APPENDIX

CODING RSA ENCRYPTION AND DECRYPTION

```
1. /*Inspired    by    http://www.trytoprogram.com/cpp-examples/cplusplus-
   program-encrypt-decrypt-string/ */
2. #include<string.h>
3. #include<math.h>
4.
5. int p, q;
6. int n, fn;
7. int i;
8. int check;
9. long int e[50]; //Untuk menqimpan Public key
10. long int d[50]; //Untuk menqimpan Private key
11. long int temp[50];
12. long int j;
13. char c[50]; //Ciphertext
14. char m[50]; //Plaintext
15. String msg = ""; //Input pesan
16. int check_prime(long int); //Untuk cek angka prima
17. long int cd(long int); //Untuk cek nilai K
18. void encryption_key();
19. void encrypt();
20. void decrypt();

21. void setup() {
22.   Serial.begin(9600);
23.   Serial.println("First Prime Number");
24.   p = 7;
25.   Serial.println(p);
26.   check = check_prime(p);
27.   if (check == 1){
28.       Serial.println("P is a Prime Number");
29.     } else{
30.        Serial.println("P is NOT a Prime Number");
31.        exit(0);
32.       }
33.
34.   Serial.println("Second Prime Number");
35.   q = 13;
36.   Serial.println(q);
37.   check = check_prime(q);
38.     if (check == 1){
39.       Serial.println("Q is a Prime Number");
40.     } else{
41.        Serial.println("Q is NOT a Prime Number");
42.        exit(0);
43.       }
44.   Serial.println("Enter Message or String To Encrypt\n");
45.   msg = "whatchamacallits";
46.   Serial.println(msg);
47.   int msg_len = msg.length()+1;
48.   char msg_arr[16];
49.   msg.toCharArray(msg_arr, msg_len);
```

```
50.    for(i=0; msg_arr[i]!=0 ; i++)
51.    {
52.        m[i] = msg [i];
53.      }
54.    n = p*q; //Untuk cari nilai N
55.    fn = ((p-1)*(q-1)); //Untuk mencari nilai dari private key
56.    encryption_key();
57.    unsigned long time1 = micros();
58.    encrypt();
59.    Serial.println("\nWaktu Encrypt : ");
60.    Serial.print(micros() - time1);
61.    unsigned long time2 = micros();
62.    decrypt();
63.    Serial.println("\nWaktu Decrypt : ");
64.    Serial.print(micros() - time2);
65. }
66.
67. void loop() {
68. }
69.
70. int check_prime(long int pri){
71.     int i;
72.     for(i = 2; i <= pri -1; i++){
73.         if(pri % i == 0)
74.           return 0;
75.       }
76.       if(i == pri)
77.         return 1;
78.   }
79.
80. //Function untuk menghasilkan public dan private key
81. void encryption_key()
82. {
83.    int k;
84.    k = 0;
85.    for(i = 2; i < fn; i++)
86.    {
87.       if(fn % i == 0)
88.          continue;
89.       check = check_prime(i);
90.       if(check == 1 && i != p && i != q)
91.       {
92.          e[k] = i;
93.          check = cd(e[k]);
94.          if(check > 0)
95.          {
96.             d[k] = check;
97.             k++;
98.          }
99.          if(k == 99)
100.         break;
101.      }
102.   }
103.}
104.long int cd (long int a){
105.    long int k = 1;
```

```
106.    while(1){
107.        k = k + fn;
108.        if(k % a == 0)
109.          return(k/a);
110.      }
111.  }
112.void encrypt(){
113.    long int pt, ct, key = e[0];
114.    long int k, len;
115.    int i = 0;
116.    len = msg.length();
117.
118.    while(i != len)
119.    {
120.      pt = m[i];
121.      pt = pt - 96; //Untuk mencegah character melebihi batas unsigned
   dan menjaga valua tetap dalam range
122.      k = 1;
123.      for(j = 0; j < key; j++){
124.          k = k * pt;
125.          k = k % n;
126.        }
127.        temp[i] = k;
128.        ct = k + 96;
129.        c[i] = ct;
130.        i++;
131.    }
132.    c[i] = -1;
133.    Serial.println("\nThe Encrypted Message Is");
134.    for(i = 0; c[i] != -1; i++){
135.      Serial.print(c[i]);
136.    }
137.    Serial.println("\nThe Encrypted Message In HEX");
138.   for (int i=0; c[i] != -1 ;i++){
139.    Serial.println(c[i]&0xFF,HEX);
140.    }
141.    Serial.println("-----END OF HEX-----");
142.  }
143.void decrypt(){
144.   long int pt, ct, key = d[0];
145.   long int k;
146.   int i = 0;
147.   while(c[i] != -1)
148.   {
149.      ct = temp[i];
150.      k = 1;
151.      for(j = 0; j < key; j++)
152.      {
153.         k = k * ct;
154.         k = k % n;
155.      }
156.      pt = k + 96;
157.      m[i] = pt;
158.      i++;
159.   }
160.   m[i] = -1;
```

```
161.    Serial.println("\nThe Decrypted Message Is");
162.    for(i = 0; m[i] != -1; i++){
163.       Serial.print(m[i]);
164.     }
165.
166.}
```

CODING AES ENCRYPTION AND DECRYPTION

```
1.  #include<AESLib.h>
2.
3.  void setup() {
4.    Serial.begin(9600);
5.    //uint8_t key[] = {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15};
6.    uint8_t                              key[]                                    =
   {'t','e','k','n','i','k','i','n','f','o','r','m','a','t','i','k'};
7.    char msg[] = "D3E2F2G2H2I2J3__"; //16 chars == 16 bytes
8.    unsigned long time1 = micros();
9.    aes128_enc_single(key, msg);
10.   Serial.print("Encrypted: ");
11.   Serial.println();
12.   Serial.println(msg);
13.   Serial.println("\nWaktu Encrypt : ");
14.   Serial.print(micros() - time1);
15.   Serial.println("\nConvert to HEX :");
16.   Serial.println();
17.   for (int i=0; msg[i] != 0 ;i++){
18.   Serial.println(msg[i]&0xFF,HEX);
19.    }
20.   unsigned long time2 = micros();
21.   aes128_dec_single(key, msg);
22.   Serial.print("Decrypted: ");
23.   Serial.println();
24.   Serial.println(msg);
25.   Serial.println("\nWaktu Decrypt : ");
26.   Serial.print(micros() - time2);
27.}
28.
29.void loop() {
30.}
```

CODING COMPRESSION ALGORITHM

```
1. /*Modified from https://stackoverflow.com/questions/14037263/how-to-
   compress-a-string-and-replace-duplicates-with-its-count-using-c */
2. #include<stdio.h>
3.
4. void setup() {
5.   Serial.begin(9600);
6.   char myStr[250] = "Sebagai informasi, Starlink merupakan proyek yang
   dikembangkan SpaceX sejak 2015.";
7.   char saved[250];
8.   int i;
9.   while(i < sizeof(myStr)-1) {
10.           saved[i] = myStr[i];
11.         i++;
12.     }
13.
14.   Serial.println("Original String is : ");
15.   Serial.print(myStr);
16.   Serial.println();
17.   Serial.println("Compressed String is : ");
18.   unsigned long time1 = micros();
19.   Serial.print(StrCompress(saved));
20.   Serial.println("\nWaktu Compress : ");
21.   Serial.print(micros() - time1);
22.   Serial.println();
23.   Serial.println("Decompressed String is : ");
24.   //String comp = "A4B3C2";
25.   String comp(saved);
26.   int len = comp.length();
27.   int x = comp.toInt();
28.   String simpan ="";
29.   String temp;
30.   int temp1;
31.   String decomp = "";
32.   unsigned long time2 = micros();
33.   for(i=0; i<len; i++){
34.       if(i % 2 == 0){
35.           simpan = comp.charAt(i);
36.         }
37.         else{
38.           temp = comp.charAt(i);
39.           temp1 = temp.toInt();
40.             for(int x=0; x< temp1; x++){
41.                 decomp = decomp + simpan;
42.               }
43.           }
44.     }
45. Serial.println(decomp);
46. Serial.println("Waktu Decompress : ");
47. Serial.print(micros() - time2);
48. }
49. void loop() {
50.
51.
52. }
```

```
53.
54. char* StrCompress(char saved[])
55. {
56.     char *s = saved;
57.     char *r, *p;
58.     int count, i;
59.
60.     while (*s)
61.     {
62.         /*Mulai dari karakter pertama*/
63.         count = 1;
64.         /*Cek apabila karakter pada posisi pointer sama dengan karakter
    selanjutnya*/
65.         while (*s && *s == *(s+1))
66.         {
67.             /*Jika iya, tambahkan di variabel count dan geser pointer
    ke karakter selanjutnya*/
68.             count++;
69.             s++;
70.         }
71.         if (count > 1) /*Jika lebih dari satu karakter yg telah di hitung
    ditemukan*/
72.         {
73.             /*Tetapkan hitungan ke kemunculan kedua dari karakter
    tertentu*/
74.             *(s - count + 2) = count + '0';
75.             /*Hapus semua kemunculan karakter lain kecuali yang pertama
    dan yang kedua menggunakan array shift*/
76.             for (i = 0; i < count - 2; i++)
77.             {
78.                 p = s + 1;
79.                 r = s;
80.
81.                 while (*r)
82.                     *r++ = *p++;
83.                 s--;
84.             }
85.         }
86.         s++;
87.     }
88.     return saved;
89. }
```

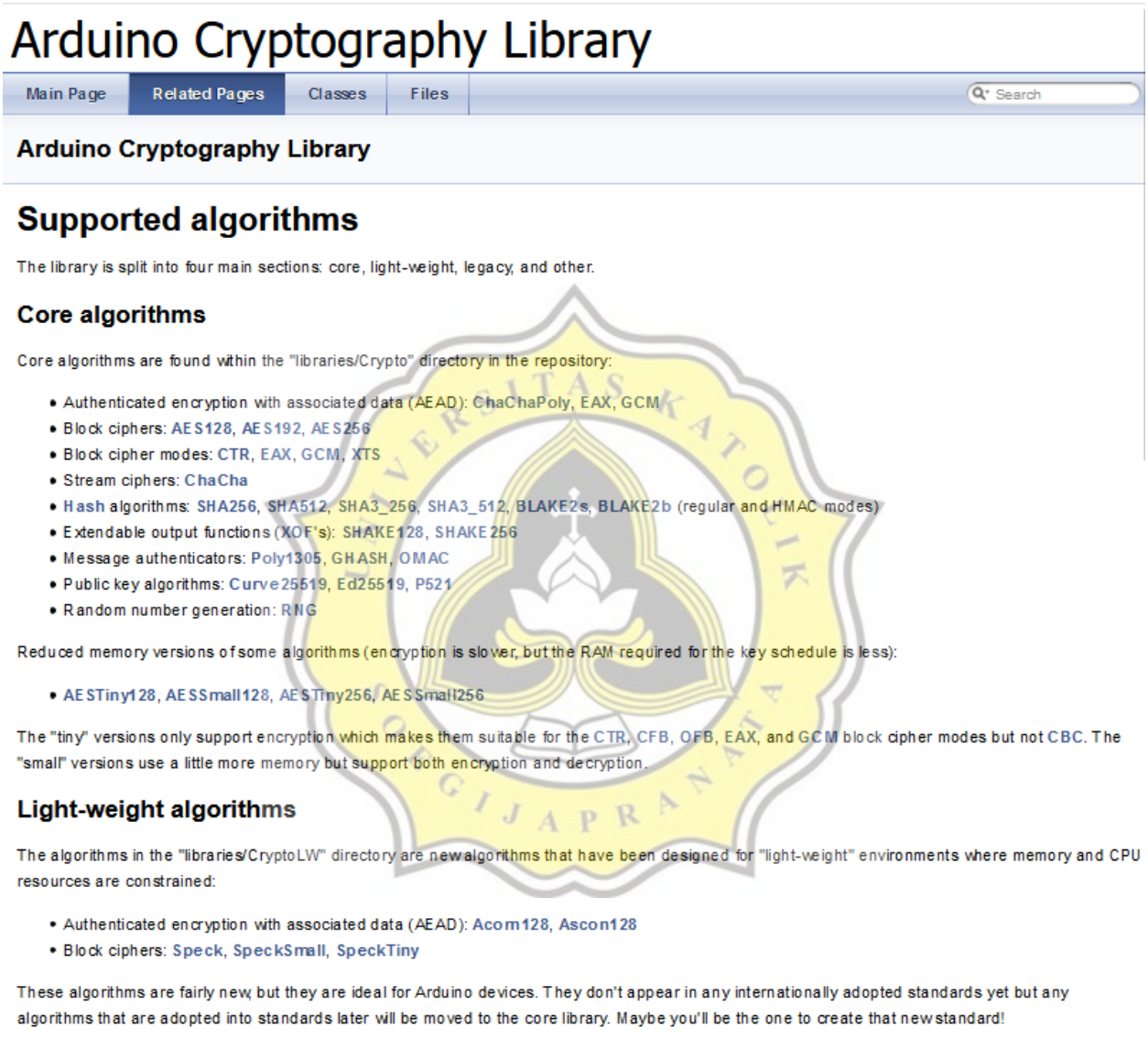
CODING RANDOM COMPRESSION

```
1. const int len = 16;
2. char string[6] = {'A','B','C','D','E','F'};
3. const byte stlen = sizeof(string) / sizeof(string[0]); //menghitung size
   dari array
4. char notes[len+1]; // ditambah 1 untuk NULL
5. unsigned long time1 = micros();
6. void setup() {
7.   Serial.begin(9600);
8.   randomSeed(analogRead(A0));
9.   for (int n = 0; n < 16 ; n++)
10.   {
```

```
    notes[n] = string[random(stlen)];
    notes[n + 1] = '\0'; //untuk terminate string
  }
}

void loop() {
  //Serial.println(string[random(3)]);
  Serial.println("Original String is : ");
  Serial.println(notes);
  Serial.println("Compressed String is : ");
  Serial.print(StrCompress(notes));
  Serial.println("\nWaktu Compress : ");
  Serial.print(micros() - time1);
  Serial.println();

  delay(1000);
  exit(0);
}

char* StrCompress(char notes[])
{
    char *s = notes;
    char *r, *p;
    int count, i;

    while (*s)
    {
        /*Mulai dari karakter pertama*/
        count = 1;
        /*Cek apabila karakter pada posisi pointer sama dengan karakter selanjutnya*/
        while (*s && *s == *(s+1))
        {
            /*Jika iya, tambahkan di variabel count dan geser pointer ke karakter selanjutnya*/
            count++;
            s++;
        }
        if (count > 1) /*Jika lebih dari satu karakter yg telah di hitung ditemukan*/
        {
            /*Tetapkan hitungan ke kemunculan kedua dari karakter tertentu*/
            *(s - count + 2) = count + '0';
            /*Hapus semua kemunculan karakter lain kecuali yang pertama dan yang kedua menggunakan array shift*/
            for (i = 0; i < count - 2; i++)
            {
                p = s + 1;
                r = s;

                while (*r)
                    *r++ = *p++;
                s--;
            }
        }
```

```
62.          s++;
63.      }
64.      return notes;
65.  }
```

BUKTI RSA TIDAK TERDAFTAR PADA LIBRARY ARDUINO



Arduino Cryptography Library

Main Page | Related Pages | Classes | Files | Search

Arduino Cryptography Library

Supported algorithms

The library is split into four main sections: core, light-weight, legacy, and other.

Core algorithms

Core algorithms are found within the "libraries/Crypto" directory in the repository:

- Authenticated encryption with associated data (AEAD): ChaChaPoly, EAX, GCM
- Block ciphers: AES128, AES192, AES256
- Block cipher modes: CTR, EAX, GCM, XTS
- Stream ciphers: ChaCha
- Hash algorithms: SHA256, SHA512, SHA3_256, SHA3_512, BLAKE2s, BLAKE2b (regular and HMAC modes)
- Extendable output functions (XOF's): SHAKE128, SHAKE256
- Message authenticators: Poly1305, GHASH, OMAC
- Public key algorithms: Curve25519, Ed25519, P521
- Random number generation: RNG

Reduced memory versions of some algorithms (encryption is slower, but the RAM required for the key schedule is less):

- AESTiny128, AESSmall128, AESTiny256, AESSmall256

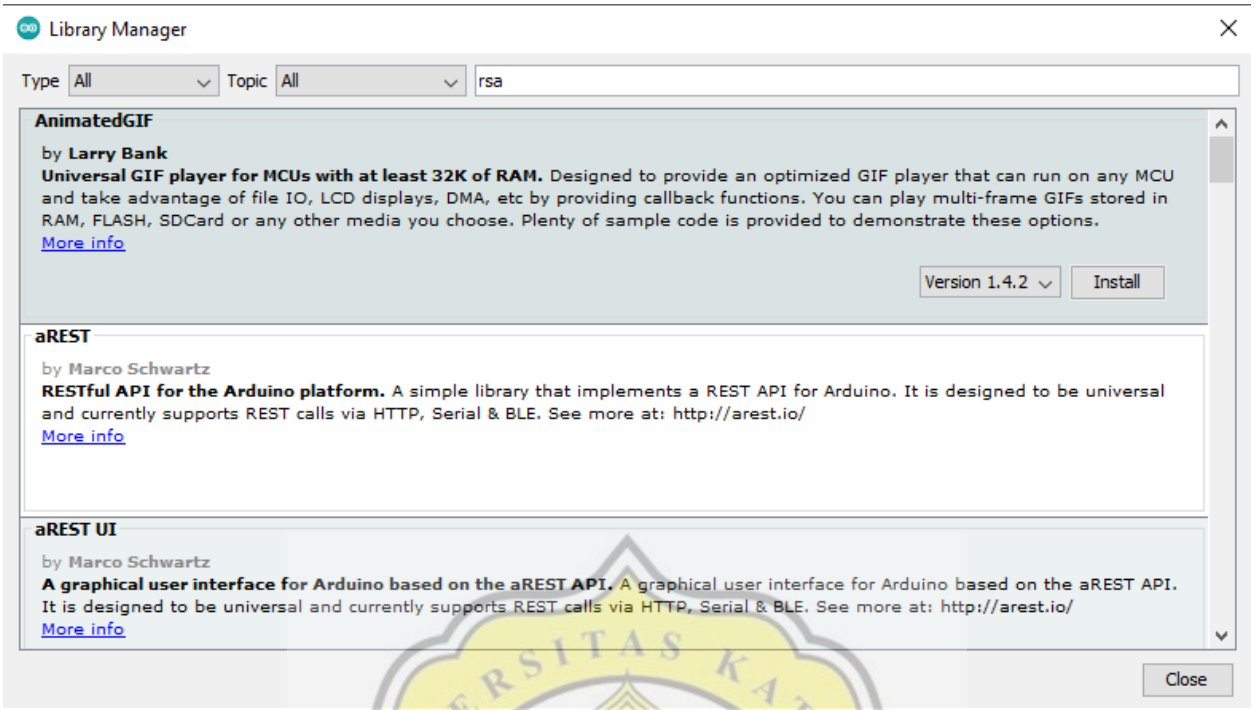
The "tiny" versions only support encryption which makes them suitable for the CTR, CFB, OFB, EAX, and GCM block cipher modes but not CBC. The "small" versions use a little more memory but support both encryption and decryption.

Light-weight algorithms

The algorithms in the "libraries/CryptoLW" directory are new algorithms that have been designed for "light-weight" environments where memory and CPU resources are constrained:

- Authenticated encryption with associated data (AEAD): Acorn128, Ascon128
- Block ciphers: Speck, SpeckSmall, SpeckTiny

These algorithms are fairly new, but they are ideal for Arduino devices. They don't appear in any internationally adopted standards yet but any algorithms that are adopted into standards later will be moved to the core library. Maybe you'll be the one to create that new standard!



BUKTI ANTIPLAGIASI

PLAGIARISM CHECK.ORG

1.42% PLAGIARISM APPROXIMATELY

Report #13361377

BAB I INTRODUCTION Background The Radio Frequency Identification Technology (RFID) was invented during the war to identify whether the planes belong to fellow British soldiers or an enemy of the British army during World War II [1]. Nowadays, RFID has come along the way to mainstream commercial usage accompanied by an advance in technology. Nevertheless, along with easy access to RFID technology today, security concerns have increased from past years. People could quickly build an RFID reader with minimal cost and read RFID tags to extract data inside the chip. For example, someone could copy and clone the data inside an RFID chip using a handmade RFID reader and use it to unlock an unsecured RFID implemented door or gate using RFID using cloned RFID tags. This scenario will create a new problem when the data inside the chip is essential, valuable information of the said victim. To secure the data inside the RFID chip when someone unauthorized tries to access, read, or clone the RFID

REPORT #133613777 CHECKED JUL 2021, 8:30 AM | AUTHOR ANDRE KURNIAWAN | PAGE 1 OF 46