6G7Z1009: Introduction to Computer Forensics and Security

Cryptography - II

Reading List

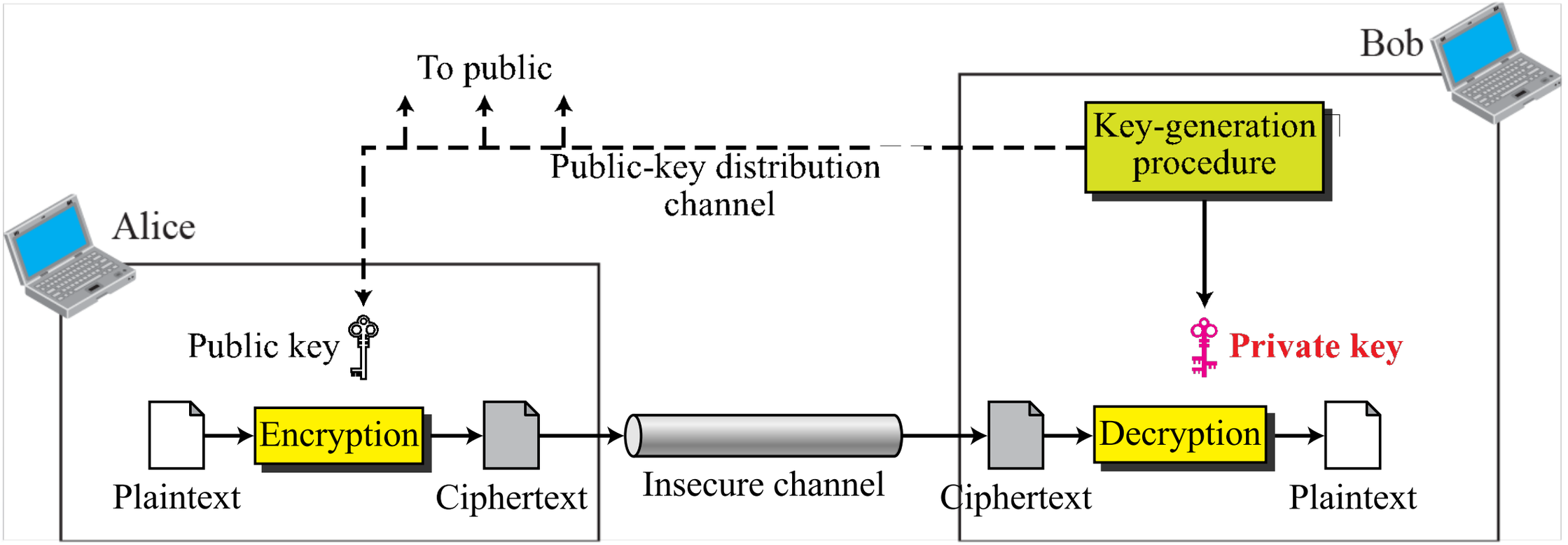
- W. Stallings, Cryptography and Network Security: Principles and Practice (7th Edition), 2016, Pearson (Chapter 9 and 10)
- Nigel P. Smart, Cryptography Made Simple (Information Security and Cryptography), 2015, Springer
- M. Stamp, Information Security. Principles and Practice (2nd Edition), 2011, John Wiley (Chapter 4)
- D. Gollmann, Computer Security, 3rd Edition, 2011, John Wiley (Chapter 14 & 15)
- Alexander Stanoyevitch, Introduction to Cryptography with Mathematical Foundations and Computer Implementations, 2010, CRC Press (Chapter 9)
- J. Erickson, Hacking: The Art of Exploitation (2nd Edition), 2008 (Chapter 7)
- Online resources:
  - Menezes,A.,van Oorschot,P.,andVanstone,S.(1996) Handbook ofApplied Cryptography.CRC Press. Chapters 8 & 14. See http://www.cacr.math.uwaterloo.ca/hac/, [Online access 11 Sep. 2017].
  - Ellis, J.H. (1987) The history of Non-Secret Encryption. See http://www.cesg.gov.uk/site/publications/
  - media/ellis.pdf, [Online 10 September 2017].

Asymmetric-key ciphers

- Also called 'public-key' ciphers
  - Symmetric key cryptography is based on shared secrecy while asymmetric key cryptography is based on personal secrecy
- Basic ideas
  - A user has two keys: a public key and a private key.
  - A message can be encrypted with the public key and decrypted with the private key to provide security.
  - A message can be encrypted with the private key and decrypted with the public key to provide signatures.

Asymmetric-key ciphers

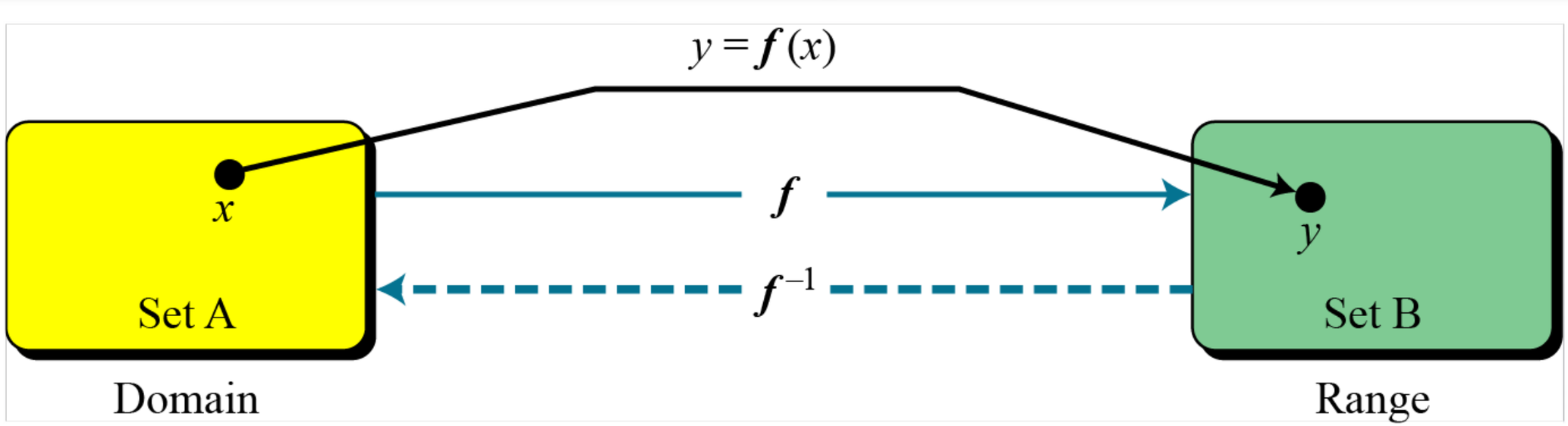
- Also called 'public-key' ciphers



Asymmetric-key ciphers

•The main idea behind asymmetric-key cryptography is the concept of the trapdoor one-way function.

•Trapdoor one-way function: a function that is easy to compute in one direction, yet difficult to compute in the opposite direction (finding its inverse) without special information, called the "trapdoor". Trapdoor functions are widely used in cryptography.



Asymmetric-key ciphers

- One-way function:
  - f is easy to compute
  - f^{-1} is difficult to compute.
- Trapdoor one-way function
  - Given y and a trapdoor, x can be computed easily.

Asymmetric-key ciphers

- An example of one-way function,

When n is large, $n = p \times q$ is a one-way function.

- Given p and q -> calculate n (easy)
- Given n ->calculate p and q (Difficult)

Asymmetric-key ciphers

- An example of trapdoor one-way function,

When n is large, the function $y = x^k \bmod n$ is a trapdoor one-way function.

- Given x, k and n -> calculate y (easy)
- Given y, k and n ->calculate x (Difficult)
- However, if we know the trapdoor, k' such that $k \times k' = 1 \bmod \Phi(n)$, we can use $x = y^{k'} \bmod n$ to find x.


RSA

- The most common public-key algorithm is the RSA cryptosystem, named for its inventors (Rivest, Shamir, and Adleman)
- RSA implements two ideas:
  - public key encryption: encryption keys are public, while the decryption keys are not, so only the person with the correct decryption key can decipher an encrypted message. Everyone has their own encryption and decryption keys. The keys must be made in such a way that the decryption key may not be easily deduced from the public encryption key.

RSA

- RSA implements two ideas:
  - Digital signatures: the receiver may need to verify that a transmitted message actually originated from the sender (signature), and didn't just come from there (authentication). This is done using the sender's decryption key, and the signature can later be verified by anyone, using the corresponding public encryption key. Signatures therefore cannot be forged. Also, no signer can later deny having signed the message.

RSA

- Basic notation: Let e, d, n be positive integers, with (e, n) as the encryption key, (d, n) the decryption key
- M is the plaintext message. C is ciphertext.
- Encryption of the message by raising it to the e^{th} power modulo n to obtain C, the ciphertext.
- Decryption of C by raising it to the d^{th} power modulo n to obtain M again. Formally, we obtain these encryption and decryption algorithms for E and D

RSA algorithm

- Publish their public encryption key $PU=\{e,n\}$
- Keep secret private decryption key $PR=\{d,n\}$
- To encrypt a message M the sender:
  - obtains public key of recipient $PU=\{e,n\}$
  - computes: $C = M^e \bmod n$, where $0 \le M < n$
- To decrypt the cipher text C
  - Uses the private key $PR=\{d,n\}$
  - Computes $M = C^d \bmod n$
- Note that the message M must be smaller than the modulus n

RSA algorithm

- Pick two primes p and q, let n=pq
- Define $\phi(n)=(p-1)(q-1)$
  - *select the encryption exponent e, where*

$$1<e<\phi(n),\ \gcd(e,\phi(n))=1$$

'gcd' refers to the greatest common divisor

- *Solve following equation to find the decryption exponent d*

$$d = e^{-1} \bmod \phi(n) \text{ and } 0 \le d \le n$$

or

$$e.d \bmod \phi(n) = 1 \text{ and } 0 \le d \le n$$

RSA

- RSA's security is based on Modular arithmetic
  - $a \equiv b \pmod{n}$ <-> there is a q such that $a-b=qn$;
  - b is the reminder after dividing a by n
  - $23 \equiv 3 \pmod{5}$
- A set {0,1,... n-1} is closed under modular addition and multiplication
- (a(mod n) +b (mod n))(mod n) = (a+b)(mod n)
- (ab)(mod n) = (a (mod n) b (mod n))(mod n)

Modular Algorithm

- Two numbers p and q are said to be relatively prime if their greatest common divisor (gcd) is 1
  - *5 and 17, 8 and 9, 10 and 21*
- *To compute gcd:*
  - *gcd(a,b) =gcd(b, a mod b) (Euclid, 300BC)*

RSA example

- Starting with two primes 17 and 11, find a public private key pair, with the encryption exponent less than the decryption exponent. Use the public key to encrypt the message who's numerical encoding is 88

RSA example

- Selected primes: p=17, q =11
- Compute n=pq=17x11=187
- Compute $\phi(n)=(p-1)(q-1)=16 \times 10=160$
- *Select* e: $\gcd(e,160)=1$; choose $e=7$
- *Determine d: d x e mod 160 =1 and d<160*
- *d x e is 1 plus an integer multiple of* $\phi(n)$ *, Hence with i=1, d=(1+ix160)/7 =23 or value is d=23 since 23 x7=161=160+1*
- *Publish public key PU=(7,187)*
- *Keep secret private key PR={23, 187}*

RSA example

- Bob chooses 7 and 11 as p and q, calculated n =77. The value of $\varnothing(n)$ =(7-1)(11-1)=60
- For, exponents, e and d, if e is chosen as 13, then d is 37, note e*d mod 60 =1 (they are inverses of each). Now imagine that Alice wants to send the plaintext 5 to Bob, she uses the public exponent 13 to encrypt 5

Plaintext: 5	$C = 5^{13} = 26 \bmod 77$	Ciphertext: 26

- Bob receives the ciphertext 26 and uses the private key 37 to decipher the ciphertext

Ciphertext: 26	$P = 26^{37} = 5 \bmod 77$	Plaintext: 5

RSA example

- Try out at here-- Online RSA: http://logos.cs.uic.edu/340%20Notes/rsa.html

Why RSA works

- Because of Euler's Theorem: $a^{\phi(n)} \bmod n = 1$ where $\gcd(a,n)=1$
- In RSA have
  $n=p.q$
  $\phi(n)=(p-1)(q-1)$
  carefully chose e & d to be inverses $\bmod \phi(n)$
  hence $e.d=1+k.\phi(n)$ for some k
- Hence

$$C^d = M^{e.d} = M^{1+k.\phi(n)} = M^1.(M^{\phi(n)})^k$$
$$= M^1.(1)^k = M^1 = M \bmod n$$

Efficient Encryption

- Encryption uses exponentiation to power e
- Hence if e small, this will be faster

often choose e=65537 (2^{16}+1)
also see choices of e=3 or e=17

- But if e too small, then it will be easily attacked

Efficient Decryption

- Decryption uses exponentiation to power d, this is likely large
- Can use the Chinese Reminder Theorem (CRT) to compute mod p and q separately. Then combine to get desired answer (4 times faster than doing directly)
- Only owner of private key who knows values of p and q can use this technique

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 Euclid's inverse algorithm to compute the other

RSA security

- Possible approaches to attacking RSA are:
  - Brute force key search (infeasible given size of numbers)
  - Mathematical attacks (based on difficulty of computing , by factorin ⌀(n) dulus n)

Other attacks

- Dictionary attack is a technique for detecting a cipher or authentication mechanism by trying to determine its decryption key or passphrase by trying hundreds or sometimes millions of likely possibilities, such as words in a dictionary
- Brute-force attacks, or exhaustive key search, is used against any encrypted data.
- Such an attack might be utilised when it is not possible to take advantage of weakness in an encryption system that would make the task easier
- It consists of systematically checking all possible keys or password until the correct one is found
- In the worst case, it involves traversing the entire search space.

RSA recommendations

- The number of bits for n should be at least 1024. This means that n should be around 2^{1024}, or 309 decimal digits.
- The two primes p and q must each be at least 512 bits.
- The values of p and q should not be very close to each other.
- Both p-1 and q-1 should have at least one large prime factor.
- The ratio p/q should not be close to a rational number with a small enumerator or denominator.

RSA recommendations

- The ratio p/q should not be close to a rational number with a small enumerator or denominator.
- The modulus n must not be shared. The value of e should be $2^{16}+1$
- If the private key d is leaked, Bob must immediately change n as well as both *e* and *d*. It has been proven that knowledge of n and one pair (e,d) can lead to the discovery of another pairs of the same modulus.
- Message must be padded with OAEP (Optimal asymmetric encryption padding). A short message in RSA makes the ciphertext vulnerable to short message attack.

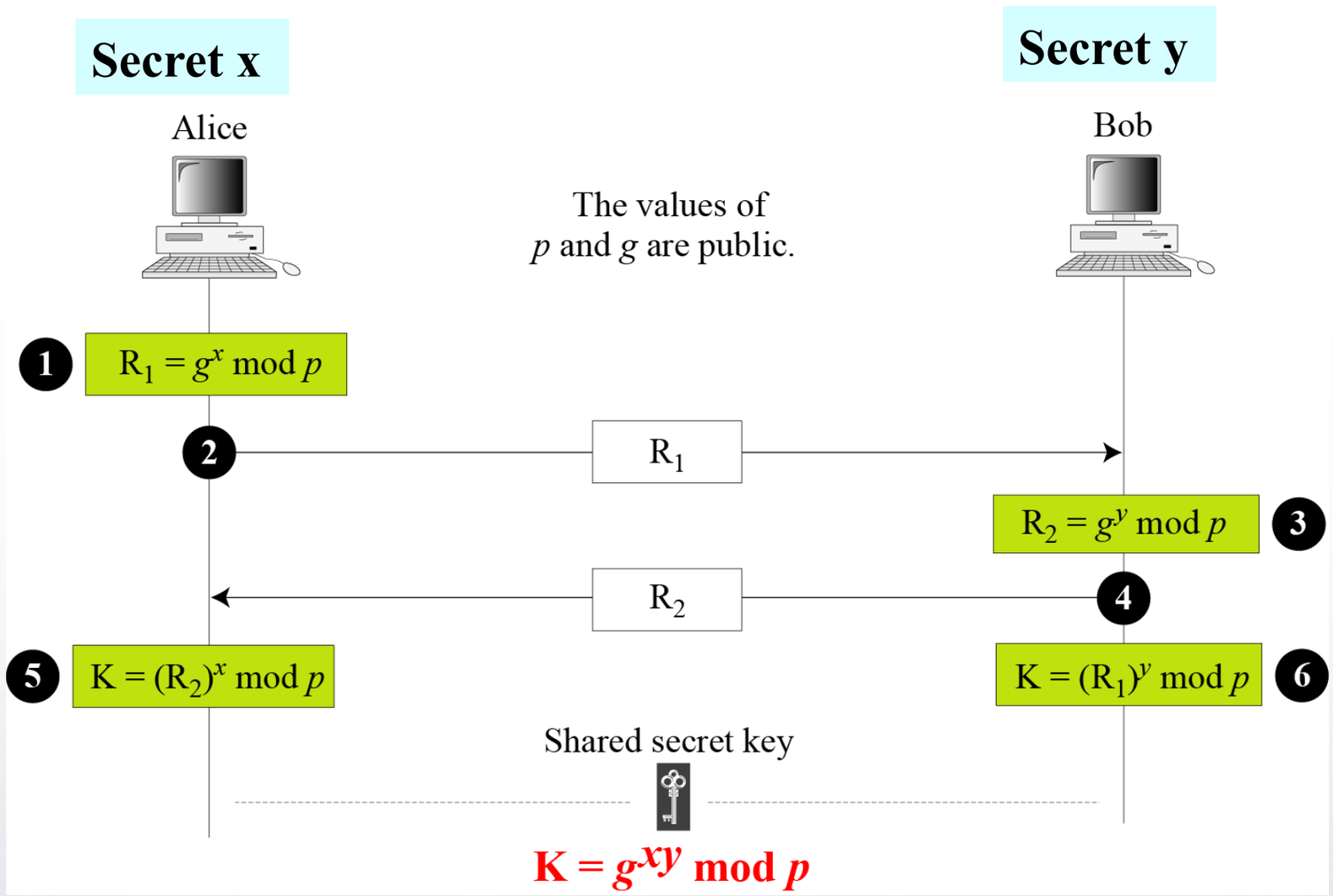
Optimal asymmetric encryption padding

- Pad the plaintext to make m-bit message M, if M is less than m-bit
- Choose a random number r of k-bits. (used only once)
- Use one-way function G that inputs r-bit integer and outputs m-bit integer. This is the mask.
- P1 = M $\oplus$ G(r)
- P2 = H(P1) $\oplus$ r, function H inputs m-bit and outputs k-bit
- C = E(P1 || P2). Use RSA encryption here.

Diffie-Hellman key exchange

- The Diffie-Hellman key exchange (exponential key exchange) is a public key system that provides a mechanism for setting up a secret but unauthenticated connection between two parties. i.e. the two can securely negotiate a secret key but neither party has any real way of knowing who the other party is.
- It does not provide encryption or digital signatures.
- It provides a method by which two parties communicating over a non-secure channel can agree on a shared secret key. The point is to agree on a key that two parties can use for a symmetric encryption, in such a way that an eavesdropper cannot obtain the key.
- Its security is based on the discrete logarithm problem.

Diffie-Hellman key exchange



Source: Cryptography and Network Security,

Diffie-Hellman key exchange

- Alice and Bob agree on a prime number p and a base g
- Alice chooses a secret number x, and sends Bob (g^x mod p)
- Bob chooses a secret number y, and sends Alice (g^y mod p)
- Alice computes k= (g^y mod p)^x mod p)
- Bob computes k=(g^x mod p)^y mod p)
- Both Alice and Bob can use this number as their key. Notice that p and g need not be protected.

Diffie-Hellman key exchange

- The symmetric (shared) key in the Diffie-Hellman method is K = g^{xy} mod p.
- An example: uses small numbers, but note that in a real situation, the numbers are very large. Assume that g = 7 and p = 23. The steps are as follows:
  - Alice chooses x = 3 and calculates R1 = 7^3 mod 23 = 21.
  - Bob chooses y = 6 and calculates R2 = 7^6 mod 23 = 4.
  - Alice sends the number 21 to Bob.
  - Bob calculates the symmetric key K = 216 mod 23 = 18.
  - The value of K is the same for both Alice and Bob;
  - g^{xy} mod p = 718 mod 35 = 18.

Diffie-Hellman key exchange

- Another example. We used a program to create a random integer of 512 bits (the ideal is 1024 bits). The integer p is a 159-digit number. We also choose g, x, and y as shown below:

p	7646242985634935721824937659550305074763380967269497489235737728609252356666607554236374233096611800333381061947301309504147387009991780436548785807987581
g	2
x	557
y	273

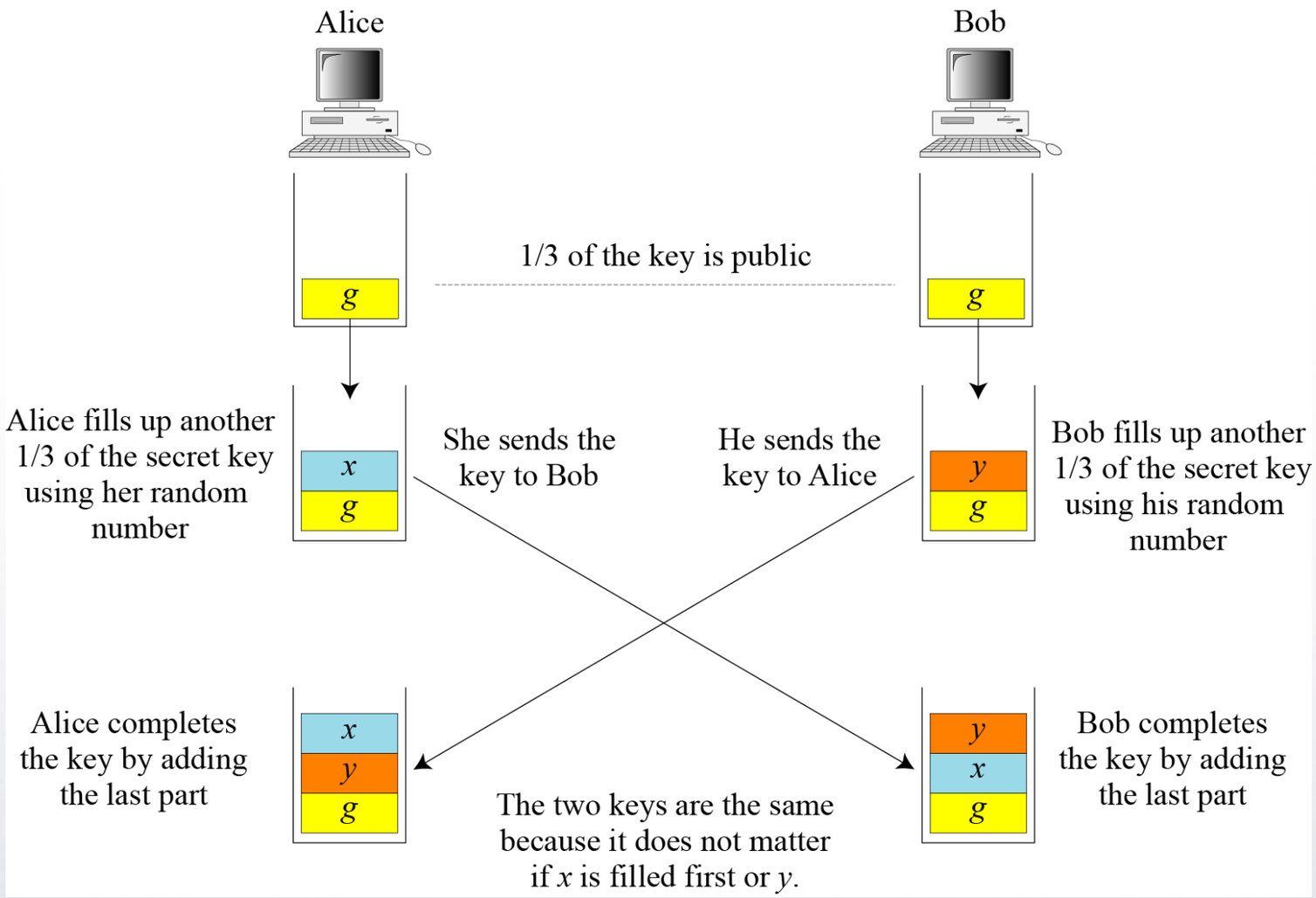
Diffie-Hellman key exchange

- The following shows the values of R1, R2, and K.

R_1	844920284205665505216172947491035094143433698520012660862863631067673 619959280828586700802131859290945140217500319973312945836083821943065 966020157955354
R_2	435262838709200379470747114895581627636389116262115557975123379218566 310011435718208390040181876486841753831165342691630263421106721508589 625520128859414 3
K	155638000664522905962258275232707652732180469444236785203204001464065 500887936651204257426776608327911017153038674561252213151610976584200 120408643361774 0

Diffie-Hellman key exchange

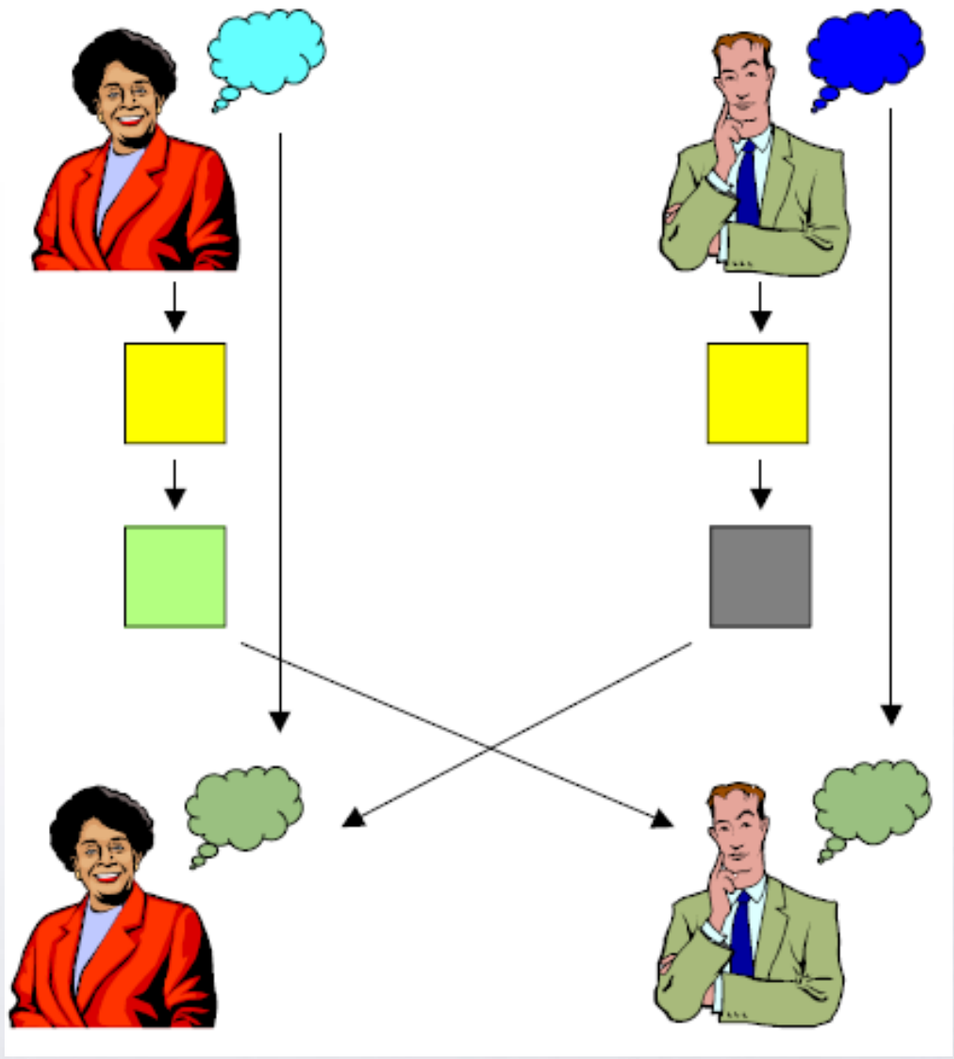
- The Diffie-Hellman idea behind



Source: Cryptography and Network Security,

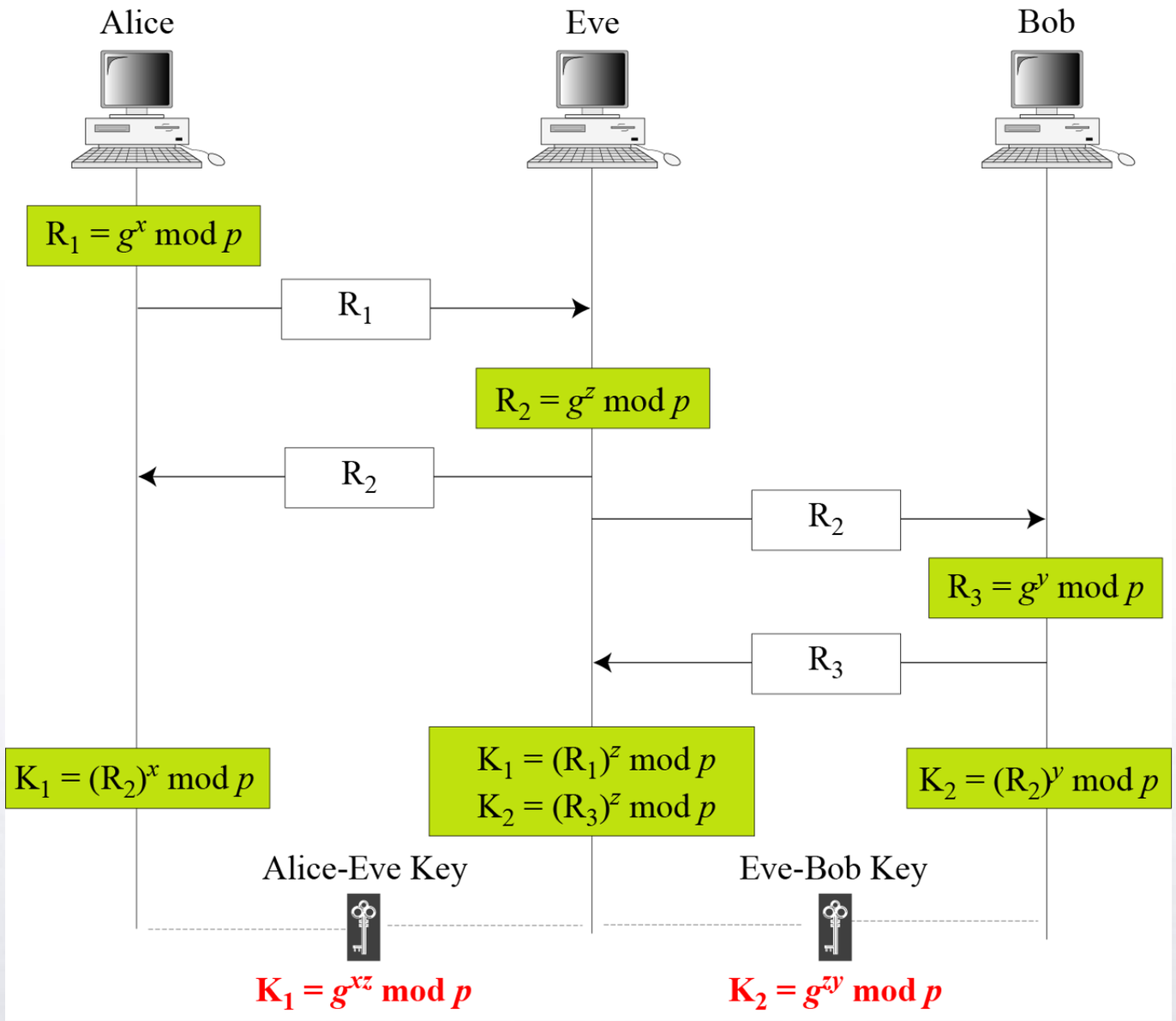
Diffie-Hellman key exchange

- Alice and Bob each think of a secret color (known only to them)
- They mix their color with yellow (agreed upon openly ahead of time) and exchange.
- They mix their color with what they've received.
- Both have the same color but observer cannot duplicate.



Source: Cryptography and Network Security,

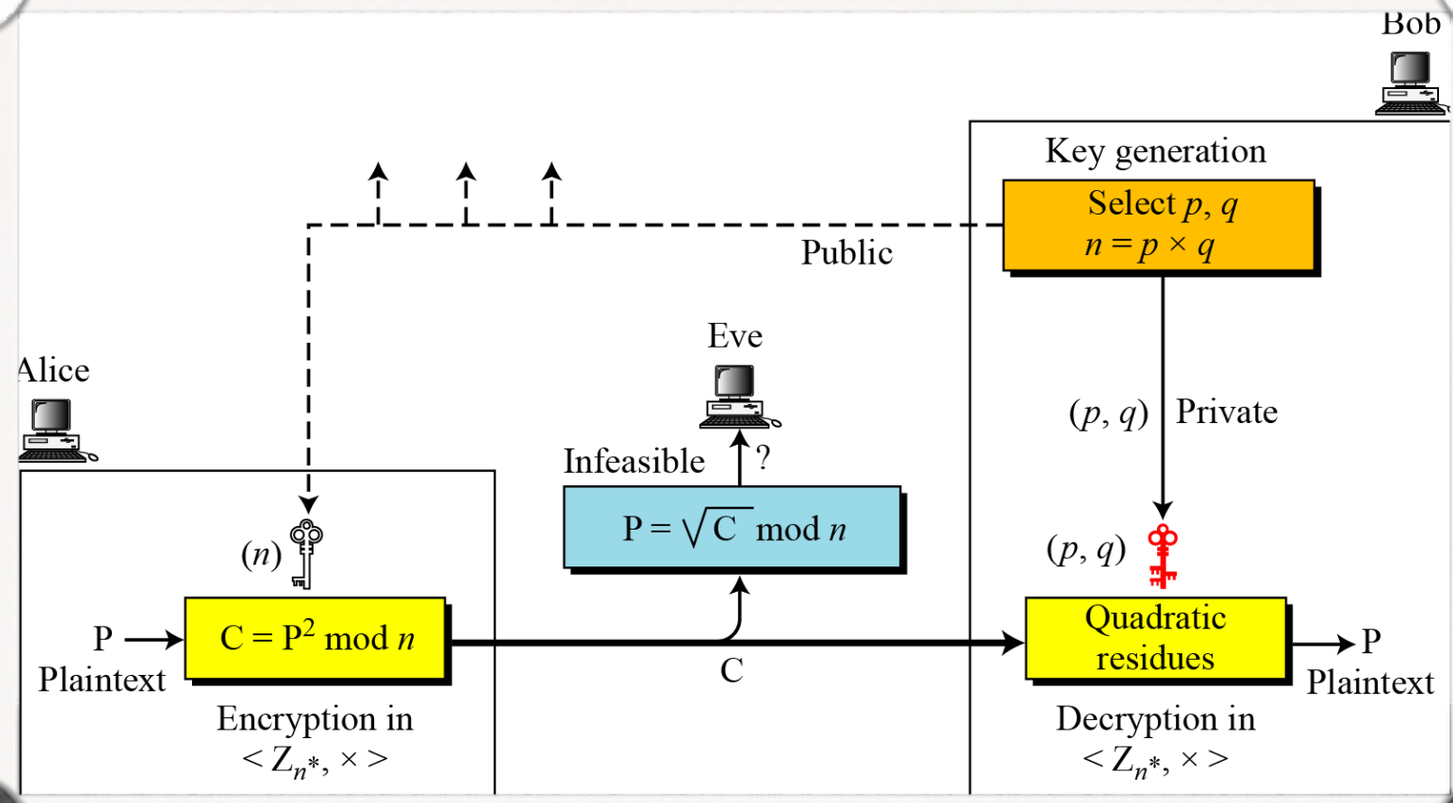
Man in the middle attack



Source: Cryptography and Network Security,

Other Cryptosystems

- The Rabin cryptosystem can be thought of as an RSA cryptosystem in which the value of e and d are fixed.
- e = 2 and d = ½
- The encryption is C ≡ P^2 (mod n) and the decryption is P ≡ $C^{1/2}$ (mod n).



Other Cryptosystems

- ElGamal Algorithm
- Is an asymmetric key encryption algorithm for public-key cryptography, which is based on the Diffie–Hellman key exchange.
- It was described by Taher Elgamal in 1984.
- ElGamal encryption is used in the free GNU Privacy Guard software, recent versions of PGP, and other cryptosystems.
- The Digital Signature Algorithm is a variant of the ElGamal signature scheme, which should not be confused with ElGamal encryption.
- ElGamal encryption can be defined over any cyclic group G. Its security depends upon the difficulty of a certain problem in G related to computing discrete logarithms

Other Cryptosystems

- Bob generates public and private keys as follows:
- He picks a large random prime p
- He finds a generator g mod p (i.e. g^x mod p gives a different answer for every value of x, which means that g^{p-1} mod p is the first time the answer is 1).
- He picks a random number a between 1 and p-1.
- He computes $y = g^a \bmod p$
- The public key is (p, g, y)
- The private key is a

Other Cryptosystems

- If Alice wants to send Bob a message, she looks up Bob's public key (p, g, y) and breaks the message up into blocks with each block less than p. Then for each message block m she takes the following steps:
- She generates a random number k between 1 and p-1.
- She computes $r = g^k \bmod p$

  $x = y^k \bmod p$, $c = (m * x) \bmod p$
- She sends Bob the values r and c.

Other Cryptosystems

- Bob receives the ciphertext (r,c) from Alice. He decrypts it as follows:
- He computes $r^a \bmod p = x$

  $\{r^a = (g^k)^a = g^{ka} = g^{ak} = (g^a)^k = y^k = x\}$
- Now he can solve $c = (m * x) \bmod p$ to find the value of m.
- Only Bob can do this because only Bob knows the value of the private key a.

Summary

- Public-key cryptography
- RSA algorithm, implementation, security
- The Diffie-Hellman