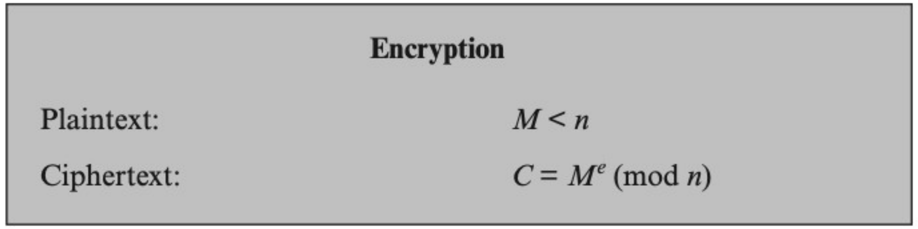
Lecture 14

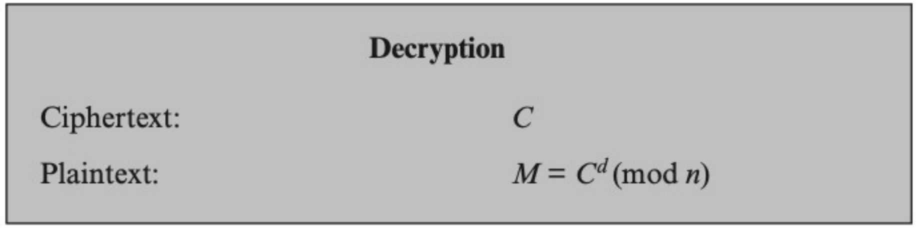
RSA use

- to encrypt a message M the sender:
  - obtains **public key** of recipient `pk={e,n}`
  - computes: $C=M^e \bmod n$, where $0 \leq M < n$
- to decrypt the ciphertext C the owner:
  - uses their private key `sk={d,p,q}`
  - computes: $M=C^d \bmod n$
- note that the message M must be smaller than the modulus n (block if needed)



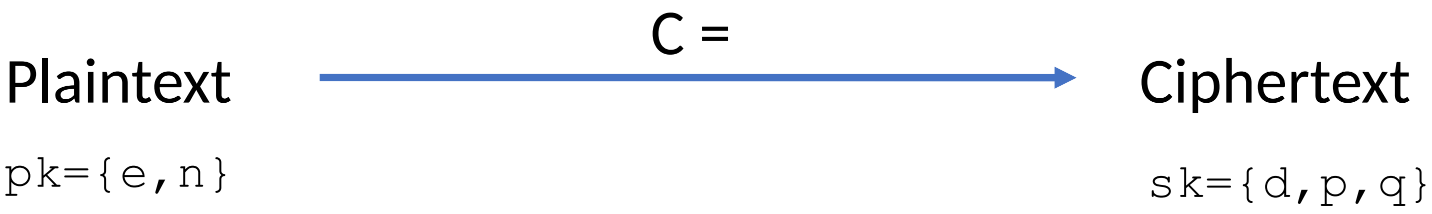
Encryption

Plaintext:	$M < n$
Ciphertext:	$C = M^e \pmod{n}$



Decryption

Ciphertext:	C
Plaintext:	$M = C^d \pmod{n}$



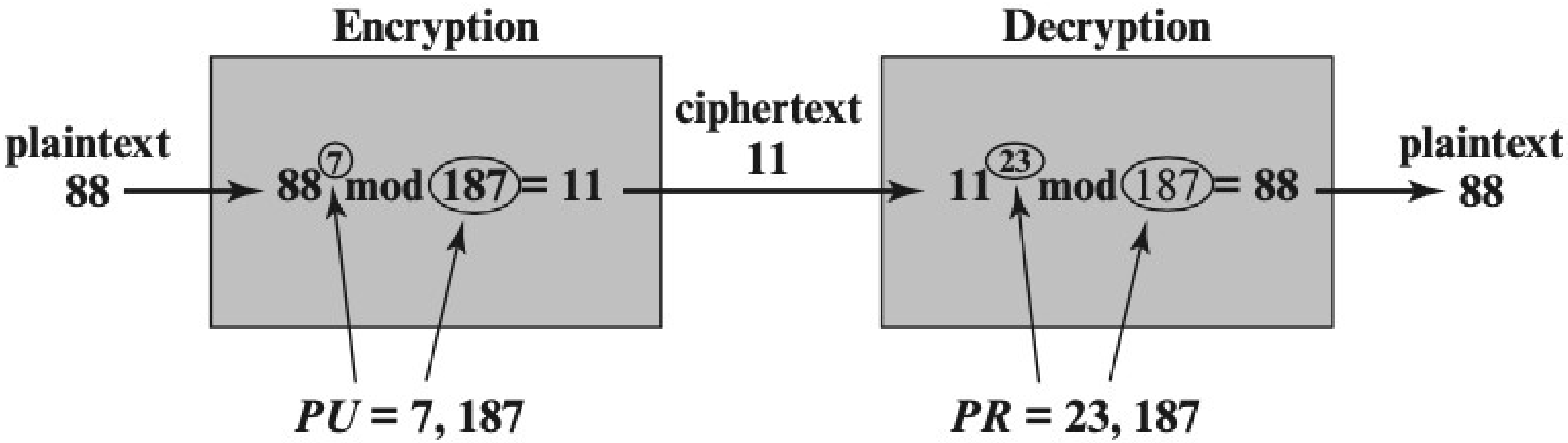
RSA example continue

- sample RSA encryption/decryption is:
- given message `M = 88` (`88<187`)
- encryption:

  $C = 88^7 \bmod 187 = 11$
- decryption:

  $M = 11^{23} \bmod 187 = 88$

Example of RSA algorithm



RSA key generation

- users of RSA must:
  - determine two primes at random - `p, q`
  - select either `e` or `d` and compute the other
- primes `p,q` must not be easily derived from modulus `n=p.q`
  - means must be sufficiently large
  - typically guess and use probabilistic test
- exponents `e`, `d` are inverses, so use Inverse algorithm to compute the other

Correctness of RSA

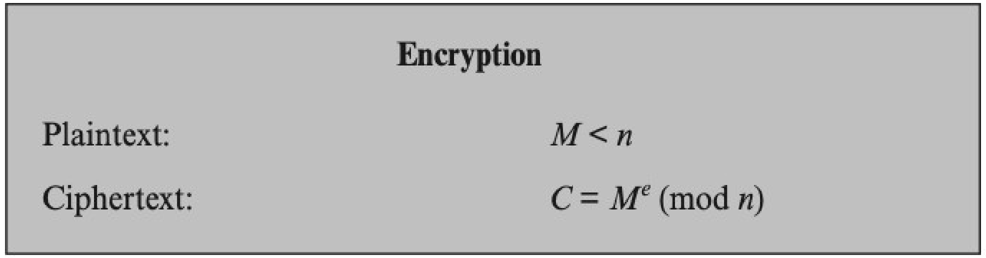
- Euler’s theorem: if gcd (M, n) = 1, then mod n. Here φ(*n*) is Euler’s totient function: the number of integers in {1, 2, . . ., *n*-1} which are relatively prime to *n*. When *n* is a prime, this theorem is just Fermat’s little theorem

M’ = mod n   mod n

mod n

¿ $[M^{\phi(n)}]^k \cdot M \, mod \, n$

= M mod n



Encryption

Plaintext:	$M < n$
Ciphertext:	$C = M^e \pmod{n}$

Attack approaches

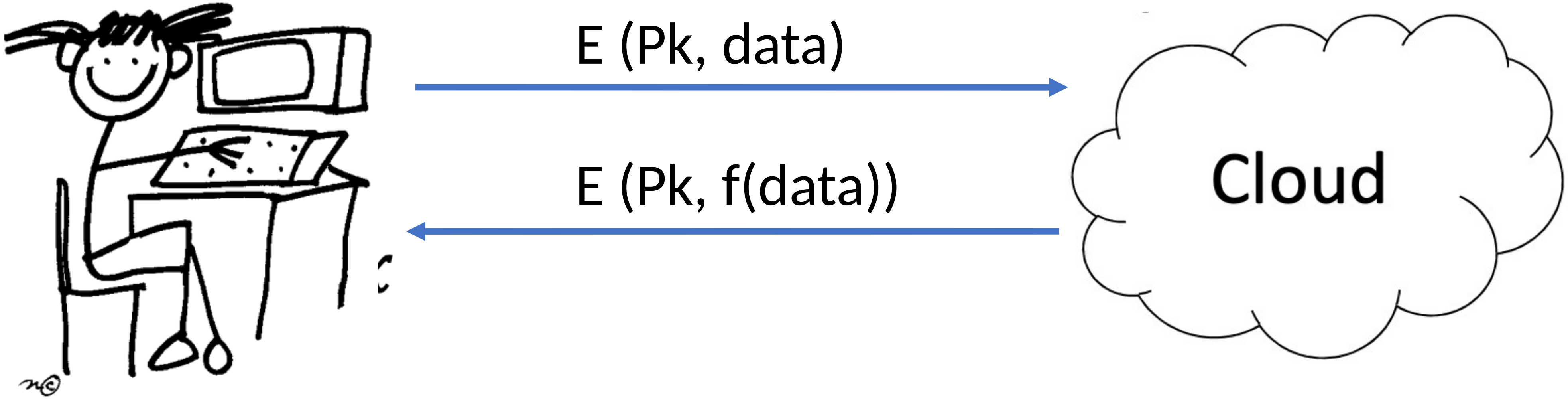
- **Mathematical attacks**: several approaches, all equivalent in effort to factoring the product of two primes. The defense against mathematical attacks is to use a large key size.
- **Timing attacks**: These depend on the running time of the decryption algorithm
- **Chosen ciphertext attacks**: this type of attacks exploits properties of the RSA algorithm by selecting blocks of data. These attacks can be thwarted by suitable padding of the plaintext, such as PKCS1 V1.5 in SSL

Homomorphic encryption

- Encryption scheme that allows computation on ciphertexts
  - i.e. a public-key encryption scheme that allows anyone in possession of the public key to perform operations on encrypted data without access to the decryption key
- Initial public-key systems that allow this for either addition or multiplication, but not both.
- Fully homomorphic encryption (FHE)

Application of homomorphic encryption

- One Use case: cloud computing
  - A weak computational device Alice (e.g., a mobile phone or a laptop) wishes to perform a computationally heavy task, beyond her computational means. She can delegate it to a much stronger (but still feasible) machine Bob (the cloud, or a supercomputer) who offers the service of doing so. The problem is that Alice does not trust Bob, who may give the wrong answer due to laziness, fault, or malice.



E (Pk, data)

E (Pk, f(data))

Cloud

RSA reading materials

- A Method for Obtaining Digital Signatures and Public-Key Cryptosystems