Computer Network Security

Lesson 8

Cryptography II

Lecturer: Dr Msagha J Mbogholi, PhD

Flashback from Lesson 7

- Cryptography is the science (or process) of securing information over a channel such that only the intended recipient can read it.
- Messages that are written in human comprehensible language are called plain text, while a message that has been changed according to some rules is called cipher text.
- Cryptographic techniques include the substitution and transposition ciphers approach.
- In symmetric encryption a shared key is used to both lock and unlock a message (text). Examples of application of symmetric encryption are DES, RC4/5, Blowfish and AES.
- In asymmetric encryption a different key is used to lock and unlock a message (text). Examples of application of asymmetric encryption are Diffie-Hellman, RSA, Rabin and ElGammal.

Content

- Steganography
- Introduction to Data integrity algorithms
- Hash functions
- Message authentication codes
- Digital signatures

Part 1

Steganography

Introduction

- Steganography is not what can be called encryption or cryptography *per se.*
- In cryptography plain text messages are encrypted using algorithms and keys in order to convert them into cipher text. Cipher text is not in human language and is therefore not comprehensible.
- In steganography there is no conversion to cipher text; rather the intended message is hidden within another message.
- How is this achieved?

Principles of Steganography

- According to Cole (2003, pg 61) there are three core principles that can be used to measure the effectiveness of a given steganography method. These are:
- Volume of hidden data – the more the better
- How hard it is to detect the hidden message – again the harder the better
- How hard it is to extract the message – if it is intercepted it is desirable that it's extremely hard to extract (find) the message.

Methods of Steganography

- In school we would play mystery games modeled on books of famous book characters. The game we enjoyed most was that of writing secret messages to one another, and then getting someone to decipher the message.
- Steganography right there, but we didn't know it. This is the most basic form of steganography.
- A letter is written telling a story; but the secret story is hidden inside the visible story. One common way of doing this was to find the secret story in every third word of the visible story. For example "I don't like Susan, Tiffany, and whatever. However, we all say students clearly like some challenges." See the story in every third word? ☺
- Other forms of steganography include…

Methods of Steganography

- Another common method is to overwrite text (plain or typed) with a writing aid like a pencil such that the overwritten text is not visible to the naked eye; one may need to use sunlight or some special ultraviolet light to see the message.
- Invisible ink is also used to hide text; in this instance the ink becomes visible under certain conditions, for example, when it is passed under a certain fluid.
- One common method used in the past was to puncture tiny holes on letters so that the true message would be read via these tiny holes.

Other Methods

- In terms of hiding data in computer based environments then there are two ways in which steganography can be applied: based on the type of file used and means by which the data has been hidden.
- Type of file used – the type of file plays a significant role. Files having different extensions (for example gif or tiff in graphics, or mp3 for music files), and this presents opportunities to apply steganography methods. Three ways of doing this are:
- Find places in the file that are likely to be ignored and write the message there (also known as injection in steganography)
- Find information in the file that can be removed (without changing the information/meaning in the file) and replacing it with the message (also known as substitution steganography)
- Change the bit representation of the file so that the message is hidden in the bits. In the case of an image it appears as if the image is the desired one but some bits have been changed in order to hide the message in it (also known as image steganography)

Part 2

Introduction to Data Integrity Algorithms

Introduction

- Data integrity has been defined earlier as the ability of the data being received by the intended recipient without any modification.
- This means there is a way of verifying that the data sent is the one that is received and has not been tampered with in any way.
- In network communication there are normally error correction and error detection mechanisms that can detect errors or correct them as they case may be.
- These mechanisms work based on some form of algorithm that checks the data at the receiving end.

Error Detection and Correction

- In both error correction and detection mechanisms the principle is to use some form of mathematical algorithm to add a bit (or bits) to the original data such that the whole data is transformed according to the algorithm. The added bits are normally called checksum in networking.
- The transformed data is sent to the recipient who then applies the algorithm to it. If the data has been modified in any way during transmission the checksum will indicate this via the algorithm.
- In the case of error detection the data is then rejected and recipient requests retransmission; in the case of error correction the algorithm attempts to 'guess' what the data should have been and corrects it accordingly.
- Understanding how this concept works makes it very easy to understand the workings of almost all data integrity algorithms.

Error Detection and Data Integrity

- From Lesson 7 we learnt that both symmetric and asymmetric encryption make use of keys in order to preserve the integrity of messages.
- In the case of symmetric a shared key is used, while asymmetric (also known as public key encryption) utilizes a public and private key. The keys are generated and sent by a trusted third party (normally a certification authority)
- We can now combine our knowledge of the way error detection mechanisms work and our knowledge of encryption systems to show the exchange of data in a communication channel.

Error Detection and Data integrity

- Figure 1 shows the flow of data across an information channel using error detection as well as encryption techniques

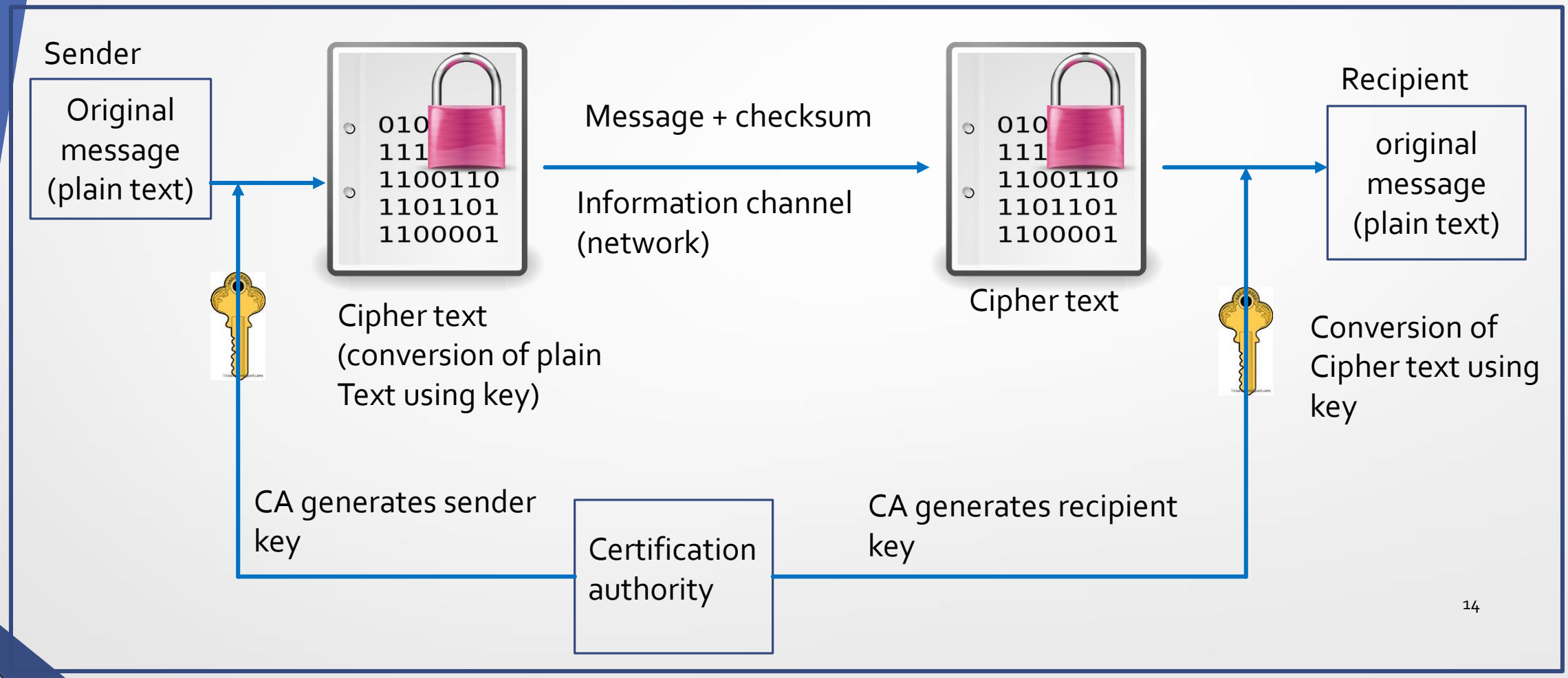

Fig 1. Implementation of Encryption and error detection system

Data Integrity with Encryption

- The keys used in figure 1 do not assume either symmetric or public key encryption (the use of either was covered in lesson 7)
- What is seen is that it is not enough to encrypt data! You may be wondering why is there need to involve checksum when the data is already encrypted?
- Well from our knowledge of the OSI model we know that data regardless of its nature is only transmitted in the form of zeroes and 1s (layer 1 transmission).
- This still exposes the data to tampering (modification) as long as the attacker can manipulate the 0s and 1s.

Data Integrity with Encryption

- Since the 0s and 1s can be manipulated across the information channel either intentionally (by an attacker) or unintentionally (due to EMI, noise or other impediments) it is necessary to add data integrity to the encrypted data itself.
- Thus the sender encrypts the data using some form of encryption algorithm; the encrypted data is then subjected to an error detection/correction mechanism (algorithm) and the resultant data and checksum are sent across the information channel.
- At the recipient end the error detection/correction mechanism is first applied to check that the data has not been tampered with; if it has not then the decryption algorithm kicks in. If it has been tampered with then appropriate measures are taken; in the case of error detection the data is rejected and sender requested to retransmit. In the case of error correction the algorithm attempts to correct the data before decryption.

Part 3

Hash Functions

Introduction

- A hash function is a function that is used to preserve data integrity.
- As seen from lesson 7 at times it is easier for a cryptanalyst to decipher a message based on its length. That is to say in a given sentence (regardless of the cryptography method used) the cryptanalyst can guess based on their knowledge of the English language (or another language) what certain characters mean; based on this they can crack the code.
- This is clearly a weakness in most systems. Hash functions purpose to overcome this weakness by ensuring that regardless of the length of the text the resultant output is always the same length.
- So, how do they do this?

The Hash function

- Consider a message M; M can be of any length. We wish to apply a hash function F to the M in order to produce a different string, let's call it h.
- The application is similar to the description in the introduction section of this lesson where we applied a checksum to a piece of data.
- Therefore mathematically the application of F to M can be shown as follows:
- F(M) = h, where F is the hash function and h is the resultant output string.
- The hashing function F is applied to M by breaking the latter down into blocks of fixed length and then producing h.
- h is referred to as the hash code or the message digest. Let us demonstrate this using Figure 2.

Message Digests

- Figure 2 demonstrates how our message M is broken down into n blocks of fixed length, and then output of each block is fed as input to the next block till the nth block (final block) which is then fed to F.

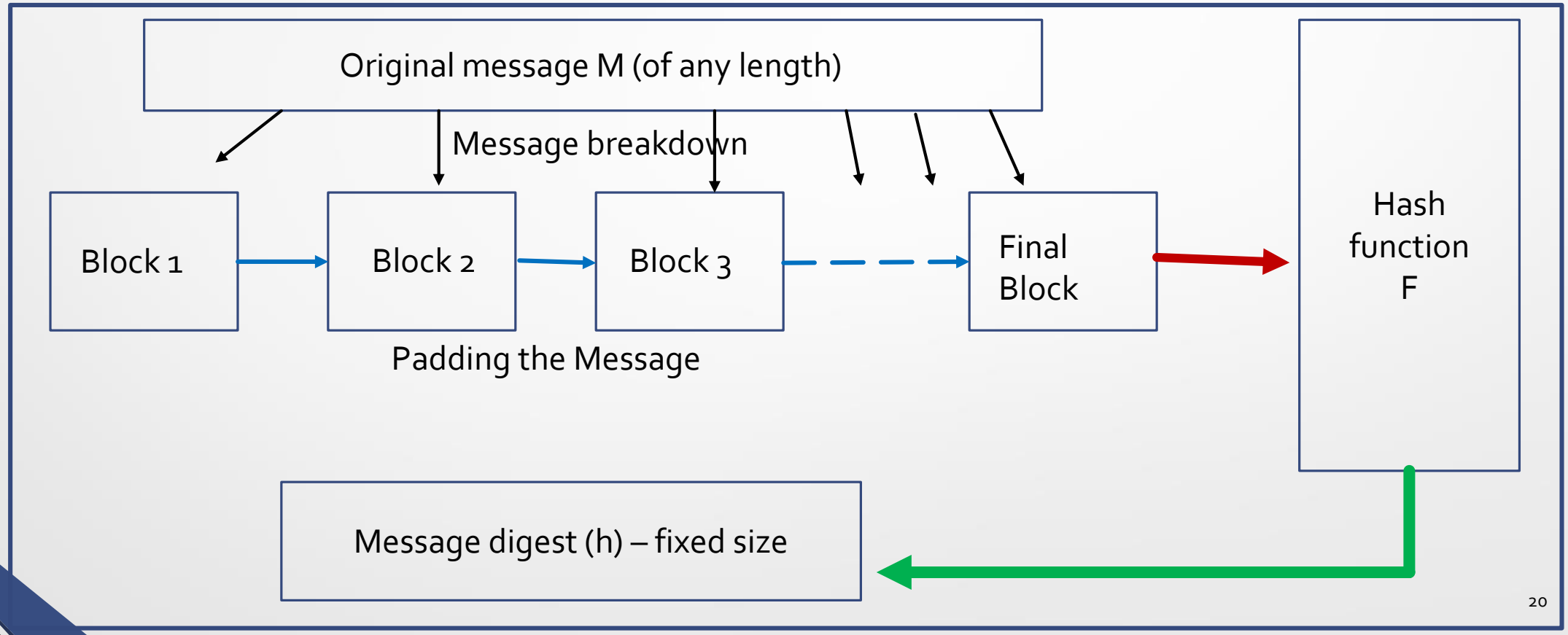

Figure 2. Message conversion to hash code (digest)

Verifying Integrity

- How does the recipient check the hash function from their end?
- First of all it is important to understand that hashing is a one way process (it is called a one way function); once the digest is created there is practically no way to reverse the process to generate the initial text. Great, right?
- This is to say that once a digest has been created from a message, the only way to create an exact similar digest is to generate it again.
- Since it is a one way function then the integrity of the message can be verified by the recipient

Verifying Integrity

- This is what happens at the recipient's end:
- The recipient receives the original message together with the message digest. This is after the message has been decrypted (in case it was encrypted before transmission)
- The recipient then takes the message and runs it through the hash function again generating a second digest.
- This second digest is then compared with the one that came with the message; if they match then it means that the message has not been tampered with, and thus message integrity is confirmed.

Hash function

- The hash function is not an encryption scheme therefore does not provide what encryption does.
- In order to provide authentication the system will have to provide/implement encryption separately.
- There are many different ways to do this and they will be briefly described in a different part of this lesson.
- Let us examine the requirements of hash functions.

Requirements

- Stallings et al., (2011) describe a list of requirements for a good cryptographic hash function H.

Requirement	Description
Variable input size	H can be applied to a block of data of any size.
Fixed output size	H produces a fixed-length output.
Efficiency	$H(x)$ is relatively easy to compute for any given x, making both hardware and software implementations practical.
Preimage resistant (one-way property)	For any given hash value h, it is computationally infeasible to find y such that $H(y) = h$.
Second preimage resistant (weak collision resistant)	For any given block x, it is computationally infeasible to find $y \neq x$ with $H(y) = H(x)$.
Collision resistant (strong collision resistant)	It is computationally infeasible to find any pair (x, y) such that $H(x) = H(y)$.
Pseudorandomness	Output of H meets standard tests for pseudorandomness.

Application Areas

- Cryptographic hash functions are used in many application areas. Some of the common hashing algorithms include:
- Secure Hash Algorithm (SHA) – it is a family of hashing algorithms (SHA - 256 being the one in common use)
- Message Digest (MD)
- Whirlpool

Part 4

Message Authentication Codes

Introduction

- As the name implies Message Authentication Code (MAC) brings in authentication into the digital integrity scene.
- The concept of MAC is similar to hash functions difference being that the authenticity of a message can also be verified.
- Further MAC also does not incorporate reversibility meaning that it is also a one way function.
- Lastly MAC is in the strict sense a cryptographic scheme unlike hash functions since it incorporates a key in the process.
- Figure 3 illustrates the MAC process.

The MAC Process

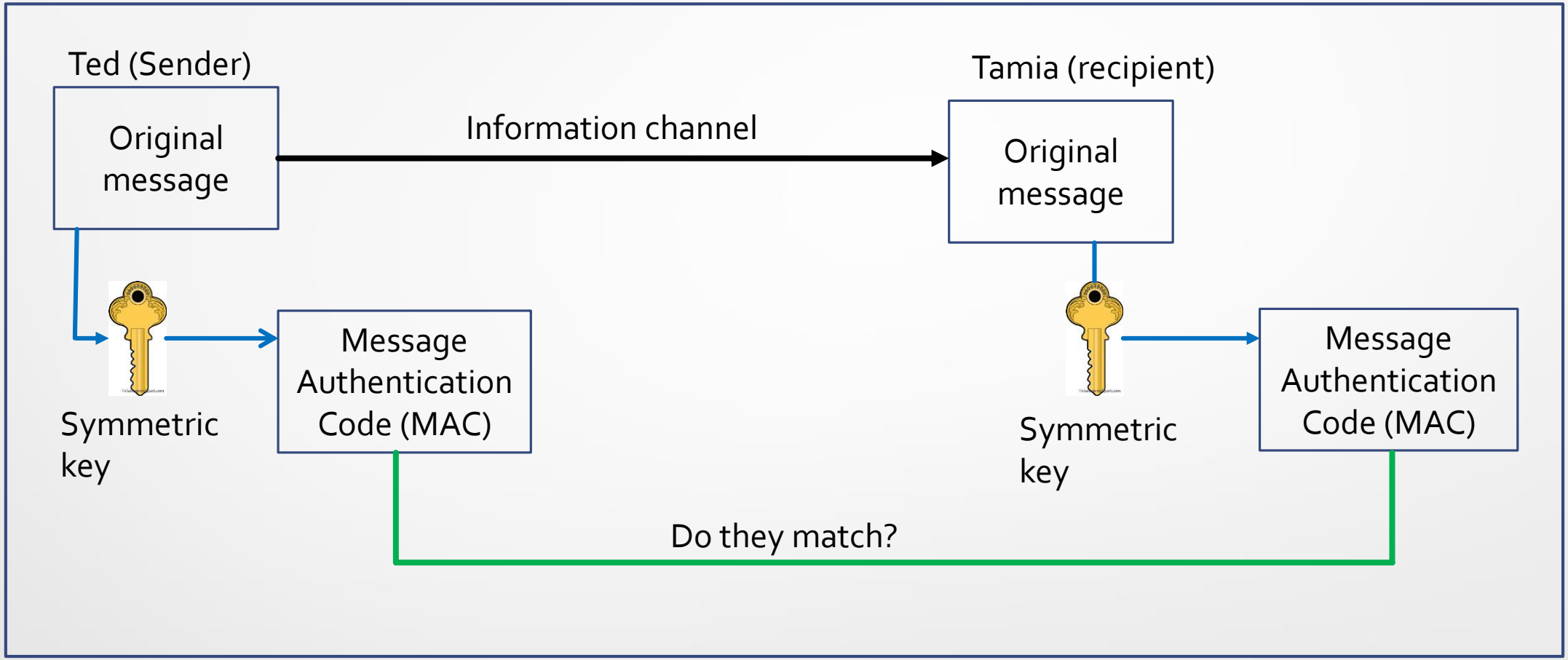

Fig 3. The Message Authentication Code (MAC) process

The MAC Process

- Figure 3 demonstrates the MAC process.
- Ted (sender) would like to send a message to Tamia (recipient).
- Ted applies a symmetric key to the message producing a MAC which we call m1. Then he sends the message together with m1 to Tamia.
- Tamia receives the message and applies the symmetric key on her end to the message, producing another MAC which we call m2.
- Tamia then compares m1 and m2 which she has generated. If the two match then the message has not been tampered with.

The MAC Process

- There are some key differences between MAC and hashing. Let us first start with the encryption process.
- In hashing there was no encryption key involved; however, a symmetric key is introduced with MAC.
- Further with MAC the encryption process isn't complete. This is so since there is encryption but no decryption. Recall that in symmetric encryption the shared key is used to encrypt the data at one end (sender) and decrypt it at the other end (recipient). In this instance the both sender and recipient only encrypt the message in order to read the MAC and compare for integrity purposes.
- How is authentication implemented? Well, the symmetric key (shared) is only known to two people, in this case Ted and Tamia. Thus when Tamia confirms the MAC she is sure it is only Ted who could have sent the message since he is the only one with the other key. The same applies to Ted who knows that it is only Tamia who can generate a MAC that will match the one he sent.

Applications

- Hash based MAC or HMAC is the best known implementation of the MAC in a networked environment.
- It works by applying the key to an already computed hash (digest).
- The rest of the process is the same as illustrated in figure 3.
- Due to the dual integrity and authenticity aspects of MACs they are used in environments that require these; a good example is in banking where confirmation of integrity and authenticity is required before authorizing some transactions (like electronic funds transfers)

Part 4

Digital Signatures

Introduction

- Anytime you go to the bank to perform a transaction you are called upon to sign a document that authenticates you. The same happens when you visit the ATM; you have a PIN that is known only to you and this is what you use to transact with your account.
- The concept of signatures serves two purposes in most scenarios; to authenticate the user and also to enforce non-repudiation.
- Digital signatures serve the same purpose; they are used to not just authenticate a user but also to enforce non-repudiation.
- Figure 4 shows the working of the digital signature scheme.

Digital Signature Scheme

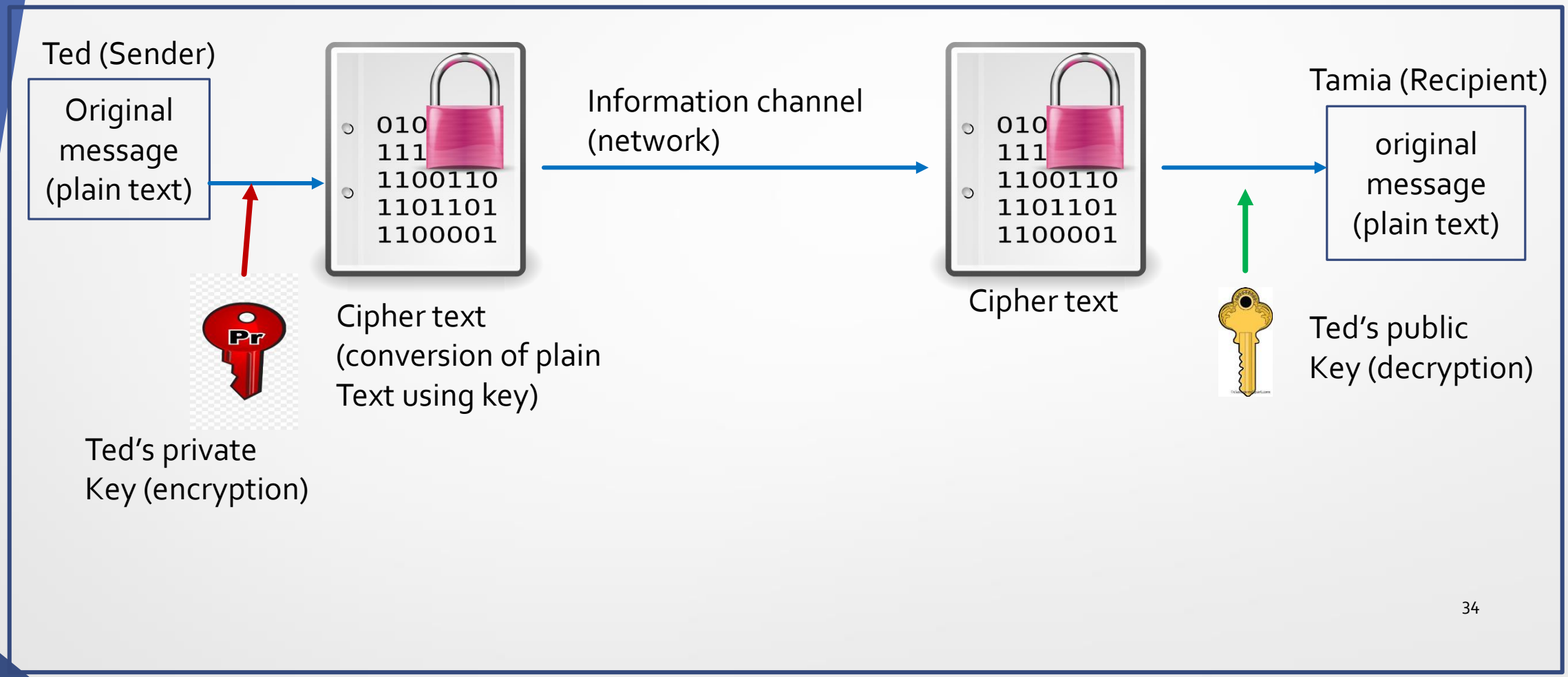

Figure 3. Digital signature scheme

34

Digital Signature Scheme

- In figure 3, Ted wishes to send a message to Tamia.
- He encrypts the message with his own private key and sends it across the information channel.
- Once the message gets to Tamia she decrypts it using Ted's public key.
- Recall that keys are generated in pairs; the public key (available and widely disseminated) and the private key (which uniquely belongs to the owner).
- When Ted encrypts the message with his private key (which uniquely identifies him) he is confirming that he is the one who has encrypted the message, and nobody else.
- Thus when the message gets to Tamia she can use Ted's public key to decrypt the message since it is widely available and anyone has access to it.
- So what's the point?

Digital Signature Scheme

- As alluded to earlier digital signatures are not about confidentiality like in other encryption schemes; rather they are about authenticity and non-repudiation.
- An attacker can intercept the message and modify (tamper) it before sending it on to Tamia since they too can use Ted's public key. However, they can't encrypt the message again using Ted's private key since it is unique to him. So when the message gets to Tamia and she attempts to open it using Ted's public key the action will fail! This tells her the message isn't from Ted.
- In the same breath since the message can only be opened by Ted's public key it means that he can't repudiate the message/text since it can only have come from him.

Summary

- Steganography is not cryptographic in nature *per se.* It is the science of hiding a message within another message.
- Data integrity algorithms purpose to preserve the integrity of data by ensuring mechanisms that confirm it has not been modified in any way whatsoever.
- Hash functions work on data by producing message digests which are then compared at the receiver's end to confirm the integrity of the data.
- Message authentication codes (MAC) combine authentication with data integrity by introducing a symmetric key in the mechanism.
- Digital signatures enforce authentication and non-repudiation by using the sender's private key at the sender's end and his/her public key at the recipient's end.

References

- Cole, E. (2003). *Hiding in plain sight steganography and the art of covert communication*. Wiley.
- Kahate, A. (2013). *Cryptography and network security*. McGraw Hill Education.
- Stallings, W. (2011). *Cryptography and network security: Principles and practice*. essay, Prentice Hall.