Private Key Encryption

Computational security

1.Security is only guaranteed against efficient adversaries that run for some feasible amount of time

2. Adversaries can potentially succeed (i.e., security can potentially fail) with some very small probability.

Computational secrecy?

- Idea: relax perfect indistinguishability

- Two approaches
  - Concrete security
  - Asymptotic security

Computational indistinguishability (concrete)

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

- (t, ε)-indistinguishability:
  - Security may fail with probability ≤ ε
  - Restrict attention to attackers running in time ≤ t
    - Or, t CPU cycles

Asymptotic security

- Introduce *security parameter* n
  - For now, think of n as the key length

A scheme is secure if any ppt adversary succeeds in breaking the scheme with at most negligible probability.

Ppt-probabilistic polynomial-time

Efficient algorithms

A function f from the natural numbers to the nonnegative real numbers is polynomially bounded

- if there is a constant c such that $f(n) < nc$ for all n.
- An algorithm A runs in polynomial time if there exists a polynomial p such that, for every input $x \in \{0,1\}*$, the computation of $A(x)$ terminates within at most $p(|x|)$ steps

Negligible success probability.

- A negligible function is one that is asymptotically smaller than any inverse polynomial function. Formally:
- *A function f from the natural numbers to the nonnegative real numbers is negligible if for every polynomial **p** there is an **N** such that for all n > N it holds that .*

$$f(n) < \frac{1}{p(n)}$$

Negligible functions

Let $negl_1$ and $negl_2$ be negligible functions. Then,

- 1. The function

  $negl_3(n) = negl_1(n) + negl_2(n)$ is negligible.

- 2. For any polynomial p, the function

  $negl_4(n) = p(n) \cdot negl_1(n)$ is negligible.

Computationally Secure Encryption

DEFINITION 3.7 *A* private-key encryption scheme *consists of three probabilistic polynomial-time algorithms* (Gen, Enc, Dec) *such that:*

1. *The* key-generation algorithm Gen *takes as input* 1^n *(i.e., the security parameter written in unary) and outputs a key* k*; we write* $k \leftarrow \mathsf{Gen}(1^n)$ *(emphasizing that* Gen *is a randomized algorithm). We assume without loss of generality that any key* k *output by* $\mathsf{Gen}(1^n)$ *satisfies* $|k| \geq n$.

2. *The* encryption algorithm Enc *takes as input a key* k *and a plaintext message* $m \in \{0,1\}^*$*, and outputs a ciphertext* c*. Since* Enc *may be randomized, we write this as* $c \leftarrow \mathsf{Enc}_k(m)$.

3. *The* decryption algorithm Dec *takes as input a key* k *and a ciphertext* c*, and outputs a message* $m \in \{0,1\}^*$ *or an error. We denote a generic error bu the sumbol* $\perp$.

$\mathsf{Gen}(1^n)$ outputs a uniform n-bit string as the key

Computational indistinguishability (asymptotic version)

- Fix a scheme Π and some adversary A
- Define a randomized exp't $\text{PrivK}_{A,\Pi}(n)$:
  1. $A(1^n)$ outputs $m_0, m_1 \in \{0,1\}^*$ of equal length
  2. $k \leftarrow \text{Gen}(1^n)$, $b \leftarrow \{0,1\}$, $c \leftarrow \text{Enc}_k(m_b)$
  3. $b' \leftarrow A(c)$

  Adversary A *succeeds* if b = b', and we say the experiment evaluates to 1 in this case

Computational indistinguishability (asymptotic version)

- Π is *computationally indistinguishable* (aka *EAV-secure*) if for all PPT attackers A, there is a negligible function ε such that

$$\Pr[\mathrm{PrivK}_{A,\Pi}(n) = 1] \leq \tfrac{1}{2} + \varepsilon(n)$$

Example 1

- Consider a scheme where Gen(1^n) generates a uniform n-bit key, and the best attack is brute-force search of the key space
  - So if A runs in time t(n), then
    $$\Pr[\mathrm{PrivK}_{A,\Pi}(n) = 1] < \tfrac{1}{2} + O(t(n)/2^n)$$

- This scheme is EAV-secure!
  For any polynomial t, the function $t(n)/2^n$ is negligible

Encryption and plaintext length

- In practice, we want encryption schemes that can encrypt arbitrary-length messages
- Encryption does not hide the plaintext length (in general)
  - The definition takes this into account by requiring m_0, m_1 to have the same length
- But beware that leaking plaintext length can often lead to problems in the real world!
  - Obvious examples…
  - Database searches
  - Encrypting compressed data

Computational secrecy

- From now on, we will assume the computational setting by default
  - Usually, the *asymptotic* setting

Semantic Security

If an EAV-secure encryption scheme (Enc,Dec) (recall that if Gen is omitted, the key is a uniform n-bit string) is used to encrypt a uniform message $m \in \{0,1\}^l$, then for any ***i*** it is infeasible for an attacker given the ciphertext to guess the ***i***th bit of m (here denoted by m^i) with probability much better than 1/2.

THEOREM 3.10 *Let* $\Pi = (\mathsf{Enc}, \mathsf{Dec})$ *be a fixed-length private-key encryption scheme for messages of length* ℓ *that is EAV-secure. Then for all* PPT *adversaries* $\mathcal{A}$ *and* $i \in \{1, \ldots, \ell\}$*, there is a negligible function* negl *such that*

$$\Pr\left[\mathcal{A}(1^n, \mathsf{Enc}_k(m)) = m^i\right] \leq \frac{1}{2} + \mathsf{negl}(n), \tag{3.2}$$

where the probability is taken over uniform $m \in \{0,1\}^\ell$ *and* $k \in \{0,1\}^n$*, the randomness of* $\mathcal{A}$*, and the randomness of* Enc.

PROOF

Fix an arbitrary PPT adversary $\mathcal{A}$ and $i \in \{1, \ldots, \ell\}$. Let $I_0 \subset \{0,1\}^\ell$ be the set of all strings whose ith bit is 0, and let $I_1 \subset \{0,1\}^\ell$ be the set of all strings whose ith bit is 1. We have

$$\begin{aligned}
&\Pr\left[\mathcal{A}(1^n, \mathsf{Enc}_k(m)) = m^i\right] \\
&= \frac{1}{2} \cdot \Pr_{m_0 \leftarrow I_0}\left[\mathcal{A}(1^n, \mathsf{Enc}_k(m_0)) = 0\right] + \frac{1}{2} \cdot \Pr_{m_1 \leftarrow I_1}\left[\mathcal{A}(1^n, \mathsf{Enc}_k(m_1)) = 1\right].
\end{aligned}$$

Construct the following eavesdropping adversary $\mathcal{A}'$:

Adversary $\mathcal{A}'(1^n)$:

1. Choose uniform $m_0 \in I_0$ and $m_1 \in I_1$. Output m_0, m_1.
2. Upon observing a ciphertext c, invoke $\mathcal{A}(1^n, c)$. If $\mathcal{A}$ outputs 0, output $b' = 0$; otherwise, output $b' = 1$.

Note that $\mathcal{A}'$ runs in polynomial time since $\mathcal{A}$ does.

PROOF

By the definition of experiment $\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A}',\Pi}(n)$, we have that $\mathcal{A}'$ succeeds if and only if $\mathcal{A}$ outputs b upon receiving $\mathsf{Enc}_k(m_b)$. So

$$\begin{aligned}
&\Pr\left[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A}',\Pi}(n) = 1\right] \\
&= \Pr\left[\mathcal{A}(1^n, \mathsf{Enc}_k(m_b)) = b\right] \\
&= \frac{1}{2} \cdot \Pr_{m_0 \leftarrow I_0}\left[\mathcal{A}(1^n, \mathsf{Enc}_k(m_0)) = 0\right] + \frac{1}{2} \cdot \Pr_{m_1 \leftarrow I_1}\left[\mathcal{A}(1^n, \mathsf{Enc}_k(m_1)) = 1\right] \\
&= \Pr\left[\mathcal{A}(1^n, \mathsf{Enc}_k(m)) = m^i\right].
\end{aligned}$$

Since $(\mathsf{Enc}, \mathsf{Dec})$ is EAV-secure, there is a negligible function negl such that $\Pr\left[\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A}',\Pi}(n) = 1\right] \leq \frac{1}{2} + \mathsf{negl}(n)$. We conclude that

$$\Pr\left[\mathcal{A}(1^n, \mathsf{Enc}_k(m)) = m^i\right] \leq \frac{1}{2} + \mathsf{negl}(n),$$

EAV-security

- EAV-security implies that no ppt adversary can learn any function ***f*** of the plaintext m from the ciphertext, regardless of the distribution $\mathcal{D}$ of m.
- If there exists an adversary who, with some probability, correctly computes $f(m)$ when given $\mathrm{Enc}_k(m)$, then there exists an adversary that can correctly compute $f(m)$ with almost the same probability *without being given the ciphertext at all* (and knowing only the distribution D of m).

THEOREM 3.11 *Let* $(\mathsf{Enc}, \mathsf{Dec})$ *be a fixed-length private-key encryption scheme for messages of length* ℓ *that is EAV-secure. Then for any* PPT *algorithm* $\mathcal{A}$ *there is a* PPT *algorithm* $\mathcal{A}'$ *such that for any distribution* $\mathcal{D}$ *over* $\{0,1\}^\ell$ *and any function* $f : \{0,1\}^\ell \to \{0,1\}$*, there is a negligible function* negl *such that:*

$$\left| \Pr\left[\mathcal{A}(1^n, \mathsf{Enc}_k(m)) = f(m)\right] - \Pr\left[\mathcal{A}'(1^n) = f(m)\right] \right| \leq \mathsf{negl}(n),$$

where the first probability is taken over choice of m *according to* $\mathcal{D}$*, uniform choice of* $k \in \{0,1\}^n$*, and the randomness of* $\mathcal{A}$ *and* Enc*, and the second probability is taken over choice of* m *according to* $\mathcal{D}$ *and the randomness of* $\mathcal{A}'$*.*

Semantic security

DEFINITION 3.12 *A private-key encryption scheme* (Enc, Dec) *is* semantically secure in the presence of an eavesdropper *if for every* PPT *algorithm* $\mathcal{A}$ *there exists a* PPT *algorithm* $\mathcal{A}'$ *such that for any* PPT *algorithm* Samp *and polynomial-time computable functions* f *and* h*, the following is negligible:*

$$\left| \Pr[\mathcal{A}(1^n, \mathsf{Enc}_k(m), h(m)) = f(m)] - \Pr[\mathcal{A}'(1^n, |m|, h(m)) = f(m)] \right|,$$

The adversary A is given the ciphertext $\mathsf{Enc}_k(m)$ as well as the external information h(m), and attempts to guess the value of f(m). Algorithm A0 also attempts to guess the value of f(m), but is given only the length of m and h(m).

h(m): arbitrary "external" information about the message m that may be available to the adversary via other means

Semantically secure

THEOREM 3.13 *A private-key encryption scheme has indistinguishable encryptions in the presence of an eavesdropper (i.e., is EAV-secure) if and only if it is semantically secure in the presence of an eavesdropper.*

Pseudorandomness

Pseudorandomness

- Important building block for computationally secure encryption
- Important concept in cryptography
- A pseudorandom generator ***G*** is an efficient, deterministic algorithm for transforming a short, uniform string (called a seed) into a longer, "uniform looking" (or "pseudorandom") output string.
- Stated differently, a pseudorandom generator uses a small amount of true randomness in order to generate a large amount of pseudorandomness.

Pseudorandom generator

- The output of a pseudorandom generator "looks like" a uniformly generated string to any efficient observer
- a pseudorandom string is just as good as a uniform string

Formal definition

DEFINITION 3.14 *Let G be a deterministic polynomial-time algorithm such that for any n and any input $s \in \{0,1\}^n$, the result $G(s)$ is a string of length $\ell(n)$. G is a* pseudorandom generator *if the following conditions hold:*

1. **(Expansion.)** *For every n it holds that $\ell(n) > n$.*
2. **(Pseudorandomness.)** *For any* PPT *algorithm D, there is a negligible function* negl *such that*

$$\left| \Pr[D(G(s)) = 1] - \Pr[D(r) = 1] \right| \leq \mathsf{negl}(n),$$

where the first probability is taken over uniform choice of $s \in \{0,1\}^n$ and the randomness of D, and the second probability is taken over uniform choice of $r \in \{0,1\}^{\ell(n)}$ and the randomness of D.

We call $\ell(n)$ the expansion factor *of G.*

The seed and its length

- The seed for a pseudorandom generator is analogous to the key used by an encryption scheme,
- As in the case of a cryptographic key—the seed **s** must be chosen uniformly and be kept secret from any adversary if we want G(s) to look random.
- **s** must be long enough so that it is not feasible to enumerate all possible seeds.

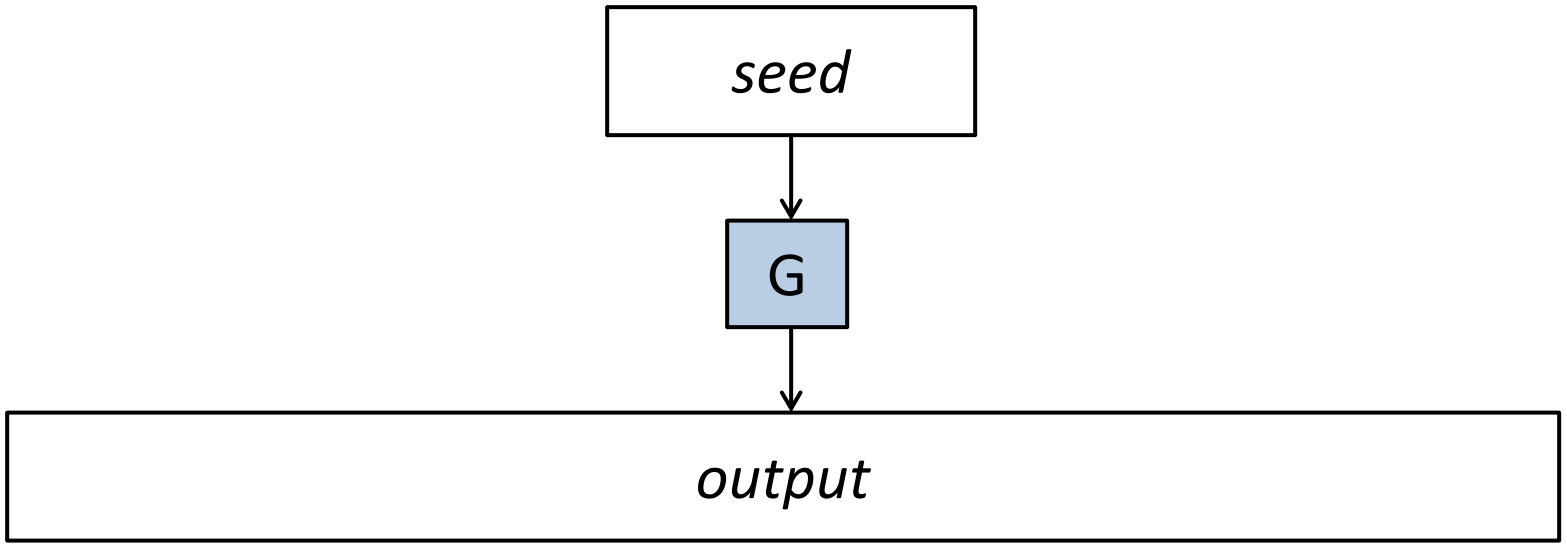
What does “uniform” mean?

- “Uniformity” is not a property of a *string*, but a property of a *distribution*

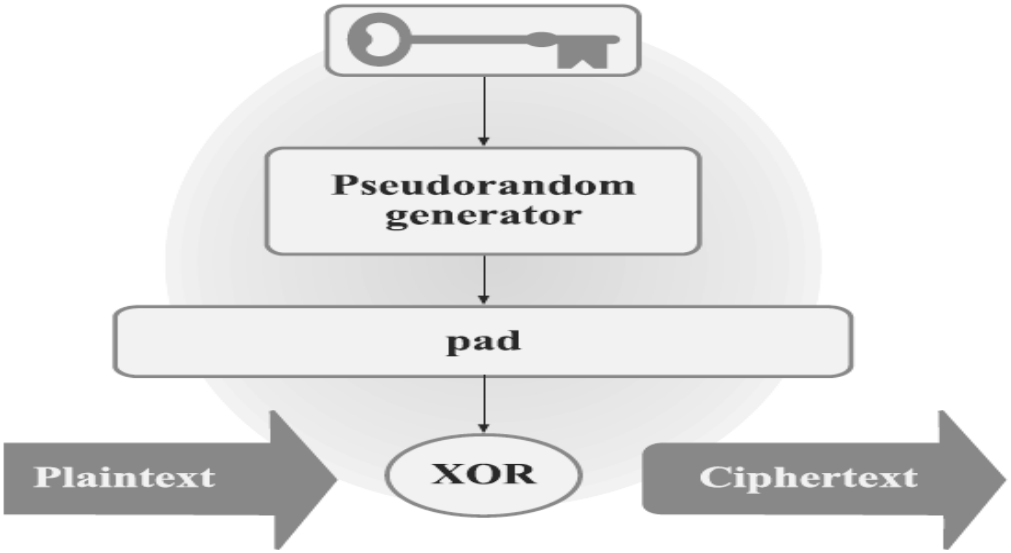
- A distribution on *n*-bit strings is a function $D: \{0,1\}^n \rightarrow [0,1]$ such that $\Sigma_x D(x) = 1$
  - The *uniform* distribution on *n*-bit strings, denoted U_n, assigns probability 2^{-n} to every $x \in \{0,1\}^n$

PRGs

- Let G be a deterministic, poly-time algorithm that is *expanding*, i.e., $|G(x)| = p(|x|) > |x|$



Encryption with a pseudorandom generator



A private-key encryption scheme based on any pseudorandom generator is

- EAV-secure
- Concretely secure