LES PROTOCOLES DE SECURITE

G. Florin

S. Natkin

Décembre 2001

Notations

A: clef d 'Alice (chiffrement symétrique)
a : clef d 'Alice (déchiffrement symétrique)
A: clef privée de Alice (déchiffrement asymétrique)
a: clef publique de Alice (chiffrement asymétrique)

$\{X\}_{Clef}^{CRY}$ Chiffrement /Déchiffrement selon le crypto sytème CRY avec la clef Clef

Crypto systèmes symétriques

$\{X\}_{a}^{SYM}$ Chiffrement $\{X\}_{A}^{SYM}$ Déchiffrement

Crypto systèmes asymétriques

$\{X\}_{a}^{ASY}$ Chiffrement (clef publique) $\{X\}_{A}^{ASY}$ Déchiffrement (clef privée)

$\{X\}^{H}$ Résumé de sécurité $\{X\}_{A}^{SIG} = \{\{X\}^{H}\}_{A}^{ASY}$ Signature de X par Alice

Notations protocolaire

Format des messages :Type, Emetteur, Destinataire, Contenu

Alice $\xrightarrow{\quad M \quad}$ Bob

Dans le protocole Alice envoie M à Bob

Les partenaires fiables

Système à clef privée: Le gardien des clefs

Alice	A	Date début/ Date fin a
Bob	B	Date début/ Date fin b
Charles	C	Date début/ Date fin c

Ce tableau est protégé en intégrité et confidentialité
Chaque participant connaît sa clef et celle du gardien G

Systèmes à clefs publiques: Annuaire de certificats

Alice	a	Date début/ Date fin a	
Bob	b	Date début/ Date fin b	
Charles	c	Date début/ Date fin c	

Ce tableau est protégé en intégrité (voir plus loin)
Chaque participant connaît sa clef privée et la clef publique de l 'annuaire

Authentification

Protocole permettant à Bob de prouver à Alice qu'il est Bob

Bob détient un secret sur lequel repose l'authentification
Bob ne doit pas révéler le secret à Alice

Il existe un tiers fiable qui a authentifié Bob
(gardien des clefs ou annuaire de certificats)

Authentification avec un crypto système symétrique

Alice *Bob*

Générer Random

Auth_Req,Alice,Bob, Random $\longrightarrow$

$X := \{Bob, Random\}_b^{SYM}$

$\longleftarrow$ Auth_ Resp, Bob, Alice, X

Gardien

Cif_Req,Alice, Gardien, Bob ,X $\longrightarrow$

$T := \{X\}_B^{SYM}$

$Z := \{T\}_a^{SYM}$

$\longleftarrow$ Cif_Resp ,Gardien,Alice,Bob,Z

$\text{Vérifier}((\text{Bob}, \text{Ramdom}) = \{Z\}_A^{SYM})$

Authentification à clef publique

Alice *Annuaire* *Bob*

Cer_Req, Alice,Annuaire,Bob

Certificat:=(Bob,b,Valid, Date,sig)

Cer_Resp, Annuaire, Alice,Certificat

Contrôler les certificats

Générer Random

Auth_Req, Alice, Bob,Random

$$X := \{Bob, Random\}_{B}^{ASY}$$

Auth_Resp, Bob,Alice,C

$$\text{Vérifier}((Bob, Ramdom) = \{Z\}_{b}^{SYM})$$

Confidentialité

Alice doit transmettre à Bob un message que eux seuls doivent connaître

Confidentialité avec chiffre symétrique

Alice

$C := \{Alice, Bob, M\}_{a}^{SYM}$

Gardien

Fwd_req,Alice, Gardien, Bob ,C

$T := \{C\}_{A}^{SYM};$

$Z := \{T\}_{b}^{SYM}$

Bob

Fwd_Ind,Gardien,Bob, Alice,Z

$(Alice, Bob, M) := \{Z\}_{B}^{SYM}$

M peut être une clef de session, qui est ensuite utilisée pour chiffrer les autres messages entre Alice et Bob

Confidentialité avec chiffre asymétrique

Alice *Annuaire* *Bob*

Cer_Req, Alice,Annuaire,Bob →

Certificat:=(Bob, b,Valid,sig)

← Cer_Resp, Annuaire, Alice,Certificat

Contrôler les certificats;

$X := \{M\}_b^{ASY}$

Data_cif_Ind, Alice, Bob, X →

$M := \{X\}_B^{ASY}$

Très peu utilisé car très lent, sert à échanger des clefs d 'algorithmes symétriques beaucoup plus rapides

On échange ainsi une clef de session pour chiffre symétrique

Signature et intégrité

Alice doit envoyer à Bob un message, tel que Bob
puisse contrôler que le message n’a pas été modifié et a bien
été créé par Alice

Signature avec chiffre symétrique

Alice

Bob

$Sig := E_A(H(M))$

Sign_Ind,Alice, Bob ,M,Sig

Gardien

$T := D_A(Sig)$
$Z := E_B(T)$

Cif_Req,Bob,Gardien, Alice,Sig

Cif_Resp, Gardien,Bob,Alice,Z

$V := H(M)$
Vérifier$(D_B(Z)=V)$

Signature à chiffrement asymétrique

Bob *Alice*

$Sig := \{Bob, Alice, M\}_{B}^{SIG}$ Bob,Alice,M,Sig

Annuaire

Cert_Req,Alice,Annuaire,Bob

Certificat:=(Bob,b,
Cert_Req, Valid,sig)

Cert_Resp, Annuaire, Alice,Certificat

Contrôle des certificats;

$V = \{Bob, Alice, M\}^{H};$

$Vérifier(V = \{Sig\}_{b}^{ASY}$

Intégrité des messages et flots de messages

Intégrité d ’un message: problème voisin de la signature
Utilisation de fonction de Hachage sécuritaire ou de MAC basé sur un chiffre symétrique en mode chaîné

Intégrité du flot de message: Possibilité de rejeu
Utilisation d ’un **Nonce** (Used Only Once), qui distingue chaque message:
Numéro de séquence sur un modulo grand
Heure
Nombre aléatoire

Gestion des clefs

Annuaire des certificats

NOM	Clef	Validité	Extensions	Signature
Alice	a	Valida	Para	$\{Alice, a, Valida, Para\}_{AC}^{SIG}$
Bob	b	Validb	Parb	$\{Bob, b, Validb, Parb\}_{AC}^{SIG}$
Charles	c	Validc	Parc	$\{Charles, c, Validc, Parc\}_{AC}^{SIG}$

AC: autorité de certification

Norme de représentation des certificats X509

Norme de protocole d'accès: LDAP

Contrôle des certificats

Toutes entités impliquées dans un schéma à clef
publique doit détenir la clef publique de l 'autorité
de certification.

Tout accès à un certificat doit être contrôlé:
Vérifier que la signature est valide
Vérifier que la date courante est dans la période de validité

Pour éviter les rejeux de certificats invalidés le serveur d 'annuaire
doit :
Soit s'authentifier
Soit dater et signer sa réponse
Soit transmettre périodiquement des listes
de révocation datée et signées

Stockage des clefs asymétriques

Clef publique de l'autorité, ne doit pas pouvoir être modifiée: Dans le code en dur , sur un support fiable (carte à puce)

Clef privée de l 'utilisateur, ne doit pas pouvoir être lue: sur un support confidentiel (carte à puce) ou un fichier chiffré avec un mot de passe (local au poste ou sur disquette)

Certificat de l 'utilisateur: Annuaire+support local ou carte ou disquette

Annuaire: Annuaire central+version locales (cache, annuaire privé

Protocole de création des certificats version répartie

Client Alice

Autorité de certification

génération de A, a, MP

Stockage $ESYM_{MP}$(Alice, A, Date)

X:= E_{RS} (Alice, a, Date)

Alice, Autorité, X →

D_{RS} (X)

Contrôle de l 'identité d 'Alice

Mise à jour de l 'annuaire

Y=Alice,a,Date, D_{RS} (Alice, a, Date)

← Autorité, Alice, Y

Protocole de création des certificats version centralisée

Autorité de certification

Alice, Autorité

Contrôle de
l'identité d 'Alice
génération de A, a, MP

ESYM_{MP}(Alice, A, Date)
fichier, disquette, carte à puces

MP (voie confidentielle)

Mise à jour de l 'annuaire
Y=Alice,a,Date, D_{RS} (Alice, a, Date)

Hiérarchie des clefs

Plus on utilise une clef plus elle est vulnérable
Clef utilisée pour chiffrer une suite de transfert de fichier vs clef utilisée pour chiffrer un numéro de carte bleue

Plus elle sert a protéger des données péreines, plus elle doit être fiable
Signature électronique d 'un article de presse vs Signature électronique d 'un testament

On peut utiliser des canaux très lents mais très fiables pour véhiculer des clefs qui seront utilisées sur des voies plus rapides et moins fiables (téléphone rouge)

Système asymétrique: Hiérarchie des autorités de certification (chaîne de certification)

Etat -------------------> Annuaire des autorités
(certificat racine) auto-signé

↓ Certificats des autorités

Autorité=> Annuaire professionnel ou commercial

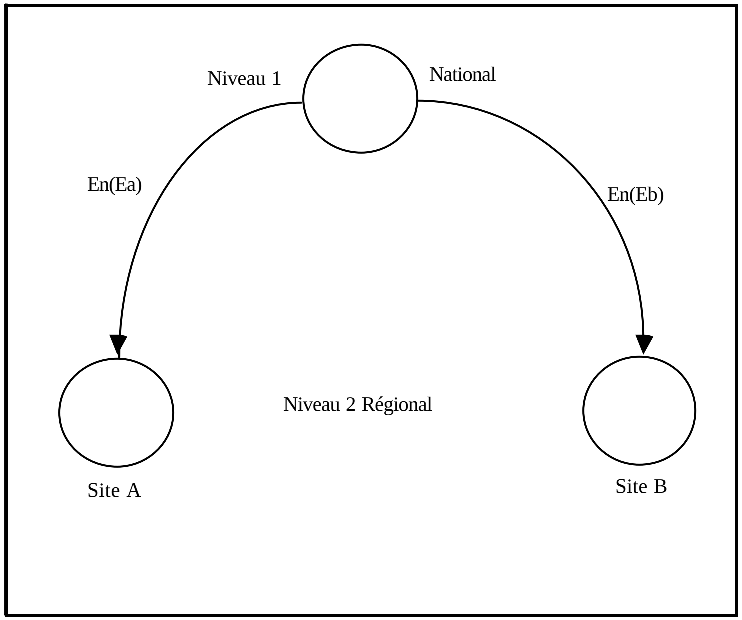
↓ Certificats des individus et systèmes

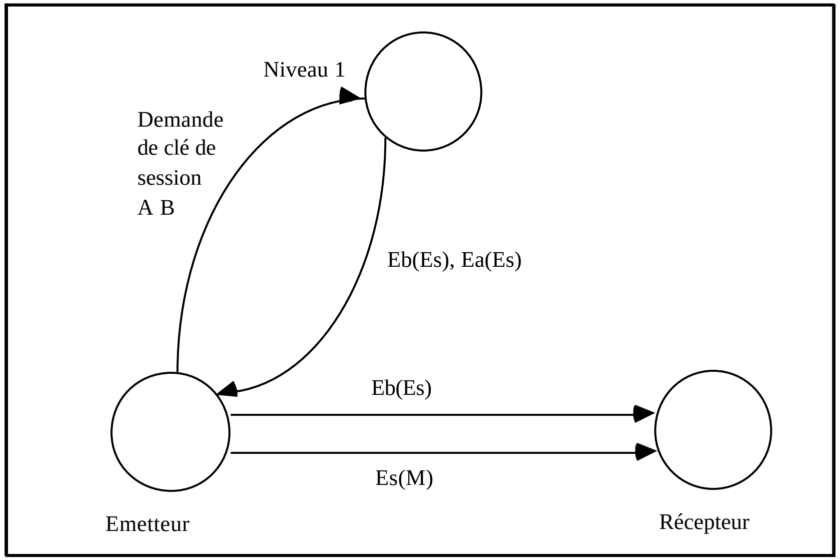
Individu ou système => Annuaire d 'objets

↓ Certificats d 'objet: Capacité

Objets=>Clefs de session

Système symétrique hiérarchie des clefs de session





L 'Authentification à apport nul de connaissance (zero knowledge protocols) Principes généraux

En utilisant les algorithmes a clefs publiques
S,D(s)-------->? E(D(S)) = S
La base est que seul celui qui doit s'authentifier sait faire D

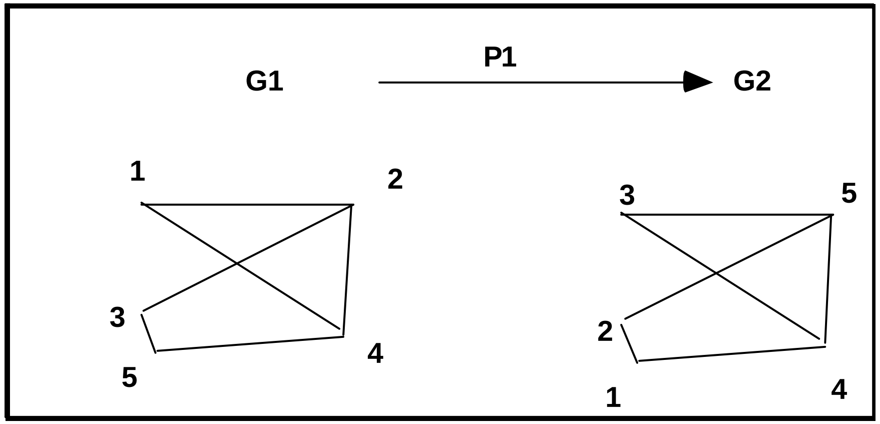
Dans les algorithmes a absence de connaissance:
Protocoles d'authentification probabilistes

Le veritable emetteur est seul a savoir répondre
à une question à coup sûr

Le pirate sait répondre avec une probabilité p
et échoue avec une probabilité 1-p

Un échec prouve une tentative d'usurpation
Après k succès la probabilité d'une tentative d'usurpation est p^k

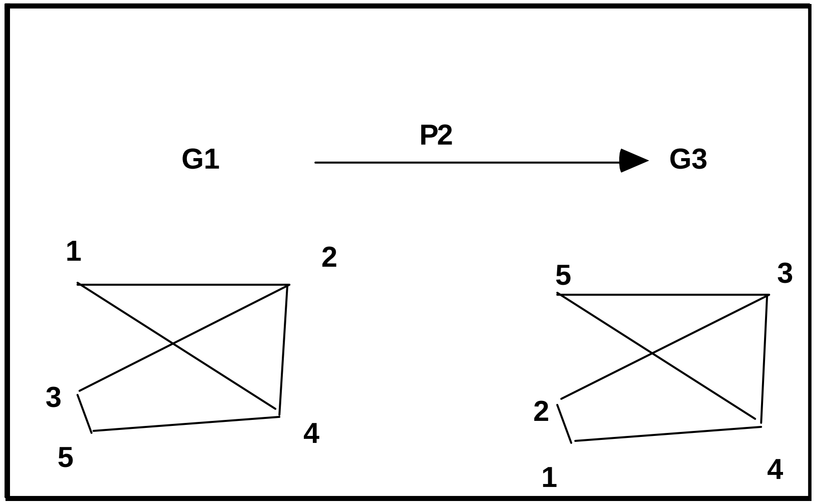
Exemple d 'école: isomorphie de graphes



G1 et G2 Sont publics
P1 (1->3, 2->5, 3->2, 5->1, 4->4) est secrète

Isomorphie de graphes (2)

Au moment de l'authentification celui qui doit s'authentifier (le prouveur) publie un troisième graphe g3 soit déduit de g1 soit déduit de g2 (il ne dit pas quel est Le graphe de départ)



P2 (1->5, 2->3, 3->2, 5->1, 4->4) est secrète

Isomorphie de graphes (3)

Celui qui cherche a vérifier l'authentification (le vérifieur) connaît les trois graphes,mais pas le processus de génération.
Il demande:

A) avec une probabilité 1/2 comment passe t'on de g1 a g3?
B) avec une probabilité 1/2 comment passe t'on de g2 a g3?

Le prouveur peut toujours répondre:

Dans l'exemple cas a il répond p2
Cas b il répond $p1^{-1}op2$
(Il connaît p1 et sait donc calculer son inverse)

Le pirate a génère un graphe soit a partir de g1 soit a partir de g2 (publics).
Supposons g1

Dans le cas a il sait répondre
Dans le cas b il ne sait pas (pb np complet)

Le protocole de Fiat-Shamir (1)

Basé sur la complexité de calcul d'une racine carrée dans une algèbre modulo N ou N est le produit de deux grands nombres premiers p et q

0) Données publiques
Le prouveur choisit un nombre S et calcule $V=S^2$ mod (N). Il publie V

1) Authentification
Le vérifieur demande au prouveur de s'authentifier
Le prouveur a choisit un nombre aléatoire R

2) Phase d'enchère:
Le prouveur calcule $X=R^2$ mod (N). Il envoie X au vérifieur

3) Phase de défi:Le vérifieur met le prouveur au défi:
Il choisit un nombre aléatoire binaire D et l'envoie au prouveur

4) Phase de preuve
Le prouveur répond en envoyant Y au vérifieur
Si D=0 Y=R
Si D=1 Y=R*S mod (N)

Le protocole de Fiat-Shamir (2)

5) Phase de vérification

Le vérifieur calcule Y^2

Il doit trouver:

Si D=0 $Y^2 = X$

Si D=1 $Y^2 = X*V \bmod (N)$

ANALYSE

Si le fraudeur connaissait des la phase 1, la question posée en 3, il pourrait toujours tromper le valideur:

Si D=0 il choisit R quelconque et calcule $X=R^2$

Si D =1 il choisit un nombre K arbitraire et pose

$X = K^2 * V \bmod (N)$

$Y = K*V \bmod (N)$

Ceci vérifie donc $Y^2 = X*V \bmod (N)$

(mais il ne connaît pas R, la racine de X)

Il est donc indispensable de procéder dans cet ordre

Donc ne connaissant pas la question le fraudeur doit a priori choisir entre les deux stratégies et a donc une chance sur deux de d'être capable de répondre

Partage d'un secret: protocoles à seuil

Certaines opérations sont suffisamment sensibles pour devoir **engager la responsabilité de plusieurs** personnes.

On peut faire **vérifier l'identité** de plusieurs usagers simultanément possesseurs d'un **mot de passe** pour engager une action.

Mais cette approche peut ensuite être encore raffinée en souhaitant **donner une part de responsabilité plus importante** selon un grade:
Ex : Il suffit de la présence du responsable financier pour ouvrir le coffre ou de trois chefs de service ou ...

-Le problème du partage d'un secret:

Comment diviser une clé d'accès représentée par **une valeur numérique V en parts** (t+1 par exemple)
De telle façon qu'un groupe de porteurs de t+1 parts peuvent reconstituer la clé alors qu'un **groupe de porteurs de t parts ne le peuvent pas**.
Les porteurs de parts **doivent pouvoir reconstituer V** dans un système informatique d'autorisation sans jamais connaître V.

Protocole de Shamir (1)

V **valeur numérique** entière

. On **génère aléatoirement t valeurs entières**

$a_1 , a_2 , ... , a_t$

. **On leur associe un polynôme** dont le terme constant est **V** :

$P(x) = a_t x^t + a_{t-1} x^{t-1} + ... + a_1 x + V$

Une part du **secret est un couple $(x_i, P(x_i))$** x_i non nul

les parts sont générées par des xi différents

Pour éviter une possible **attaque force brute** par un groupe de porteurs agissant par essais et erreurs pour compléter leur connaissance:
on choisit **un entier premier** N grand,
les calculs sont faits en arithmétique modulo N

Protocole de Shamir (2)

V **valeur numérique** entière

. On **génère aléatoirement t valeurs entières**

$a_1 , a_2 , ... , a_t$

. **On leur associe un polynôme** dont le terme constant est **V** :

$P(x) = a_t\, x^t + a_{t-1}\, x^{t-1} + ... + a_1\, x + V$

Une part du **secret est un couple (x_i,P(x_i))** x_i non nul

les parts sont générées par des xi différents

Pour éviter une possible **attaque force brute** par un groupe de porteurs agissant par essais et erreurs pour compléter leur connaissance:
on choisit **un entier premier** N grand,
les calculs sont faits en arithmétique modulo N

Protocole de Shamir (3)

Tout groupe d'au moins t+1 possesseurs de parts peut résoudre le système linéaire de détermination des coefficients du polynôme et ainsi trouver V:

$$(a_t , a_{t-1} , \ldots , a_1 , V) \begin{pmatrix} x_1^{t} & & x_{t+1}^{t} \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ x_1 & & x_t \\ 1 & & 1 \end{pmatrix} = (P(x_1) , \ldots , P(x_{t+1}))$$

Comme les xi sont différents et non nuls la matrice est régulière
Tout sous groupe de porteurs dont la somme des parts est inférieure ou égale à t ne peut déterminer V.