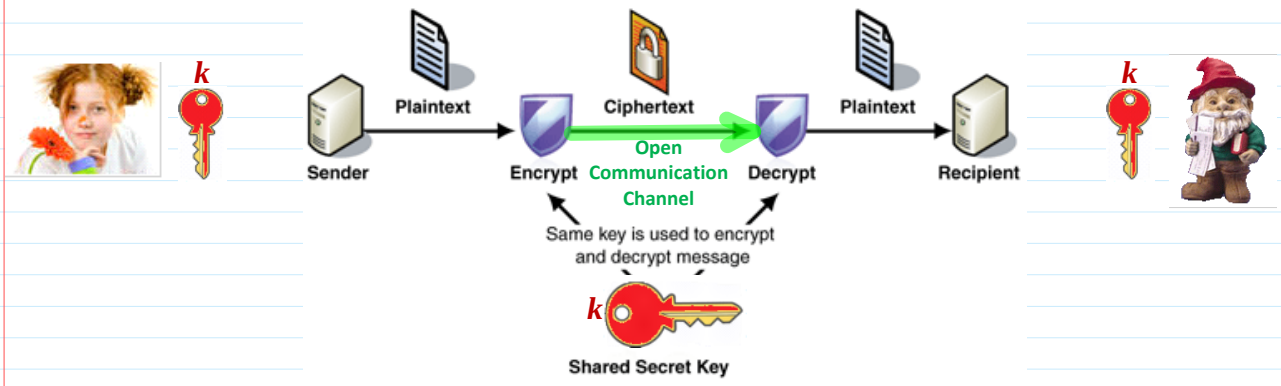
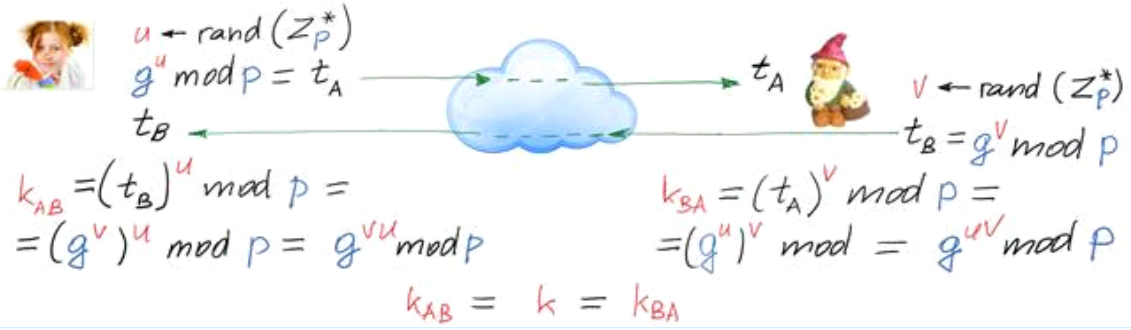


Using the same secret key Alice can encrypt a large message using block cipher for example AES-128, 192, 256.

Diffie-Hellman Key Agreement Protocol (DH KAP)

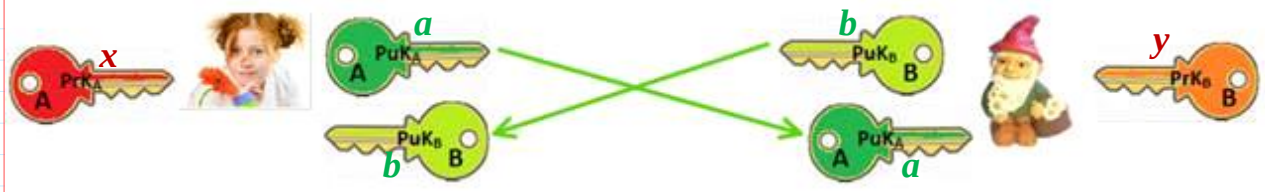
Public Parameters PP=(*p*,*g*)



AKAP: ElG-Enc + ElG-Sig = AKAP

Hybrid enc-dec method.

1. Parties must agree on common symmetric secret key k.
   for symmetric block cipher, e.g. AES-128, 192, 256.



A: $PrK_A = x$; $PuK_A = a = g^x \bmod p$.
$PuK_B = b$.

B: $PrK_B = y$; $PuK_B = b$.
$PuK_A = a$.

1) $k \leftarrow randi(2^{128})$
$i_k \leftarrow randi(2^{128})$

1) $k \leftarrow randi(2^{128})$
$i_k \leftarrow randi(2^{128})$

$Enc(PuK_B = b, i_k, k) = c = (E, D)$

2) M - large file to be encrypted

$E_k(M) = AES_k(M) = G$

3) Signs ciphertext c

3.1) $h = H(c)$

3.2) $Sign(PrK_A = x, h) = \sigma = (r, s)$

$c, G, \sigma, PuK_A, Cert_A$

1.1. Verify if PuK_A and $Cert_A$ are valid?

1.2. Verify if σ on $h = H(c)$ is valid?

$h' = H(c)$

$Ver(PuK_A, \sigma, h') = True$

2. $Dec(PrK_B, c) = k$

3. $D_k(G) = AES_k(G) = M.$

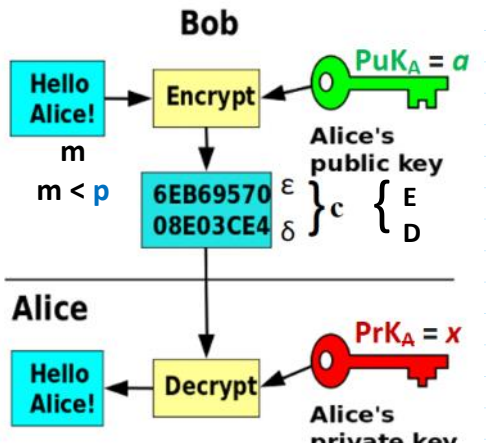
A was using so called encrypt-and-sign (E-&-S) paradigm. (E-&-S) paradigm is recomended to prevent so called choosen ciphertext Attacks - CCA: it is most strong attack but most complex in realization.

```
>> p=int64(268435019)
p = 268435019
>> g=2;
```

```
>> x=int64(randi(2^28-1))
x = 189519825
>> a=mod_exp(g,x,p)
a = 80162083
>> k=int64(randi(2^28))
k = 147364121
```

```
>> y=int64(randi(2^28-1))
y = 162527967
>> b=mod_exp(g,y,p)
b = 76401416
```

Asymmetric Encryption - Decryption
c=Enc(PuK_A, m)
m=Dec(PrK_A, c)

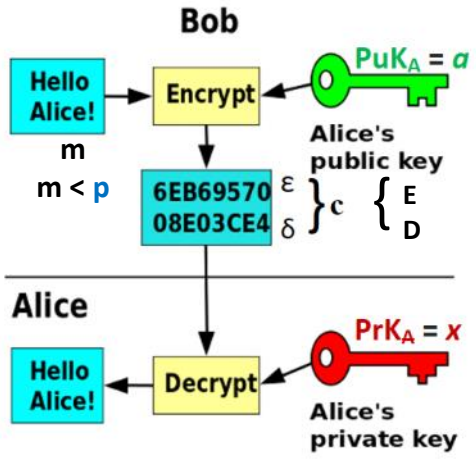


Alice is encrypting key k using El Gamal encryption

$Enc(PuK_B, i_k, k) = c = (E, D)$

$E = k \cdot b^{i_k} \bmod p \; ; \; D = g^{i_k} \bmod p.$

```
>> b_ik=mod_exp(b,ik,p)
b_ik = 262733343
>> E=mod(k*b_ik,p)
E = 75434686
>> D=mod_exp(g,ik,p)
D = 2879528
```



$E = k \cdot b^{i_k} \bmod p \; ; \; D = g^{i_k} \bmod p.$

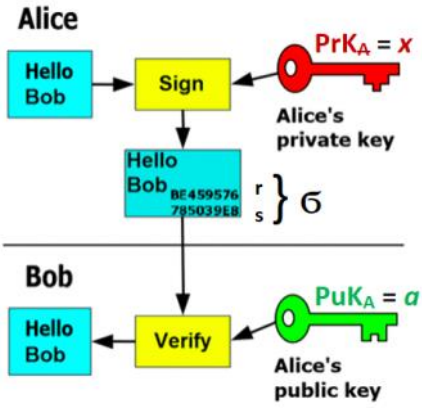
```
>> b_ik=mod_exp(b,ik,p)
b_ik = 262733343
>> E=mod(k*b_ik,p)
E = 75434686
>> D=mod_exp(g,ik,p)
D = 2879528
```

Signature creation for message $M > p$.

$h = sha('E, D')$

$j = randi(2^{28})$

$Sign(PrK_A, j, h) = \sigma = (r, s)$



1. Compute decimal h-value $h=H('E,D')$; $h<p$.
2. Generate >> j =int64(randi(p-1)) % such that gcd(j,p-1)=1.
3. Compute j^{-1} mod (p-1). You can use the function
   >> j_m1=mulinv(i, p-1);
4. Compute $r=g^j \bmod p$.
5. Compute $s=(h-xr)j^{-1}$ mod (p-1).
6. Signature on h-value h is $\sigma = (r,s)$
   Sign(x,h) = $\sigma = (r,s)$.

```
>> h=hd28('75434686,2879528')
h = 185959935
>> j=int64(randi(p-1))
j = 91638421
>> gcd(j,p-1)
ans = 1
>>  j_m1=mulinv(j, p-1)
j_m1 = 123035687
>> mod(j*j_m1,p-1)
ans = 1

>> r=mod_exp(g,j,p)
r = 190075158
>> xr=mod(x*r,p-1)
xr = 47387312
>> hmxr=mod(h-xr,p-1)
hmxr = 138572623
>> s=mod(hmxr*j_m1,p-1)
s = 163470271
```

$A: \quad c = (E, D), G, \sigma = (r, s) \rightarrow B:$

$PuK_A, Cert_A$

1. computes $h = H('E, D')$

```
>> h=hd28('75434686,2879528')
h = 185959935
```

2. Verifies signature σ on h

2.***Signature Verification***

A signature $\sigma=(r,s)$ on message ***M*** is verified using Public Parameters **PP**=(**p**, **g**) and $\mathbf{PuK_A}=\mathbf{a}$.

1. Bob computes **h=H(M)**.
2. Bob verifies if 1<**r**<**p-1** and 1<**s**<**p-1.**
3. Bob calculates $\mathbf{V1}=\mathbf{g^h}$ **mod p** and $\mathbf{V2}=\mathbf{a^r r^s}$ **mod p,** and verifies if **V1=V2.**

The verifier Bob accepts a signature if all **conditions** are satisfied during the signature creation and rejects it otherwise.

```
>> V1=mod_exp(g,h,p)
V1 = 7325643
>> a_r=mod_exp(a,r,p)
a_r = 32677501
>> r_s=mod_exp(r,s,p)
r_s = 68068424
>> V2=mod(a_r*r_s,p)
V2 = 7325643
```

Let message ***m~*** needs to be encrypted, then it must be encoded in decimal number ***m***: 1< ***m*** < ***p***.
E.g. ***m*** = 111222. Then ***m*** **mod** ***p*** = ***m***.
Since key **k** is encrypted then 1< **k** < ***p***.

3. Decrypts ciphertext $C=(E,D)$ to find k. using his PrK_B.

3.1. $D^{-x \bmod (p-1)} \bmod p$

3.2. $E \cdot D^{-x} \bmod p = k$. **k** = 147364121