs 1 if $b = b'$, 0 otherwise.

$$|\Pr[D(r) = 1] - \Pr[D(G(s)) = 1]| = |\Pr[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},OTP} = 1] -$$
$$\Pr[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},E} = 1]| = |1/2 - \Pr[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},E} = 1]| \leq \mathsf{negl}(n)$$

□

Stream Ciphers

- Terminology is not standard: it is either considered to be practical instantiations of pseudo-random generators or the encryption scheme which uses it.
- They produce as many random-looking bits as exactly needed.
- They are more flexible (no upper bound on the number of bits) and efficient (each application takes the exact number of random-looking bits that it requests)

Stream Ciphers

- A stream cipher consists of two main deterministic algorithms:
- Init(s, IV): takes a seed s and an optional *initialization vector* IV, and outputs an initial state st_0
- GetBits(st_i): takes the i-th state information st_i and outputs a bit y and an updated state, i.e. st_{i+1}

Stream Ciphers and PRGs

Construction of a PRG G_ℓ:

$\mathsf{st}_0 \leftarrow \mathsf{Init}(s, IV)$

for $i = 1, \cdots, \ell$;

$\quad (y_i, \mathsf{st}_i) \leftarrow \mathsf{GetBits}(\mathsf{st}_{i-1})$

return $y_1, \cdots, y_\ell$

A stream cipher is secure if:

- it takes no IV
- for any positive polynomial $\ell(n) \in \mathbb{N}[n]$ with $\ell(n) > n$, G_ℓ is a secure PRG

Stream Ciphers and PRGs

Construction of a PRG G_ℓ:

$\mathsf{st}_0 \leftarrow \mathsf{Init}(s, IV)$

for $i = 1, \cdots, \ell$;

$\quad (y_i, \mathsf{st}_i) \leftarrow \mathsf{GetBits}(\mathsf{st}_{i-1})$

return $y_1, \cdots, y_\ell$

A stream cipher is secure if:

- it takes no IV
- for any positive polynomial $\ell(n) \in \mathbb{N}[n]$ with $\ell(n) > n$, G_ℓ is a secure PRG

Examples of Stream Ciphers

- Linear-Feedback Shift Registers (LFSRs)
- RC4 by Ron Rivest 1987 (recent attack: AlFardan et al. 2013)
- eStream: Salsa 20, ChaCha (2008), and SOSEMANUK
- eStream competition page:
  `http://competitions.cr.yp.to/estream.html`

RC4

- The state in RC4 consists of the triplet (S, i, j). S is a 256-byte array that contains a permutation of the numbers $0, \cdots, 255$. Both $i, j \in \{0, \cdots, 255\}$.
- The key can be up to 256 byte long.

RC4- the Init() algorithm

Input: a 16-byte key

Output: Initial state (S, i, j)

for $i = 1, \cdots, 255$:

$S[i] \leftarrow i$ ▶ it sets S to the identity permutation

$k[i] \leftarrow k[i \mod 16]$ ▶ it expands the key to 256 bytes by repetition

$j = 0$

for $i = 1, \cdots, 255;$

$j \leftarrow j + S[i] + k[i] \mod 256$

Swap $S[i]$ and $S[j]$ ▶ "pseudo-random" swapping of S's elements

$i \leftarrow 0, j \leftarrow 0$

return (S, i, j)

RC4- the GetBits() algorithm

Input: Current state (S, i, j)

Output: byte y**, updated state** (S, i, j)

$i \leftarrow i + 1 \mod 256$

$j \leftarrow j + S[i] \mod 256$ ▶ changing j in a “pseudo-random” way

Swap $S[i]$ and $S[j]$

$t \leftarrow S[i] + S[j] \mod 256$

$y \leftarrow S[t]$

return $(S, i, j), y$

RC4: Security Analysis

- Biases in the second output byte of RC4: the probability that it is 0 is $1/128$ instead of $1/256$ for $S = 256$.
- Biases in further bytes
- Conclusion: not secure, nevertheless, "its usage is still running at about 30% of all TLS traffic" (Garman et al. March 2015)

RC4: Security Analysis [AlFardan et al. 2013]

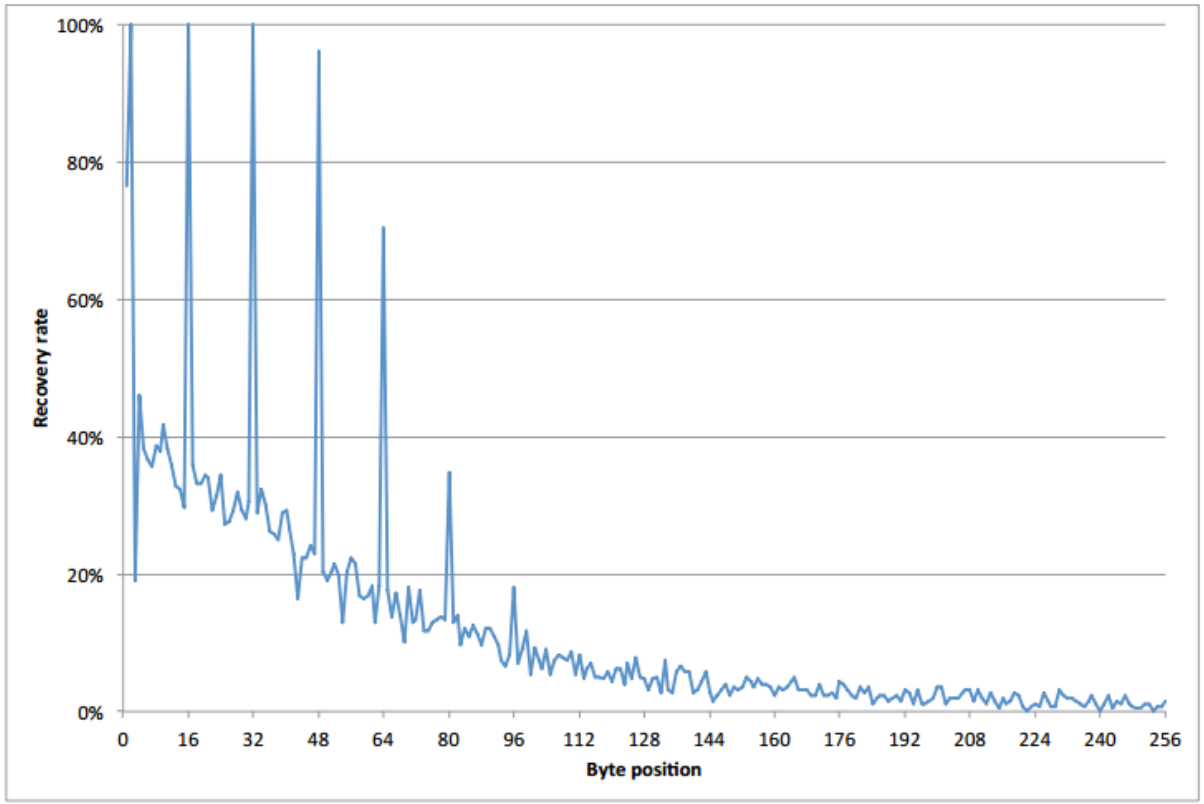

Figure: Recovery rate of the single-byte bias attack (based on 256 experiments)

Outline

More security definitions are needed...

- Encryption schemes from secure PRGs are computationally indistinguishable
- But what about multiple encryptions?
- What if the adversary wants to be challenged on two vectors of messages instead of two single messages?
- Obviously, he can trivially win the game (why?)
- Conclusion: deterministic encryption schemes are NOT secure under the multiple encryptions model.

More security definitions are needed...

- Encryption schemes from secure PRGs are computationally indistinguishable
- But what about multiple encryptions?
- What if the adversary wants to be challenged on two vectors of messages instead of two single messages?
- Obviously, he can trivially win the game (why?)
- Conclusion: deterministic encryption schemes are NOT secure under the multiple encryptions model.

CPA Indistinguishability Experiment $\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},E}$

Challenger Ch		**Adversary** $\mathcal{A}$
$k = \mathsf{KeyGen}(n)$		Access to an oracle $\mathsf{Enc}(k,\cdot)$
	$\xleftarrow{m_0,m_1,\lvert m_0\rvert=\lvert m_1\rvert}$	
$b \leftarrow \{0,1\}$		
	$\xrightarrow{c=\mathsf{Enc}(k,m_b)}$	Access to an oracle $\mathsf{Enc}(k,\cdot)$
		Outputs their guess b'

Definition

An encryption scheme E is CPA-secure if for all PPT $\mathcal{A}$ it holds:

$$\mathsf{Adv}^{\mathsf{cpa}}_{\mathcal{A},E}(n) = \Pr[\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},E}(n) = 1] \leq 1/2 + \mathsf{negl}(n)$$

Where $\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},E}(n) = 1$ if $b' = b$, and 0 otherwise.

CPA-security for multiple encryptions

- A PPT adversary $\mathcal{A}$ has access to an oracle $\mathrm{LR}_{k,b}$, where k is a key and b is a random bit.
- $\mathcal{A}$ queries the oracle on pair of messages $(m_{0,1}, m_{1,1})$, $(m_{0,2}, m_{1,2})$, $\ldots$, receiving $\mathrm{Enc}(k, m_{b,1})$, $\mathrm{Enc}(k, m_{b,2})$, $\ldots$
- $\mathcal{A}$ submits a guess $b' \in \{0, 1\}$

Theorem

If a private-key encryption scheme E is CPA-secure, it is CPA-secure for multiple encryptions.

The encryption cannot be deterministic ($\mathcal{A}$ can just query on (m, m) and (m', m))

CPA-security for multiple encryptions

- A PPT adversary $\mathcal{A}$ has access to an oracle $\mathrm{LR}_{k,b}$, where k is a key and b is a random bit.
- $\mathcal{A}$ queries the oracle on pair of messages $(m_{0,1}, m_{1,1})$, $(m_{0,2}, m_{1,2})$, $\ldots$, receiving $\mathrm{Enc}(k, m_{b,1})$, $\mathrm{Enc}(k, m_{b,2})$, $\ldots$
- $\mathcal{A}$ submits a guess $b' \in \{0, 1\}$

Theorem

If a private-key encryption scheme E is CPA-secure, it is CPA-secure for multiple encryptions.

The encryption cannot be deterministic ($\mathcal{A}$ can just query on (m, m) and (m', m))

CPA-security for multiple encryptions

- A PPT adversary $\mathcal{A}$ has access to an oracle $\mathrm{LR}_{k,b}$, where k is a key and b is a random bit.
- $\mathcal{A}$ queries the oracle on pair of messages $(m_{0,1}, m_{1,1})$, $(m_{0,2}, m_{1,2})$, $\ldots$, receiving $\mathrm{Enc}(k, m_{b,1})$, $\mathrm{Enc}(k, m_{b,2})$, $\ldots$
- $\mathcal{A}$ submits a guess $b' \in \{0, 1\}$

Theorem

If a private-key encryption scheme E is CPA-secure, it is CPA-secure for multiple encryptions.

The encryption cannot be deterministic ($\mathcal{A}$ can just query on (m, m) and (m', m))

CCA Indistinguishability Experiment $\mathsf{PrivK}^{\mathsf{cca}}_{\mathcal{A},E}$

Challenger Ch

$k = \mathsf{KeyGen}(n)$

$\xleftarrow{m_0, m_1, |m_0| = |m_1|}$

$b \leftarrow \{0,1\}$

$\xrightarrow{c = \mathsf{Enc}(k, m_b)}$

Adversary $\mathcal{A}$

Access to two oracles $\mathsf{Enc}(k,\cdot), \mathsf{Dec}(k,\cdot)$

Access to two oracles $\mathsf{Enc}(k,\cdot), \mathsf{Dec}(k,\cdot)^c$

Outputs their guess b'

Definition

An encryption scheme E is CCA-secure if for all PPT $\mathcal{A}$ it holds:

$$\mathsf{Adv}^{\mathsf{cca}}_{\mathcal{A},E}(n) = \Pr[\mathsf{PrivK}^{\mathsf{cca}}_{\mathcal{A},E}(n) = 1] \leq 1/2 + \mathsf{negl}(n)$$

Outline

Pseudo-Random Functions

- A generalisation of the notion of pseudo-random generators: we now consider a “random-looking” function.
- It is the pseudo-randomness of a distribution on functions.
- We are interested in *keyed functions*, i.e.

$$F : \{0,1\}^{\ell_{key}} \times \{0,1\}^{\ell_{in}} \to \{0,1\}^{\ell_{out}}$$

  Once k is chosen $F_k : \{0,1\}^{\ell_{in}} \to \{0,1\}^{\ell_{out}}$, $x \mapsto F(k,x)$ is a single-input function.
- F is *length-preserving* if the $\ell_{key} = \ell_{in} = \ell_{out}$.
- F is pseudo-random if the function F_k, for a uniform key k, is indistinguishable from a function chosen uniformly at random from the set of all functions with the same domain and range.

Pseudo-Random Functions

Definition

Let $Func[X, Y]$ be the set of all functions from X to Y.
$F : K \times X \to Y$ is a secure Pseudo-Random Function (PRF) if F

- *is efficiently computable*
- *for all PPT distinguishers $\mathcal{A}$*

$$\mathsf{Adv}^{\mathrm{prf}}_{F,\mathcal{A}}(n) = |\Pr[\mathcal{A}^{f()}(n) = 1] - \Pr[\mathcal{A}^{F_k()}(n) = 1]| \leq \mathsf{negl}(n)$$

where $f \in Func[X, Y]$, $k \in K$, and $\mathcal{A}$ has access to the function in question, i.e. either $f()$ and $F_k()$.

Note that $|Func[X, Y]| = |X|^{|Y|}$.

Pseudo-Random Permutations

A pseudo-random function $F : K \times X \to Y$ is an efficient pseudo-random permutation if the following hold:

- F_k is injective and $|X| = |Y|$
- F is deterministic and efficiently computable
- F_k^{-1} is efficiently computable

In practice: $F : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, where;

- 3DES: $n = 64$ bits, $k = 168$ bits
- AES: $n = 128$ bits, $k = 128, 192, 256$ bits

Pseudo-Random Permutations

A pseudo-random function $F : K \times X \to Y$ is an efficient pseudo-random permutation if the following hold:

- F_k is injective and $|X| = |Y|$
- F is deterministic and efficiently computable
- F_k^{-1} is efficiently computable

In practice: $F : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, where;

- 3DES: $n = 64$ bits, $k = 168$ bits
- AES: $n = 128$ bits, $k = 128, 192, 256$ bits

Strong Pseudo-Random Permutations

Definition

Let Perm_n *be the set of all permutations from* $\{0,1\}^n$ *to* $\{0,1\}^n$. *Let* $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ *be an efficient length-preserving, keyed permutation.* F *is a strong pseudo-random permutation if, for all PPT distinguishers* D, *there exists a negligible function* $\mathsf{negl}(n)$ *such that*

$$|\Pr[D^{f(),f^{-1}()}(n) = 1] - \Pr[D^{F_k(),F_k^{-1}()}(n) = 1]| \leq \mathsf{negl}(n)$$

where $f \in \mathsf{Perm}_n$, $k \in \{0,1\}^k$, *and* D *has access to the functions in question, i.e. either* $f(), f^{-1}()$ *or* $F_k(), F_k^{-1}()$.

- Strong PRP $\Rightarrow$ secure PRP
- $|\mathsf{Perm}_n| = 2^n!$

Encryption using PRP

Let $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a pseudo-random permutation. We define the following encryption scheme $E = (\text{KeyGen}, \text{Enc}, \text{Dec})$:

- KeyGen : it takes n and outputs a key $k \in \{0,1\}^n$.
- Enc: it takes a key $k \in \{0,1\}^n$ and a message m, it picks a random $r \leftarrow \{0,1\}^n$, and outputs

$$(c_0, c_1) \leftarrow (r, F_k(r) \oplus m).$$

- Dec: it takes a key k and a ciphertext $c = (c_0, c_1)$ and outputs

$$m \leftarrow (F_k(c_0) \oplus c_1).$$

Theorem

If F is a secure pseudo-random permutation then the encryption scheme E is CPA-secure.

Proof.

Let E' be a variant of E with a random permutation instead of F_k.

Suppose that a PPT adversary $\mathcal{A}$ can make $q(n)$ queries to the encryption oracle ($q(n) \in \mathcal{O}(p(n))$).

Let r_c be the first component of the challenge cipher-text.

- case 1: r_c didn't appear in any of the encryption queries. Then r_c is a uniform string and the probability to win the game is 1/2 (OTP is perfectly secret)
- r_c appeared in *at least* one of the queries. The probability of this event is at most $q(n)/2^n$.

Thus $\Pr[\text{PrivK}^{\text{cpa}}_{\mathcal{A},E'}(n) = 1] \leq 1/2 + q(n)/2^n$.

□

Proof.

We use $\mathcal{A}$ to construct a distinguisher D for the PRP.

On a query $m \in \{0,1\}^n$, D queries the oracle $\mathcal{O}$ on a uniform $r \in \{0,1\}^n$, receiving y. Then they reply with $(r, y+m)$.

If $\mathcal{A}$ wins the game, D outputs 1 (0 otherwise).

$$\Pr[D^{f()}(n) = 1] = \Pr[\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},E'}(n) = 1]$$

$$\Pr[D^{F_k()}(n) = 1] = \Pr[\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},E}(n) = 1]$$

Since F is a secure PRP we have

$$\left|\Pr[D^{F_k()}(n) = 1] - \Pr[D^{f()}(n) = 1]\right| \leq \mathsf{negl}(n)$$

Thus $\Pr[\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},E}(n) = 1] \leq 1/2 + q(n)/2^n + \mathsf{negl}(n)$. □

Notes:

- the sum of two negligible functions is a negligible function
- the product of a positive polynomial in $\mathbb{N}[n]$ and a negligible function is a negligible function

Further Reading (1)


Nadhem J AlFardan, Daniel J Bernstein, Kenneth G Paterson, Bertram Poettering, and Jacob CN Schuldt.
On the security of RC4 in TLS.
In *USENIX Security*, pages 305–320, 2013.


Boaz Barak and Shai Halevi.
A model and architecture for pseudo-random generation with applications to/dev/random.
In *Proceedings of the 12th ACM conference on Computer and communications security*, pages 203–212. ACM, 2005.


Daniel J Bernstein.
The Salsa20 Family of Stream Ciphers.
In *New stream cipher designs*, pages 84–97. Springer, 2008.

Further Reading (2)


Lenore Blum, Manuel Blum, and Mike Shub.
A simple unpredictable pseudo-random number generator.
SIAM Journal on computing, 15(2):364–383, 1986.


Christian Cachin.
Entropy measures and unconditional security in cryptography.
PhD thesis, SWISS FEDERAL INSTITUTE OF TECHNOLOGY ZURICH, 1997.


Scott Fluhrer, Itsik Mantin, and Adi Shamir.
Weaknesses in the key scheduling algorithm of RC4.
In *Selected areas in cryptography*, pages 1–24. Springer, 2001.

Further Reading (3)

Christina Garman, Kenneth G Paterson, and Thyla van der Merwe.
Attacks only get better: Password recovery attacks against RC4 in TLS.
2015.

Itsik Mantin and Adi Shamir.
A practical attack on broadcast RC4.
In *Fast Software Encryption*, pages 152–164. Springer, 2002.