Relaxing Security

We will relax security in 2 ways:

1. We will only claim security against *polynomial time* adversaries.
2. Even an efficient adversary might have some very (very very) small advantage in distinguishing an encryption of m_0 from an encryption of m_1.

Relaxing Security

We will relax security in 2 ways:

1. We will only claim security against *polynomial time* adversaries.
2. Even an efficient adversary might have some very (very very) small advantage in distinguishing an encryption of m_0 from an encryption of m_1.

Relaxing Security

We will relax security in 2 ways:

1. We will only claim security against *polynomial time* adversaries.
2. Even an efficient adversary might have some very (very very) small advantage in distinguishing an encryption of m_0 from an encryption of m_1.

Relaxing Security

We will relax security in 2 ways:

1. We will only claim security against *polynomial time* adversaries.
2. Even an efficient adversary might have some very (very very) small advantage in distinguishing an encryption of m_0 from an encryption of m_1.

We could quantify this *concretely*:
An encryption scheme is (t, ϵ)-secure if, after running for t computational steps (or cycles), the best adversary has an ϵ advantage in distinguishing two encryptions.

Relaxing Security

We will relax security in 2 ways:

1. We will only claim security against *polynomial time* adversaries.
2. Even an efficient adversary might have some very (very very) small advantage in distinguishing an encryption of m_0 from an encryption of m_1.

We could quantify this *concretely*:
An encryption scheme is (t, ϵ)-secure if, after running for t computational steps (or cycles), the best adversary has an ϵ advantage in distinguishing two encryptions.

For example, if our encryption scheme is $(2^{120}, 2^{-40})$-secure, an adversary that can perform 2^{60} operations per second from the time of the big bang, would gain less than a one-in-a-trillion advantage.

Relaxing Security

We will relax security in 2 ways:

1. We will only claim security against *polynomial time* adversaries.
2. Even an efficient adversary might have some very (very very) small advantage in distinguishing an encryption of m_0 from an encryption of m_1.

We could quantify this *concretely*:
An encryption scheme is (t, ϵ)-secure if, after running for t computational steps (or cycles), the best adversary has an ϵ advantage in distinguishing two encryptions.

For example, if our encryption scheme is $(2^{120}, 2^{-40})$-secure, an adversary that can perform 2^{60} operations per second from the time of the big bang, would gain less than a one-in-a-trillion advantage.

The drawback to concrete security is that a "timestep" or "cycle" is platform dependent. We will instead use asymptotic notation for quantifying the adversary's advantage.

Relaxing Security

We will relax security in 2 ways:

1. We will only claim security against *polynomial time* adversaries.
2. Even an efficient adversary might have some very (very very) small advantage in distinguishing an encryption of m_0 from an encryption of m_1.

We could quantify this *concretely*:
An encryption scheme is (t, ϵ)-secure if, after running for t computational steps (or cycles), the best adversary has an ϵ advantage in distinguishing two encryptions.

For example, if our encryption scheme is $(2^{120}, 2^{-40})$-secure, an adversary that can perform 2^{60} operations per second from the time of the big bang, would gain less than a one-in-a-trillion advantage.

The drawback to concrete security is that a "timestep" or "cycle" is platform dependent. We will instead use asymptotic notation for quantifying the adversary's advantage.

Recall: A polynomial time algorithm (in our case, adversary) has a runtime that grows polynomially in its input size. **What is our input here?**

Relaxing Security

We will relax security in 2 ways:

1. We will only claim security against *polynomial time* adversaries.
2. Even an efficient adversary might have some very (very very) small advantage in distinguishing an encryption of m_0 from an encryption of m_1.

We could quantify this *concretely*:
An encryption scheme is (t, ϵ)-secure if, after running for t computational steps (or cycles), the best adversary has an ϵ advantage in distinguishing two encryptions.

For example, if our encryption scheme is $(2^{120}, 2^{-40})$-secure, an adversary that can perform 2^{60} operations per second from the time of the big bang, would gain less than a one-in-a-trillion advantage.

The drawback to concrete security is that a "timestep" or "cycle" is platform dependent. We will instead use asymptotic notation for quantifying the adversary's advantage.

Recall: A polynomial time algorithm (in our case, adversary) has a runtime that grows polynomially in its input size. What is our input here? **Key length!**

Relaxing Security

We will relax security in 2 ways:

1. We will only claim security against *polynomial time* adversaries.
2. Even an efficient adversary might have some very (very very) small advantage in distinguishing an encryption of m_0 from an encryption of m_1.

We could quantify this *concretely*:
An encryption scheme is (t, ϵ)-secure if, after running for t computational steps (or cycles), the best adversary has an ϵ advantage in distinguishing two encryptions.

For example, if our encryption scheme is $(2^{120}, 2^{-40})$-secure, an adversary that can perform 2^{60} operations per second from the time of the big bang, would gain less than a one-in-a-trillion advantage.

The drawback to concrete security is that a "timestep" or "cycle" is platform dependent. We will instead use asymptotic notation for quantifying the adversary's advantage.

Recall: A polynomial time algorithm (in our case, adversary) has a runtime that grows polynomially in its input size. What is our input here? Key length!

We will use a security parameter, n, as input to Gen.
We want to build encryption schemes that:
For any adversary with runtime that increases as poly(n) ,
The adversary's distinguishing advantage is negl(n).

Negligible Functions

A negligible function approaches 0 faster than any inverse polynomial.

Negligible Functions

A negligible function approaches 0 faster than any inverse polynomial.

Negligible Functions

A function f from the natural numbers to the non-negtive real numbers is **negligible** if for every positive polynomial p, there exists a natural number N, such that for all integers $n > N$, $f(n) < 1/p(n)$.

Negligible Functions

A negligible function approaches 0 faster than any inverse polynomial.

Negligible Functions

A function f from the natural numbers to the non-negtive real numbers is **negligible** if for every positive polynomial p, there exists a natural number N, such that for all integers $n > N$, $f(n) < 1/p(n)$.

Examples:
2^{-n}, $2^{-\frac{n}{2}}$, $2^{-\sqrt{n}}$, $n^{-\log n}$

Negligible Functions

A negligible function approaches 0 faster than any inverse polynomial.

Negligible Functions

A function f from the natural numbers to the non-negtive real numbers is **negligible** if for every positive polynomial p, there exists a natural number N, such that for all integers $n > N$, $f(n) < 1/p(n)$.

Examples:
2^{-n}, $2^{-\frac{n}{2}}$, $2^{-\sqrt{n}}$, $n^{-\log n}$

Where do these become smaller than $\frac{1}{n^5}$?

$$2^{-n} < \frac{1}{n^5} \text{ for } n \geq 23$$

$$2^{-\frac{n}{2}} < \frac{1}{n^5} \text{ for } n \geq 59$$

$$2^{-\sqrt{n}} < \frac{1}{n^5} \text{ for } n \geq 3454$$

$$n^{-\log n} < \frac{1}{n^5} \text{ for } n \geq 33$$

Negligible Functions

A negligible function approaches 0 faster than any inverse polynomial.

Negligible Functions

A function f from the natural numbers to the non-negtive real numbers is **negligible** if for every positive polynomial p, there exists a natural number N, such that for all integers $n > N$, $f(n) < 1/p(n)$.

Examples:
2^{-n}, $2^{-\frac{n}{2}}$, $2^{-\sqrt{n}}$, $n^{-\log n}$

Where do these become smaller than $\frac{1}{n^5}$? For $n = 2^8 = 256$, $\frac{1}{n^5} = 2^{-40}$.

$$2^{-n} < \frac{1}{n^5} \text{ for } n \geq 23 \qquad n = 40$$

$$2^{-\frac{n}{2}} < \frac{1}{n^5} \text{ for } n \geq 59 \qquad n = 80$$

$$2^{-\sqrt{n}} < \frac{1}{n^5} \text{ for } n \geq 3454 \qquad n = 1600$$

$$n^{-\log n} < \frac{1}{n^5} \text{ for } n \geq 33 \qquad n \approx 80$$

Negligible Functions

A negligible function approaches 0 faster than any inverse polynomial.

Negligible Functions

A function f from the natural numbers to the non-negtive real numbers is **negligible** if for every positive polynomial p, there exists a natural number N, such that for all integers $n > N$, $f(n) < 1/p(n)$.

Examples:
2^{-n}, $2^{-\frac{n}{2}}$, $2^{-\sqrt{n}}$, $n^{-\log n}$

Where do these become smaller than $\frac{1}{n^5}$? For $n = 2^8 = 256$, $\frac{1}{n^5} = 2^{-40}$.

$$2^{-n} < \frac{1}{n^5} \text{ for } n \geq 23 \qquad n = 40$$

$$2^{-\frac{n}{2}} < \frac{1}{n^5} \text{ for } n \geq 59 \qquad n = 80$$

$$2^{-\sqrt{n}} < \frac{1}{n^5} \text{ for } n \geq 3454 \qquad n = 1600$$

$$n^{-\log n} < \frac{1}{n^5} \text{ for } n \geq 33 \qquad n \approx 80$$

Note: $2^{-\sqrt{n}} < n^{-\log n}$ for every $n > 65,536$.

Adding Negligible Functions

Negligible Functions

A function f from the natural numbers to the non-negtive real numbers is **negligible** if for every positive polynomial p, there exists a natural number N, such that for all integers $n > N$, $f(n) < 1/p(n)$.

Adding Negligible Functions

Negligible Functions

A function f from the natural numbers to the non-negtive real numbers is **negligible** if for every positive polynomial p, there exists a natural number N, such that for all integers $n > N$, $f(n) < 1/p(n)$.

Suppose we have 2 negligible functions: $\text{negl}_1(x)$ and $\text{negl}_2(x)$.

Claim: The function $f(x) = \text{negl}_1(x) + \text{negl}_2(x)$ is negligible.

Adding Negligible Functions

Negligible Functions

A function f from the natural numbers to the non-negtive real numbers is **negligible** if for every positive polynomial p, there exists a natural number N, such that for all integers $n > N$, $f(n) < 1/p(n)$.

Suppose we have 2 negligible functions: $\text{negl}_1(x)$ and $\text{negl}_2(x)$.

Claim: The function $f(x) = \text{negl}_1(x) + \text{negl}_2(x)$ is negligible.

We have to show that:
for any polynomial $p(x)$, $\exists N \in \mathbb{N}$ such that $\forall n > N, f(n) < \frac{1}{p(n)}$.

Adding Negligible Functions

Negligible Functions

A function f from the natural numbers to the non-negtive real numbers is **negligible** if for every positive polynomial p, there exists a natural number N, such that for all integers $n > N$, $f(n) < 1/p(n)$.

Suppose we have 2 negligible functions: $\text{negl}_1(x)$ and $\text{negl}_2(x)$.

Claim: The function $f(x) = \text{negl}_1(x) + \text{negl}_2(x)$ is negligible.

We have to show that:
for any polynomial $p(x)$, $\exists N \in \mathbb{N}$ such that $\forall n > N, f(n) < \frac{1}{p(n)}$.

Let $q(x) = 2p(x)$. Because $\text{negl}_1(x)$ is a negligible function, there exists some N_1 such that $\forall n > N_1, \text{negl}_1(n) < \frac{1}{q(n)}$.

Adding Negligible Functions

Negligible Functions

A function f from the natural numbers to the non-negtive real numbers is **negligible** if for every positive polynomial p, there exists a natural number N, such that for all integers $n > N$, $f(n) < 1/p(n)$.

Suppose we have 2 negligible functions: $\text{negl}_1(x)$ and $\text{negl}_2(x)$.

Claim: The function $f(x) = \text{negl}_1(x) + \text{negl}_2(x)$ is negligible.

We have to show that:
for any polynomial $p(x)$, $\exists N \in \mathbb{N}$ such that $\forall n > N, f(n) < \frac{1}{p(n)}$.

Let $q(x) = 2p(x)$. Because $\text{negl}_1(x)$ is a negligible function, there exists some N_1 such that $\forall n > N_1, \text{negl}_1(n) < \frac{1}{q(n)}$.

Let $q(x) = 2p(x)$. Because $\text{negl}_2(x)$ is a negligible function, there exists some N_2 such that $\forall n > N_2, \text{negl}_2(n) < \frac{1}{q(n)}$.

Adding Negligible Functions

Negligible Functions

A function f from the natural numbers to the non-negtive real numbers is **negligible** if for every positive polynomial p, there exists a natural number N, such that for all integers $n > N$, $f(n) < 1/p(n)$.

Suppose we have 2 negligible functions: $\text{negl}_1(x)$ and $\text{negl}_2(x)$.

Claim: The function $f(x) = \text{negl}_1(x) + \text{negl}_2(x)$ is negligible.

We have to show that:
for any polynomial $p(x)$, $\exists N \in \mathbb{N}$ such that $\forall n > N, f(n) < \frac{1}{p(n)}$.

Let $q(x) = 2p(x)$. Because $\text{negl}_1(x)$ is a negligible function, there exists some N_1 such that $\forall n > N_1, \text{negl}_1(n) < \frac{1}{q(n)}$.

Let $q(x) = 2p(x)$. Because $\text{negl}_2(x)$ is a negligible function, there exists some N_2 such that $\forall n > N_2, \text{negl}_2(n) < \frac{1}{q(n)}$.

Define $N = \max(N_1, N_2)$.
$\forall n > N, \text{negl}_1(n) + \text{negl}_2(n) < \frac{2}{q(n)} = \frac{2}{2p(n)} = \frac{1}{p(n)}$.

Multiplying a Negligible Function by a Polynomial

Negligible Functions

A function f from the natural numbers to the non-negtive real numbers is **negligible** if for every positive polynomial p, there exists a natural number N, such that for all integers $n > N$, $f(n) < 1/p(n)$.

Suppose we have negligible function: negl(x) and a polynomial, $p(x)$.

Claim: The function $f(x) = \text{negl}(x) \cdot \text{p}(x)$ is negligible.

Multiplying a Negligible Function by a Polynomial

Negligible Functions

A function f from the natural numbers to the non-negtive real numbers is **negligible** if for every positive polynomial p, there exists a natural number N, such that for all integers $n > N$, $f(n) < 1/p(n)$.

Suppose we have negligible function: negl(x) and a polynomial, $p(x)$.

Claim: The function $f(x) = \text{negl(x)} \cdot \text{p(x)}$ is negligible.

We have to show that:
for any polynomial $q(x)$, $\exists N \in \mathbb{N}$ such that $\forall n > N, f(n) < \frac{1}{q(n)}$.

Multiplying a Negligible Function by a Polynomial

Negligible Functions

A function f from the natural numbers to the non-negtive real numbers is **negligible** if for every positive polynomial p, there exists a natural number N, such that for all integers $n > N$, $f(n) < 1/p(n)$.

Suppose we have negligible function: negl(x) and a polynomial, $p(x)$.

Claim: The function $f(x) = \text{negl(x)} \cdot \text{p(x)}$ is negligible.

We have to show that:
for any polynomial $q(x)$, $\exists N \in \mathbb{N}$ such that $\forall n > N, f(n) < \frac{1}{q(n)}$.

Consider the polynomial $\frac{q(x)}{p(x)}$. Since negl(x) is negligible, there exists $N \in \mathbb{N}$ such that $\forall n > N, \text{negl(n)} < \frac{\text{q(n)}}{\text{p(n)}}$

Multiplying a Negligible Function by a Polynomial

Negligible Functions

A function f from the natural numbers to the non-negtive real numbers is **negligible** if for every positive polynomial p, there exists a natural number N, such that for all integers $n > N$, $f(n) < 1/p(n)$.

Suppose we have negligible function: negl(x) and a polynomial, $p(x)$.

Claim: The function $f(x) = \text{negl(x)} \cdot \text{p(x)}$ is negligible.

We have to show that:
for any polynomial $q(x)$, $\exists N \in \mathbb{N}$ such that $\forall n > N, f(n) < \frac{1}{q(n)}$.

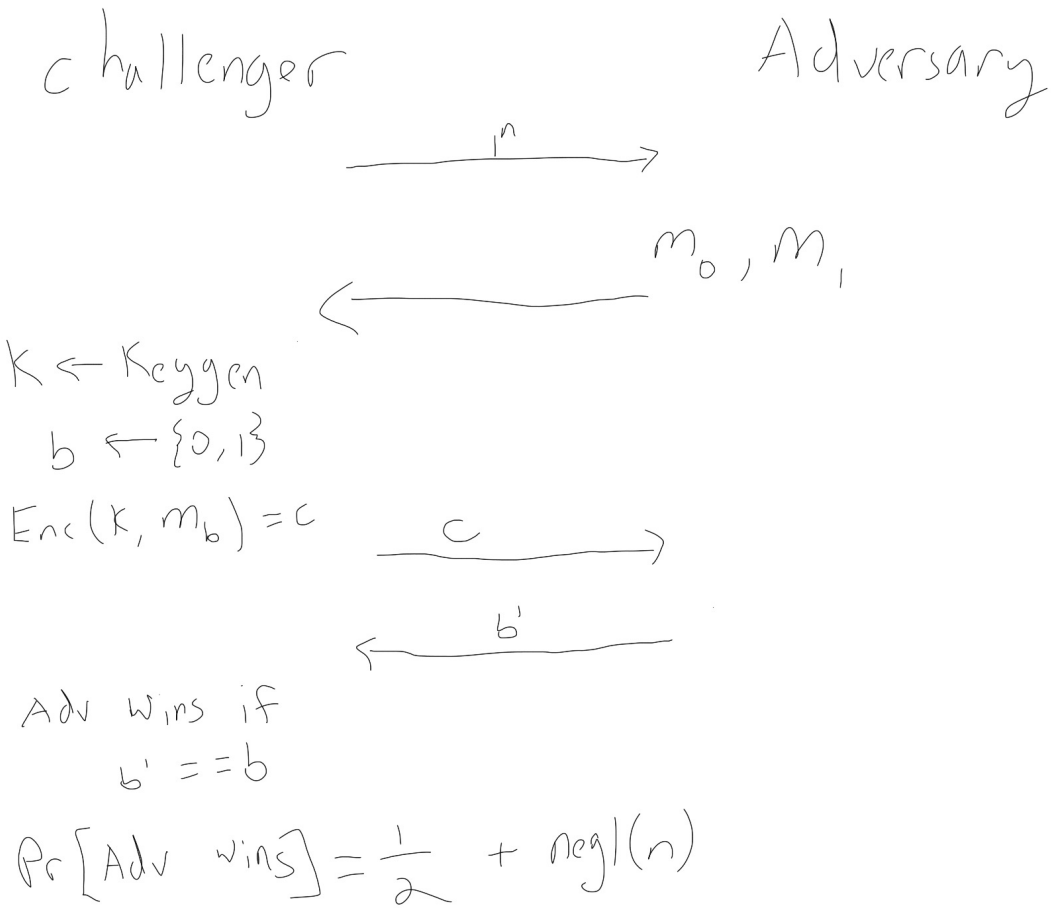
Consider the polynomial $\frac{q(x)}{p(x)}$. Since negl(x) is negligible, there exists $N \in \mathbb{N}$ such that $\forall n > N, \text{negl(n)} < \frac{\text{q(n)}}{\text{p(n)}}$

It follows that $\forall n > N, p(n) \cdot \text{negl(n)} = \text{f(n)} < \text{q(n)}$

Relaxing Security

We will relax security in 2 ways:

1. We will only claim security against *polynomial time* adversaries.
2. Even an efficient adversary might have some very (very very) small advantage in distinguishing an encryption of m_0 from an encryption of m_1.



Relaxing Security

We will relax security in 2 ways:

1. We will only claim security against *polynomial time* adversaries.
2. Even an efficient adversary might have some very (very very) small advantage in distinguishing an encryption of m_0 from an encryption of m_1.

These relaxations are both necessary:

1. Given ciphertext c, try decrypting c using every key $k \in \mathcal{K}$.
   Denote the resulting set by $M(c)$.
   Since $|\mathcal{K}| < |\mathcal{M}|$, there must be some $m \in \mathcal{M}$ such $m \notin M(c)$.
   We have learned something about the encrypted plaintext!

Relaxing Security

We will relax security in 2 ways:

1. We will only claim security against *polynomial time* adversaries.
2. Even an efficient adversary might have some very (very very) small advantage in distinguishing an encryption of m_0 from an encryption of m_1.

These relaxations are both necessary:

1. Given ciphertext c, try decrypting c using every key $k \in \mathcal{K}$.
   Denote the resulting set by $M(c)$.
   Since $|\mathcal{K}| < |\mathcal{M}|$, there must be some $m \in \mathcal{M}$ such $m \notin M(c)$.
   We have learned something about the encrypted plaintext!

   Consider the following known plaintext attack, where the adversary $\mathcal{A}$ knows that $\{c_1 = \mathrm{Enc}(k, m_1), \ldots, c_\ell = \mathrm{Enc}(k, m_\ell)\}$ (without knowing k)
   Try all k until you find one that is consistent with those ℓ values. Full key recovery!

Relaxing Security

We will relax security in 2 ways:

1. We will only claim security against *polynomial time* adversaries.
2. Even an efficient adversary might have some very (very very) small advantage in distinguishing an encryption of m_0 from an encryption of m_1.

These relaxations are both necessary:

1. Given ciphertext c, try decrypting c using every key $k \in \mathcal{K}$.
   Denote the resulting set by $M(c)$.
   Since $|\mathcal{K}| < |\mathcal{M}|$, there must be some $m \in \mathcal{M}$ such $m \notin M(c)$.
   We have learned something about the encrypted plaintext!

   Consider the following known plaintext attack, where the adversary $\mathcal{A}$ knows that $\{c_1 = \mathrm{Enc}(k, m_1), \ldots, c_\ell = \mathrm{Enc}(k, m_\ell)\}$ (without knowing k)
   Try all k until you find one that is consistent with those ℓ values. Full key recovery!

2. A polynomial-time adversary can try a few values of k. If she succeeds, then great, and if not, then she can guess b' at random.
   Small advantage, but non-zero.

Unary security parameters

Intuitively, if we double our key space, we should double our security.

Unary security parameters

Intuitively, if we double our key space, we should double our security.

If we add 1 bit to our key length, we double our key space.

Unary security parameters

Intuitively, if we double our key space, we should double our security.

If we add 1 bit to our key length, we double our key space.

Technicality: If we represent our key length (i.e. security parameter) in binary, then the input *length* grows *logarithmically* in the key length.
For example: input 10000000 indicates a 128-bit key.
input 11111111 indicates a 255-bit key.
The adversary has input length 7 in both cases!
It's runtime and its guessing advantage would both be measured by poly(7).

Unary security parameters

Intuitively, if we double our key space, we should double our security.

If we add 1 bit to our key length, we double our key space.

Technicality: If we represent our key length (i.e. security parameter) in binary, then the input *length* grows *logarithmically* in the key length.
For example: input 10000000 indicates a 128-bit key.
input 11111111 indicates a 255-bit key.
The adversary has input length 7 in both cases!
It's runtime and its guessing advantage would both be measured by poly(7).

We specify our security parameter in unary.
$\text{Gen}(1^n)$ outputs an n-bit key.
Each increase in n allows the adversary's runtime to grow. It still doubles the keyspace, ensuring that its advantage diminishes.

Privacy against eavesdroppers

Indistinguishability in the presence of an eavesdropper:

$\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n)$:

1. $\mathcal{A}$ is given 1^n and outputs m_0 and m_1 such that $|m_0| = |m_1|$.
2. $k \leftarrow \mathit{Gen}(1^n)$, $b \leftarrow \{0,1\}$, and $c \leftarrow \mathit{Enc}(k, m_b)$.
   Then c is given to $\mathcal{A}$.
3. $\mathcal{A}$ outputs a bit b'.
4. The outcome of the experiment is 1 if $b = b'$ and 0 otherwise.

Definition

A private-key encryption scheme $\Pi = (\mathit{Gen}, \mathit{Enc}, \mathit{Dec})$ has indistinguishable encryptions in the presence of an eavesdropper, or is EAV-secure, if for all probabilistic polynomial-time adversaries $\mathcal{A}$, there is a negligible function negl such that, for all n,

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