RSA Encryption

Lecture 23: From hard roots to hidden messages

Lecture overview

RSA lets a receiver recover a random secret. Turning that secret into a secure encryption scheme requires another idea: the first natural attempt leaks a publicly testable function of the message.

Based on the instructor's Lecture 23, pp. 1–12 [Maj26].

Abstract and lecture roadmap

We follow the original lecture's progression: transport a secret, use it as a multiplicative pad, break the resulting scheme, and investigate randomized padding.

1. Recall the RSA permutation and the receiver's trapdoor.
2. Separate **correct decryption** from **message privacy**.
3. Recover m^e from the first ciphertext and explain two attacks.
4. Ask what random padding fixes—and what it does not prove.

The question that drives the lecture

If recovering the random value r is hard, why is using r to mask a message not automatically secure?

A labeled extension at the end gives hashed RSA and a short connection to RSA-OAEP.

Source sequence: [Maj26, pp. 2–12]; extension: [BS23, §11.4].

Three results to keep in view

Correctness: the trapdoor recovers the random secret

For $r \in \mathbb{Z}_N^*$ and $ed \equiv 1 \pmod{\varphi(N)}$, we have $(r^e)^d \equiv r \pmod{N}$.

Attack: multiplicative masking exposes a message tag

From $(y, c) = (r^e, mr)$, anyone can compute

$$\boxed{c^e y^{-1} \equiv m^e \pmod{N}.}$$

Distinct unit messages have distinct tags, so the attack distinguishes them exactly.

Conditional repair: hashed RSA

With an ideal random oracle H, the ciphertext $(r^e,\ m \oplus H(r))$ hides equal-length messages under the RSA assumption.

First two: source pp. 3–9 [Maj26]. Third: [BS23, Thm. 11.3, p. 445].

Recall: the public map and its secret inverse

Let p, q be **distinct** n-bit odd primes. Define

$$N = pq, \qquad \varphi(N) = (p-1)(q-1), \qquad \mathbb{Z}_N^* = \{x \in \{0, \ldots, N-1\} : \gcd(x, N) = 1\}.$$

Public key

Choose e with $\gcd(e, \varphi(N)) = 1$.

$\mathsf{pk} = (N, e), \qquad f_{N,e}(x) = x^e \bmod N.$

Anyone can evaluate this map.

Secret key

Compute $d = e^{-1} \bmod \varphi(N)$.

$\mathsf{sk} = d, \qquad f_{N,e}^{-1}(y) = y^d \bmod N.$

Keep $d, p, q, \varphi(N)$ secret.

The security parameter n is public; N has $2n-1$ or $2n$ bits.

[HPS14, §§3.1–3.2, pp. 117–125]; [BS23, §10.3, pp. 403–406].

A theorem, followed by an assumption

Proof idea. Write $ed = 1 + k\varphi(N)$ and use Euler's theorem.

For every $r \in \mathbb{Z}_N^*$,

$$(r^e)^d = r^{ed} = r\big(r^{\varphi(N)}\big)^k \equiv r \pmod{N}.$$

Thus exponentiation by e is a permutation, with inverse exponent d.

RSA assumption for a specified key generator Gen

For every efficient A, given a generated (N, e) and $y = r^e$ for uniform $r \in \mathbb{Z}_N^*$,

$$\Pr[A(N, e, y) = r] \leq \mathsf{negl}(n).$$

The probability includes key generation, r, and the attacker's coins; Gen is specified in the final technical-input slides.

This is hardness of recovering the *whole* r, not a claim that every function of r is hidden.

[HPS14, Prop. 3.5, pp. 120–121]; [BS23, Attack Game 10.3, Def. 10.5, pp. 404–405].

Two different meanings of “inverse”

Multiplicative inverse: public and efficiently computable

Given $a \in \mathbb{Z}_N^*$, extended Euclid finds $u, v \in \mathbb{Z}$ with

$$ua + vN = 1, \qquad \text{so } a^{-1} \equiv u \pmod{N}.$$

Neither the factors of N nor $\varphi(N)$ are needed.

RSA inverse: an e-th root

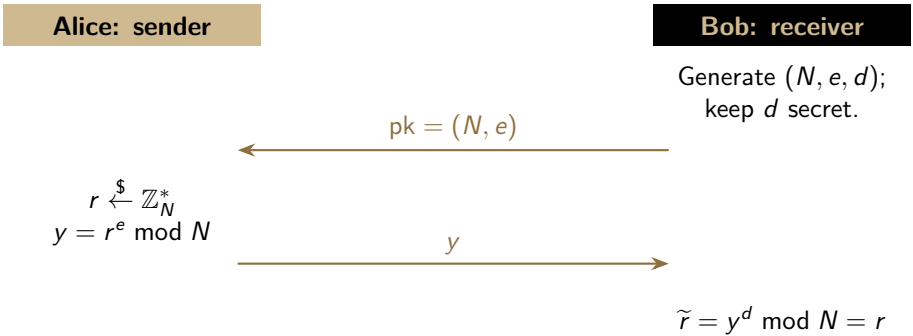
Given y, find r satisfying $r^e \equiv y \pmod{N}$. The secret exponent gives $r = y^d \bmod N$.

For example, $13^{-1} \equiv 28 \pmod{33}$ because $13 \cdot 28 - 11 \cdot 33 = 1$.

Correction to source p. 5: computing inv(r) does *not* require knowing $\varphi(N)$.

[HPS14, Thm. 1.11, pp. 16–17; Fig. 1.3, p. 19]; source: [Maj26, p. 5].

RSA transports a random secret to Bob



A passive observer sees (N, e, y). Recovering the entire r is the RSA inversion task.

As in the source, we assume Alice has Bob's authentic public key. This diagram does not establish authentication.

Source protocol: [Maj26, p. 4]; compare [BS23, §10.3.1, p. 405].

A multiplicative one-time pad really is secure—alone

Let $r \xleftarrow{\$} \mathbb{Z}_N^*$ be independent of $m \in \mathbb{Z}_N^*$ and completely hidden from the observer. Alice sends

$$c = mr \bmod N, \qquad m = cr^{-1} \bmod N.$$

Perfect secrecy from a counting argument

For each fixed $m, c \in \mathbb{Z}_N^*$, exactly one pad works: $r = m^{-1}c$. Hence

$$\Pr[C = c \mid M = m] = \frac{1}{\varphi(N)}.$$

Thus C alone hides m perfectly.

The catch when we transport r using RSA

The observer now sees $(C, Y) = (mr, r^e)$. The proof above only analyzes C.

Source motivation: [Maj26, pp. 4–5]; perfect secrecy: [BS23, §2.1.2].

Putting the two together: the first attempt

Alice: $\mathrm{Enc}_{N,e}(m)$

Input: $m \in \mathbb{Z}_N^*$.

Choose fresh $r \xleftarrow{\$} \mathbb{Z}_N^*$.

$$y = r^e \bmod N,$$
$$c = mr \bmod N.$$

Send the pair (y, c).

Bob: $\mathrm{Dec}_d(y, c)$

Use the secret exponent to recover

$$\widetilde{r} = y^d \bmod N.$$

Then use extended Euclid to compute

$$\widetilde{m} = c\widetilde{r}^{-1} \bmod N.$$

Return $\widetilde{m}$.

There are two public ciphertext components. Security must account for their *joint* distribution.

The first-attempt construction in source pp. 5–7 [Maj26].

Correctness is easy; security is a separate question

Proof idea. First undo RSA exponentiation. Then cancel the recovered pad.

Correctness proof

For every $m, r \in \mathbb{Z}_N^*$,

$$\widetilde{r} = (r^e)^d \equiv r \pmod{N}, \qquad \widetilde{m} = (mr)\widetilde{r}^{-1} \equiv m \pmod{N}.$$

Thus $\mathrm{Dec}_d(\mathrm{Enc}_{N,e}(m)) = m$ for every encryption randomness r.

What this proof does not show

It proves that Bob gets the message back. It says nothing about whether an observer can distinguish two candidate messages.

Domain detail: $1 \le m < 2^{n-1}$ is certainly a unit because $m < p, q$. The integer 0 is not a unit.

Source correctness and domain discussion: [Maj26, pp. 5–7].

A worked example, continuing Lecture 22

Use $N = 33$, $e = 3$, $d = 7$. Take message $m = 2$ and randomness $r = 13$.

Alice encrypts

$$y = 13^3 \bmod 33 = 19,$$
$$c = 2 \cdot 13 \bmod 33 = 26.$$

The ciphertext is $(19, 26)$.

Bob decrypts

$$\widetilde{r} = 19^7 \bmod 33 = 13,$$
$$\widetilde{r}^{-1} = 28 \bmod 33,$$
$$\widetilde{m} = 26 \cdot 28 \bmod 33 = 2.$$

Everything works as intended for the receiver. Now give the observer only $N = 33$, $e = 3$, and $(y, c) = (19, 26)$.

A worked instance of the source construction [Maj26, p. 6]; arithmetic checked independently.

The public calculation that removes the randomness

The observer knows N, e, y, c, where $y = r^e$ and $c = mr$ modulo N.

Proof idea. Raise the masked message to e, then divide by the public RSA image of the pad.

The entire attack is one identity

$$\boxed{z := c^e y^{-1} \equiv (mr)^e (r^e)^{-1} \equiv m^e \pmod{N}.}$$

Both operations are efficient: repeated squaring computes c^e; extended Euclid computes y^{-1}.

In our example,

$$26^3 \equiv 20, \qquad 19^{-1} \equiv 7, \qquad z \equiv 20 \cdot 7 \equiv 8 = 2^3 \pmod{33}.$$

The source's attack identity: [Maj26, p. 9].

A perfect message test—for every message size

Choose distinct $m_0, m_1 \in \mathbb{Z}_N^*$. The challenger encrypts m_b, for a uniform bit b.

An observer's algorithm

Compute $z = c^e y^{-1} \bmod N$ and the two public tags

$$t_0 = m_0^e \bmod N, \qquad t_1 = m_1^e \bmod N.$$

Output the unique b for which $z = t_b$.

Why the test never makes a mistake

RSA exponentiation is injective, so $m_0 \neq m_1$ implies $t_0 \neq t_1$. The observer wins with probability 1, even with fresh randomness for every encryption.

The same tag detects repeated messages across ciphertexts and supports dictionary attacks. No small-message assumption is needed.

Source equality attack, strengthened to an explicit two-message test: [Maj26, p. 9]; compare [BS23, §11.3.1, pp. 438–439].

Integer roots and modular roots are different problems

Over the nonnegative integers

Given $z = x^3$, the function $x \mapsto x^3$ is increasing.

Binary search finds the unique integer root using $O(\log(z+1))$ comparisons.

For example, $\sqrt[3]{125} = 5$.

Modulo N

The map $x \mapsto x^3 \bmod N$ is not increasing.

For example, modulo 33,

$$3^3 \equiv 27, \qquad 4^3 \equiv 31, \qquad 5^3 \equiv 26.$$

Wraparound destroys the order needed by binary search.

Efficiency is measured in the *bit length* of the input, not its numerical value. Trying every integer up to z is not efficient.

Source's integer-versus-modular-root discussion: [Maj26, p. 8].

When there is no wraparound, recover the message

Small-message recovery

Represent m by an integer $0 \le m < N$. If $m^e < N$, then the tag is an equality of ordinary integers:

$$z = m^e, \qquad m = \sqrt[e]{z}.$$

An exact integer-root algorithm recovers m without the trapdoor.

For $N = 33$, $e = 3$, the ciphertext $(19, 26)$ yields $z = 8$. Since $2^3 = 8 < 33$, the integer cube root immediately gives $m = 2$.

Two attacks, not one

Small messages: recover m by an integer root.
Arbitrary unit messages: distinguish candidates by comparing their tags.

[Maj26, pp. 8–9]. The candidate-message attack does not require $m < N^{1/e}$.

Why this does not contradict the RSA assumption

What RSA promised

Given a generated public key and $y = r^e$ for random r, recovering the entire r is hard.

What the attack actually does

It uses the *additional* ciphertext component $c = mr$ and computes a public function of m:

$$c^e y^{-1} = m^e.$$

There is no need to recover a random RSA preimage.

The pad r is uniform before y is published. For a fixed public key, y uniquely determines r mathematically, even when finding it is computationally hard. The ordinary one-time-pad proof does not apply to the joint view (y, c).

Source construction and attack: [Maj26, pp. 4–9]; RSA assumption: [BS23, Def. 10.5, p. 405].

The source's next idea: randomize the payload

Let $n \geq 4$ be even and $t = n/2$. For $m \in \{0,1\}^t$, choose $s \xleftarrow{\$} \{0,1\}^t$ and encode

$$x = s \parallel m = 2^t s + m \qquad (0 \leq x < 2^n < N).$$

The padded first attempt

Choose $r \xleftarrow{\$} \mathbb{Z}_N^*$ and send

$$y = r^e \bmod N, \qquad c = xr \bmod N.$$

Bob computes $x = c(y^d)^{-1} \bmod N$, then reads the last t bits.

Motivation. The exposed value is now $(s \parallel m)^e$. Fresh s changes the payload even when m repeats.

Domain clarification: x may be a nonunit. Canceling the unit r still works; full-domain RSA correctness is proved in the appendix.

Source padding construction: [Maj26, p. 10]; full-domain RSA: [HPS14, Prop. 3.5, pp. 120–121].

Fresh padding makes repeated payloads unlikely

Encrypt a fixed message m a total of Q times, using independent t-bit strings $s_1, \ldots, s_Q$.

Proof idea. Each pair of pads agrees with probability 2^{-t}. Sum over pairs.

Birthday bound

$$\Pr[\exists i < j : \; s_i = s_j] \leq \binom{Q}{2} 2^{-t}.$$

For fixed m, equal payloads—and equal RSA images—occur exactly when the pads agree.

What the calculation proves

For $Q \ll 2^{t/2}$, exact repetitions are unlikely. This is *not* an indistinguishability proof for two different messages.

Birthday-bound prompt from source p. 10 [Maj26]; the bound follows directly by a union bound.

Random padding usually creates wraparound

Keep $t = n/2$ and take $e \geq 3$. For a fixed message m, let $X = 2^t S + m$ with uniform $S \in \{0, \ldots, 2^t - 1\}$.

A quantitative version of the source's intuition

$$\Pr[X^e < N] \leq \frac{N^{1/e}}{2^n} + 2^{-n/2} < 2^{-n(1-2/e)} + 2^{-n/2}.$$

For $e = 3$, this is at most $2^{-n/3} + 2^{-n/2}$.

Only the payloads below $N^{1/e}$ avoid wraparound. They occur at spacing 2^t, so there are few possible padding strings that produce them.

A limited conclusion

This makes the *ordinary integer-root attack* unlikely. It does not rule out other attacks on the randomized encoding.

Source intuition: [Maj26, p. 10]. Counting proof and endpoint check are in the appendix.

The optimized form: encrypt the payload directly

The observer can still compute

$$c^e y^{-1} \equiv (s \parallel m)^e \pmod{N}.$$

This motivates the source's one-component version:

Encryption

Choose $s \xleftarrow{\$} \{0,1\}^t$.

$$x = 2^t s + m, \qquad C = x^e \bmod N.$$

Send C.

Decryption

Recover $x = C^d \bmod N$.

$$m = x \bmod 2^t.$$

For an honest ciphertext, $x < 2^n$.

Example: $p = 41$, $q = 47$, $e = 3$, $d = 1227$, $t = 3$. With $s = 5$ and $m = 3$, the payload is 43 and $C = 43^3 \bmod 1927 = 500$.

Source's optimized construction: [Maj26, p. 11]. This is still a candidate scheme, not a security theorem.

Randomized padding is an idea, not yet a proof

What we established for the source's prefix padding

Correct decryption; a small probability of an unwrapped payload; and a birthday bound for repeated payloads.

What remains unproved here

That encryptions of arbitrary equal-length messages are computationally indistinguishable under the RSA assumption.

Neither "the ciphertext changes" nor "the power wraps around" establishes this.

Added connection. We now separate two ways forward: derive a pad by hashing the random RSA preimage, or use an analyzed RSA encoding such as OAEP.

The source's name "Final Optimized RSA Encryption" is retained as motivation, not as a claim that concatenating random bits is a secure standard.

Source pp. 10–11 [Maj26]; analyzed constructions: [BS23, §11.4; §12.8.4].

A principled repair: hash the RSA secret

Added connection. Let $H : \mathbb{Z}_N^* \to \{0,1\}^\ell$ be an ideal random oracle, public to everyone. Messages are ℓ-bit strings.

Alice encrypts

Choose fresh $r \xleftarrow{\$} \mathbb{Z}_N^*$.

$$y = r^e \bmod N, \qquad k = H(r).$$

Send

$$(y, C) = (r^e,\ m \oplus k).$$

Bob decrypts

Recover $r = y^d \bmod N$.

Recompute the same pad and output

$$m = C \oplus H(r).$$

Correctness: $(m \oplus H(r)) \oplus H((r^e)^d) = m$.

Inputs to H use a fixed, unambiguous encoding. Here its domain is described for one public key.

OTP specialization of [BS23, §11.4, pp. 441–445, Thm. 11.3].

The security claim—including its model

What an ideal random oracle means

On a new input, H returns an independent uniform ℓ-bit string. Repeating an input returns the same answer. The attacker may query H at any chosen inputs.

Conditional theorem: hashed RSA hides the message

For each efficient single-challenge IND-CPA attacker A, an efficient RSA inverter B satisfies

$$\left|\Pr[\widehat{b} = b] - \frac{1}{2}\right| \leq \mathsf{RSAadv}(B).$$

Here $\mathsf{RSAadv}(B)$ is the probability that B inverts a random RSA challenge.

Proof idea. Replace $H(r)$ by an independent pad. A query at r reveals the substitution—and supplies the RSA root.

Specialized unit-domain proof of [BS23, Thm. 11.2, eqs. (11.3)–(11.7), pp. 442–444; Thm. 11.3, p. 445].

Why distinguishing would give an RSA root

1. **Ideal experiment.** Given the RSA challenge (N, e, y), choose independent $K \xleftarrow{\$} \{0,1\}^{\ell}$ and encrypt the selected message as $C = m_b \oplus K$. Answer hash queries with an independent lazy-sampled table.

2. **Recognize the decisive query.** For every hash input u, test

$$u^e \equiv y \pmod{N}.$$

   If it passes, output u: it is the unique RSA preimage r.

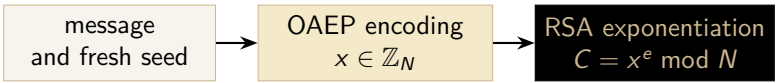
3. **Compare the experiments.** Real and ideal views agree until such a query. The ideal view hides b perfectly. Thus the distinguishing advantage is at most the probability of that query.

There is no need to guess which hash query is correct: the public RSA map checks every candidate. The test also covers queries made before the challenge ciphertext.

Game comparison and extraction: [BS23, pp. 442–444, eqs. (11.5)–(11.7)].

RSA-OAEP: randomized encoding with a security analysis

Added connection. RSA-OAEP is not the concatenation $s \| m$. It mixes a random seed and a structured data block using hash-based masks.



The precise distinction

RSA-OAEP has a chosen-ciphertext security analysis in the random-oracle model. Arbitrary random-prefix padding does not inherit that theorem.

The RSAES-OAEP specification defines the encoding, decoding checks, and message-length limit. These details are part of the scheme, not optional decoration.

[MKJR16, §7.1, pp. 19–26; §7.1.1, pp. 22–24]; [BS23, §12.8.4, pp. 508–510].

Looking ahead: implementing the RSA arithmetic

The three tasks in the source lecture

1. Generate distinct n-bit primes p, q and compute $N = pq$.
2. Choose e relatively prime to $(p-1)(q-1)$.
3. Compute $d = e^{-1} \bmod (p-1)(q-1)$ by extended Euclid.

Operations needed when encrypting and decrypting

Modular exponentiation by repeated squaring; sampling fresh randomness; modular inverses when using a multiplicative mask; and the chosen encoding or key-derivation procedure.

The arithmetic is only one layer. A complete encryption specification must also say how messages are encoded and how invalid ciphertexts are handled.

Original implementation roadmap: [Maj26, p. 12]; [HPS14, §1.3.2; §3.2, Table 3.1]; [MKJR16, §§5.1,7.1].

The lecture in three equations

Correctness

$(r^e)^d \equiv r \pmod{N}$: the secret exponent recovers the transported value.

The failed construction

$$(y, c) = (r^e, mr) \quad \Longrightarrow \quad c^e y^{-1} = m^e \pmod{N}.$$

The observer learns a deterministic, publicly testable message tag.

A repair with an explicit security model

$$(y, C) = (r^e,\ m \oplus H(r)).$$

RSA hardness plus an ideal random oracle gives message indistinguishability.

Source: [Maj26, pp. 3–11]; added theorem: [BS23, Thm. 11.3, p. 445].

Check your understanding

1. An observer knows N and $y \in \mathbb{Z}_N^*$. Can she compute y^{-1} without factoring N? Does this also give an e-th root of y?

2. In the first attempt, what public calculation distinguishes encryptions of $m_0 = 2$ and $m_1 = 5$ when $N = 33$ and $e = 3$?

3. Does fresh randomness r stop the observer from recognizing repeated messages? What changes after adding a random prefix s?

4. What extra assumption or model is used by the hashed-RSA proof? Why is collision resistance alone not the assumption stated there?

For each answer, separate an algebraic identity, a probability calculation, and a computational security claim.

Answers to the check

1. Yes: extended Euclid computes y^{-1}. It solves $yu = 1$, not $u^e = y$; the latter is the RSA inversion problem.

2. Compute $z = c^e y^{-1} \bmod 33$. The two tags are $2^3 = 8$ and $5^3 \equiv 26 \pmod{33}$, so the test is exact.

3. No: m^e is independent of r. A fresh prefix changes the exposed payload to $s \parallel m$; for a fixed message, repeated payloads occur with probability at most $\binom{Q}{2} 2^{-t}$ across Q encryptions.

4. The proof models H as an ideal random oracle. Collision resistance only concerns finding equal hash outputs; it does not supply the independent random answers used in this argument.

Definitions and construction: [BS23, §8.10.2; §11.4].

Appendix: RSA works on all residues, including zero

Proof idea. Prove $x^{ed} = x$ separately modulo p and modulo q, then use the Chinese remainder theorem.

The two cases modulo p

If $x \equiv 0 \pmod{p}$, then $x^{ed} \equiv x \equiv 0 \pmod{p}$. Otherwise, Fermat's theorem gives $x^{p-1} \equiv 1 \pmod{p}$. Since $ed \equiv 1 \pmod{p-1}$, again $x^{ed} \equiv x \pmod{p}$.

Repeat modulo q. The Chinese remainder theorem gives $x^{ed} \equiv x \pmod{pq}$ for every $x \in \mathbb{Z}_N$.

Do not conflate the domains

The full-domain correctness theorem does not change the unit-domain RSA hardness assumption used in the key-transport argument.

[HPS14, Prop. 3.5 and eq. (3.3), pp. 120–121; Thm. 2.24, pp. 84–85].

Appendix: counting the payloads that do not wrap

Fix $m \in \{0, \ldots, 2^t - 1\}$ and let $A = N^{1/e}$. The event $X^e < N$ is

$$2^t S + m < A, \qquad 0 \le S < 2^t.$$

Count the possible padding strings

There are at most $\lceil A/2^t \rceil \le A/2^t + 1$ such integers S. Therefore

$$\Pr[X^e < N] \le \frac{A}{2^{2t}} + 2^{-t} = \frac{N^{1/e}}{2^n} + 2^{-n/2}.$$

Using $N < 2^{2n}$ gives the stated bound for $e \ge 3$.

Endpoint check: the $+1$ covers rounding and the possible payload $X = 0$. No assumption that X is a unit was used, and no rejection sampling is being performed.

Detailed calculation for the source's wraparound intuition [Maj26, p. 10].

Appendix: why the multiplier is redundant for unit payloads

For $x \in \mathbb{Z}_N^*$, the ciphertext (r^e, xr) and the single value $z = x^e$ can be converted into one another without knowing x.

Pair to single value

Given (y, c), output

$$z = c^e y^{-1} \bmod N.$$

This equals x^e.

Single value to a simulated pair

Given $z \in \mathbb{Z}_N^*$, choose $c \xleftarrow{\$} \mathbb{Z}_N^*$ and set

$$y = c^e z^{-1} \bmod N.$$

Output (y, c).

Distribution check. Put $r = x^{-1}c$. Then r is uniform, $c = xr$, and $y = c^e(x^e)^{-1} = r^e$. The simulated pair has exactly the real distribution.

This two-way statement is restricted to unit payloads. The public tag identity itself also holds for nonunits.

An explicit justification of the redundancy observation in source p. 11 [Maj26].

Reading guide

The undergraduate mathematics reference

Hoffstein–Pipher–Silverman, second edition [HPS14]:
§1.2 and Fig. 1.3, p. 19: extended Euclid; §1.3.2, pp. 24–26: fast exponentiation; §§3.1–3.2, pp. 117–125: roots modulo pq and the RSA cryptosystem.

The cryptographic formulation and repair

Boneh–Shoup, version 0.6 [BS23]:
§10.3, pp. 403–406: RSA assumption; §11.3.1, pp. 438–439: why deterministic encryption fails; §11.4, pp. 441–445: hashed trapdoor encryption; §12.8.4, pp. 508–510: OAEP.

For the precise RSAES-OAEP format, read RFC 8017, §7.1 [MKJR16]. The original RSA construction appears in [RSA78].

Page numbers above refer to the printed pages of the editions listed, not PDF-viewer indices.

References

[BS23] Dan Boneh and Victor Shoup.
A graduate course in applied cryptography, 2023. Version 0.6, January 2023. URL: https://toc.cryptobook.us/book.pdf.

[HPS14] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman.
An Introduction to Mathematical Cryptography. Undergraduate Texts in Mathematics. Springer, New York, second edition, 2014. doi:10.1007/978-1-4939-1711-2.

[Maj26] Hemanta K. Maji.
CS355, Lecture 23: RSA Encryption. Instructor's lecture slides, Purdue University, 2026. Source file supplied by the instructor: 23.pdf, 12 slides.

[MKJR16] Kathleen Moriarty, Burt Kaliski, Jakob Jonsson, and Andreas Rusch.
PKCS #1: RSA Cryptography Specifications Version 2.2. RFC 8017, RFC Editor, November 2016. Kathleen Moriarty, editor; Section 7.1 specifies RSAES-OAEP. URL: https://www.rfc-editor.org/rfc/rfc8017, doi:10.17487/RFC8017.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman.
A method for obtaining digital signatures and public-key cryptosystems. ***Communications of the ACM***, 21(2):120–126, 1978. doi:10.1145/359340.359342.

Technical inputs: arithmetic and probability

Arithmetic facts used in the proofs

Extended Euclid: find u, v with $ua + vN = \gcd(a, N)$ in polynomial time. When the gcd is 1, this yields $a^{-1} \bmod N$.
Exponentiation: compute $a^j \bmod N$ with $O(\log j)$ modular multiplications for $j \geq 2$.
RSA inverse formula: for distinct primes p, q and $ed \equiv 1 \pmod{(p-1)(q-1)}$, one has $x^{ed} \equiv x \pmod{pq}$ for every integer x.

[HPS14, Thm. 1.11, pp. 16–17; Fig. 1.3, p. 19; §1.3.2, pp. 24–26; Prop. 3.5, pp. 120–121].

Elementary probability input

For events $E_1, \ldots, E_k$, $\Pr[\bigcup_i E_i] \leq \sum_i \Pr[E_i]$. This is the only probability inequality needed for the birthday bound. The wraparound estimate is direct counting.

Technical input: the key distribution and RSA assumption

The key-generation convention inherited from Lecture 22

Sample distinct n-bit primes p, q uniformly. Set $N = pq$, choose $e \xleftarrow{\$} \mathbb{Z}^*_{\varphi(N)}$, and compute $d = e^{-1} \bmod \varphi(N)$.

Correctness holds for every valid exponent; hardness concerns this key *distribution*.

RSA hardness for precisely this generator

For every efficient attacker B,

$$\mathsf{RSAadv}(B) := \Pr[B(N, e, r^e \bmod N) = r] \leq \mathsf{negl}(n),$$

where $(N, e, d) \leftarrow \mathsf{Gen}(1^n)$ and $r \xleftarrow{\$} \mathbb{Z}^*_N$.

Boneh–Shoup instead use fixed e and the full residue domain. Our reduction matches its assumption exactly.

Compare [BS23, Attack Game 10.3, Def. 10.5, pp. 404–405; Thm. 11.3, p. 445].

Technical input: the model behind the encryption claims

Ideal random oracle: the input to our hashed-RSA proof

H gives independent uniform ℓ-bit outputs on distinct inputs, with consistent replies to repeated inputs. The attacker has oracle access.

This is an idealized model, not merely a collision-resistance assumption.

[BS23, §8.10.2; §11.4, p. 441].

Scope: distinguish the two encryption claims

Hashed RSA: our proof gives single-challenge IND-CPA security, with no decryption oracle. It does not give chosen-ciphertext security: flipping a bit of C flips that plaintext bit.
RSA-OAEP: the stronger CCA result is an external input, not proved here; RFC 8017, §7.1, pp. 19–21, discusses its RSA-specific random-oracle analysis.

[BS23, Thm. 11.3, p. 445]; [MKJR16, §7.1, pp. 19–21].