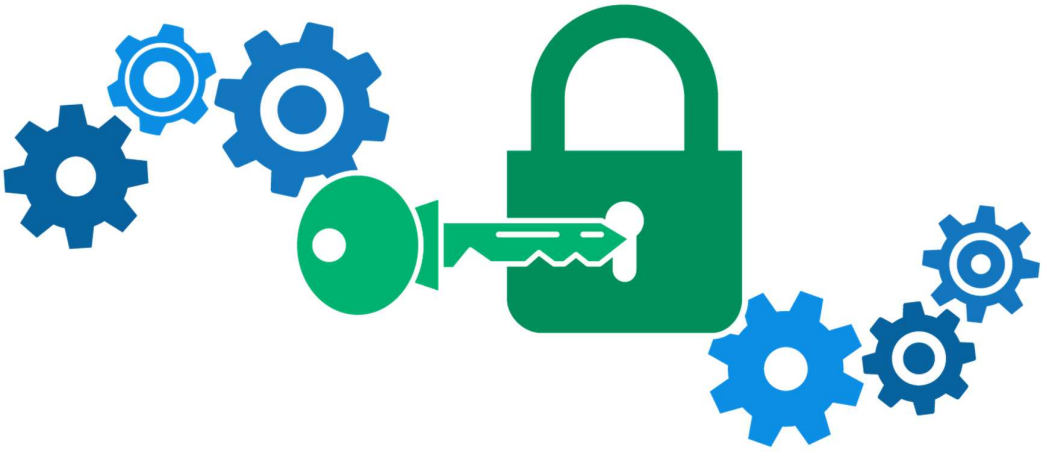
Faculty of engineering

2nd communication

Digital Logic

Labview



Network Encryption

Encryption

In cryptography, encryption is the process of encoding a message or information in such a way that only authorized parties can access it and those who are not authorized cannot.

Encryption does not itself prevent interference but denies the intelligible content to a would-be interceptor. In an encryption scheme, the intended information or message, referred to as plaintext, is encrypted using an encryption algorithm – a cipher – generating ciphertext that can be read only if decrypted.

For technical reasons, an encryption scheme usually uses a pseudo-random encryption key generated by an algorithm. It is in principle possible to decrypt the message without possessing the key, but, for a well-designed encryption scheme, considerable computational resources and skills are required. An authorized recipient can easily decrypt the message with the key provided by the originator to recipients but not to unauthorized users.

Types:

Symmetric key / Private key:

In symmetric-key schemes, the encryption and decryption keys are the same. Communicating parties must have the same key in order to achieve secure communication.

Public key:

In public-key encryption schemes, the encryption key is published for anyone to use and encrypt messages. However, only the receiving party has access to the decryption key that enables messages to be read

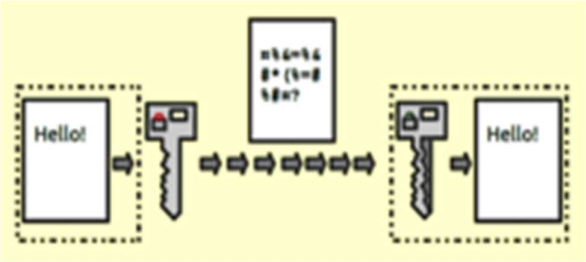


Illustration of how encryption is used within servers

Public key encryption

Uses:

Encryption has long been used by militaries and governments to facilitate secret communication.

It is now commonly used in protecting information within many kinds of civilian systems. For example, the Computer Security Institute reported that in 2007, 71% of companies surveyed utilized encryption for some of their data in transit, and 53% utilized encryption for some of their data in storage.[6]

Encryption can be used to protect data "at rest", such as information stored on computers and storage devices (e.g. USB flash drives).Encrypting such files at rest helps protect them if physical security measures fail.

Digital rights management systems, which prevent unauthorized use or reproduction of copyrighted material and protect software against reverse engineering is another somewhat different example of using encryption on data at rest.[7]

Encryption is also used to protect data in transit, for example data being transferred via networks (e.g. the Internet, e-commerce), mobile telephones, wireless microphones, wireless intercom systems, Bluetooth devices and bank automatic teller machines

Data should also be encrypted when transmitted across networks in order to protect against eavesdropping of network traffic by unauthorized users.

Message verification:

Encryption, can protect the confidentiality of messages, but other techniques are still needed to protect the integrity and authenticity of a message; for example, verification of a message authentication code (MAC) or a digital signature.

Standards for cryptographic software and hardware to perform encryption are widely available, but successfully using encryption to ensure security may be a challenging problem. A single error in system design or execution can allow successful attacks.

Digital signature and encryption must be applied to the ciphertext when it is created (typically on the same device used to compose the message) to avoid tampering; otherwise any node between the sender and the encryption agent could potentially tamper with it. Encrypting at the time of creation is only secure if the encryption device itself has not been tampered with.

Data erasure:

Conventional methods for deleting data permanently from a storage device involve overwriting its whole content with zeros, ones or other patterns – a process which can take a significant amount of time, depending on the capacity and the type of the medium.

Cryptography offers a way of making the erasure almost instantaneous. This method is called crypto-shredding. An example implementation of this method can be found on iOS devices, where the cryptographic key is kept in a dedicated 'Effaceable Storage'.[18] Because the key is stored on the same device, this setup on its own does not offer full confidentiality protection in case an unauthorised person gains physical access to the device.

XOR cipher

In cryptography, the **simple XOR cipher** is a type of additive cipher, an encryption algorithm that operates according to the principles:

$A \oplus 0 = A$
$A \oplus A = 0$
$(A \oplus B) \oplus C = A \oplus (B \oplus C)$
$(B \oplus A) \oplus A = B \oplus 0 = B$

where $\oplus$ denotes the exclusive disjunction (XOR) operation. This operation is sometimes called modulus 2 addition (or subtraction, which is identical). With this logic, a string of text can be encrypted by applying the bitwise XOR operator to every character using a given key. To decrypt the output, merely reapplying the XOR function with the key will remove the cipher.

Example

For example, the string "Wiki" (01010111 01101001 01101011 01101001 in 8-bit ASCII) can be encrypted with the repeating key 11110011 as follows:

```
  01010111 01101001 01101011 01101001
⊕ 11110011 11110011 11110011 11110011
  ----------------------------------
= 10100100 10011010 10011000 10011010
```

And conversely, for decryption:

```
  10100100 10011010 10011000 10011010
⊕ 11110011 11110011 11110011 11110011
  ----------------------------------
= 01010111 01101001 01101011 01101001
```

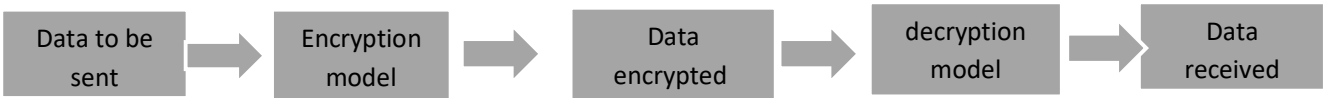
Use and security

The XOR operator is extremely common as a component in more complex ciphers. If the content of any message can be guessed or otherwise known then the key can be revealed. Its primary merit is that it is simple to implement, and that the XOR operation is computationally inexpensive. A simple repeating XOR (i.e. using the same key for xor operation on the whole data) cipher is therefore sometimes used for hiding information in cases where no particular security is required.

If the key is random and is at least as long as the message, the XOR cipher is much more secure than when there is key repetition within a message. In any of these ciphers, the XOR operator is vulnerable to a known-plaintext attack, since plaintext $\oplus$ciphertext = key. It is also trivial to flip arbitrary bits in the decrypted plaintext by manipulating the ciphertext. This is called malleability.

Network encryption simple model using cipher implementation in LABVIEW

1st block diagram



a) Data to be sent
   - Its type is positive numbers only
   - It is a numeric control

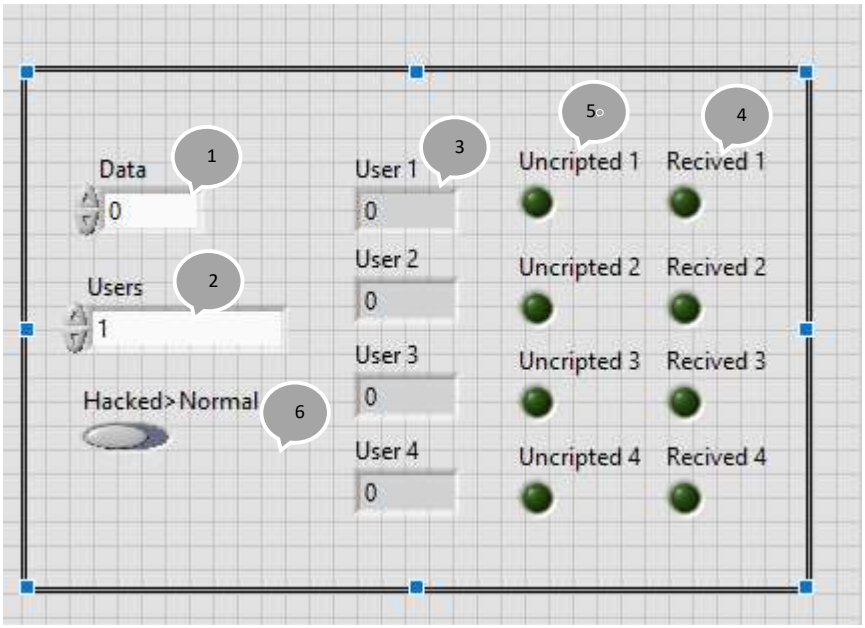
b) Encryption model
   - It is simple cipher model
   - It consists of data that user had entered converted to array of Booleans XORed with user to be sent to key

c) Data encrypted
   - It is an array of Booleans
   - Using case structure, data determines its path according to whom to be sent to key

d) Decryption model
   - It is like the encryption one
   - The difference is that data is XORed with a constant distinct key for each user

e) Data received
   - Its type is positive numbers only
   - It is a numeric indicator

2nd user interface



1) Numeric control: user uses it to enter data
2) Enum: user uses it to choose who will receive data
3) Numeric indicator: it shows data received
4) Boolean leds: it indicates if this user received something or not
5) Boolean leds; it indicates if this user decrypted the data correctly or not
6) Boolean switch: it chooses between two modes (hacked -most worst case- and normal)

Modes of operation

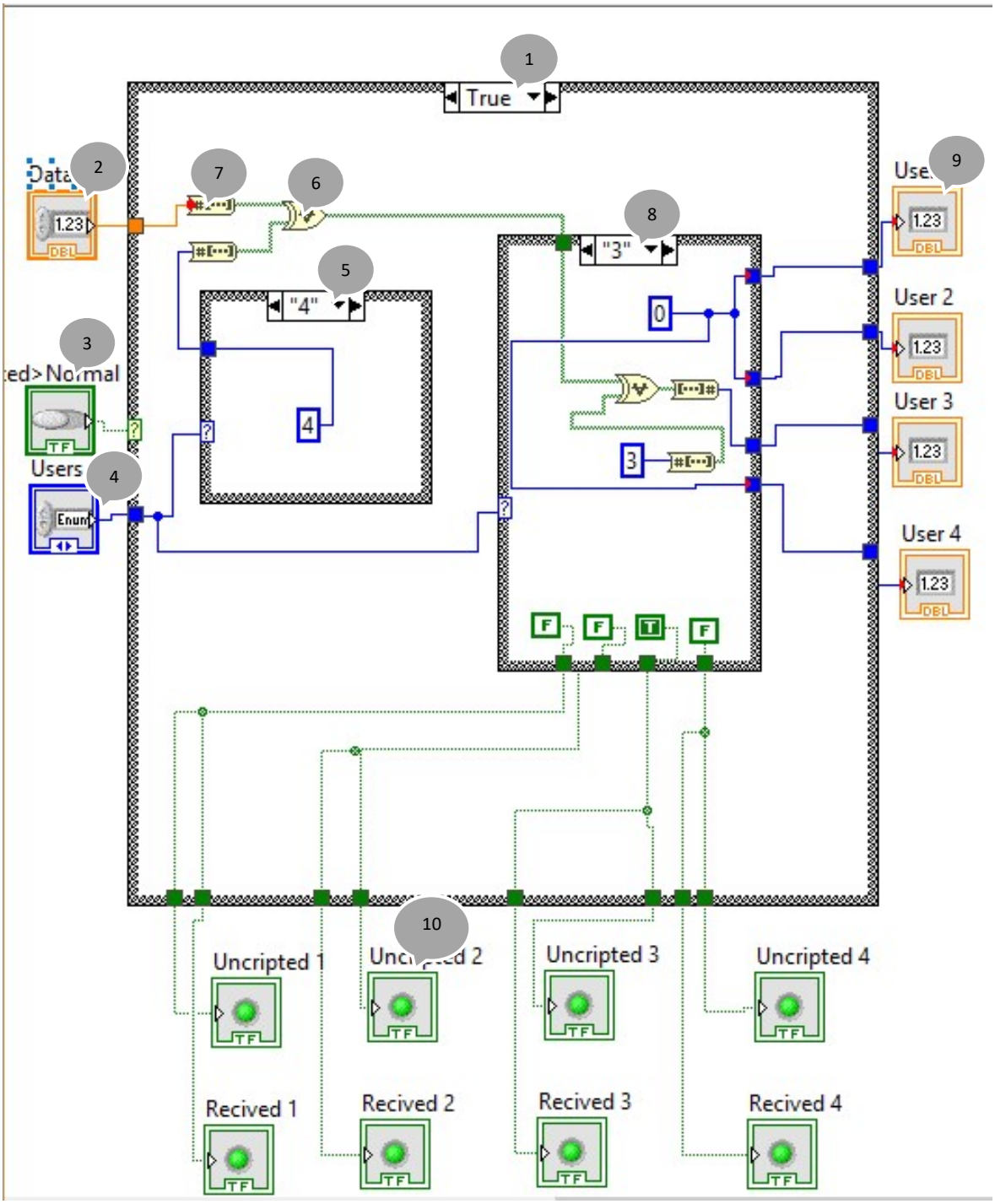
a) Normal mode
   - User enters data in a numeric control
   - User chooses whom data to be sent to
   - User hits run
   - The data is sent to only one user who had been chosen former
   - Only one led from received leds will light up
   - Only one led from uncrypted leds will light up

Note: this mode is the normal one as the data should be delivered to only one person who had been chosen

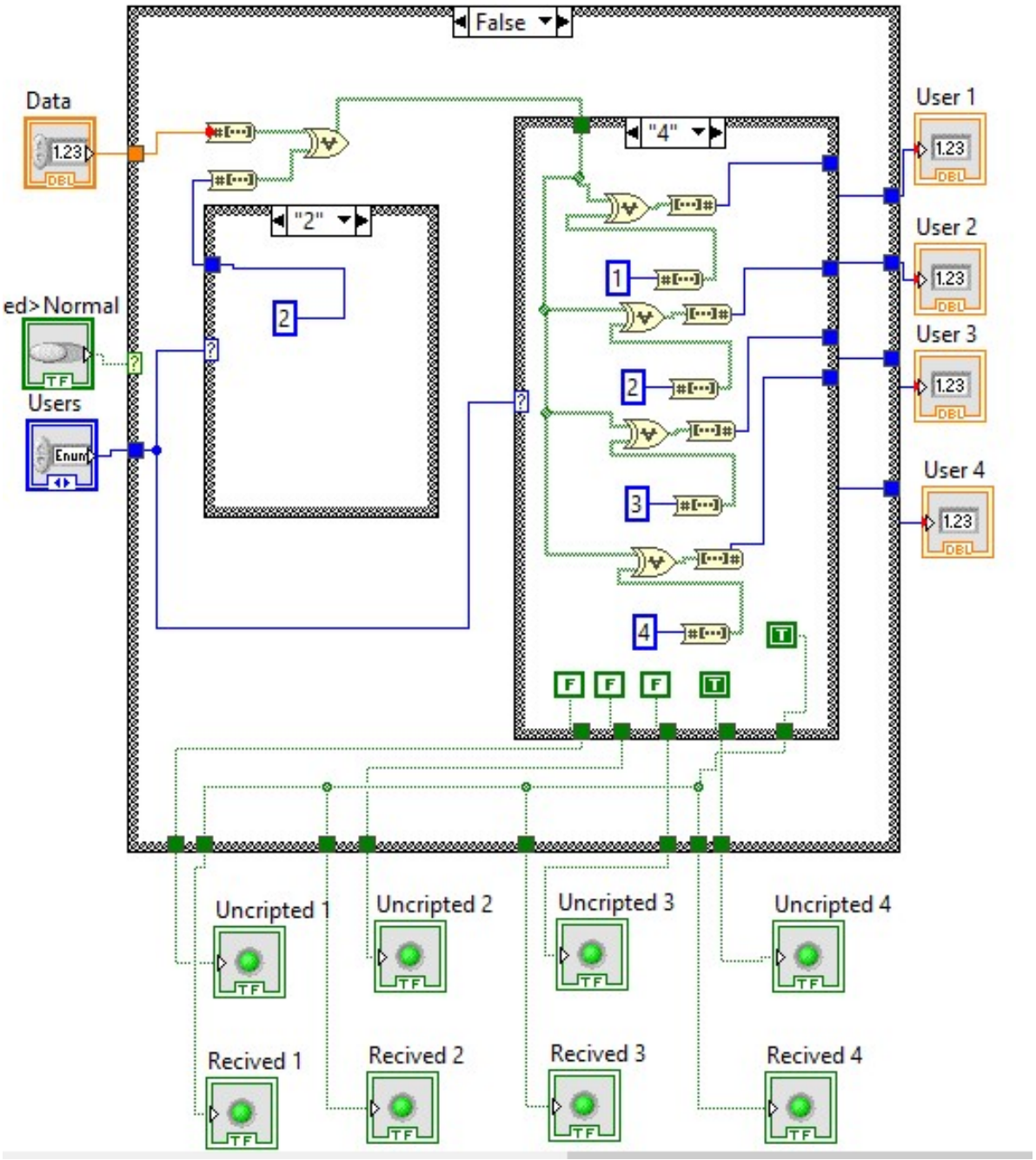
b) Hacked mode
   - User enters data in a numeric control
   - User chooses whom data to be sent
   - User hits run
   - The data is sent to all users neglecting the chosen one
   - All leds of received leds will light up
   - Only one led from uncrypted leds will light up

Note: this mode is the worst scenario will happen in any network that the data is delivered to all users neglecting the chosen one, so the encryption algorithm in this bad case should secure data from been read by anyone

3rd block diagram LABVIEW



1) Case structure: it determines which mode will work (hacked or normal) according to (3)
2) Numeric control: it contains data which will be encrypted
3) Boolean switch: according to it (1) will choose the mode
4) Enum: it is used to choose user
5) Case structure: it assigns key for each user beside it chooses who will receive data
6) XOR: it is the encryption stage at where the cipher encryption happens
7) Number to Boolean array converter: it is used to pass data to XOR as XOR understands only Boolean type
8) Case structure: it differs between hacked and normal mode as in normal mode it passes data only for one user and decrypts it using a former assigned key, but in hacked mode it passes data for all users and decrypts data to each one of them according to the former assigned key so only one will read the data right
9) Numeric indicator: to display data received to verify our idea
10) Boolean Leds: it indicates if data is decrypted or not
11) Boolean Leds: it indicates if data is received or not



Team(65):

Mina nabil wahib_ 296 Andrew Morkos Fahmy _ 71

Engy Achraf _ 70 Youstina Abd Elmsih Ibrahim _ 325

Sylvia Atif _ 141