



Index of Security Definitions

One-time uniform ciphertexts for symmetric-key encryption (Definition 2.5):

$\mathcal{L}_{\text{ots-real}}^\Sigma$
$\text{CTXT}(m \in \Sigma.\mathcal{M}):$ $k \leftarrow \Sigma.\text{KeyGen}$ $c \leftarrow \Sigma.\text{Enc}(k, m)$ return c

$\mathcal{L}_{\text{ots-rand}}^\Sigma$
$\text{CTXT}(m \in \Sigma.\mathcal{M}):$ $c \leftarrow \Sigma.C$ return c

One-time secrecy for symmetric-key encryption (Definition 2.6):

$\mathcal{L}_{\text{ots-L}}^\Sigma$
$\text{EAVESDROP}(m_L, m_R \in \Sigma.\mathcal{M}):$ $k \leftarrow \Sigma.\text{KeyGen}$ $c \leftarrow \Sigma.\text{Enc}(k, m_L)$ return c

$\mathcal{L}_{\text{ots-R}}^\Sigma$
$\text{EAVESDROP}(m_L, m_R \in \Sigma.\mathcal{M}):$ $k \leftarrow \Sigma.\text{KeyGen}$ $c \leftarrow \Sigma.\text{Enc}(k, m_R)$ return c

t -out-of- n secret sharing (Definition 3.3):

$\mathcal{L}_{\text{tsss-L}}^\Sigma$
$\text{SHARE}(m_L, m_R \in \Sigma.\mathcal{M}, U):$ if $ U \geq \Sigma.t$: return err $s \leftarrow \Sigma.\text{Share}(m_L)$ return $\{s_i \mid i \in U\}$

$\mathcal{L}_{\text{tsss-R}}^\Sigma$
$\text{SHARE}(m_L, m_R \in \Sigma.\mathcal{M}, U):$ if $ U \geq \Sigma.t$: return err $s \leftarrow \Sigma.\text{Share}(m_R)$ return $\{s_i \mid i \in U\}$

Pseudorandom generator (Definition 5.1):

$\mathcal{L}_{\text{prg-real}}^G$
$\text{QUERY}():$ $s \leftarrow \{0, 1\}^\lambda$ return $G(s)$

$\mathcal{L}_{\text{prg-rand}}^G$
$\text{QUERY}():$ $r \leftarrow \{0, 1\}^{\lambda+\ell}$ return r

Pseudorandom function (Definition 6.1):

$\mathcal{L}_{\text{prf-real}}^F$
$k \leftarrow \{0, 1\}^\lambda$ $\text{LOOKUP}(x \in \{0, 1\}^{\text{in}}):$ return $F(k, x)$

$\mathcal{L}_{\text{prf-rand}}^F$
$T := \text{empty assoc. array}$ $\text{LOOKUP}(x \in \{0, 1\}^{\text{in}}):$ if $T[x]$ undefined: $T[x] \leftarrow \{0, 1\}^{\text{out}}$ return $T[x]$

Pseudorandom permutation (Definition 6.6):

$\mathcal{L}_{\text{prp-real}}^F$
$k \leftarrow \{0, 1\}^\lambda$
$\text{LOOKUP}(x \in \{0, 1\}^{\text{blen}}):$ return $F(k, x)$

$\mathcal{L}_{\text{prp-rand}}^F$
$T := \text{empty assoc. array}$
$\text{LOOKUP}(x \in \{0, 1\}^{\text{blen}}):$ if $T[x]$ undefined: $T[x] \leftarrow \{0, 1\}^{\text{blen}} \setminus T.\text{values}$ return $T[x]$

Strong pseudorandom permutation (Definition 6.13):

$\mathcal{L}_{\text{sprp-real}}^F$
$k \leftarrow \{0, 1\}^\lambda$
$\text{LOOKUP}(x \in \{0, 1\}^{\text{blen}}):$ return $F(k, x)$
$\text{INVLOOKUP}(y \in \{0, 1\}^{\text{blen}}):$ return $F^{-1}(k, y)$

$\mathcal{L}_{\text{sprp-rand}}^F$
$T, T_{\text{inv}} := \text{empty assoc. arrays}$
$\text{LOOKUP}(x \in \{0, 1\}^{\text{blen}}):$ if $T[x]$ undefined: $y \leftarrow \{0, 1\}^{\text{blen}} \setminus T.\text{values}$ $T[x] := y; T_{\text{inv}}[y] := x$ return $T[x]$
$\text{INVLOOKUP}(y \in \{0, 1\}^{\text{blen}}):$ if $T_{\text{inv}}[y]$ undefined: $x \leftarrow \{0, 1\}^{\text{blen}} \setminus T_{\text{inv}}.\text{values}$ $T_{\text{inv}}[y] := x; T[x] := y$ return $T_{\text{inv}}[y]$

CPA security for symmetric-key encryption (Definition 7.1, Section 8.2):

$\mathcal{L}_{\text{cpa-L}}^\Sigma$
$k \leftarrow \Sigma.\text{KeyGen}$
$\text{EAVESDROP}(m_L, m_R \in \Sigma.\mathcal{M}):$ if $ m_L \neq m_R $ return err $c := \Sigma.\text{Enc}(k, m_L)$ return c

$\mathcal{L}_{\text{cpa-R}}^\Sigma$
$k \leftarrow \Sigma.\text{KeyGen}$
$\text{EAVESDROP}(m_L, m_R \in \Sigma.\mathcal{M}):$ if $ m_L \neq m_R $ return err $c := \Sigma.\text{Enc}(k, m_R)$ return c

CPA\$ security for symmetric-key encryption (Definition 7.2, Section 8.2):

$\mathcal{L}_{\text{cpa\$-real}}^\Sigma$
$k \leftarrow \Sigma.\text{KeyGen}$
$\text{CHALLENGE}(m \in \Sigma.\mathcal{M}):$ $c := \Sigma.\text{Enc}(k, m)$ return c

$\mathcal{L}_{\text{cpa\$-rand}}^\Sigma$
$\text{CHALLENGE}(m \in \Sigma.\mathcal{M}):$ $c \leftarrow \Sigma.\mathcal{C}(m)$ return c

CCA security for symmetric-key encryption (Definition 9.1):

$\mathcal{L}_{\text{cca-L}}^\Sigma$	$\mathcal{L}_{\text{cca-R}}^\Sigma$
$k \leftarrow \Sigma.\text{KeyGen}$	$k \leftarrow \Sigma.\text{KeyGen}$
$\mathcal{S} := \emptyset$	$\mathcal{S} := \emptyset$
<u>EAVESDROP($m_L, m_R \in \Sigma.\mathcal{M}$):</u>	<u>EAVESDROP($m_L, m_R \in \Sigma.\mathcal{M}$):</u>
if $ m_L \neq m_R $ return err	if $ m_L \neq m_R $ return err
$c := \Sigma.\text{Enc}(k, m_L)$	$c := \Sigma.\text{Enc}(k, m_R)$
$\mathcal{S} := \mathcal{S} \cup \{c\}$	$\mathcal{S} := \mathcal{S} \cup \{c\}$
return c	return c
<u>DECRYPT($c \in \Sigma.\mathcal{C}$):</u>	<u>DECRYPT($c \in \Sigma.\mathcal{C}$):</u>
if $c \in \mathcal{S}$ return err	if $c \in \mathcal{S}$ return err
return $\Sigma.\text{Dec}(k, c)$	return $\Sigma.\text{Dec}(k, c)$

CCA\$ security for symmetric-key encryption (Definition 9.2):

$\mathcal{L}_{\text{cca\$-real}}^\Sigma$	$\mathcal{L}_{\text{cca\$-rand}}^\Sigma$
$k \leftarrow \Sigma.\text{KeyGen}$	$k \leftarrow \Sigma.\text{KeyGen}$
$\mathcal{S} := \emptyset$	$\mathcal{S} := \emptyset$
<u>CTXT($m \in \Sigma.\mathcal{M}$):</u>	<u>CTXT($m \in \Sigma.\mathcal{M}$):</u>
$c := \Sigma.\text{Enc}(k, m)$	$c \leftarrow \Sigma.\mathcal{C}(m)$
$\mathcal{S} := \mathcal{S} \cup \{c\}$	$\mathcal{S} := \mathcal{S} \cup \{c\}$
return c	return c
<u>DECRYPT($c \in \Sigma.\mathcal{C}$):</u>	<u>DECRYPT($c \in \Sigma.\mathcal{C}$):</u>
if $c \in \mathcal{S}$ return err	if $c \in \mathcal{S}$ return err
return $\Sigma.\text{Dec}(k, c)$	return $\Sigma.\text{Dec}(k, c)$

MAC (Definition 10.2):

$\mathcal{L}_{\text{mac-real}}^\Sigma$	$\mathcal{L}_{\text{mac-fake}}^\Sigma$
$k \leftarrow \Sigma.\text{KeyGen}$	$k \leftarrow \Sigma.\text{KeyGen}$
	$\mathcal{T} := \emptyset$
<u>GETTAG($m \in \Sigma.\mathcal{M}$):</u>	<u>GETTAG($m \in \Sigma.\mathcal{M}$):</u>
return $\Sigma.\text{MAC}(k, m)$	$t := \Sigma.\text{MAC}(k, m)$
	$\mathcal{T} := \mathcal{T} \cup \{(m, t)\}$
<u>CHECKTAG($m \in \Sigma.\mathcal{M}, t$):</u>	return t
return $t \stackrel{?}{=} \Sigma.\text{MAC}(k, m)$	<u>CHECKTAG($m \in \Sigma.\mathcal{M}, t$):</u>
	return $(m, t) \stackrel{?}{\in} \mathcal{T}$

Collision resistance (Definition 11.1):

$\mathcal{L}_{\text{cr-real}}^{\mathcal{H}}$
$s \leftarrow \{0, 1\}^\lambda$
<u>GETSALT():</u> return s
<u>TEST($x, x' \in \{0, 1\}^*$):</u> if $x \neq x'$ and $H(s, x) = H(s, x')$: return true return false

$\mathcal{L}_{\text{cr-fake}}^{\mathcal{H}}$
$s \leftarrow \{0, 1\}^\lambda$
<u>GETSALT():</u> return s
<u>TEST($x, x' \in \{0, 1\}^*$):</u> return false

Digital signatures (Definition 13.6):

$\mathcal{L}_{\text{sig-real}}^{\Sigma}$
$(vk, sk) \leftarrow \Sigma.\text{KeyGen}$
<u>GETVK():</u> return vk
<u>GETSIG(m):</u> return $\Sigma.\text{Sign}(sk, m)$
<u>VERSIG(m, σ):</u> return $\Sigma.\text{Ver}(vk, m, \sigma)$

$\mathcal{L}_{\text{sig-fake}}^{\Sigma}$
$(vk, sk) \leftarrow \Sigma.\text{KeyGen}$ $\mathcal{S} := \emptyset$
<u>GETVK():</u> return vk
<u>GETSIG(m):</u> $\sigma := \Sigma.\text{Sign}(sk, m)$ $\mathcal{S} := \mathcal{S} \cup \{(m, \sigma)\}$ return σ
<u>VERSIG(m, σ):</u> return $(m, \sigma) \in \mathcal{S}$

Key agreement (Definition 14.4):

$\mathcal{L}_{\text{ka-real}}^{\Sigma}$
<u>QUERY():</u> $(t, K) \leftarrow \text{EXECROT}(\Sigma)$ return (t, K)

$\mathcal{L}_{\text{ka-rand}}^{\Sigma}$
<u>QUERY():</u> $(t, K) \leftarrow \text{EXECROT}(\Sigma)$ $K' \leftarrow \Sigma.\mathcal{K}$ return (t, K')

Decisional Diffie-Hellman assumption (Definition 14.5):

$\mathcal{L}_{\text{dh-real}}^{\mathbb{G}}$
<u>QUERY():</u> $a, b \leftarrow \mathbb{Z}_n$ return (g^a, g^b, g^{ab})

$\mathcal{L}_{\text{dh-rand}}^{\mathbb{G}}$
<u>QUERY():</u> $a, b, c \leftarrow \mathbb{Z}_n$ return (g^a, g^b, g^c)

CPA security for public-key encryption (Definition 15.1):

$\mathcal{L}_{\text{pk-cpa-L}}^{\Sigma}$	$\mathcal{L}_{\text{pk-cpa-R}}^{\Sigma}$
$(pk, sk) \leftarrow \Sigma.\text{KeyGen}$	$(pk, sk) \leftarrow \Sigma.\text{KeyGen}$
<u>GETPK():</u> return pk	<u>GETPK():</u> return pk
<u>CHALLENGE($m_L, m_R \in \Sigma.\mathcal{M}$):</u> return $\Sigma.\text{Enc}(pk, m_L)$	<u>CHALLENGE($m_L, m_R \in \Sigma.\mathcal{M}$):</u> return $\Sigma.\text{Enc}(pk, m_R)$

CPA\$ security for public-key encryption (Definition 15.2):

$\mathcal{L}_{\text{pk-cpa\$-real}}^{\Sigma}$	$\mathcal{L}_{\text{pk-cpa\$-rand}}^{\Sigma}$
$(pk, sk) \leftarrow \Sigma.\text{KeyGen}$	$(pk, sk) \leftarrow \Sigma.\text{KeyGen}$
<u>GETPK():</u> return pk	<u>GETPK():</u> return pk
<u>CHALLENGE($m \in \Sigma.\mathcal{M}$):</u> return $\Sigma.\text{Enc}(pk, m)$	<u>CHALLENGE($m \in \Sigma.\mathcal{M}$):</u> $c \leftarrow \Sigma.\mathcal{C}$ return c

One-time secrecy for public-key encryption (Definition 15.4):

$\mathcal{L}_{\text{pk-ots-L}}^{\Sigma}$	$\mathcal{L}_{\text{pk-ots-R}}^{\Sigma}$
$(pk, sk) \leftarrow \Sigma.\text{KeyGen}$ $count := 0$	$(pk, sk) \leftarrow \Sigma.\text{KeyGen}$ $count := 0$
<u>GETPK():</u> return pk	<u>GETPK():</u> return pk
<u>CHALLENGE($m_L, m_R \in \Sigma.\mathcal{M}$):</u> $count := count + 1$ if $count > 1$: return null return $\Sigma.\text{Enc}(pk, m_L)$	<u>CHALLENGE($m_L, m_R \in \Sigma.\mathcal{M}$):</u> $count := count + 1$ if $count > 1$: return null return $\Sigma.\text{Enc}(pk, m_R)$