Лабораторная работа № 1

Тема: Стандарты кодирования данных.

Цель: ознакомиться со стандартами кодирования данных.

Некоторые из наиболее распространенных стандартов кодирования данных:

1. **Base64** — стандарт кодирования двоичных данных при помощи только 64 символов ASCII. Алфавит кодирования содержит латинские символы A-Z, a-z, цифры 0-9 (всего 62 знака) и 2 дополнительных символа, зависящих от системы реализации. Каждые 3 исходных байта кодируются четырьмя символами (увеличение на ⅓).
2. **Base32**: Аналогично Base64, Base32 представляет двоичные данные в виде текста, используя 32 символа алфавита (A-Z и цифры 2-7). В отличие от Base64, Base32 не использует символы, которые могут быть неудобными в некоторых контекстах, таких как + и /.
3. **ASCII85**: Также известный как Base85, ASCII85 представляет данные как последовательность ASCII-символов. Он использует 85 символов (33-117 в таблице символов ASCII) для представления данных. ASCII85 часто используется в формате кодирования PostScript и PDF для сжатия данных и представления двоичных данных в текстовом формате.
4. **UTF-8**: (Unicode Transformation Format 8-bit) является стандартом кодирования символов Unicode. Он представляет символы Unicode в виде последовательности байтов. UTF-8 использует переменную длину кодирования, что позволяет представлять как ASCII-символы (1 байт), так и символы Unicode (от 2 до 4 байтов). UTF-8 широко используется в веб-разработке и во многих других приложениях для поддержки мультиязычности.
5. **URL-кодирование**: URL-кодирование используется для представления специальных символов в URL-адресах. Он заменяет некоторые символы на последовательности символов, начинающиеся с процента (%), за которыми следуют шестнадцатеричные коды символов. Например, пробелы кодируются как %20, а символы "@" кодируются как %40. URL-кодирование позволяет передавать и обрабатывать URL-адреса, которые содержат специальные символы или несовместимы с URL-форматом.

Base64

Base64 - это способ кодирования данных, который позволяет представить двоичные данные в виде текста, состоящего только из ASCII-символов. Он получил свое название от того, что использует 64 символа алфавита (A-Z, a-z, 0-9 и символы + и /) для представления данных.
Основная цель кодирования Base64 - это представление двоичных данных (байтов) в текстовой форме, чтобы они могли быть переданы или сохранены в текстовом формате, который не поддерживает двоичные данные. Например, Base64 широко используется в электронной почте для кодирования вложенных файлов или в веб-протоколах для

передачи данных, которые могут содержать специальные символы или несовместимы с текстовыми форматами.
Процесс кодирования Base64 работает следующим образом:

1. Двоичные данные (байты) разбиваются на группы по 3 байта.
2. Каждая группа байтов преобразуется в набор из 4 символов ASCII, используя таблицу соответствия Base64.
3. Если исходные данные не делятся на 3 без остатка, производится дополнение нулевыми байтами или добавлением специальных символов заполнения ('=') в конце закодированной строки.
4. Полученные символы объединяются в одну строку, которая представляет закодированные данные.

Процесс декодирования Base64 выполняется в обратном направлении:

1. Закодированная строка разбивается на группы по 4 символа.
2. Каждая группа символов декодируется в соответствующие байты, используя обратную таблицу соответствия Base64.
3. Если в конце закодированной строки присутствуют символы заполнения ('='), они игнорируются или используются для восстановления исходного числа байтов.
4. Полученные байты объединяются в исходные двоичные данные (байты).

Важно понимать, что кодирование Base64 не обеспечивает шифрование или сжатие данных. Оно просто предоставляет способ представления данных в другой форме. Кодированные данные могут быть безопасно переданы в текстовом формате, но они всё равно могут быть прочитаны и использованы любым получателем.

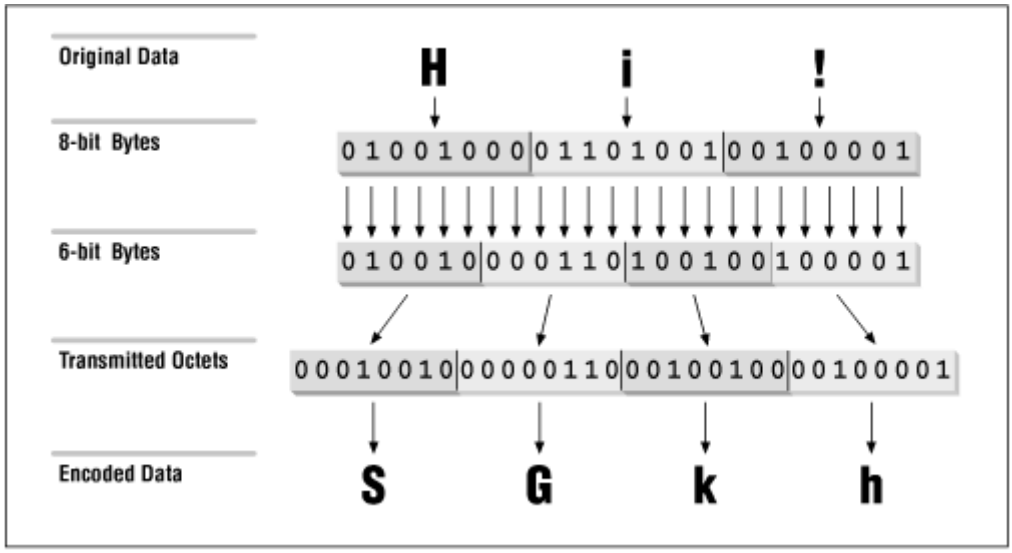
Символ	Значение				Символ	Значение				Символ	Значение				Символ	Значение			
	10	2	8	16		10	2	8	16		10	2	8	16		10	2	8	16
A	0	000000	00	00	Q	16	010000	20	10	g	32	100000	40	20	w	48	110000	60	30
B	1	000001	01	01	R	17	010001	21	11	h	33	100001	41	21	x	49	110001	61	31
C	2	000010	02	02	S	18	010010	22	12	i	34	100010	42	22	y	50	110010	62	32
D	3	000011	03	03	T	19	010011	23	13	j	35	100011	43	23	z	51	110011	63	33
E	4	000100	04	04	U	20	010100	24	14	k	36	100100	44	24	0	52	110100	64	34
F	5	000101	05	05	V	21	010101	25	15	l	37	100101	45	25	1	53	110101	65	35
G	6	000110	06	06	W	22	010110	26	16	m	38	100110	46	26	2	54	110110	66	36
H	7	000111	07	07	X	23	010111	27	17	n	39	100111	47	27	3	55	110111	67	37
I	8	001000	10	08	Y	24	011000	30	18	o	40	101000	50	28	4	56	111000	70	38
J	9	001001	11	09	Z	25	011001	31	19	p	41	101001	51	29	5	57	111001	71	39
K	10	001010	12	0A	a	26	011010	32	1A	q	42	101010	52	2A	6	58	111010	72	3A
L	11	001011	13	0B	b	27	011011	33	1B	r	43	101011	53	2B	7	59	111011	73	3B
M	12	001100	14	0C	c	28	011100	34	1C	s	44	101100	54	2C	8	60	111100	74	3C
N	13	001101	15	0D	d	29	011101	35	1D	t	45	101101	55	2D	9	61	111101	75	3D
O	14	001110	16	0E	e	30	011110	36	1E	u	46	101110	56	2E	+	62	111110	76	3E
P	15	001111	17	0F	f	31	011111	37	1F	v	47	101111	57	2F	/	63	111111	77	3F

При декодировании Base64 могут возникать следующие проблемы:

1. Некорректные символы: если в строке для декодирования Base64 присутствуют некорректные символы, которые не являются символами Base64, процесс декодирования может вызвать ошибку.
   Решение: перед декодированием убедитесь, что строка содержит только допустимые символы Base64. Если в строке есть недопустимые символы, удалите

их или осуществите предварительную обработку строки, чтобы оставить только допустимые символы Base64.

2. Неправильная длина строки: для успешного декодирования Base64 строка должна иметь правильную длину. Длина строки Base64 должна быть кратна 4, и в конце строки может быть дополнительный символ "=". Если длина строки не соответствует этим требованиям, декодирование может завершиться ошибкой.
   Решение: Проверьте, что длина строки Base64 кратна 4. Если длина отличается, добавьте соответствующее количество символов "=" в конец строки, или нули в начале, чтобы сделать ее длину кратной 4.
3. Некорректное заполнение символами "=": в некоторых случаях строка Base64 может содержать символы "=" внутри строки, а не только в конце. Это может привести к ошибке декодирования.
   Решение: перед декодированием убедитесь, что символы "=" присутствуют только в конце строки Base64. Если символы "=" встречаются внутри строки, удалите их или осуществите предварительную обработку строки, чтобы символы "=" находились только в конце строки.
4. Неправильный выбор кодировки: при декодировании Base64 важно использовать правильную кодировку. Если используется неправильная кодировка, результат декодирования может быть некорректным или содержать ошибки.
   Решение: убедитесь, что используемая кодировка соответствует кодировке, которая использовалась при кодировании строк в Base64. В Python стандартная кодировка UTF-8 часто используется для работы с Base64, но в некоторых случаях может потребоваться использование других кодировок, особенно если работа выполняется с не ASCII-символами.



Задание:

1. Скачать дамп lab1.pcap, проанализировать трафик в программе Wireshark, найти секретное послание.
2. Составить отчет.

Отчет должен содержать:
- скриншоты с этапами поиска флагов;
- секретное послание;
- выводы.