DAFTAR PUSTAKA

Akbar, A., & Pangestu, A. (2018, Maret 16). *Algoritma blok cipher OE-CK*. Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung. https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2017-2018/Makalah1-2018/Makalah1-Kripto-2018-10b.pdf

Aziiz, A. K., & Pakereng, M. A. I. (2020). Perancangan teknik kriptografi block cipher berbasis pola batik ceplok Yogyakarta. *Jurnal Sistem dan Teknologi Informasi*, 8(1): Artikel e26208989. http://dx.doi.org/10.26418/justin.v8i1.37135

Buchanan, B. (2021, Oktober 18). So what is PKCS#7. *Medium*. So What Is PKCS#7?. A symmetric key block cipher, such as… | by Prof Bill Buchanan OBE | ASecuritySite: When Bob Met Alice | Medium

Fauzi, R. R., & Wellem, T. (2021). Perancangan kriptografi *block cipher* berbasis pola *dribbling practice*. *Jurnal Teknologi Informasi*, 18(1): Artikel e26157128. https://doi.org/10.24246/aiti.v18i2.158-172

Harahap, M.K. (2016). Analisis perbandingan algoritma kriptografi klasik vigenera cipher dan one time pad. *Jurnal Nasional Informatika dan Teknologi Jaringan*, 1(1): Artikel e25407600. http://dx.doi.org/10.26418/justin.v8i1.37135

Henry., Kridalaksana, A.H., Arifin, Z. (2016). Kriptografi AES mode CBC pada citra digital berbasis android. *Prosiding Seminar Ilmu Komputer dan Teknologi Informasi*, 1(1): Artikel e25407902. https://e-journals.unmul.ac.id/index.php/SAKTI/article/view/150

Hernowo, M.B., Seta, H.B., & Widi. I. W.P. (2020, November 19-20). *Pengamanan soal ujian sekolah computer based test (CBT) dengan algoritma advance encryption standard (AES) dan metode steganografi End of File (EoF)*. Seminar Nasional Informatika, Sistem Informasi dan Keamanan Siber. https://conference.upnvj.ac.id/index.php/seinasikesi/article/download/775/pdf

Hidayatullah, A., & Insanudin, E. (2016). Pengenalan kriptografi dan pemakaianya sehari-hari. *Jurnal Pengenalan kriptografi dan pemakaianya sehari-hari*. https://www.researchgate.net/profile/publication/303374537_PENGENALAN_KRIPTOGRAFI_DAN_PEMAKAIANYA_SEHARI-HARI

Hidayatulloh, N.W., Tahir, M., Amalia, H., Basyar, N.A., Prianggara, A. F., & Yasin, M. (2023). Mengenal *Advance Encryption Standard* (AES) sebagai algoritma kriptografi dalam mengamankan data. *Digital Transformation Teknology (Digitech)*, 3(1): Artikel e28079000. https://doi.org/1047709/digitech.v3i1.2293

Ibad, M. N., Alqoroni, S., Ridho, M. A., & Holle, K. F. H. (2020). Pengembangan aplikasi *Computer Based Test* dengan protokol *two central facilities*. *JISKa*, 4(3): Artikel e25280074. http://repository.uin-malang.ac.id/5929/1/5929.pdf

Kennedy. (2021). *Implementasi teknologi blokchain untuk pengiriman jawaban CBT menggunakan RSA-SHA 256*. Repository Universitas Hasanuddin. http://repository.unhas.ac.id/12795/2/H071171516_skripsi_02-12-2021%20bab%201-2.pdf

Kurniawan, F. (2018). *Diktat teknik digital : sistem bilangan dan representasi data*. Sekolah Tinggi Teknologi Adisutjipto. https://www.researchgate.net/publication/324152018_Diktat_Teknik_Digital_Sistem_Bilangan_dan_Representasi_Data

Marino, S. (2007). *Perbandingan antara kriptografi modern dengan kriptografi kuantum*. Institut Teknologi Bandung. https://www.academia.edu/23996047/Perbandingan_Antara_Kriptografi_Modern_dengan_Kriptografi_Kuantum

Murniati, E. (2015). *Computer Based Test (CBT) sebagai alternatif instrument evaluasi pembelajaran*. Universitas Sebelas Maret Surakarta. https://jurnal.fkip.uns.ac.id/index.php/snpe/article/download/10647/7893

Rosmasari., Agus, F., & Khairumman, F. (2020). Implementasi algoritma blowfish pada aplikasi CBT berbasis mobile android (studi kasus : FKTI

Universitas Mulawarman). *Jurnal Teknologi Informasi*, 4(1): Artikel e2579-8790. https://e-journals.unmul.ac.id/index.php/INF/article/view/4514

Sajati, H., Ayuningtyas, A., & Kholistyanto, D. (2017). Penerapan eigenface untuk computer based test (CBT) penerimaan mahasiswa baru Sekolah Tinggi Teknologi Adisutjipto. *Compiler*, 6(2). https://ejournals.itda.ac.id/index.php/compiler/article/view/228

Senjaya, W.F., & Rahardjo, B. (2015). Implementasi dan pengukuran kinerja operasi aritmatika *finite field* berbasis polinomial biner. *Jurnal Teknik Informasi dan Sistem Informasi,* 1(2): Artikel e24432229. https://media.neliti.com/media/publications/134562-ID-implementasi-dan-pengukuran-kinerja-oper.pdf

Syaifuddin, R., Faza., Diah., & Hamidah. N. (2022). Analisis pemanfaatan aplikasi CBT sebagai sarana tes di MI Badrussalam Surabaya. *Jurnal PTK dan Pendidikan*, 8(2), 79-84. https://doi.org/10.18592/ptk.v%vi%i.7569

LAMPIRAN A

ASCII

Dec	Heksa	Bin	Char	Dec	Heksa	Bin	Char
1	1	0000 0001		129	81	1000 0001	
2	2	0000 0010		130	82	1000 0010	
3	3	0000 0011		131	83	1000 0011	
4	4	0000 0100		132	84	1000 0100	
5	5	0000 0101		133	85	1000 0101	
6	6	0000 0110		134	86	1000 0110	
7	7	0000 0111		135	87	1000 0111	
8	8	0000 1000		136	88	1000 1000	
9	9	0000 1001		137	89	1000 1001	
10	A	0000 1010		138	8A	1000 1010	
11	B	0000 1011		139	8B	1000 1011	
12	C	0000 1100		140	8C	1000 1100	
13	D	0000 1101		141	8D	1000 1101	
14	E	0000 1110		142	8E	1000 1110	
15	F	0000 1111		143	8F	1000 1111	
16	10	0001 0000		144	90	1001 0000	
17	11	0001 0001		145	91	1001 0001	
18	12	0001 0010		146	92	1001 0010	
19	13	0001 0011		147	93	1001 0011	
20	14	0001 0100		148	94	1001 0100	
21	15	0001 0101		149	95	1001 0101	
22	16	0001 0110		150	96	1001 0110	
23	17	0001 0111		151	97	1001 0111	
24	18	0001 1000		152	98	1001 1000	
25	19	0001 1001		153	99	1001 1001	
26	1A	0001 1010		154	9A	1001 1010	
27	1B	0001 1011		155	9B	1001 1011	
28	1C	0001 1100		156	9C	1001 1100	
29	1D	0001 1101		157	9D	1001 1101	
30	1E	0001 1110		158	9E	1001 1110	
31	1F	0001 1111		159	9F	1001 1111	
32	20	0010 0000		160	A0	1010 0000	
33	21	0010 0001	!	161	A1	1010 0001	
34	22	0010 0010	"	162	A2	1010 0010	
35	23	0010 0011	#	163	A3	1010 0011	
36	24	0010 0100	$	164	A4	1010 0100	
37	25	0010 0101	%	165	A5	1010 0101	
38	26	0010 0110	&	166	A6	1010 0110	
39	27	0010 0111	`	167	A7	1010 0111	
40	28	0010 1000	(	168	A8	1010 1000	

41	29	0010 1001	)	169	A9	1010 1001	
42	2A	0010 1010	*	170	AA	1010 1010	
43	2B	0010 1011	+	171	AB	1010 1011	
44	2C	0010 1100	,	172	AC	1010 1100	
45	2D	0010 1101	-	173	AD	1010 1101	
46	2E	0010 1110	.	174	AE	1010 1110	
47	2F	0010 1111	/	175	AF	1010 1111	
48	30	0011 0000	0	176	B0	1011 0000	
49	31	0011 0001	1	177	B1	1011 0001	
50	32	0011 0010	2	178	B2	1011 0010	
51	33	0011 0011	3	179	B3	1011 0011	
52	34	0011 0100	4	180	B4	1011 0100	
53	35	0011 0101	5	181	B5	1011 0101	
54	36	0011 0110	6	182	B6	1011 0110	
55	37	0011 0111	7	183	B7	1011 0111	
56	38	0011 1000	8	184	B8	1011 1000	
57	39	0011 1001	9	185	B9	1011 1001	
58	3A	0011 1010	:	186	BA	1011 1010	
59	3B	0011 1011	;	187	BB	1011 1011	
60	3C	0011 1100	<	188	BC	1011 1100	
61	3D	0011 1101	"	189	BD	1011 1101	
62	3E	0011 1110	>	190	BE	1011 1110	
63	3F	0011 1111	?	191	BF	1011 1111	
64	40	0100 0000	@	192	C0	1100 0000	
65	41	0100 0001	A	193	C1	1100 0001	
66	42	0100 0010	B	194	C2	1100 0010	
67	43	0100 0011	C	195	C3	1100 0011	
68	44	0100 0100	D	196	C4	1100 0100	
69	45	0100 0101	E	197	C5	1100 0101	
70	46	0100 0110	F	198	C6	1100 0110	
71	47	0100 0111	G	199	C7	1100 0111	
72	48	0100 1000	H	200	C8	1100 1000	
73	49	0100 1001	I	201	C9	1100 1001	
74	4A	0100 1010	J	202	CA	1100 1010	
75	4B	0100 1011	K	203	CB	1100 1011	
76	4C	0100 1100	L	204	CC	1100 1100	
77	4D	0100 1101	M	205	CD	1100 1101	
78	4E	0100 1110	N	206	CE	1100 1110	
79	4F	0100 1111	O	207	CF	1100 1111	
80	50	0101 0000	P	208	D0	1101 0000	
81	51	0101 0001	Q	209	D1	1101 0001	
82	52	0101 0010	R	210	D2	1101 0010	
83	53	0101 0011	S	211	D3	1101 0011	
84	54	0101 0100	T	212	D4	1101 0100	

85	55	0101 0101	U	213	D5	1101 0101	
86	56	0101 0110	V	214	D6	1101 0110	
87	57	0101 0111	W	215	D7	1101 0111	
88	58	0101 1000	X	216	D8	1101 1000	
89	59	0101 1001	Y	217	D9	1101 1001	
90	5A	0101 1010	Z	218	DA	1101 1010	
91	5B	0101 1011	[	219	DB	1101 1011	
92	5C	0101 1100	\	220	DC	1101 1100	
93	5D	0101 1101	]	221	DD	1101 1101	
94	5E	0101 1110	'	222	DE	1101 1110	
95	5F	0101 1111	_	223	DF	1101 1111	
96	60	0110 0000	`	224	E0	1110 0000	
97	61	0110 0001	a	225	E1	1110 0001	
98	62	0110 0010	b	226	E2	1110 0010	
99	63	0110 0011	c	227	E3	1110 0011	
100	64	0110 0100	d	228	E4	1110 0100	
101	65	0110 0101	e	229	E5	1110 0101	
102	66	0110 0110	f	230	E6	1110 0110	
103	67	0110 0111	g	231	E7	1110 0111	
104	68	0110 1000	h	232	E8	1110 1000	
105	69	0110 1001	i	233	E9	1110 1001	
106	6A	0110 1010	j	234	EA	1110 1010	
107	6B	0110 1011	k	235	EB	1110 1011	
108	6C	0110 1100	l	236	EC	1110 1100	
109	6D	0110 1101	m	237	ED	1110 1101	
110	6E	0110 1110	n	238	EE	1110 1110	
111	6F	0110 1111	o	239	EF	1110 1111	
112	70	0111 0000	p	240	F0	1111 0000	
113	71	0111 0001	q	241	F1	1111 0001	
114	72	0111 0010	r	242	F2	1111 0010	
115	73	0111 0011	s	243	F3	1111 0011	
116	74	0111 0100	t	244	F4	1111 0100	
117	75	0111 0101	u	245	F5	1111 0101	
118	76	0111 0110	v	246	F6	1111 0110	
119	77	0111 0111	w	247	F7	1111 0111	
120	78	0111 1000	x	248	F8	1111 1000	
121	79	0111 1001	y	249	F9	1111 1001	
122	7A	0111 1010	z	250	FA	1111 1010	
123	7B	0111 1011	{	251	FB	1111 1011	
124	7C	0111 1100	|	252	FC	1111 1100	
125	7D	0111 1101	}	253	FD	1111 1101	
126	7E	0111 1110		254	FE	1111 1110	
127	7F	0111 1111		255	FF	1111 1111	
128	80	1000 0000					

LAMPIRAN B

KODE ENKRIPSI MODE ECB

```
from cryptography.hazmat.primitives.ciphers import Cipher,
algorithms, modes
from cryptography.hazmat.backends import default_backend
from cryptography.hazmat.primitives import padding
import base64

def pad(data, block_size):
    padder = padding.PKCS7(block_size * 8).padder()
    padded_data = padder.update(data) + padder.finalize()
    return padded_data

def panjang_kunci(kunci_awal, length):
    # Jika kunci sudah sesuai panjang, gunakan langsung
    if len(kunci_awal) == length // 8:
        return kunci_awal
    # Jika kunci terlalu panjang, potong sesuai panjang yang
diinginkan
    elif len(kunci_awal) > length // 8:
        return kunci_awal[:length // 8]
    # Jika kunci terlalu pendek, lakukan padding dengan nol
    else:
        return kunci_awal.ljust(length // 8, b'\x00')

def enkripsi_ECB_PKCS7(plaintext, key):
    block_size = algorithms.AES.block_size // 8  # AES block
size in bytes
    cipher = Cipher(algorithms.AES(key), modes.ECB(),
backend=default_backend())

    # Padding plaintext using PKCS7
    padded_plaintext = pad(plaintext, block_size)
```

```
    enkripsi = cipher.encryptor()
    ciphertext    =    enkripsi.update(padded_plaintext)    +
enkripsi.finalize()

    return base64.b64encode(ciphertext)

# Contoh penggunaan
plaintext = b'ABCDEFAB'
kunci_awal = b'00010111'

# Pilih panjang bit kunci yang diinginkan (misalnya, 128,
192, atau 256)
panjang_bit_kunci = 128

# Derivasi kunci
kunci_asli = panjang_kunci(kunci_awal, panjang_bit_kunci)

cipher_text = enkripsi_ECB_PKCS7(plaintext, kunci_asli)
      print("Cipher Text:", cipher_text.decode('utf-8'))
```

LAMPIRAN C
KODE ENKRIPSI MODE CBC

```
import hashlib
from cryptography.hazmat.primitives.ciphers import Cipher,
algorithms, modes
from cryptography.hazmat.primitives import padding
kunci = b'00010111'
kunci_asli = hashlib.sha256(kunci).digest()
iv = b'123456789O987654' #yang menghasilkan vektor
inisialisasi 128-bit secara acak
cipher = Cipher(algorithms.AES(kunci_asli), modes.CBC(iv))
enkripsi = cipher.encryptor()
pad = padding.PKCS7(128).padder()
plainteks = b'ABCDEFAB'
padding_plainteks = pad.update(plainteks) + pad.finalize()
enkripsi_plainteks = enkripsi.update(padding_plainteks) +
enkripsi.finalize()
print("enkripsi plainteks mode CBC : ", enkripsi_plainteks)
```

LAMPIRAN D

KODE ENKRIPSI MODE CFB

```
import hashlib
from cryptography.hazmat.primitives.ciphers import Cipher,
algorithms, modes
from cryptography.hazmat.primitives import padding
kunci = b'00010111'
kunci_asli = hashlib.sha256(kunci).digest()
iv = b'123456789O987654' #yang menghasilkan vektor
inisialisasi 128-bit secara acak
cipher = Cipher(algorithms.AES(kunci_asli), modes.CFB(iv))
enkripsi = cipher.encryptor()
pad = padding.PKCS7(128).padder()
plainteks = b'ABCDEFAB'
padding_plainteks = pad.update(plainteks) + pad.finalize()
enkripsi_plainteks = enkripsi.update(padding_plainteks) +
enkripsi.finalize()
print("enkripsi plainteks mode CFB : ", enkripsi_plainteks)
```

LAMPIRAN E
KODE ENKRIPSI MODE OFB

```
import hashlib
from cryptography.hazmat.primitives.ciphers import Cipher,
algorithms, modes
from cryptography.hazmat.primitives import padding
kunci = b'00010111'
kunci_asli = hashlib.sha256(kunci).digest()
iv = b'123456789O987654' #yang menghasilkan vektor
inisialisasi 128-bit secara acak
cipher = Cipher(algorithms.AES(kunci_asli), modes.OFB(iv))
enkripsi = cipher.encryptor()
pad = padding.PKCS7(128).padder()
plainteks = b'ABCDEFAB'
padding_plainteks = pad.update(plainteks) + pad.finalize()
enkripsi_plainteks = enkripsi.update(padding_plainteks) +
enkripsi.finalize()
print("enkripsi plainteks mode OFB : ", enkripsi_plainteks)
```