Lecture note 4: Network Security Principles

Jialong Zhou, Zhiyuan Sun

April 17, 2023

In this lecture, we mainly discuss the content of network security principles. Firstly we recall the public key encryption, RSA algorithm, and digital signature. Then we elaborate on the authentication key exchange, focusing on the explanations of three protocols. Finally we introduce the public key infrastructure and certification authorities.

1 Recall RSA and Digital Signature

1.1 Public key encryption

We first recall the RSA. Starting with the public key encryption, G is a generator, and we have two parties. They have A and B respectively, where $A \leftarrow g^a$, $B \leftarrow g^b$. They pass A and B to each other, so they can have shared secret key, which is K shown in Figure 1 [CHK03].

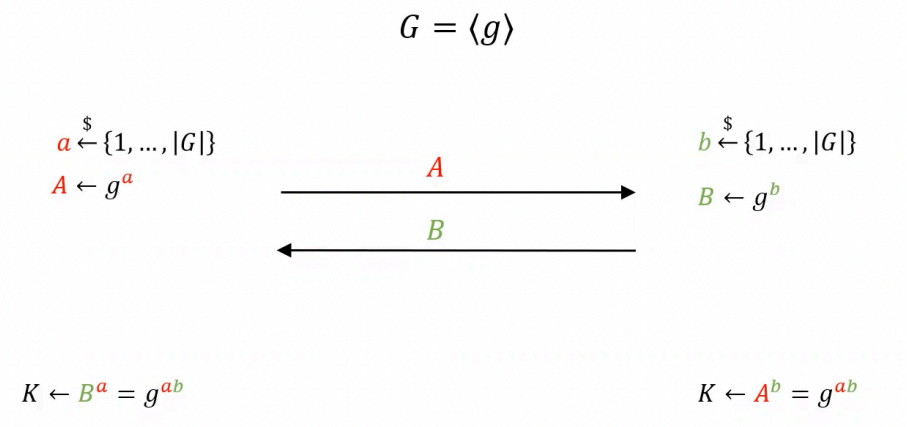

Figure 1: Diffie-Hellman

Public key encryption is a type of encryption that uses two keys: a public key and a private key, to secure communications. The public key is used to encrypt the data, while the private key is used to decrypt it.

In this encryption scheme, the public key is distributed widely to anyone who wants to communicate securely with the owner of the private key. The private key is kept secret and

only accessible to the owner.

When a sender wants to send a message to the receiver, it encrypts the message using the receiver's public key. This ensures that only the receiver, who holds the corresponding private key, can decrypt the message and read its contents. This is a secure way to communicate over an insecure channel, such as the Internet.

WhatsApp is a good example. Different from WeChat, WhatsApp is an app in which even the company does not know what you arer talking about with others. The process is using the principle of Diffie-Hellman. Since the server has only A_0 and B_0, it can not get the messages between two users. The users have secret key, so they can know the messages.

Then we recall the ElGamal. Part of ElGamal is Diffie-Hellman. To be more specific, The generator is g, the secret key is b, and the public key is B. For encryption, since it knows the public key and it owns has A, it can get K and C. For decryption, $M = C/K$". The difference with the original Diffie-Hellman key exchange is C, which is $K \cdot M$. ElGamal is IND-CPA under DDH assumption. The whole process is shown in Figure 2.

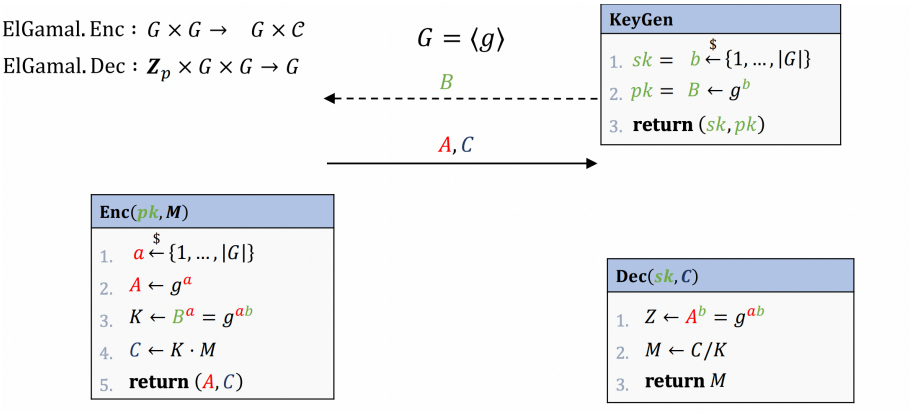

Figure 2: ElGamal

Public key encryption is widely used for securing online transactions, such as e-commerce, banking, and email. It is also used for secure file sharing, remote access, and virtual private networks [BDPR98].

Overall, public key encryption is a powerful tool for ensuring secure communication and protecting sensitive information. By using two keys instead of just one, it provides an extra layer of security that makes it difficult for hackers and cybercriminals to intercept and access confidential data.

1.2 RSA

In the last class, we also learn RSA. The RSA algorithm works on the principle of prime factorization. It generates a pair of keys - a public key and a private key - using large prime

numbers. The public key is used to encrypt data, while the private key is used to decrypt it. The keys are related mathematically, but it is computationally infeasible to derive the private key from the public key.

Definition 1. *The RSA encryption scheme:*

$$c = E(m) = m^e (mod \quad n), \tag{1}$$

We introduce the Euler's Theorem. Euler's Theorem, named after the mathematician Leonhard Euler. This theorem is a fundamental result in number theory and has important applications in cryptography, as it is used in the RSA algorithm for public-key encryption.

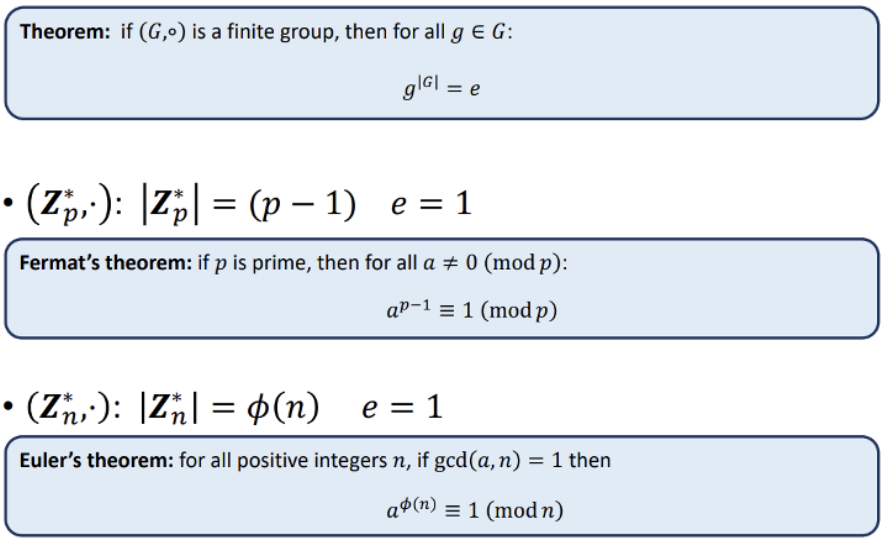

Figure 3: Euler's Theorem, what is needed in RSA Enc

From Figure 3, we must know: for all positive integers n, if $gcd(a, n) = 1$ then $a^{\phi(n)} \equiv 1 \pmod n$. This theorem is useful in RSA.

For RSA Enc, we first have M. Through the encryption, we have $C \leftarrow M^e \pmod n$. Through the decryption, we have $M \leftarrow C^d \pmod n$. Finally, what we actually have is $M \leftarrow M^{ed} \pmod n$. According to the Euler's Theorem, we have $a^{\phi(n)} \equiv 1 \pmod n$ and $ed = 1 \pmod{\phi(n)}$, so $M \leftarrow M^{ed} \pmod n$ is right, which means RSA Enc works.

The strength of the RSA algorithm lies in the difficulty of factoring large prime numbers. The security of the RSA algorithm is based on the fact that it is currently computationally infeasible to factor the product of two large prime numbers. This means that an attacker cannot derive the private key from the public key, making the encryption scheme secure.

RSA algorithm has various applications, including secure communication, digital signatures, and secure key exchange. It is used in various protocols such as HTTPS, SSL/TLS,

and SSH to provide secure communication over the internet.

However, the RSA algorithm is not without its limitations. The key size used in the algorithm determines the strength of the encryption. As computing power increases, the key size required to ensure security also increases. Additionally, the RSA algorithm is vulnerable to attacks such as side-channel attacks and timing attacks.

Then we recall Textbook RSA. Textbook RSA is a widely used public-key cryptosystem that uses the mathematical properties of large prime numbers for encryption and decryption. In this method, the sender uses the recipient's public key to encrypt the message, which can only be decrypted using the recipient's private key [Zho22].

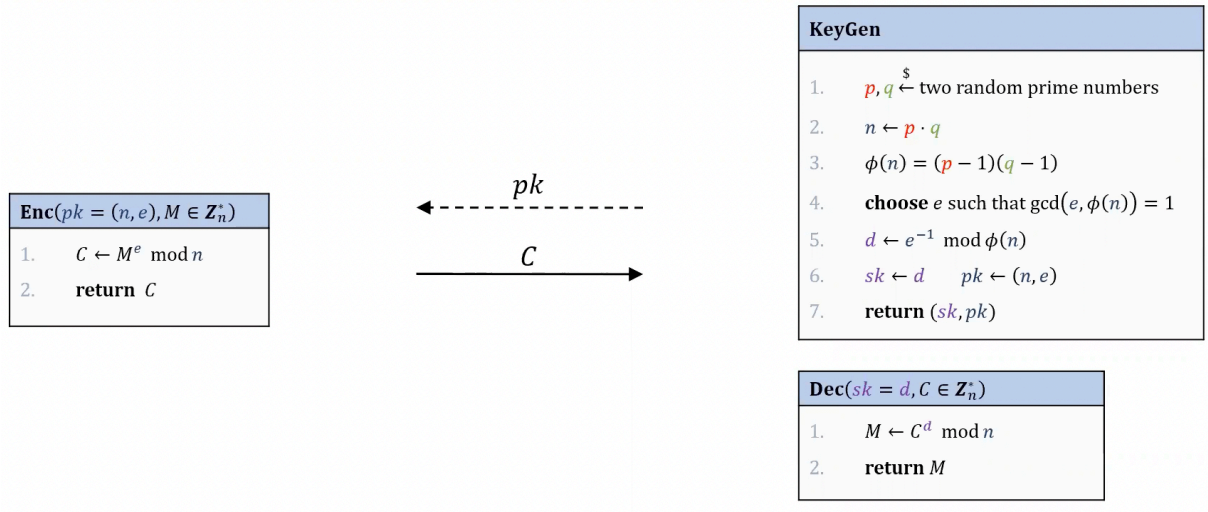

Figure 4: Textbook RSA

In Figure 4, the public key is n. e has some common choices, like 3, 17, and 65537.

Since Textbook RSA is deterministic, it can not be IND-CPA secure. Since it is not secure, Textbook RSA is not used much in practice anymore. In practice, RSA can be used with padding message with random data before applying RSA function, or using PKCS#1v1.5 and RSA-OAEP. Figure 5 shows one method.

1.3 Digital Signature

We first review the definition of digital signature. A digital signature, also known as a digital signature certificate or simply a signature, is a mathematical technique used to validate the authenticity and integrity of electronic documents, messages, or software. It is essentially an electronic version of a handwritten signature that can be used to verify the identity of the sender and ensure that the data has not been tampered with during trans-

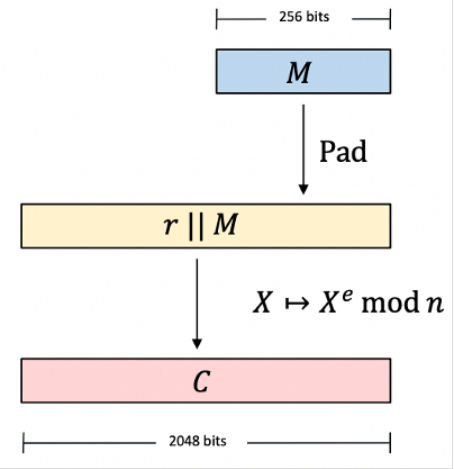

Figure 5: RSA in practice

mission [MS21, Nis92].

A digital signature is a mathematical scheme for verifying the authenticity of digital messages or documents. The sender uses their private key to encrypt the document or message, and the receiver uses the sender's public key to verify that the message was indeed sent by the claimed sender and that it has not been altered during transmission.

To be more specific, in Figure 6, we can see Alice has the secret key, and Bob has the public key, so they do not have the same shared secret key, which is different with MACs. In this situation, digital signatures can be verified by anyone with the public key.

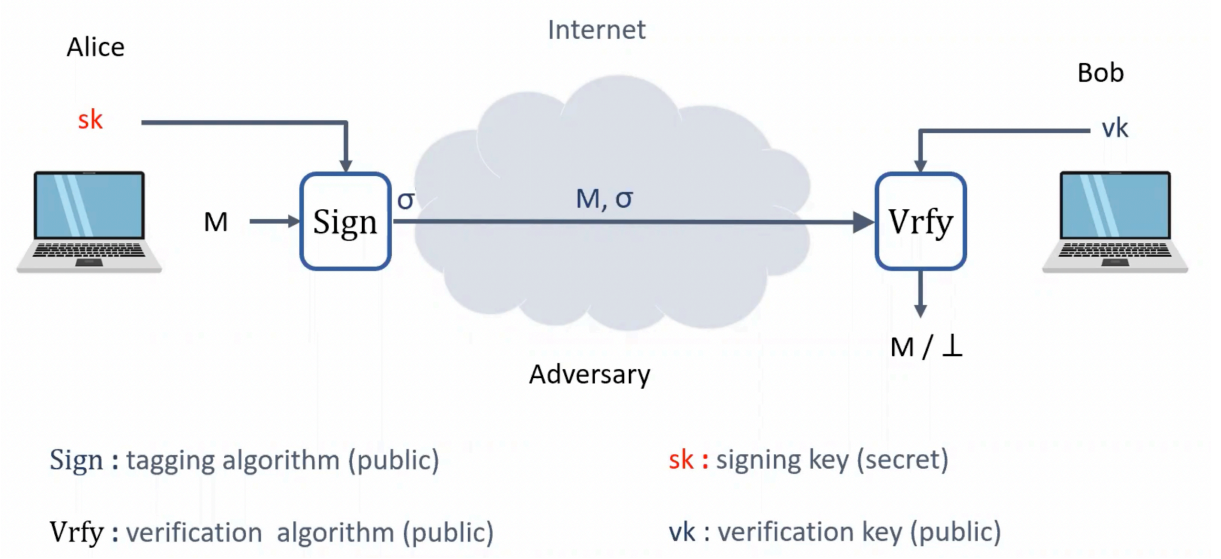

Figure 6: Digital Signatures

A digital signature scheme is a tuple of algorithms $\Sigma = (KeyGen, Sign, Vrfy)$. $KeyGen$ will give Alice the secret key, and give Bob the verified key.

The use of digital signatures provides a high level of security and trust for electronic transactions and communications, making them an essential tool in e-commerce, online banking, and other digital applications. They are also commonly used in government, legal, and healthcare settings where secure and accurate electronic document verification is critical.

Formally, a digital signature scheme is a triple of probabilistic polynomial time algorithms, $(KeyGen, Sign, Vrfy)$, satisfying:

KeyGen (key-generator) generates a public key (pk), and a corresponding private key (sk), on input 1^n, where n is the security parameter.

Sign (signing) returns a tag, (σ), on the inputs: the private key (sk), and a string (M).

Vrfy (verifying) outputs accepted or rejected on the inputs: the public key (pk), a string (M), and a tag (σ).

For correctness, S and V must satisfy

$$Pr[(pk, sk) \leftarrow KeyGen(1^n), Vrfy(pk, x, Sign(sk, M)) = accepted] = 1. \tag{2}$$

A digital signature scheme is secure if for every non-uniform probabilistic polynomial time adversary, A

$$Pr[(pk, sk) \leftarrow KeyGen(1^n), (M, \sigma) \leftarrow A^{Sign(sk,\cdot)}(pk, 1n), M \notin Q, Vrfy(pk, M, \sigma) = accepted] < negl(n), \tag{3}$$

where $A^{Sign(sk,\cdot)}$ denotes that A has access to the oracle, $Sign(sk, \cdot)$, Q denotes the set of the queries on S made by A, which knows the public key, pk, and the security parameter, n, and MQ denotes that the adversary may not directly query the string, M, on S.

Signatures must have unforgeability.If an adversary gives a new message M^* to Alice, and he gets a valid signature, we can say the adversary succeeds.

Definition 2. *The $UF - CMA -$ advantage of an adversary A is*

$$Adv_{\Sigma}^{uf-sma}(A) = Pr[Exp_{\Sigma}^{uf-cma}(A) \Rightarrow 1] \tag{4}$$

We also review RSA Signatures. Different from RSA Encryption, it is in another pattern. At this time, e is the public key and d is the secret key. Through the $Sign$, the signature of M is $\sigma \leftarrow M^d \pmod n$. Through the $Vrfy$, it will output 0 or 1 by Euler's Theorem. According to the Euler's theorem, this is right.Textbook RSA is very similar.

Textbook RSA is insecure. Since the adversary can get $\sigma_1 = M_1^d$ and $\sigma_2 = M_2^d$. Then $\sigma_1\sigma_2 = (M_1M_2)^d \pmod n$, so the adversary can get the signature of a new message, which

means Textbook RSA is insecure.

We also review RSA-FDH: Hash-then sign paradigm, which is a revision of Textbook RSA. The difference is the message M should go through the hash function H first.

Theorem 1. *For any UF-CMA adversy A against hashed RSA making q $SIGN_sk(\cdot)$ queries, there is an algorithm B solving the RSA-problem:*

$$Adv^{uf-cma}_{RSA,H}(A) <= q * Adv^{RSA}_{n,e}(B) \tag{5}$$

From the view of attack, given $\sigma_1 = H(M_1)^d$ and $\sigma_2 = H(M_2)^d$, we have $\sigma_1\sigma_2 = (H(M_1)H(M_2))^d \pmod n$. The goal of the adversary is find m such that $H(M) = H(M_1)H(M_2)$, but it is impossible due to the one-wayness of hash function H.

We learn discrete-log-based signatures, ECDSA, and Schnorr. For Schnorr, it is elegant design, and it has formal security proof (based on DLOG problem and H assumed perfect), which has been patented (expired in February 2008). Since it has been patented, it is not widely used. For (EC)DSA, it is a non-patented alternative, and derived from ElGamal-based signature scheme. It is more complicated design than Schnorr and used widely, but it do not have security proof.

In summary, a digital signature is a powerful tool that helps ensure the authenticity, integrity, and security of electronic documents and messages, providing a reliable way to verify the identity of the sender and protect against tampering and forgery.

2 Authenticated Key Exchange

In this section, we first learn Diffie-Hellman Key Exchange. We define what is active adversary. Active adversary acts like a man-in-the-middle attack. Adversary has complete control of the network. They can modify, inject and delete packets. Moreover, some Internet users are honest and some are corrupt. Corrupt users are controlled by the adversary. Key exchange with corrupt users should not affect other sessions [VOW$^+$92].

At this time, we need Authenticated Key Exchange (AKE), which secures against active adversaries. AKE protocol should allow two users to establish a shared key, and ensure that they are talking with whom they plan to talk with. Also, we have a new concept: Trusted Third Party. All AKE protocols require a TTP to certify user identities [PCD18].

The registration process is shown in Figure 7. Alice and Bank can get the secret key $Cert_{alice}$ and $Cert_{bank}$ from TTP, respectively.

We also learn the basic AKE security. Assuming Bank is not corrupt, for Alice, we need Authenticity, Secrecy, and Consistency. Authenticity means Alice's key is only shared with

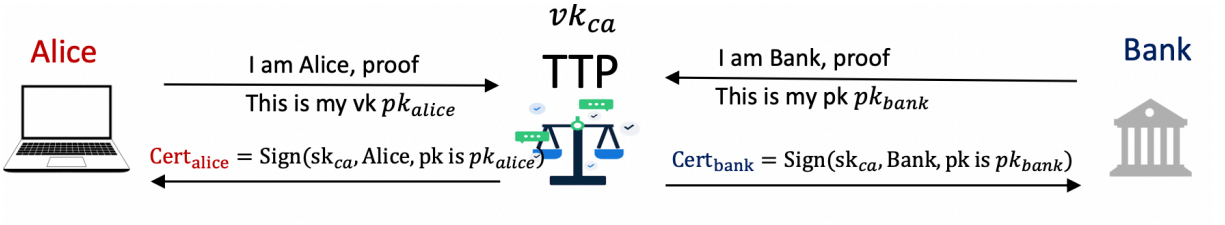

Figure 7: The Registration Process of Trusted Third Party

Bank. Secrecy means, to the adversary, Alices key K is indistinguishable from random. Consistency means if Bank completes AKE then it obtains $(K, Alice)$.

We know three levels security of AKE: static security, forward secrecy and hardware security module (HSM). Static security is the basic one. Compared with static security, forward security is if the adversary learns sk_{bank} at time T then all sessions with Bank before T remains secret.

Static security in AKE protocols refers to the property of a protocol that ensures that an adversary who has only observed the protocol execution cannot recover the secret key or impersonate one of the protocol participants. In other words, static security guarantees that the protocol remains secure even if an attacker intercepts and observes the communication between the participants.

To achieve static security, AKE protocols typically rely on the use of long-term secret keys, such as shared secrets or public keys, in combination with cryptographic primitives such as digital signatures, message authentication codes, and encryption. These primitives are carefully designed and implemented to ensure that an attacker cannot exploit any weaknesses in the protocol to compromise its security.

In summary, static security is an important property of AKE protocols that ensures that the communication between participants remains confidential, authentic, and secure, even in the face of a passive attacker. The problem of protocol 1 is no forward secrecy.

We learn One-side AKE, since only big organizations like PolyU have TTP certifications.

Then we introduce some protocols [Res18, Gol01] in section 2.1, 2.2 and 2.3.

2.1 Protocol #1: Building blocks

First we have some asumptions and definitions. Bank have $Cert_{bank}$ contians pk_{bank}.

Definition 3. *Enc_{bank}: IND-CCA secure PKE using Bank's public key. Bank keeps sk_{bank} as the secret encryption key.*

Definition 4. *$Sign_{alice}$/$Sign_{bank}$: UF-CMA secure signature of Alice/Bank.*

Here is the whole process of Protocol #1 shown in Figure 8.

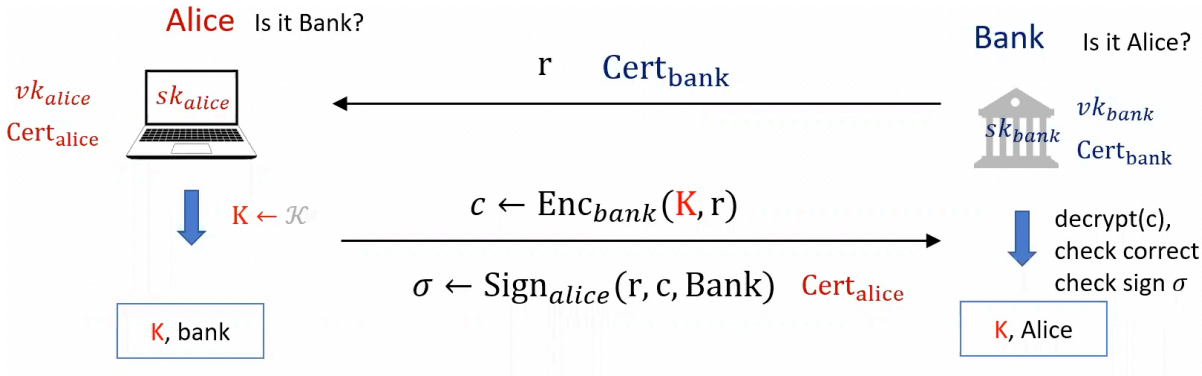

Figure 8: Protocol #1: Building blocks

Bank creates a random string r, and then it gives Alice r and $Cert_{bank}$. Alice verifies them and gives c and σ back to Bank, since she has the secret key. Bank decrypts c, and check whether σ is correct. Alice believes only Bank can decrypt K and Bank believes K come from Alice. This is the whole process.

Protocol #1 is statically secure AKE, but it is not secure. If the adversary obtains sk_{bank} one year later, the adversary can decrypt all traffic records since it has the secret key, which can get K easily.

2.2 Protocol #2: HSM Security

HSM Security is forward secrecy, and n queries to HSM should compromise at most n sessions [Wol10].

HSMs are specialized devices designed to securely store and manage digital keys and cryptographic operations. HSMs provide a high level of security by physically protecting sensitive keys and cryptographic materials from unauthorized access, tampering, and theft.

HSMs use advanced security measures such as tamper-resistant casing, secure boot process, and firmware encryption to ensure that the device itself is secure. The keys and cryptographic operations are also protected through measures such as PINs, passwords, and biometric authentication.

In addition to secure key storage and management, HSMs provide a range of cryptographic functions such as key generation, encryption, decryption, digital signatures, and key wrapping. These functions can be used to secure a variety of applications such as online transactions, financial transactions, identity verification, and secure communications.

The process of protocol #2 is shown in Figure 9. Different from protocol #1, Alice first has a pk from Gen. This is secure since sk has been deleted later, so the adversary can not

have K.

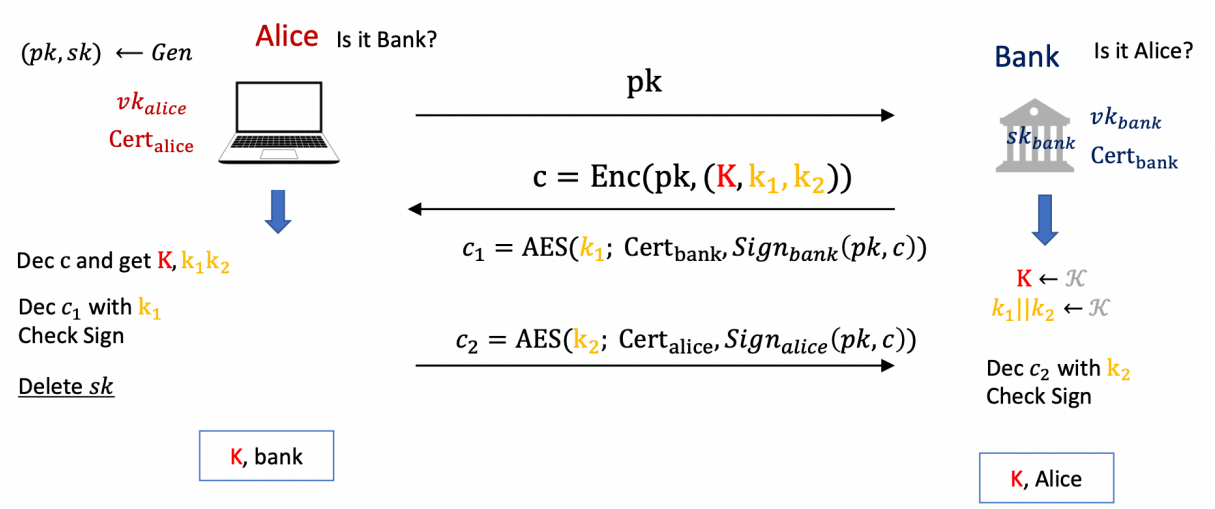

Figure 9: Protocol #2: HSM Security

2.3 Protocol #4: one side-use Diffie-Hellman instead of PKE

Similar to Protocol #2, Protocol #4 has the same theory, but replaces PKE with Diffie Hellman. It uses g^a and g^b, which is shown in Figure 10.

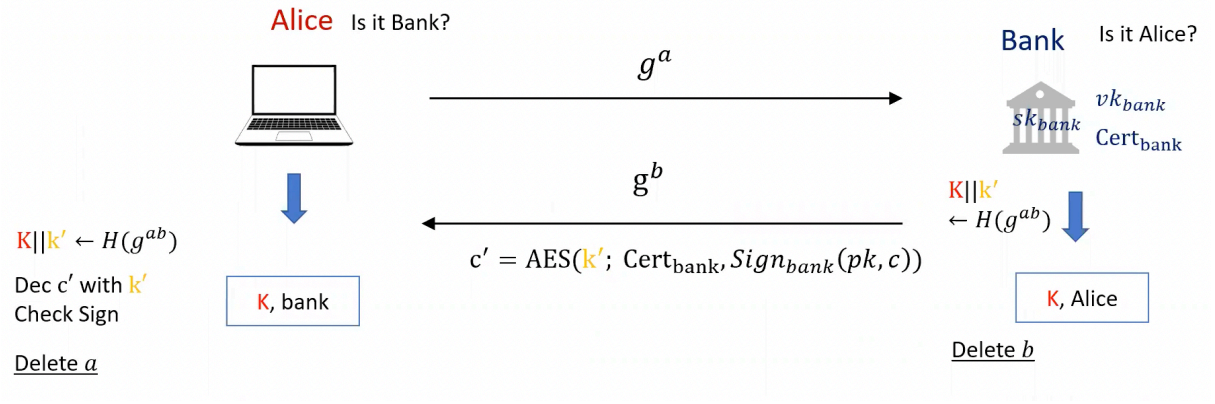

Figure 10: Protocol #4: one side-use Diffie-Hellman instead of PKE

We can build secure AKE via PKE, signature, and/or, AES.

3 Public Key Infrastructure

Public Key Infrastructure (PKI) is a security framework that enables secure communication over the internet. It is a system that uses a combination of public and private keys to encrypt and decrypt data, ensuring the confidentiality, integrity, and authentication of digital communications.

The foundation of PKI is the use of asymmetric cryptography, also known as public key cryptography. In this system, each user has a pair of keys: a public key that can be freely distributed and a private key that is kept secret. Data encrypted with a user's public key can only be decrypted with that user's private key, ensuring that only the intended recipient can access the information.

There are several parts in Certificate Authority (CA), such as: subject name, issuer name, and public key information. The way to check the root CA in MACs and Windows are not the same. Also we can find root CAs in web browser. [Ann19, KL20, BS20].

The infrastructure that supports the use of these keys includes several key components. The first component is a CA, which is responsible for issuing digital certificates that contain the public key and other identifying information about the user. These certificates are used to verify the authenticity of the public key, and they can be obtained through a process called Certificate Enrollment.

Another component of PKI is the Registration Authority (RA), which is responsible for verifying the identity of users before a certificate is issued. This process involves validating the user's identity through various means, such as ID checks or biometric authentication.

A third component of PKI is the Certificate Revocation List (CRL), which contains a list of all certificates that have been revoked or invalidated. This helps ensure that certificates that have been compromised or are no longer valid cannot be used to decrypt data.

Overall, PKI is a critical infrastructure for enabling secure communication over the internet, ensuring that sensitive information can be transmitted and stored safely. With its combination of public and private keys, digital certificates, and other components, PKI provides a powerful and flexible security framework that is essential for the modern digital world.

4 Certification Authorities

CAs are entities responsible for verifying the identities of individuals, organizations, and devices in a digital environment. They issue digital certificates, which are electronic documents that serve as proof of identity and are used to secure online communications and transactions. CAs play a crucial role in ensuring the authenticity, confidentiality, and integrity of online communications, as they are responsible for validating the identities of parties involved in

such communications. Some of the most well-known CAs include Comodo, DigiCert, and GlobalSign.

To solve the problem of single point failure, we have multiple CAs, and for each user, they may have multiple certifications.

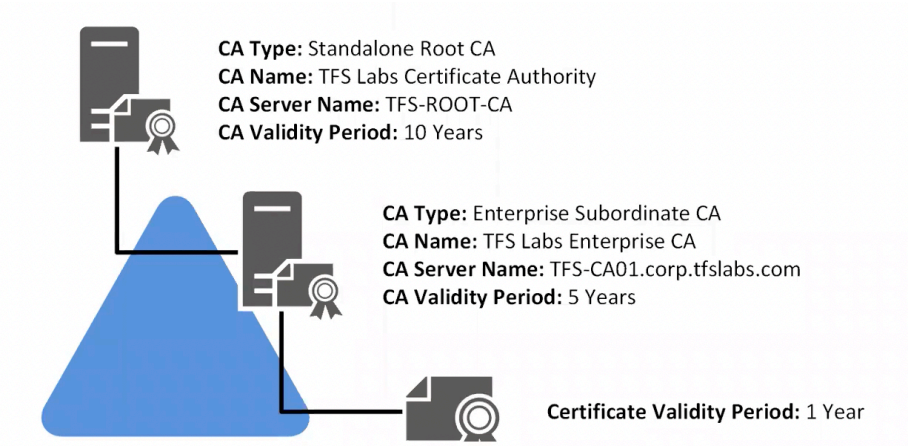

Figure 11: Authentication Chain

In Figure 11, it is an example of authentication chain. In this chain, if one of them has been trusted, the others will be trusted [Gol01].

The authentication chain refers to a sequence of authentication steps used to verify the identity of a user. Each step in the chain represents a layer of security designed to prevent unauthorized access to a system or application. The chain may include a variety of authentication methods, such as passwords, biometric scans, security tokens, or smart cards. By requiring multiple factors for authentication, the chain provides an extra layer of protection against identity theft and cyber-attacks. The authentication chain is commonly used in enterprise-level security systems and is an essential component of any robust cybersecurity strategy.

For example, Root CA in Mac OS refers to the trusted root certificate authorities that are pre-installed on Apple's macOS operating system. These certificate authorities are responsible for verifying the authenticity of digital certificates issued by various websites and other online services. By default, Mac OS trusts a set of root CAs that are regularly updated to ensure the security of the operating system. Users can view and manage the list of trusted root CAs in the Keychain Access utility on their Mac.

References

[Ann19] Robert Annessi. *Securing group communication in critical Infrastructures*. PhD thesis, Wien, 2019.

[BDPR98] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations among notions of security for public-key encryption schemes. In *Advances in CryptologyCRYPTO'98: 18th Annual International Cryptology Conference Santa Barbara, California, USA August 23–27, 1998 Proceedings 18*, pages 26–45. Springer, 1998.

[BS20] Dan Boneh and Victor Shoup. A graduate course in applied cryptography. *Draft 0.5*, 2020.

[CHK03] Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. In *Advances in CryptologyEUROCRYPT 2003: International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4–8, 2003 Proceedings 22*, pages 255–271. Springer, 2003.

[Gol01] Oded Goldreich. *Foundations of Cryptology: Basic Tools*. Cambridge, 2001.

[KL20] Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography*. CRC press, 2020.

[MS21] Stephen Mason and Daniel Seng. *Electronic Evidence and Electronic Signatures*. University of London, 2021.

[Nis92] Corporate Nist. The digital signature standard. *Communications of the ACM*, 35(7):36–40, 1992.

[PCD18] Andre EA Pacheco, Louella M Mesquita Colaco, and Shaba Desai. Secure dynamic data storage with third party arbitration in cloud. In *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 118–122. IEEE, 2018.

[Res18] Eric Rescorla. The transport layer security (tls) protocol version 1.3. Technical report, 2018.

[VOW+92] Paul C Van Oorschot, Michael J Wiener, et al. Authentication and authenticated key exchanges. *Des. Codes Cryptogr*, 2:107–125, 1992.

[Wol10] Marko Wolf. The evita hardware security module (hsm). In *Deliverable D1. 2.5. 1: Presentation slides from the EVITA project workshop*, page 34, 2010.

[Zho22] Yutong Zhong. An overview of rsa and oaep padding. *Highlights in Science, Engineering and Technology*, 1:82–86, 2022.