,,,,
The University of Sydney
School of Computer Science
Dr Suranga Seneviratne
Senior Lecturer - Security

THE UNIVERSITY OF SYDNEY

Asymmetric Cryptography

Recommended Reading

Cryptography and Network Security (7^{th} Edition - William Stallings)

Chapter 9 – Public-key Cryptography and RSA
Chapter 10 - Other Public-key Cryptosystems.

1 Asymmetric/Public-key cryptography

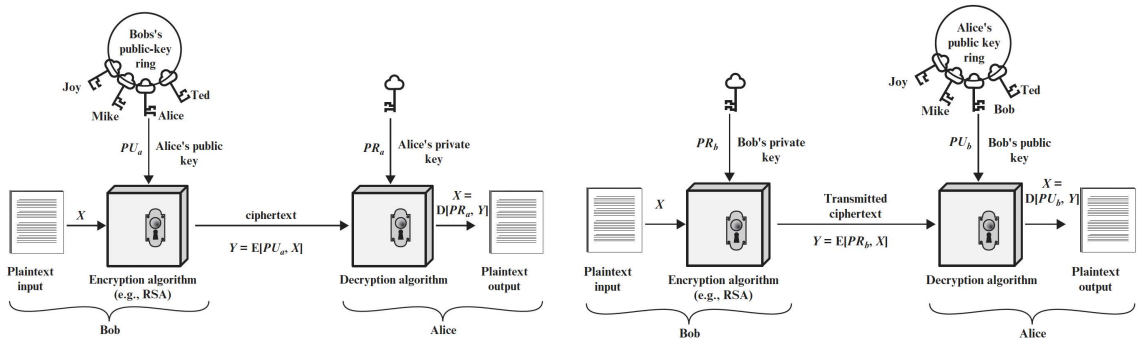
Symmetric cryptography uses the same shared key for both encryption and decryption relying on the principles of substitution (confusion) and permutation (diffusion) to secure data. In contrast, public-key cryptography utilizes two distinct keys for encryption and decryption (hence also known as asymmetric cryptography), relying on mathematical functions with specific properties beyond substitution and permutation. Asymmetric algorithms have the following important characteristics.

- It is computationally infeasible to determine the decryption key given only knowledge of the cryptographic algorithm and the encryption key.
- In some algorithms like RSA, from a mathematical point of view, either of the two related keys can be used for encryption while the other can be used for decryption.

1.1 Public-key Cryptosystems

The essential steps of using public-key cryptography are as follows.

1. Each user gets a pair of keys (private key and public key)
2. Each user places one key (public key) as a publicly accessible file. The companion key (private key) is kept confidential.
3. If Bob wishes to send a **confidential message** to Alice, Bob encrypts the message with Alice's public key before sending it, and upon receiving the message, Alice decrypts it using her private key as shown in Figure 1a. Since only Alice has access to her private key, only Alice can decrypt this message; therefore, confidentiality is achieved.
4. If Bob wants to send a message to Alice guaranteeing that the message was sent by him (**origin authentication**), Bob encrypts the message with his own private key as shown



(a) Confidentiality (b) Origin Authentication

Figure 1: Public-key Cryptography
Source: Cryptography and Network Security (Seventh Edition - William Stallings)

in Figure 1b. Upon receiving the message Alice can decrypt it using Bob's public key. **Digital signatures**; a key application of public-key cryptography follows this approach. In this case, not only Alcie but anyone who has access to Bob's public key can verify the origin of the message.

It is important to note that we refer to the conceptual usage of keys here in the confidentiality and origin authentication settings. In practice, we don't simultaneously use the same key pair for both purposes.
Overall, in public key cryptography, all participants in the system have access to all other participants' public keys, while each participant keeps the private keys confidential. Table 1 summarizes the applications of public-key cryptosystems.

Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Elliptic Curve	Yes	Yes	Yes
Diffie-Hellman	No	No	Yes
DSS	No	Yes	No

Table 1: Applications for Public-key Cryptosystems

1.2 Requirements for Public-key Cryptography

Following conditions that should be met by algorithms that can be used in public-key cryptography.

1. It is computationally easy for a party B to generate a key pair (public key PU_b, private key PR_b).

2. It is computationally easy for a sender A, knowing the public key and the message to be encrypted, M, to generate the corresponding ciphertext:
$$C = E(PU_b, M)$$

3. It is computationally easy for the receiver B to decrypt the resulting ciphertext using the private key to recover the original message:
$$M = D(PR_b, C) = D[PR_b, E(PU_b, M)]$$

4. It is computationally infeasible for an adversary who knows the public key, PU_b, to determine the private key, PR_b.
5. It is computationally infeasible for an adversary who knows the public key, PU_b, and a ciphertext, C, to recover the original message M, without knowing the private key PR_b.

   We may introduce a sixth requirement, which, while beneficial, may not be essential for all public-key applications.
6. The two keys can be applied in either order:
$$M = D[PU_b, E(PR_b, M)] = D[PR_b, E(PU_b, M)]$$

The above requirements can be aggregated to the requirement of a **trap-door one-way function**.

One-way function: A function that maps a domain into a range such that every function value has a unique inverse, with the condition that the calculation of the function is easy, whereas the calculation of the inverse is infeasible:

$Y = f(X)$ –> Easy

$X = f^{-1}(Y)$ –>Infeasible

Trap-door one-way function: A function that is relatively easy to compute in one direction but computationally difficult to invert without the knowledge of additional information called the "*trapdoor*".

$Y = f_k(X)$ –> Easy if k and X are known

$X = f_k^{-1}(Y)$ –> Easy if k and Y are known

$X = f_k^{-1}(Y)$ –>Infeasible if k is unknown

Classic trap-door candidate problems:

1. **Discrete logarithms in modular arithmetic**
   It is computationally infeasible to compute the discrete logarithm modulo p for certain p. (Diffie and Hellman, 1976)

   Discrete Logarithm Problem (DLP): Find x, given p (a prime number), g (a primitive root of p), and y satisfying the equation,
   $$y = g^x \bmod p$$
2. **RSA problem**
   It is computationally impossible to compute the e^{th} root of an integer modulo n, for certain n.

RSA Problem: Find m, given known values c, e, and n satisfying the equation,

$$c = m^e \bmod n$$

Both the Discrete Logarithm problem and the RSA problem have the property that 'computationally infeasible' becomes 'easy to compute' with additional information.

1.3 RSA Algorithm

The Rivest-Shamir-Adleman (RSA) [2] scheme was developed by Ron Rivest, Adi Shamir, and Len Adleman at MIT in 1977 and is the most widely accepted and implemented general-purpose approach to public-key encryption. RSA algorithm encrypts plaintext in blocks with each block having a binary value less than some number n.

RSA algorithm generates keys (values of e, d, and n) as follows.

Key Generation by Sender

Select p, q	p and q both prime, $p \neq q$
Calculate $n = p$ x q	
Calculate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e) = 1; 1 < e < \phi(n)$
Calculate d	$d \equiv e^{-1}(\bmod \phi(n))$
Public Key: $PU = \{e, n\}$	
Private Key: $PR = \{d, n\}$	

Note that in the above key generation process, $\phi(x)$ denotes *Euler's phi function* that counts elements in $\mathbb{Z}_n^*$.

Encryption by Sender

Plaintext: $M < n$
Ciphertext: $C = M^e \bmod n$

Note that RSA encryption is defined as a function from $\mathbb{Z}/n\mathbb{Z}$ to $\mathbb{Z}/n\mathbb{Z}$. The intuition behind this choice is that, as exponentiation in $\mathbb{Z}/n\mathbb{Z}$ is fast to compute, computing $C = M^e \bmod n$ is very easy (satisfies the second requirement above for public-key encryption). However, getting $\sqrt[e]{C}$ to get the original M (decryption without knowing d) is believed to be **infeasibly hard**.

Decryption by Receiver

Ciphertext: C
Plaintext: $M = C^d \bmod n$

RSA Algorithm - Example

Key generation:
Select two prime numbers as $p = 17$ and $q = 11$.
Obtain $n = pq = 17 * 11 = 187$
Calculate $\phi(n) = (p-1)(q-1) = 16 * 10 = 160$
Select e such that e is coprime to $\phi(n) = 160$ and $e < \phi(n) \rightarrow e = 7$
Determine d such that $d * 7 \equiv 1 \bmod 160$ and $d < 160$.
$23 * 7 = 161 = (1 * 160) + 1 \rightarrow d = 23$

$PU = \{7, 187\}$, $PR = \{23, 187\}$

Encryption:
Encrypt plaintext $M = 88$.
$C = 88^7 \bmod 187 = 11$

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

It should be noted that while encrypting numerical values with RSA is straightforward, non-numerical characters must be mapped to numerical values before encryption.

1.3.1 Security of RSA

The security of RSA depends on the following:

- Getting $\sqrt[e]{C}$ to get the original M is believed to be **infeasibly hard** unless d is known.
- Factoring $n = pq$, for very large p and q, is computationally infeasible. Otherwise, d can be easily calculated.

The defense against brute forcing d is to use a large key space for d. The larger the number of bits in d, the more secure the system becomes. However, it is important to recognize that increasing the key size can also slow down the system due to the more complex calculations required for key generation, encryption, and decryption.

To make factoring $n = pq$ computationally infeasible,

- p and q, p must be **very large**.
- p and q, p must not be too close together

There are many other factors that contribute to the security of a key. Consequently, it is not recommended to implement RSA yourself for use in production environments.

Five possible approaches to attacking RSA are,

- **Brute force**: This involves trying all possible private keys – feasible only against smaller keys.
- **Mathematical attacks**: There are several approaches, all equivalent in effort to factoring the product of two primes.
- **Timing attacks**: These depend on the running time of the decryption algorithm.
- **Hardware fault-based attack**: This involves inducing hardware faults in the processor that is generating digital signatures.
- **Chosen ciphertext attacks**: This type of attack exploits properties of the RSA algorithm (e.g. Malleability).

Malleability of RSA

First, Eve listens for a ciphertext that she wants to crack:

$$c = m^e \mod n$$

Next, she takes this ciphertext, multiply it by a random value raised to the power of Bob's e value) and get Bob to decrypt it:

$$c' = c \times r^e \mod n$$

If Eve can determine the decrypted value for this ciphertext, she can determine the original message as:

$$(c')^d = (c \times r^e)^d = (m^e \times r^e)^d = m^{e \times d} \times r^{e \times d} = m \times r$$

since $(m^e)^d \mod n$ must equal $m^1 \mod n$.

Eve obtains the original plaintext by dividing the resulting plaintext by r.

Note the Chosen-Ciphertext-Attack (CCA) assumption here. We assume that the attacker has the ability to get c' decrypted.

RSA and Padding: The vanilla form of RSA is not secure. For example, it is vulnerable to CCA because of malleability. Basic RSA encryption does not provide semantic security due to its lack of randomness. Since RSA is deterministic—meaning the same plaintext encrypted with the same public key will always produce the same ciphertext—it is vulnerable to attacks like Chosen Plaintext Attacks (CPA). In such attacks, an adversary can encrypt likely messages and compare the results to a given ciphertext to infer the original plaintext. To achieve semantic security, modern RSA implementations incorporate padding schemes that introduce randomness, ensuring that the same plaintext encrypted multiple times results in different ciphertexts. Today, we use OAEP (Optimal Asymmetric Encryption Padding) for this.

1.4 Hybrid Encryption

Public-key cryptography has the following limitations.

- Public-key cryptography is slow, particularly when long keys are used to enhance security

- It requires a mechanism to map numbers to characters for practical use

To address these limitations, most applications employ hybrid encryption. In this approach, public-key encryption like RSA is used to quickly establish a symmetric key, and subsequent communications are secured using symmetric key encryption.

2 Diffie-Hellman Key Exchange

Diffie–Hellman algorithm is the first published public-key algorithm published by Diffie and Hellman [1] in 1976 and its best-known application is secure key exchange which can subsequently be used for symmetric-key encryption. Other applications of the Diffie–Hellman algorithm include *ElGamal encryption/decryption* and *Digital Signature Standard (DSA/DSS).*

The security of the Diffie–Hellman algorithm relies on the difficulty of computing discrete logarithms. Specifically, it is considered **computationally infeasible to compute a discrete logarithm over a finite field**, provided the prime number used to define the field is sufficiently large. However, if additional information, such as the private key, is available, this computation becomes computationally feasible.

Recap: Discrete Logarithm Problem (DLP): Can we find x, given p (a prime number), g (a primitive root of p) and y satisfying the equation,

$$y = g^x \bmod p$$

2.1 Diffie-Hellman Algorithm

Figure 2 summarizes the Diffie-Hellman algorithm for exchanging a shared key between two users A and B who have already shared two publicly known values p (a prime number) and g (a primitive root of q). Accordingly, each user selects a random integer a and b such that they are less than p and computes $X = g^a \bmod p$ and $Y = g^b \bmod p$, respectively. The a and b are kept private (i.e., private keys of A and B), and the X and Y values are publicly exchanged by the two users(i.e., X and Y are the public keys of the two users).

After the exchange, each user can separately and independently calculate the same key; i.e., A will compute $Y^a \mod p$, and B will compute $X^b \mod p$. Both of these values will be equal and that will be the shared key; $K = Y^a \mod p = X^b \mod p$.

$$\begin{aligned} K &= (Y)^a \mod p \\ &= (g^b \mod p)^a \mod p \\ &= (g^{ba}) \mod p \\ &= (g^a \mod p)^b \mod p \\ &= (X)^b \mod p \end{aligned}$$

Through this process, the two parties have exchanged a secret value K which is typically used as a shared key.

An adversary who can only passively observe the key exchange will only know the values p, g, X, and Y and will be forced to take a discrete logarithm to determine one of the private keys

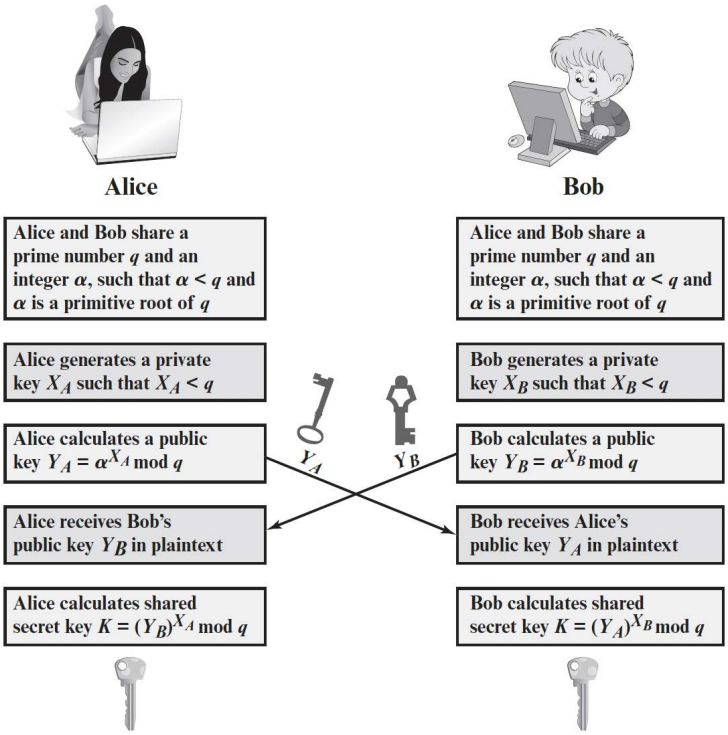


Figure 2: Diffie-Hellman Algorithm
Source: Cryptography and Network Security (Seventh Edition - William Stallings)

a or b to calculate the shared key K.

For instance, if calculating b, the adversary needs to compute b such that $Y = g^b \mod p$, which is solving the DLP. For large primes, this task is computationally infeasible. Hence, the shared secret key K is considered secure from a passive observer.

2.2 Person-in-the-middle-attack

While the Diffie-Hellman algorithm in Figure 2 is secure against passive eavesdroppers, it is vulnerable to person-in-the-middle-attacks as shown in Figure 3. Accordingly, the adversary first observes the publicly shared values of p and g, and then generates two separate sets of private and public key pairs. The attacker then positions themselves between the two communicating users, *Alice* and *Bob*, intercepting and replacing the messages transmitted during the key exchange. By doing this, the attacker performs two independent key exchanges: one with *Alice* and another with *Bob*. At the end of this process, the attacker establishes separate secret keys K_2 and K_1 with *Alice* and *Bob* respectively and can use these keys to perform a person-in-the-middle-attack (intercept and replace original messages) on future communications between Alice and Bob.

Person-in-the-middle attacks in Diffie-Hellman key exchange can be overcome by using digital signatures, where each party signs their public key with their private key, allowing the other party to verify the authenticity of the key using the corresponding public key, ensuring that the exchanged keys have not been tampered with by an attacker.

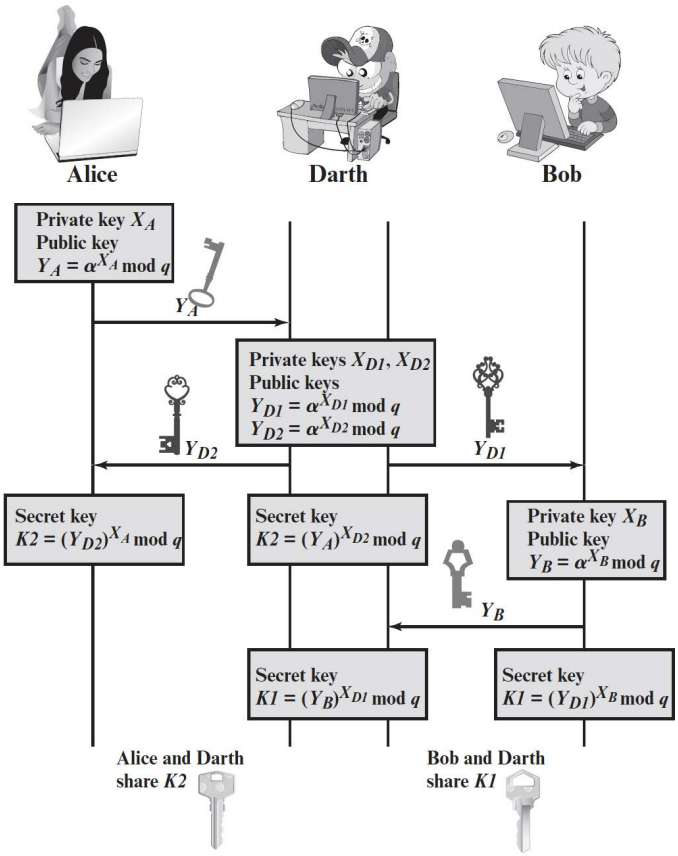


Figure 3: Person-in-the-middle-attack on Diffie-Hellman key exchange

2.3 Post-Quantum Cryptography (PQC)

PQC focuses on developing algorithms that remain secure even in the presence of quantum computational power. Some of the key concepts of PQC include,

- **Lattice-Based Cryptography** is based on the hardness of problems related to lattice structures in high-dimensional spaces.
- **Code-Based Cryptography** relies on the hardness of decoding random linear codes.
- **Hash-Based Cryptography** leverages the security properties of hash functions for cryptographic tasks.

3 Practice Quiz

Question 1: Select all that is TRUE for the RSA algorithm.

a) One factor that makes RSA secure is the difficulty in prime number factorisation.
b) RSA is still considered secure irrespective of the key size.
c) Padding is required to make RSA secure.
d) RSA is a symmetric key encryption algorithm
e) You can choose any two prime numbers to construct an RSA scheme

Explanation: The security of RSA lies on two factors. The first is the difficulty of prime factorization. The second is the difficulty of finding the e_{th} root of a number under modulo n. Therefore **a)** is TRUE.

b) is FALSE. These days we consider only key sizes above 2048 bits as secure for RSA.

c) is TRUE. A semantically secure cryptosystem is one in which only a small amount of information about the plaintext can be extracted from the ciphertext. Textbook RSA doesn't have any randomness. Therefore, it doesn't have any semantic security. As a result, we add padding.

d) in FALSE. RSA is an asymmetric cryptosystem.

e) in FALSE. We can't select any two prime numbers. For starters, we can't select two small prime numbers. Even for large prime numbers, there are restrictions. We can't select prime numbers that are close to each other. Then the factorization problem becomes easier.

Question 2: For $p = 11$ and $q = 17$ and choose $e = 7$. Apply the RSA algorithm to find the ciphertext when the plaintext message is 88.

a) 23
b) 64
c) 11
d) 54

In the RSA notation, we have been given p=11, q=17 and e=7. We can find,

$n = pq = 187$

$\phi(n) = (p-1)(q-1) = 160$ (As e is given we don't need it. But note that e is less than $\phi(n)$.)

$C = M^e \mod n = 88^7 \mod 187 = 11$

Question 3: In an RSA system the public key of a given user is e = 31, n = 3599. What is the private key of this user?

a) 3031
b) 2412
c) 2432
d) 1023

Explanation: In the RSA notation, we have been given e=131 and n=3599. We can find

p= 59 and q = 61 (By finding the prime factors of 3599) and n = pq = 3599

$\phi(n) = (p-1)(q-1) = 58 \times 60 = 3480$

We know the private and the public key have the property.

$ed \equiv 1 \mod (\phi(n))$ i.e. $31d \equiv 1 \mod (3480)$

Check them one by one.

31 x 3031 mod (3480) = 1. Therefore, 3031 is the correct answer. You can verify the rest.

E.g. 31 x 2412 mod (3480) = 1692.

Question 4: The ..... protocol enables two users to establish a secret key using a public-key scheme based on discrete logarithms.

a) Micali-Schnorr
b) Elgamal-Fraiser
c) Diffie-Hellman
d) Miller-Rabin

Explanation: We learned only one public key cryptography-based key exchange scheme in the class. That is the Diffe-hellman key exchange. The rest are either made up or not related. For example, Micali-Schnorr is a random number generator algorithm, and Miller-Rabin is a primality test.

Question 5: The basic Diffe-Hellamn key exchange protocol is vulnerable to a ..... attack because it does not authenticate the participants.
a) one-way function
b) time complexit
c) chosen ciphertext
d) person-in-the-middle

Explanation: An attacker can store and replay messages in the Diffe-Hellman key exchange and conduct an active person-in-the-middle attack. Diffe-Hellman key exchange is secure only against a passive attacker. For protection against active attackers, a modified version of the process is required, called as the signed Diffe-Hellman key exchange.

Question 6: The Diffie-Hellman algorithm depends on the difficulty of computing discrete logarithms for its effectiveness. TRUE or FALSE.
a) TRUE
b) FALSE

Explanation: There are two key mathematical problems that have trapdoor properties. The discrete logarithm problem and the RSA problem. DH key exchange relies on the discrete logarithm problem.

Question 7: Read this article on how the advances in quantum computing can affect symmetric and asymmetric cryptography differently.

`https://builtin.com/cybersecurity/post-quantum-cryptography`

Based on your reading, which cryptographic scheme will become more vulnerable when quantum computing becomes feasible?

a) Symmetric cryptography
b) Asymmetric cryptography

Explanation: Once quantum cryptography becomes available, some of the previously infeasible problems become feasible. As a result, asymmetric cryptography becomes vulnerable.

Question 8: Asymmetric and symmetric encryption have their own advantages and disadvantages. For example, asymmetric cryptography is slower and not suitable for real-time communications. On the other hand, symmetric cryptography is faster and more suitable for real-time data encryption and bulk data encryption. However, key distribution is an issue with symmetric encryption. As a result, most of the applications, especially in communications settings, use hybrid cryptography. Watch the following video on hybrid cryptography and decide whether the following statement is TRUE or FALSE.

`https://www.youtube.com/watch?v=VPvZbMXfv_0`

"Hybrid cryptography uses asymmetric cryptography to establish a symmetric key between two parties that can be used for subsequent communications." TRUE or FALSE

a) TRUE
b) FALSE

Explanation: The answer is TRUE. Hybrid cryptography assumes the receiver's public key is already shared with the sender. The sender generates a session key (a symmetric key), encrypts it with the receiver's public key, and sends it to the receiver. The receiver can decrypt the message using their private key and recover the session key. Now, both parties have access to the shared session key, which can be used for any subsequent communication.

Question 9: A considerably larger key size can be used for ECC compared to RSA. TRUE or FALSE.

a) TRUE
b) FALSE

Explanation: The statement is False. In general, ECC key sizes are much shorter than RSA, which is an advantage of ECC. This is because the elliptic curve logarithm problem is more difficult than the RSA problem.

Question 10: Consider the elliptic curve $E_7(2,1)$; that is, the curve is defined by $y^2 = x^3+2x+1$ with a modulus of $p = 7$. Which of the following is NOT a point in $E_7(2,1)$? Not examined - for information only

a) (0,1)
b) (1,5)
c) (3,4)

d) (1,2)

The equation we are after is $y^2 \mod 7 = x^3 + 2x + 1 \mod 7$. We check each answer separately as shown in the following table.

x	x3+2 x+1	(x3+2 x+1)m od 7	y	y^2	y^2mo d7
0	1	1	1	1	1
1	4	4	5	25	4
3	34	6	4	16	2
1	1	4	2	4	4

Figure 4: Points on an Elliptic Curve

Only (3,4) is not satisfying the equation.

References

[1] W. Diffie and M. E. Hellman, "Multiuser cryptographic techniques," in *Proceedings of the June 7-10, 1976, national computer conference and exposition*, 1976, pp. 109–112.

[2] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.