Definition: $\mathsf{w-Factor}_{\mathcal{A}}$ (The Weak Factoring Experiment):Choose two uniform n-bit integers, x_1, x_2. Let $N := x_1 \cdot x_2$. Given N, $\mathcal{A}$ outputs $x_1', x_2' > 1$. $\mathcal{A}$ succeeds iff $x_1' \cdot x_2' = N$. (Need not equal x_1, x_2.)

Theorem: Prime Number Theorem: The number of primes less than x is $\pi(x) \approx \frac{x}{\log(x)}$

Theorem: Bertrand's Postulate: For any $n > 1$, at least $\frac{1}{3n}$ of the n-bit integers are prime.

Definition: Miller-Rabin Test: An algorithm which takes 2 inputs, integer p and parameter t, and attempts to determine if p is prime. The Miller-Rabin test runs in polynomial time in $||p||$ and t.

Theorem: If p is prime, the Miller-Rabin test always outputs "prime". Else, it outputs composite, except with probability at most 2^{-t}.

Algorithm: Random Prime: Given a length n, run the following $3n^2$ times: $p' \leftarrow \{0,1\}^{n-1}$, $p = 1||p$, run Miller-Rabin on p, parameter $t = 1^n$. Return p if MR says prime. Else, retry. If all attempts fail, return fail.

Theorem: Let $\mathbb{G}$ be a finite group, and $\mathbb{H} \subseteq \mathbb{G}$. Assume $\mathbb{H}$ is nonempty, and for all $a, b \in \mathbb{H}$, $ab \in \mathbb{H}$. Then $\mathbb{H}$ is a subgroup of $\mathbb{G}$.

Theorem: Let $\mathbb{H}$ be a strict subgroup of a finite group $\mathbb{G}$ (Strict: $\mathbb{H} \neq \mathbb{G}$.). Then $|\mathbb{H}| \leq |\mathbb{G}|/2$.

Definition: GenModulus: A polynomial-time algorithm, given 1^n, outputs (N, p, q), $N = pq$, $p, q = n$-bit primes except with probability negligible in n.

Definition: $\mathsf{Factor}_{\mathcal{A},\mathsf{GenModulus}(n)}(n)$: (The Factoring Experiment): Run $\mathsf{GenModulus}(1^n)$ to obtain (N, p, q). Give N to $\mathcal{A}$, get $p', q' > 1$. Output of the experiment is 1 iff $p' \cdot q' = N$. Note: Unless GenModulus fails, then success means $\{p', q'\} = \{p, q\}$.

Definition: Factoring Assumption: Factoring is hard relative to GenModulus if for all PPT algorithms $\mathcal{A}$, $\Pr[\mathsf{Factor}_{\mathcal{A},\mathsf{GenModulus}(n)}(n) = 1] \leq \mathtt{negl}(n)$.

Definition: GenRSA: A PPT algorithm that, given 1^n, outputs a modulus N that is the product of 2 primes, and integers $e, d > 0$ wich $\gcd(e, \phi(N)) = 1$ and $ed = 1 \mod \phi(N)$.

Definition: $\mathsf{RSA-inv}_{\mathcal{A},\mathsf{GenRSA}(n)}(n)$: (The RSA Experiment): $(N, e, d) \leftarrow \mathsf{GenRSA}(1^n)$. Choose a uniform $y \in \mathbb{Z}_N^*$. Give N, e, y to $\mathcal{A}$, get $x \in \mathbb{Z}_N^*$. $\mathcal{A}$ succeeds iff $x^e = y \mod N$.

Definition: The RSA problem is hard relative to GenRSA if for all PPT algorithms $\mathcal{A}$, $\Pr[\mathsf{RSA-inv}_{\mathcal{A},\mathsf{GenRSA}(n)}(n) = 1] \leq \mathtt{negl}(n)$.

Definition: RSA Assumption: There exists as GenRSA algorithm relative to which the RSA problem is hard.

Algorithm: GenRSA: (Vanilla version): Given 1^n, $(N, p, q) \leftarrow \mathsf{GenModulus}(1^n)$. $\phi(N) = (p-1)(e-1)$. Choose e such that $\gcd(e, \phi(N)) = 1$. Compute $d := [e^{-1} \mod \phi(N)]$. Return N, e, d.

Theorem: There is a PPT algorithm that, given a composite N and e, d with $ed = 1 \mod \phi(N)$, outputs a factor of N except with probability negligible in $||N||$.

Definition: Order: Let $\mathbb{G}$ be a finite group, $g \in \mathbb{G}$. The order of g is the smallest positive integer i such that $g^i = 1$.

Theorem: Let $\mathbb{G}$ be a finite group, $g \in \mathbb{G}$ with order i. Then for any integer x, $g^x = g^{[x \mod i]}$.

Theorem: Let $\mathbb{G}$ be a finite group, $g \in \mathbb{G}$ with order i. Then $g^x = g^y$ iff $x = y \mod i$.

Definition: Generator: An element $g \in \mathbb{G}$, such that order(g) =order$(\mathbb{G})$. Notation: $\langle g \rangle = \mathbb{G}$.

Definition: Cyclic: A group $\mathbb{G}$ such that some $g \in \mathbb{G}$ is a generator of $\mathbb{G}$.

Theorem: Let $\mathbb{G}$ be a finite group order m, $g \in \mathbb{G}$ with order i. Then $i \mid m$.

Theorem: If $\mathbb{G}$ is a group of prime order p, then $\mathbb{G}$ is cyclic, and all elements except the identity are generators of $\mathbb{G}$.

Theorem: If p is prime, then $\mathbb{Z}_p^*$ is a cyclic group of order $p-1$.

Definition: $\mathcal{G}$: Group Generation Algorithm. On input 1^n, gives (description of) cyclic group $\mathbb{G}$, order q, and generator g.

Definition: $\mathsf{DLog}_{\mathcal{A},\mathcal{G}}(n)$: Run $\mathcal{G}(1^n)$ to obtain $(\mathbb{G}, q, g)$. Choose a uniform $h \in \mathbb{G}$. $\mathcal{A}$ is given G, q, g, h, and outputs $x \in \mathbb{Z}_q$. $\mathcal{A}$ succeeds iff $g^x = h$.

Definition: The Discrete-Logarithm Problem is hard relative to $\mathcal{G}$ if for all PPT algorithms $\mathcal{A}$, $\Pr[\mathsf{DLog}_{\mathcal{A},\mathcal{G}}(n) = 1] \leq \mathtt{negl}(n)$.

Definition: DDH Problem: The Decisional Diffie-Hellman Problem: It is hard relative to $\mathcal{G}$ if for all PPT algorithms $\mathcal{A}$, $|\Pr[\mathcal{A}(\mathbb{G}, q, g, g^x, g^y, g^z) = 1] - \Pr[\mathcal{A}(\mathbb{G}, q, g, g^x, g^y, g^{xy}) = 1]| \leq \mathtt{negl}(n)$.

Theorem: Let $p = rq + 1$, with p, q prime. Then $G \stackrel{\text{def}}{=} \{[h^n \mod p] \mid h \in \mathbb{Z}_p^*\}$ is a subgroup of $\mathbb{Z}_p^*$ of order q.

Algorithm: Group Generation: A basic $\mathcal{G}$. Input 1^n, $\ell = \ell(n)$. Generate a uniform n-bit prime q. Generate an ℓ-bit prime p such that $q \mid (p-1)$. Choose a uniform $h \in \mathbb{Z}_p^*$ with $h \neq 1$. Set $g := [h^{(p-1)/q} \mod p]$. Return p, q, g.

Construction: Let $\mathcal{G}$ be as above. Define a fixed-length hash function (Gen, H) as follows: Gen: Given 1^n, get $(\mathbb{G}, q, g) \leftarrow \mathcal{G}(1^n)$, select uniform $h \in \mathbb{G}$. Output $s := \langle \mathbb{G}, q, g, h \rangle$ as the key. H: Given s, and input $(x_1, x_2) \in \mathbb{Z}_q \times \mathbb{Z}_q$, output $H^s(x_1, x_2) := g^{x_1} h^{x_2} \in \mathbb{G}$.

Theorem: If the discrete-logarithm problem is hard relative to $\mathcal{G}$, the above construction is collision-resistant.

Theorem: If the discrete-logarithm problem is hard, then collision-resistant hash functions exist.

Definition: $\mathsf{KE}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n)$: The Key-Exchange Experiment: Two parties with 1^n execute protocol Π. This results in a transcript trans containing all the messages sent by the parties, and a key k output by each of the parties. A uniform bit $b \in \{0,1\}$ is chosen. If $b = 0$, set $\hat{k} := k$. Else, choose $\hat{k} \in \{0,1\}^n$ at random. $\mathcal{A}$ is given trans and $\hat{k}$, outputs b'. Success if and only if $b' = b$.

Definition: KE EAV secure: (Security in the presence of an eavesdropper): A key-exchange protocol such that for all PPT adversaries $\mathcal{A}$, $\Pr[\mathsf{KE}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathtt{negl}(n)$.

Construction: Diffie-Hellman: Alice gets $(\mathbb{G}, q, g) \leftarrow \mathcal{G}(1^n)$. She chooses a uniform $x \in \mathbb{Z}_q$, computes $h_A := g^x$. Alice sends $(\mathbb{G}, q, g, h_A)$ to Bob. Bob chooses a uniform $y \in \mathbb{Z}_q$, and computes $h_B := g^y$. Bob sends h_B to Alice and outputs the key $k_B := h_A^y$. Alice outputs the key $k_A := h_B^x$. Note: $k_A = k_B$.

Theorem: If the DDH problem is hard relative to $\mathcal{G}$, then the Diffie-Hellman key-exchange protocol Π is secure in the presence of an eavesdropper.

It is not, however, even remotely secure in the presence of an active attacker, who can simulate Bob, and intercept and decrypt all messages.

Definition: Public Key Cryptography: A party generates a pair of keys; the public key is widely distributed, and the private key is kept secret. The public key can encrypt messages to the party, which are only decryptable with the private key.

Definition: Signature: Some function of a message is computed (usually a hash), then encrypted using the private key. It is then sent with the message, as the signature can be decoded by anyone, but only produced by the person with the private key.

Definition: Non-repudiation: Verification that a document was indeed signed by the person the signature claims it is from.

Definition: Public-Key Encryption Scheme: A triple of probabilistic polynomial-time algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ such that: Gen: takes 1^n and outputs a par of keys (pk, sk), there pk is public key and sk is secret (or private) key. Assume both are length $\geq n$, and n can be determined from them. Enc: takes public key pk and message m, and outputs $c \leftarrow \mathsf{Enc}_{pk}(m)$. Dec: Deterministic, takes secret key sk and c, and $m := \mathsf{Dec}_{sk}(c)$. Outputs $\perp$ on failure.

Definition: $\mathsf{PubK}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n)$: $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$. $\mathcal{A}$ is given pk, outputs $m_0, m_1 \in \mathcal{M}$, with $|m_0| = |m_1|$. $b \in \{0,1\}$ is chosen uniformly, and $c \leftarrow \mathsf{Enc}_{pk}(m_b)$ is computed and returned to $\mathcal{A}$ as the challenge ciphertext. $\mathcal{A}$ outputs b'. Success iff $b' = b$.

Definition: Public Key EAV Security: A public-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ such that for all PPT adversaries $\mathcal{A}$, $\Pr[\mathsf{PubK}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathtt{negl}(n)$.

Theorem: If a public-key encryption scheme has indistinguishable encryptions in the presences of an eaves-dropper (is EAV-secure), then it is CPA-secure.

Theorem: No deterministic public-key encryption scheme is CPA-secure.

Definition: LR: Left-or-right oracle, on input a pair of equal-length messages m_0, m_1, outputs $c \leftarrow \mathsf{LR}_{pk,b}(m_0, m_1) := \mathsf{Enc}_{pk}(m_b)$.

Definition: $\mathsf{PubK}^{\mathsf{LR-cpa}}_{\mathcal{A},\Pi}(n)$: $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$. $b \in \{0,1\}$ is chosen uniformly. $\mathcal{A}$ is given pk and oracle access to $\mathsf{LR}_{pk,b}(\cdot,\cdot)$. $\mathcal{A}$ outputs a bit b'. Success iff $b' = b$.

Definition: Pub LR-cpa secure: A public-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ has indistinguishable multiple encryptions if for all PPT adversaries $\mathcal{A}$, $\Pr[\mathsf{PubK}^{\mathsf{LR-cpa}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathtt{negl}(n)$.

Theorem: If a public-key encryption scheme is CPA-secure, then it also has indistinguishable multiple encryptions.

Theorem: Given $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a fixed-length 1-bit encryption scheme. Define $\Pi = (\mathsf{Gen}, \mathsf{Enc}', \mathsf{Dec}')$ for arbitrary length messages where $\mathsf{Enc}'_{pk}(m) = \mathsf{Enc}_{pk}(m_1), \ldots, \mathsf{Enc}_{pk}(m_\ell)$, where $m = m_1 \cdots m_\ell$, and Dec' is constructed similarly. Then if Π is CPA-secure, so is Π'.

Definition: $\mathsf{PubK}^{\mathsf{CCA}}_{\mathcal{A},\Pi}(n)$: $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$. $\mathcal{A}$ is given access to pk and decryption oracle $\mathsf{Dec}_{sk}(\cdot)$. Outputs m_1, m_2, of same length. $b \in \{0,1\}$ is chosen uniformly. $c \leftarrow \mathsf{Enc}_{pk}(m_b)$ is given to $\mathcal{A}$. $\mathcal{A}$ can call $\mathsf{Dec}_{sk}(\cdot)$ again, but not with c. $\mathcal{A}$ outputs b'. Success iff $b' = b$.

Definition: Pub CCA-secure: A public-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ such that for all PPT adversaries $\mathcal{A}$, $\Pr[\mathsf{PubK}^{\mathsf{CCA}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathtt{negl}(n)$.

Definition: KEM (Key Encapsulation Mechanism): A tuple of PPT algorithms $(\mathsf{Gen}, \mathsf{Encaps}, \mathsf{Decaps})$ such that: Gen takes 1^n and outputs a public/private key pair, (pk, sk). Assume pk, sk have length $\geq n$, and n can be determined from them. Encaps takes pk and 1^n. Outputs c and $k \in \{0,1\}^{\ell(n)}$, where ℓ is the key length. Decaps takes sk and c and outputs a key k. If failure, outputs $\perp$.

Construction: Let $\Pi = (\mathsf{Gen}, \mathsf{Encaps}, \mathsf{Decaps})$ be a KEM with key length n, and let $\Pi = (\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ be a private-key encryption scheme. Construct public key encryption scheme $\Pi^{\mathsf{hy}} = (\mathsf{Gen}^{\mathsf{hy}}, \mathsf{Enc}^{\mathsf{hy}}, \mathsf{Dec}^{\mathsf{hy}})$ as follows: $\mathsf{Gen}^{\mathsf{hy}}$ On input 1^n, run $\mathsf{Gen}(1^n)$ and use (pk, sk) from that. $\mathsf{Enc}^{\mathsf{hy}}$: On input pk and $m \in \{0,1\}^*$, $(c, k) \leftarrow \mathsf{Encaps}_{pk}(1^n)$, $c' \leftarrow\leftarrow \mathsf{Enc}'_k(m)$, output $\langle c, c' \rangle$. $\mathsf{Dec}^{\mathsf{hy}}$ On input $sk, \langle c, c' \rangle$, compute $k := \mathsf{Decaps}_{sk}(c)$, output $m := \mathsf{Dec}'_k(c')$.

Definition: $\mathsf{KEM}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n)$: $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$. $(c, k) \leftarrow \mathsf{Encaps}_{pk}(1^n)$. $b \in \{0,1\}$ chosen uniformly. If $b = 0$, $\hat{k} := k$, else $\hat{k} \in \{0,1\}$ uniformly at random. Give $(pk, c, \hat{k})$ to $\mathcal{A}$, which outputs b'. Success iff $b' = b$.

Definition: CPA-Secure KEM: A key-encapsulation mechanism Π such that for all PPT adversaries $\mathcal{A}$, $\Pr[\mathsf{KEM}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathtt{negl}(n)$.

Theorem: If Π is a CPA-secure KEM and Π' is a private-key encryption scheme that has indistinguishable encryptions in the presence of an eavesdropped, then Π^{hy} as above is a CPA-secure public key encryption scheme.

Definition: $\mathsf{KEM}^{\mathsf{cca}}_{\mathcal{A},\Pi}(n)$: $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$. $(c, k) \leftarrow \mathsf{Encaps}_{pk}(1^n)$. $b \in \{0,1\}$ chosen uniformly. If $b = 0$, $\hat{k} := k$, else $\hat{k} \in \{0,1\}^n$ uniformly at random. Give $(pk, c, \hat{k})$ to $\mathcal{A}$. $\mathcal{A}$ can access $\mathsf{Decaps}_{sk}(\cdot)$ oracle, but not on c itself. $\mathcal{A}$ outputs b'. Success iff $b' = b$.

Definition: KEM CCA-Secure: A key-encapsulation mechanism Π such that for all PPT adversaries $\mathcal{A}$, $\Pr[\mathsf{KEM}^{\mathsf{cca}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathtt{negl}(n)$.

Theorem: If Π is a CCA-secure KEM and Π' is a CCA-secure private-key encryption scheme, then Π^{hy} as above is a CCA-secure public-key encryption scheme.

Theorem: Let $\mathbb{G}$ be a finite group and let $m \in \mathbb{G}$ be arbitrary. Choose a uniform $k \in \mathbb{G}$. For any $\hat{g} \in \mathbb{G}$, $\Pr[k \cdot m = \hat{g}] = 1/|\mathbb{G}|$.

Construction: El Gamal: Let $\mathcal{G}$ be as in the Group Generation algorithm. Define a public-key encryption scheme as follows: Gen: On input 1^n, $(\mathbb{G}, q, g) \leftarrow \mathcal{G}(1^n)$. Then choose a uniform $x \in \mathbb{Z}_q$ and compute $h := g^x$. The public key is $\langle \mathbb{G}, q, g, h \rangle$ and the secret (private) key is $\langle \mathbb{G}, q, g, x \rangle$. The message space is $\mathbb{G}$. Enc: Given pk, m, choose a uniform $y \in \mathbb{Z}_q$ and output the ciphertext $\langle g^y, h^y \cdot m \rangle$. Dec: Given $sk, c = \langle c_1, c_2 \rangle$, output $\hat{m} := c_2 / c_1^x$.

Theorem: If the DDH problem is hard relative to $\mathcal{G}$, then the El Gamal encryption scheme is CPA-secure.

Construction: Let $\mathcal{G}$ be as in the Group Generation algorithm. Define a KEM as follows: Gen: On input 1^n, $(\mathbb{G}, q, g) \leftarrow \mathcal{G}(1^n)$. Choose a uniform $x \in \mathbb{Z}_q$, set $h := g^x$. Specify a function $H : \mathbb{G} \to \{0,1\}^{\ell(n)}$. for some function ℓ. $pk = \langle \mathbb{G}, q, g, h, H \rangle$, $sk = \langle \mathbb{G}, q, g, x \rangle$. Encaps: On input pk, choose uniform $y \in \mathbb{Z}_q$, and output ciphertext g^y and key $H(h^y)$. Decaps: On input $sk, c \in \mathbb{G}$, output key $H(c^x)$.

Theorem: If the DDH problem is hard relative to $\mathcal{G}$, and H is as specified in the text, then the above construction is a CPA-secure KEM.

Theorem: If the CDH problem is hard relative to $\mathcal{G}$, and H is modeled as a random oracle, then the above construction is CPA-secure.

Theorem: If the gap-CDH problem is hard relative to $\mathcal{G}$, and H is modeled as a random oracle, then the above construction is a CCA-secure KEM.

Construction: DHIES/ECIES: Let $\mathcal{G}$ be as in the Group Generation algorithm. Let $\Pi_E = (\mathsf{Enc}', \mathsf{Dec}')$ be a private-key encryption scheme, and let $\Pi_M =$

$(\mathsf{Mac}, \mathsf{Vrfy})$ be a message authentication code. Define a public-key encryption scheme as follows: Gen: On input 1^n, $(\mathbb{G}, q, g) \leftarrow \mathcal{G}(1^n)$. Choose a uniform $x \in \mathbb{Z}_q$, set $h := g^x$, and specify $H : \mathbb{G} \to \{0,1\}^{2n}$. $pk = \langle \mathbb{G}, q, g, h, H \rangle$, $sk = \langle \mathbb{G}, q, g, x, H \rangle$. Enc: On input pk, message m, choose a uniform $y \in \mathbb{Z}_q$, set $k_E || k_M := H(h^y)$. Compute $c' \leftarrow \mathsf{Enc}'_{k_E}(m)$, output ciphertext $\langle g^y, c', \mathsf{Mac}_{k_M}(c') \rangle$. Dec: On input sk, and ciphertext $\langle c, c', t \rangle$, output $\perp$ if $c \notin \mathbb{G}$. Else, compute $k_E || k_M := H(c^x)$. If $\mathsf{Vrfy}_{k_M}(c', t) \neq 1$ output $\perp$. Else, output $\mathsf{Dec}'_{k_E}(c')$.

Theorem: Let Π_E be a CPA-secure private-key encryption scheme, and let Π_M be a strongly secure MAC. If the gap-CDH problem is hard relative to $\mathcal{G}$, and h is modeled as a random oracle, then the above construction is a CCA-secure public-key encryption scheme.

Algorithm: GenRSA: Input 1^n. $(N, p, q) \leftarrow \mathsf{GenModulus}(1^n)$. $\phi(N) = (p-1)(q-1)$. Choose $e > 1$ such that $\gcd(e, \phi(N)) = 1$. Compute $d := [e^{-1} \mod \phi(N)]$. Return N, e, d.

Construction: Plain RSA: Let GenRSA be as above. Define a public-key encryption scheme as follows: Gen: On input 1^n, $(N, e, d) \leftarrow \mathsf{GenRSA}(1^n)$. $pk = \langle N, e \rangle$, $sk = \langle N, d \rangle$. Enc: On input pk, message $m \in \mathbb{Z}_N^*$, output ciphertext $c := [m^e \mod N]$. Dec: On input sk, $c \in \mathbb{Z}_N^*$, compute message $m : [c^d \mod N]$.

Algorithm: RSA FAIL: Input $pk = \langle N, e \rangle$, ciphertext c. Set $T := 2^{\alpha n}$, where $\alpha \in (\frac{1}{2}, 1)$ is some fixed constant, and n is the security parameter. For integer $r \in [1, T]$, $x_r := [c/r^e \mod N]$. Sort the pairs $\{(r, x_r)\}_{r=1}^{T}$ by their second component. For $s = 1$ to T: If $x_r \stackrel{?}{=} [s^e \mod N]$ for some r, return $[r \cdot s \mod N]$.

Theorem: Let $p(x)$ be a polynomial of degree e. Then in time $\mathtt{poly}(||N||, e)$ one can find all m such that $p(m) = 0 \mod N$ and $|m| \leq N^{1/e}$.

Construction: Padded RSA: Let GenRSA be as above, and let ℓ be a function with $\ell(n) \leq 2n - 4$ for all n. Define a public-key encryption scheme as follows: Gen: On input 1^n, $(N, e, d) \leftarrow \mathsf{GenRSA}(1^n)$. Output $pk = \langle N, e \rangle$, $sk = \langle N, d \rangle$. Enc On input pk and $m \in \{0,1\}^{||N|| - \ell(n) - 2}$, choose a uniform string $r \in \{0,1\}^{\ell(n)}$ and interpret $\hat{m}$ as an element of $\mathbb{Z}_N^*$. Output the ciphertext $c := [\hat{m}^e \mod N]$. Dec: On input sk, and ciphertext c, compute $\hat{m} := [c^d \mod N]$, and output the $||N|| - \ell(n) - 2$ least-significant bits of $\hat{m}$.

Definition: lsb: Least Significant Bit.

Definition: $\mathsf{RSA\text{-}lsb}_{\mathcal{A}, \mathsf{GenRSA}}(1^n)$: (The RSA hard-core predicate experiment): $(N, e, d) \leftarrow \mathsf{GenRSA}(1^n)$. Choose a uniform $x \in \mathbb{Z}_N^*$ ad compute $y := [x^e \mod N]$. $\mathcal{A}$ is given N, e, y and outputs a bit b. The output of the experiment is 1 iff $\mathtt{lsb}(x) = b$.

Theorem: If the RSA problem is hard relative to GenRSA, then for all PPT algorithms $\mathcal{A}$, $\Pr[\mathsf{RSA\text{-}lsb}_{\mathcal{A}, \mathsf{GenRSA}}(n) = 1] \leq \frac{1}{2} + \mathtt{negl}(n)$.

Construction: Let GenRSA be as usual, and define a public-key encryption scheme as follows: Gen: On input 1^n, $(N, e, d) \leftarrow \mathsf{GenRSA}(1^n)$. Output $pk = \langle N, e \rangle$, $sk = \langle N, d \rangle$. Enc: On input pk, $m \in \{0,1\}$, choose a uniform $r \in \mathbb{Z}_N^*$ subject to the constraint that $\mathtt{lsb}(r) = m$. Output ciphertext $c := [r^e \mod N]$. Dec: On input sk and ciphertext c, compute $r := [c^d \mod N]$ and output $\mathtt{lsb}(r)$.

Theorem: If the RSA problem is hard relative to GenRSA, then the above construction is CPA-secure.

Construction: Let GenRSA be as usual, and define a KEM as follows: Gen: On input 1^n, $(N, e, d) \leftarrow \mathsf{GenRSA}(1^n)$. Comute $d' = [d^n \mod \phi(N)]$. Output $pk = \langle N, e \rangle$, and $sk = \langle N, d' \rangle$. Encaps: On input pk and 1^n, choose a uniform $c_1 \in \mathbb{Z}_N^*$. Then for $i = 1, \ldots, n$, compute $k_i = \mathtt{lsb}(c_i)$ and $c_{i+1} := [c_i^e \mod N]$. Output ciphertext c_{n+1} and key $k = k_1 \cdots k_n$. Decaps On input sk and ciphertext c, compute $c_1 := [c^{d'} \mod N]$. Then for $i = 1, \ldots, n$, compute $k_i := \mathtt{lsb}(c_i)$, and $c_{i+1} := c_i^e \mod N$]. Output the key $k = k_1 \ldots k_n$.

Theorem: If the RSA problem is hard relative to GenRSA, then the above construction is a CPA-secure KEM.

Construction: RSA-OAEP: Let GenRSA be as usual, and $\ell = \ell(n)$, $k_0 = k_0(n)$, and $k_1 = k_1(n)$ be integer valued functions with $k_0(n), k_1(n) = \Phi(n)$, and such that $\ell(n) + k_0(n) + k_1(n)$ is less than the minimum bit-length of moduli output by $\mathsf{GenRSA}(1^n)$. Let $G : \{0,1\}^{k_0} \to \{0,1\}^{\ell + k_1}$ and $H : \{0,1\}^{\ell + k_1} \to \{0,1\}^{k_0}$ be functions. Construct a public-key encryption scheme as follows: Gen On input 1^n, $(N, e, d) \leftarrow \mathsf{GenRSA}(1^n)$. $pk = \langle N, e \rangle$, $sk = \langle N, d \rangle$. Enc: On input pk and message $m \in \{0,1\}^{\ell}$, set $m' := m || 0^{k_1}$ and choose a uniform $r \in \{0,1\}_0^k$. Then compute $s := m' \oplus G(r)$ and $t := r \oplus H(s)$. and set $\hat{m} := s || t$. Output the ciphertext $c := [\hat{m}^e \mod N]$. Dec: On input pk and ciphertext $c \in \mathbb{Z}_N^*$, compute $\hat{m} := [c^d \mod N]$. If $||\hat{m}|| > \ell + k_0 + k_1$, output $\perp$. Else, parse $\hat{m}$ as $s || t$ with $s \in \{0,1\}^{\ell + k_1}$ and $t \in \{0,1\}^{k_0}$. Compute $r := H(s) \oplus t$ and $m' := G(r) \oplus s$. If the least-significant k_1 bits of m' are no all 0, output $\perp$. Otherwise, output the ℓ most significant bits of $\hat{m}$.

Construction: CCA-secure KEM: (In the random-oracle model) Let GenRSA be as usual and construct a KEM as follows. Gen: On input 1^n, $(N, e, d) \leftarrow \mathsf{GenRSA}(1^n)$. $pk = \langle N, e \rangle$, $sk = \langle N, d \rangle$. As part of the key generation, a function $H : \mathbb{Z}_N^*$ is specified, but the book leaves it implicit. Encaps: On input pk, 1^n, choose uniform $r \in \mathbb{Z}_N^*$. Output the ciphertext $c := [r^e \mod N]$, and key $k := H(r)$. Decaps: On input pk and ciphertext $c \in \mathbb{Z}_N^*$, compute $r := [c^d \mod N]$ and output key $k := H(r)$.

Theorem: If the RSA problem is hard relative to GenRSA and H is modeled as a random oracle, then the above construction is CCA-secure.

Theorem: If the RSA problem is hard relative to GenRSA and H is modeled as a random oracle, then $\Pr[\mathtt{Query}]$ is negligible, where Query is the event that, at any point during it's executing, a PPT adversary $\mathcal{A}$ queries r to the random oracle H.

Definition: Signature Scheme: A set of PPT algorithms $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ such that: Key generation algorithm Gen takes security parameter 1^n and outputs a pair of keys (pk, sk). Assume each has length $\geq n$ and that n can be determined from them. Signing algorithm Sign takes sk and message m for some message space, and outputs $\sigma \leftarrow \mathsf{Sign}_{sk}(m)$. Deterministic verification algorithm Vrfy takes pk, m, and σ, and outputs b, with $b = 1$ meaning valid and $b = 0$ meaning invalid. $b := \mathsf{Vrfy}_{pk}(m, \sigma)$. Except with negl probability over (pk, sk), it must be that $\mathsf{Vrfy}_{pk}(m, \mathsf{Sign}_{sk}(m)) = 1$ for all legal m.

Definition: $\mathsf{Sig\text{-}forge}_{\mathcal{A}, \Pi}(n)$: $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$. $\mathcal{A}$ is given pk and access to oracle $\mathsf{Sign}_{sk}(\cdot)$. $\mathcal{A}$ outputs m, σ. $\mathcal{A}$ succeeds iff $\mathsf{Vrfy}_{pk}(m, \sigma) = 1$ *and* $m \notin Q$, where q is the set of all queries that $\mathcal{A}$ asked its oracle.

Definition: Secure (Existentially unforgeable under an adaptive chosen-message attack): A signature scheme Π such that for all PPT adversaries $\mathcal{A}$, $\Pr[\mathsf{Sig\text{-}forge}_{\mathcal{A}, \Pi}(n) = 1] \leq \mathtt{negl}(n)$.

Construction: Hash-And-Sign: Let $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ be a signature scheme for messages of length $\ell(n)$. Let $\Pi_H = (\mathsf{Gen}_H, H)$ be a hash function with output length $\ell(n)$. Construct signature scheme $\Pi' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vrfy}')$ as follows: Gen′: On input 1^n, $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$, $s \leftarrow \mathsf{Gen}_H(1^n)$. Public key $\langle pk, s \rangle$, secret key $\langle sk, s \rangle$. Sign′: On input a secret key and message $m \in \{0,1\}^*$, output $\sigma = \mathsf{Sign}_{sk}(H^s(m))$. Vrfy′ On input public key, m, and σ, output 1 iff $\mathsf{Vrfy}_{pk}(H^s(m), \sigma) \stackrel{?}{=} 1$.

Theorem: If Π is a secure signature scheme for messages of length ℓ and Π_H is collision resistant, then the above construction is a secure signature scheme (for arbitrary-length messages).

Construction: Plain RSA Signatures: Let GenRSA be usual. Define a signature scheme as follows: Gen: On input 1^n, $(N, e, d) \leftarrow \mathsf{GenRSA}(1^n)$. $pk = \langle N, e \rangle$, $sk = \langle N, d \rangle$. Sign: On input sk, message $m \in \mathbb{Z}_N^*$, compute $\sigma := [m^d \mod N]$. Vrfy: On input pk, $m \in \mathbb{Z}_N^*$, and $\sigma \in \mathbb{Z}_N^*$, output 1 iff $m \stackrel{?}{=} [\sigma^e \mod N]$.

To break the above, choose σ, and compute $m := [\sigma^e \mod N]$.

Construction: RSA-FDH: Let GenRSA be as usual, and construct a signature scheme as follows: Gen: On input 1^n, $(N, e, d) \leftarrow \mathsf{GenRSA}(1^n)$. $pk = \langle N, e \rangle$, $sk = \langle N, d \rangle$. As part of this key generation, a function $H : \{0,1\}^* \to \mathbb{Z}_N^*$ is specified, but the textbook leaves it implicit. Sign: On input sk, message $m \in \{0,1\}^*$, compute $\sigma := [H(m)^d \mod N]$. Vrfy: On input pk, m, σ, output 1 iff $\sigma^e \stackrel{?}{=} H(m) \mod N$.

Theorem: If the RSA problem is hard relative to GenRSA and H is modeled as a random oracle, then the above construction is secure.