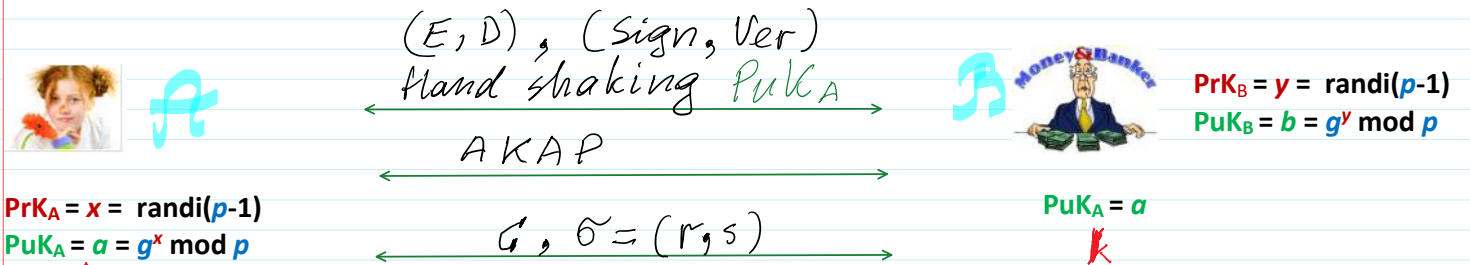


Mini-https

Confidential, Integral, Authentic

Public Parameters  $PP = (p, g)$ .  
 $p = 268435019$ ;  $g = 2$ ;



$E(k, T) = C$   
 Encrypt & sign paradigm  
 Chosen ciphertext security  
 CCS  
 $h_c = H(C)$   
 $Sign(PrK=x, h_c) = \sigma = (r, s)$

1.  $Ver(PuK_A=a, \sigma) = \{T, F\}$
2.  $D(k, C) = T$
3. Performs money transf.

By realizing Schnorr - sign  
 $i \leftarrow \text{randi}(p-1)$   
 $r = g^i \text{ mod } p$   
 $h_c = H(C || r)$   
 $s = (i + x \cdot h_c) \text{ mod } p$

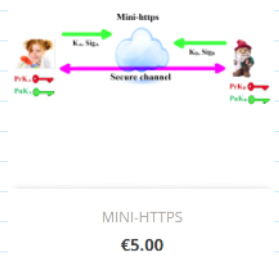
After receiving  $M'$  and  $\sigma$ , Bob according to (2.20) computes  $h' = H(M' || r)$ , and verifies if

$$g^s \text{ mod } p = r a^{h'} \text{ mod } p. \quad (2.22)$$

V1      V2

Symbolically this verification function we denote by  $Ver(a, \sigma, h') = V \in \{True, False\} = \{1, 0\}$ . (2.23)

This function yields True if (2.22) is valid if:  $h = h'$  and  $PuK_A = a = F(PrK_A) = g^x \text{ mod } p$  and:  $M = M'$



1. Mentor sends you Public Parameters ( $p = 15728303$ ;  $g = 5$ ) of 24 bits length. Generate public and private keys  $PrK_A = x$  and  $PuK_A = a$ . Send public key [a] to the Mentor.

```

>> p=int64(15728303)
p = 15728303
>> g=5;
>> x=int64(randi(p-1))
x = 9712179
>> a=mod_exp(g,x,p)
13777229
    
```

12677229

$PuK_A = a$ . Send public key [a] to the Mentor.

12677229

2. Compute random secret number  $u$  of 24 bit length and compute session public parameter  $K_A$ . Sign  $K_A$  with Schnorr signature scheme by computing two components of signature  $Sig_A = (r, s)$ . Be aware that parameter  $h = hd24(concat(K_A, r))$ . Send  $[K_A, r, s]$  to the Mentor.

$h$  of 28

14438572,14538340,13707662

```
>> x=int64(randi(p-1))
x = 9712179
>> a=mod_exp(g,x,p)
a = 12677229
```

```
>> u=int64(randi(p-1))
u = 7085010
>> KA=mod_exp(g,u,p)
KA = 14438572
>> i=int64(randi(p-1))
i = 5104007
>> r=mod_exp(g,i,p)
r = 14538340
>> conc=concat(KA,r)
conc = 1443857214538340

hd28
>> h=int64(hd24(conc))
h = 12720923
>> hh=int64(hd24(concat(KA,r)))
ans = 12720923
>> xh=mod(x*h,p-1)
xh = 8603655
>> s=mod(i+xh,p-1)
s = 13707662
```

3. Mentor sends you ( $PuK_B=4670305, K_B=918922, R=8070925, S=6944326$ ). Verify Mentor's signature  $Sig_B = (R, S)$  on  $K_B$ . If signature is valid then compute secret session symmetric key  $K_{AB} = K$ . Transform  $K$  to the hexadecimal form  $K_h$ . Create the string of message variable  $m = 'MMDD'$  consisting of the month and day of your birth. Encrypt message  $m$  using 1 round of AES128 cipher with key  $K_h$  by computing ciphertext  $C = AES128(m, K_h, 1, 'e')$ . Send  $[C]$  to the Mentor for decryption and for resending message  $m$  to your friend Bob2.  $C$  should be entered within ".

```
>> PuKB=4670305;
>> KB=872215;
>> R=2705869;
>> S=4402137;
>> hd=int64(hd24(concat(KB,R)))
hd = 3684952
>> rez=sig_ver(p,g,R,S,PuKB,hd)
rez = TRUE: Signature correct
Val= 11780819
```

}  $\sigma_m = (R, S)$

```
>> K=mod_exp(KB,u,p)
K = 10027187
>> key=dec2hex(K,32)
key = 000000000000000000000000009900B3
>> m='0501'
m = 0501
>> C=AES128(m,key,1,'e')
new= .....
C = 8c000e708c0069008cf20e00eb990eb3
```

'8c000e708c0069008cf20e00eb990eb3'

4. Ok, Bob2 informed me that all the sum of money he received from the Knowledge Bank he dedicates to buy the gift for your birthday. This sum I am sending you as a ciphertext ( $C_M = '8c000e3c8c0075008c8d0e008c990eb3'$ ). Please decrypt and check it and then encrypt it again with added string 'ok' right after the sum by computing ciphertext  $C_1$ . Send  $[C_1]$  to the Mentor.  $C_1$  should be entered within ".

```
>> CM='8c000e3c8c0075008c8d0e008c990eb3'
CM = 8c000e3c8c0075008c8d0e008c990eb3
```

```
>> M=AES128(CM,key,1,'d')
Out = 00000000000000000000000000003437
M = 47
>> Mok='47ok'
>> C1=AES128(Mok,key,1,'e')
new = .....
C1 = 8c000e028c00c5008c870e00f7990eb3
```

'8c000e028c00c5008c870e00f7990eb3'

'8c000e708c0069008cf20e00eb990eb3'

Realize 10 rounds of encryption for the same plaintexts.

Success! You have finished the task. Great job!

Get reward

