**Radix 64**

**LEARNING OUTCOME**

**By the end of the lesson the student will be able to:**

a) explain the concept of radix 64  
b) to encode a text to ascii code and then to radix 64.

c) to decode radix64 to ascii code and back to text.

Let us go an overview of cryptographic history. Go back to the year 1920 right after WW1. We have mechanical typewriter and then electromechanical crypto machine. In 1940, WW2 we will have electronic typewriter. Popular machines are German Enigma, Purple of Japan. From 1920 to 2000, CryptoAG is still the main producer of crypto machine.

A picture containing shape

Description automatically generated

Figure 1. Enigma has an electromechanical 3-rotor mechanism that scrambles 26 alphabets.

In 1960 a computer comes into picture only after WW2. Computer operates on bytes or ASCII code.

Base64 is a group of similar binary-to-text encoding schemes that represent binary data in an ASCII string format by translating it into a radix-64 representation. The term Base64 originates from a specific MIME content transfer encoding.

Ideally, a crypto message will be encoded in radix-64. It will be written in a symbol of 6 bits. This is the most common text in public key certificate and ciphertext.

Base64 implementation uses A–Z, a–z, and 0–9 for the first 62 values.

**The Base64 index table:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Value** | **Char** |  | **Value** | **Char** |  | **Value** | **Char** |  | **Value** | **Char** |
| 0 | A | 16 | Q | 32 | g | 48 | w |
| 1 | B | 17 | R | 33 | h | 49 | x |
| 2 | C | 18 | S | 34 | i | 50 | y |
| 3 | D | 19 | T | 35 | j | 51 | z |
| 4 | E | 20 | U | 36 | k | 52 | 0 |
| 5 | F | 21 | V | 37 | l | 53 | 1 |
| 6 | G | 22 | W | 38 | m | 54 | 2 |
| 7 | H | 23 | X | 39 | n | 55 | 3 |
| 8 | I | 24 | Y | 40 | o | 56 | 4 |
| 9 | J | 25 | Z | 41 | p | 57 | 5 |
| 10 | K | 26 | a | 42 | q | 58 | 6 |
| 11 | L | 27 | b | 43 | r | 59 | 7 |
| 12 | M | 28 | c | 44 | s | 60 | 8 |
| 13 | N | 29 | d | 45 | t | 61 | 9 |
| 14 | O | 30 | e | 46 | u | 62 | + |
| 15 | P | 31 | f | 47 | v | 63 | / |

When the number of bytes to encode is not divisible by three (that is, if there are only one or two bytes of input for the last 24-bit block), then the following action is performed:

Add extra bytes with value zero so there are three bytes, and perform the conversion to base64.

If there is only one significant input byte (e.g., 'M'), all 8 bits will be captured in the first two base64 digits (12 bits).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Text content** | **M** | | | | | | | |  | | | | | | | |  | | | | | | | |
| **ASCII** | 77 (0x4d) | | | | | | | | 0 (0x00) | | | | | | | | 0 (0x00) | | | | | | | |
| **Bit pattern** | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Index** | 19 | | | | | | 16 | | | | | | 0 | | | | | | 0 | | | | | |
| **Base64-encoded** | **T** | | | | | | **Q** | | | | | | **=** | | | | | | **=** | | | | | |

If there are two significant input bytes (e.g., 'Ma'), all 16 bits will be captured in the first three base64 digits (18 bits). '=' characters might be added to make the last block contain four base64 characters.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Text content** | **M** | | | | | | | | **a** | | | | | | | | r | | | | | | | |
| **ASCII** | 77 (0x4d) | | | | | | | | 97 (0x61) | | | | | | | | 0 (0x00) | | | | | | | |
| **Bit pattern** | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| **Index** | 19 | | | | | | 22 | | | | | | 5 | | | | | | 50 | | | | | |
| **Base64-encoded** | **T** | | | | | | **W** | | | | | | **F** | | | | | | **y** | | | | | |

**Output padding**

The final '==' sequence indicates that the last group contained only one byte, and '=' indicates that it contained two bytes. The example below illustrates how truncating the input of the above quote changes the output padding:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Length** | **Input** | **Length** | **Output** | **Padding** |
| 20 | *any carnal plea****sure.*** | 28 | YW55IGNhcm5hbCBwbGVh**c3VyZS4=** | 1 |
| 19 | *any carnal plea****sure*** | 28 | YW55IGNhcm5hbCBwbGVh**c3VyZQ==** | 2 |
| 18 | *any carnal plea****sur*** | 24 | YW55IGNhcm5hbCBwbGVh**c3Vy** | 0 |

In [cryptography](https://en.wikipedia.org/wiki/Cryptography), **X.509** is a standard that defines the format of [public key certificates](https://en.wikipedia.org/wiki/Public_key_certificate). X.509 certificates are used in many Internet protocols, including [TLS/SSL](https://en.wikipedia.org/wiki/Transport_Layer_Security), which is the basis for [HTTPS](https://en.wikipedia.org/wiki/HTTPS), the secure protocol for browsing the [web](https://en.wikipedia.org/wiki/World_Wide_Web). They're also used in offline applications, like [electronic signatures](https://en.wikipedia.org/wiki/Electronic_signature). An X.509 certificate contains a public key and an identity (a hostname, or an organization, or an individual), and is either signed by a [certificate authority](https://en.wikipedia.org/wiki/Certificate_authority) or self-signed. When a certificate is signed by a certificate authority, or validated by another means, someone holding that certificate can rely on the public key it contains to establish secure communications with another party, or validate documents [digitally signed](https://en.wikipedia.org/wiki/Digital_signature) by the corresponding [private key](https://en.wikipedia.org/wiki/Private_key).

Besides the format for certificates themselves, X.509 specifies certificate [revocation lists](https://en.wikipedia.org/wiki/Revocation_list) as a means to distribute information about certificates that are no longer valid, and a [certification path validation algorithm](https://en.wikipedia.org/wiki/Certification_path_validation_algorithm), which allows for certificates to be signed by intermediate CA certificates, which are in turn signed by other certificates, eventually reaching a [trust anchor](https://en.wikipedia.org/wiki/Trust_anchor).

X.509 is defined by the [International Telecommunications Union's](https://en.wikipedia.org/wiki/International_Telecommunication_Union) Standardization sector (ITU-T), and is based on [ASN.1](https://en.wikipedia.org/wiki/Abstract_Syntax_Notation_One), another ITU-T standard.

**Structure of a certificate** X.509

The structure foreseen by the standards is expressed in a formal language, [Abstract Syntax Notation One](https://en.wikipedia.org/wiki/Abstract_Syntax_Notation_One) (ASN.1).

The structure of an X.509 v3 [digital certificate](https://en.wikipedia.org/wiki/Digital_certificate) is as follows:

* Certificate
  + Version Number
  + Serial Number
  + Signature Algorithm ID
  + Issuer Name
  + Validity period
    - Not Before
    - Not After
  + Subject name
  + Subject Public Key Info : MyKAD number
    - Public Key Algorithm: RSA 1024
    - Subject Public Key in radix64.
  + Issuer Unique Identifier (optional)
  + Subject Unique Identifier (optional)
  + Extensions (optional)
    - ...
* Certificate Signature Algorithm
* Certificate Signature

Certificate:

Data:

Version: 3 (0x2)

Serial Number:

10:e6:fc:62:b7:41:8a:d5:00:5e:45:b6

Signature Algorithm: sha256WithRSAEncryption

Issuer: C=BE, O=GlobalSign nv-sa, CN=GlobalSign Organization Validation CA - SHA256 - G2

Validity

Not Before: Nov 21 08:00:00 2016 GMT

Not After : Nov 22 07:59:59 2017 GMT

Subject: C=US, ST=California, L=San Francisco, O=Wikimedia Foundation, Inc., CN=\*.wikipedia.org

Subject Public Key Info:

Public Key Algorithm: id-ecPublicKey

Public-Key: (256 bit)

pub:

04:c9:22:69:31:8a:d6:6c:ea:da:c3:7f:2c:ac:a5:

af:c0:02:ea:81:cb:65:b9:fd:0c:6d:46:5b:c9:1e:

ed:b2:ac:2a:1b:4a:ec:80:7b:e7:1a:51:e0:df:f7:

c7:4a:20:7b:91:4b:20:07:21:ce:cf:68:65:8c:c6:

9d:3b:ef:d5:c1

ASN1 OID: prime256v1

NIST CURVE: P-256

X509v3 extensions:

X509v3 Key Usage: critical

Digital Signature, Key Agreement

Authority Information Access:

CA Issuers - URI:http://secure.globalsign.com/cacert/gsorganizationvalsha2g2r1.crt

OCSP - URI:http://ocsp2.globalsign.com/gsorganizationvalsha2g2

X509v3 Certificate Policies:

Policy: 1.3.6.1.4.1.4146.1.20

CPS: https://www.globalsign.com/repository/

Policy: 2.23.140.1.2.2

X509v3 Basic Constraints:

CA:FALSE

X509v3 CRL Distribution Points:

Full Name:

URI:http://crl.globalsign.com/gs/gsorganizationvalsha2g2.crl

Exercise: Write the first 12 characters of your full name including spaces.

Convert the name into 4\*4 = 16 radix64 symbols.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Text content** | **N** | | | | | | | | u | | | | | | | | r | | | | | | | |
| **ASCII** | 78 (0x4e) | | | | | | | | 117 (0x75) | | | | | | | | 114 (0x72) | | | | | | | |
| **Bit pattern** | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| **Index** | 19 | | | | | | 39 | | | | | | 21 | | | | | | 50 | | | | | |
| **Base64-encoded** | **T** | | | | | | **n** | | | | | | **V** | | | | | | **y** | | | | | |

For an example my name starts from Nur as the first 3 characters.

**LAB / TUTORIAL 1: Radix 64**

a) Take the first 40 characters of your name AND high school name.

b) Convert each character into ascii codes.

c) Encode the ascii codes into Radix64

d) Decode back into 40 visual characters

Make sure write your name and ID and Submit in Word Document

Submit : 10pm on 20.10.2022