"KRUPTOR" BIBLIOTHÈQUE DE FONCTIONS DE BASE POUR TESTER AVEC XCAS QUELQUES NOTIONS CRYPTOGRAPHIQUES

Liste récapitulative des entêtes des procédures communes
kruptor_commun_2.1.pdf
Version 2.1 du 10/03/2016
Auteur : Ainigmatias Cruptos
Association AcrypTA : acrypta "at" acrypta "point" com
http://www.acrypta.com

Fonctions générales : Kruptor_commun_2.1.map

Fonction rralea(x)
Entrée : un nombre de bits x
Sortie : un nombre aléatoire ayant exactement x bits

Fonction toutalea(x)
Entrée : un nombre de bits x
Sortie : un nombre aléatoire ayant au plus x bits

Fonction taille(x)
Entrée : un nombre entier positif x
Sortie : nombre de bits de x

Fonction auhasard(x)
Entrée : un nombre entier x>=1
Sortie : un nombre au hasard >=0 et <x.

Fonction nbp(x)
Entrée : un nombre de bits x
Sortie : un nombre premier au hasard ayant exactement x bits
On tire un nombre impair au hasard jusqu'à obtenir un nombre premier

Fonction nbpfast(x)
Entrée : un nombre de bits x
Sortie : un nombre premier au hasard ayant exactement x bits
on tire un nombre impair au hasard on incrémente de 2 jusqu'à obtenir un
nombre premier (moins propre mais plus rapide que la fonction précédente).

Fonction trouvepqk(x,y)
Entrée : un nombre de bits x, un nombre de bits y, avec x<y
Sortie : un nombre premier q de x bits, un nombre premier p de y bits
et un nombre k
tels que p=2kq+1, ainsi qu'un compteur de nombre d'essais [p,q,k,compteur]

Fonction prim(p,q,k)
Entrée : deux nombres premiers p et q et un nombre k tels que p:=2kq+1
Sortie : un élément primitif de Z/pZ
Attention, k ne doit pas être trop grand car il faut pouvoir le factoriser.

Fonction sophiegermain(x)
Entrée : un nombre de bits x
Sortie : p et q premiers tels que p=2q+1, p ayant x bits

Fonction zp(x)
Entrée : un nombre de bits x (x grand, au moins 1024 bits)
sortie : un nombre premier p de x bits
ainsi qu'un élément primitif de Z/pZ : [p,alpha]

Fonctions en relation avec le chiffrement et la signature RSA : kruptor_rsa_2.1.map

Fonction rsagene(x)
Entrée : taille des deux nombres premiers p et q
dont le produit donne le module n
Sortie : [p,q,n,eul,ee,d] ou eul est la
fonction d'Euler du module n, ee l'exposant de
chiffrement et d l'exposant de déchiffrement.

Fonction rsapk(u)
Entrée : la sortie u de la fonction rsagene
Sortie : la clé publique v=[n,ee]

Fonction rsask(u)
Entrée : la sortie u de la fonction rsagene
Sortie : la clé privée w=[n,d]

Fonction rsacrypt(x,v)
Entrée : le message x<n, la clé publique v=[n,ee]
Sortie : le message chiffré y

Fonction rsadecrypt(y,w)
Entrée : le message chiffre y, la clé privée w=[n,d]
Sortie : le texte clair x

Fonction rsatag(x,w)
Entrée : le message condensé x a signer, la clé privée w=[n,d]
Sortie : l'appendice s de la signature

Fonction rsabezout(u)
Entrée : la sortie u=[p,q,n,eul,ee,d] de rsagene
Sortie : un tableau v=[a,b] tel que ap+bq=1

Fonction rsaspeedtag(x,u,v)
Entrée : le message condensé x a signer, la sortie u de rsagene
la sortie v de rsabezout
Sortie : l'appendice s de la signature

Fonction rsaverif(x,s,v)
Entrée : le message condense x, l'appendice de signature 0<=s<n
la clé publique v=[n,ee]
Sortie 0 si la signature est vérifiée, 1 sinon

Fonction rsatrouvepq(n,ee,d)
Entrée : le module n, la clé publique ee et la clé privée d
d'un système RSA
Sortie : tableau donnant la factorisation de n : [p,q]

Fonctions en relation avec le chiffrement d'Elgamal et la signature DSA : kruptor_elgamal_2.1.map

Fonction compression(x)
Entrée : un nombre x ayant au plus 1024 bits
Sortie : un nombre s ayant au plus 160 bits

Fonction ElGamalgene()
Entrée : vide
Sortie : un tableau [p,q,k,alpha,a,b] où p et q sont des nombres
premiers (p a 1024 bits, q a 160 bits) p=2kq+1, alpha un élément
générateur du sous-groupe multiplicatif d'ordre q, a la clé privée
(0 <=a <=q-2), b=alpha^a mod p la clé publique.

Fonction ElGamalpk(v)
Entrée : la sortie v de ElGamalgene
Sortie : la clé publique pk=[p,q,alpha,b]

Fonction ElGamalsk(v)
Entrée : la sortie v de ElGamalgene
Sortie : la clé privée sk=[p,q,alpha,a]

Fonction ElGamalcrypt(x,pk)
Entrée : un message x <q, la clé publique pk
Sortie : le chiffre y=[y1,y2]

Fonction ElGamaldecrypt(c,sk)
Entrée : un chiffré c, la clé privée sk
Sortie : le texte clair x

Fonctions dsagene(), dsapk(v), dsask(v)
pour la signature dsa sont exactement les mêmes
que les fonctions correspondantes du
chiffrement d'ElGamal

Fonction dsatag(x,sk)
Entrée : le message condensé x a signer, la clé privée sk=[p,q,alpha,a]
Sortie : l'appendice t=[r,s] de la signature

Fonction dsaverif(x,t,pk)
Entrée : le message condensé x, l'appendice t de signature
la clé publique pk
Sortie 0 si la signature est verifiée, 1 sinon

Fonctions relative au hachage : kruptor_hash_2.1.map
(utilise pari, lancer pari() à la ligne de commande)

Fonctions purement techniques pour définir sha256;

Fonction constantes()
Stockage d'un tableau de constantes
Entrée : vide
Sortie : le tableau de 64 constantes de 32 bits

Fonction valinit
Valeurs initiales
Entrée : vide
Sortie : Une ligne contenant 8 constantes de 32 bits

Fonctions Ch(x,y,z),Maj(x,y,z),shr32(x,n),shl32(x,n),
rotr32(x,n),bsig0(x),bsig1(x),ssig0(x),ssig1(x)
Procédures de manipulations de bits
sur des mots de 32 bits
Entrée : des mots de 32 bits x,y,z, un entier n
Sortie : un mot de 32 bits

Fonction padding(m,longueur)
Entrée : un entier m, une taille longueur supérieure
a la taille de l'entier m
Sortie : un tableau [t,n]
t est un entier calculé suivant le standard sha256 de taille 512*n bits

Fonction parsing(u)
Entrée : u=[t,n] est la sortie de la fonction padding
Sortie : un tableau de n lignes de 16 nombres de 32 bits
chaque ligne est en fait un bloc de 512 bits de t (découpé
en 16 morceaux de 32 bits)

Fonction hascomp(v) calcule le digest
Entrée : v est la sortie de parsing
Sortie : digest, l'empreinte du message

Fonction hascomp(v) calcule le digest
Entrée : v est la sortie de parsing
Sortie : digest, l'empreinte du message

Fonctions utiles.

Fonction stringtoint(chaine)
Entree : chaine est une chaine de caracteres
Sortie : la sortie est [k,t] ou s est l'entier
represente par la chaine lors d'une traduction en ascii
et t le nombre de bits du bloc.

Fonction sha256string(chaine)
hachage d'une chaîne de caractères
Entrée : une chaîne de caractères
Sortie : l'empreinte

Fonctions test256court(), test256multi()
Tests sur des valeurs données par le NIST
(les résultats sont corrects!)

Fonction sha256int(nb,t)
hachage d'un entier
Entrée : un entier et un nombre de bits >= à la taille de l'entier
Sortie : l'empreinte

Fonction kdf(state,l)
Generateur de masque
entree : un nombre d'octets l
sortie : une suite pseudoaleatoire de l octets
Le germe « state » peut être pris dans une variable « germe » de 256 bits qui se trouve dans un fichier
seed fourni. On peut évidemment affecter state par tout autre moyen.