Chapitre 6: Conception de protocoles de sécurité

Polytech Nancy

Année 2021/2022

UNIVERSITÉ DE LORRAINE

Conception de protocoles de sécurité

Point route

Point route

Rôle des protocoles de sécurité

Sécuriser la transmission d'informations pour

- le commerce électronique,
- la communication sans fil.

Les protocoles de sécurité sont cruciaux
dans la sécurité des systèmes distribués.

Motivation scientifique pour l'analyse de ces protocoles

- Petits programmes, mais systèmes à nombre infini d'états...
- *Souvent faux :* ne satisfont pas les propriétés de sécurité qu'ils prétendent assurer (secret, authentification, anonymat, non répudiation,...).

Nombreuses catégories de petits protocoles :

- à clefs symétriques (ISO, Andrew),
- d'authentification par des fonctions de hachage (ISO),
- à clefs symétriques utilisant un serveur sûr (Denning-Sacco, Otway-Rees, Yahalom, Woo-Lam),
- à signatures avec clefs conventionnelles (NSSP),
- à authentification répétée par clefs symétriques (Kerberos, Neuman-Stubblebine, Kao-Chow),
- à clefs publiques (NSPK, ISO, Diffie-Hellman),
- à clefs publiques utilisant un serveur sûr (NSPK-KS),
- ...

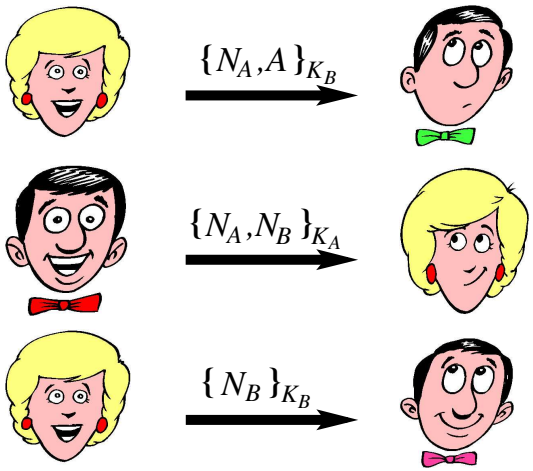
Point route

Exemple : Needham-Schroeder

Protocole d'authentification :

1. $A \rightarrow B : \quad \{N_A, A\}_{PK_B}$
2. $B \rightarrow A : \quad \{N_A, N_B\}_{PK_A}$
3. $A \rightarrow B : \quad \{N_B\}_{PK_B}$

Signification :



"C'est Alice ! et voici un nonce N_A."

"Voici ton nonce. Comme j'ai pu le lire, je suis bien Bob. Et voici un nonce N_B."

"Merci pour N_B. Comme seule Alice peut le lire, et que le voici, je suis bien Alice !"

Les protocoles sont en général petits et convaincants ...

Les protocoles sont en général petits et convaincants ...

... mais aussi souvent faux !

Attaque sur Needham-Schroeder

1. $A \rightarrow B : \quad \{N_A, A\}_{PK_B}$
2. $B \rightarrow A : \quad \{N_A, N_B\}_{PK_A}$
3. $A \rightarrow B : \quad \{N_B\}_{PK_B}$

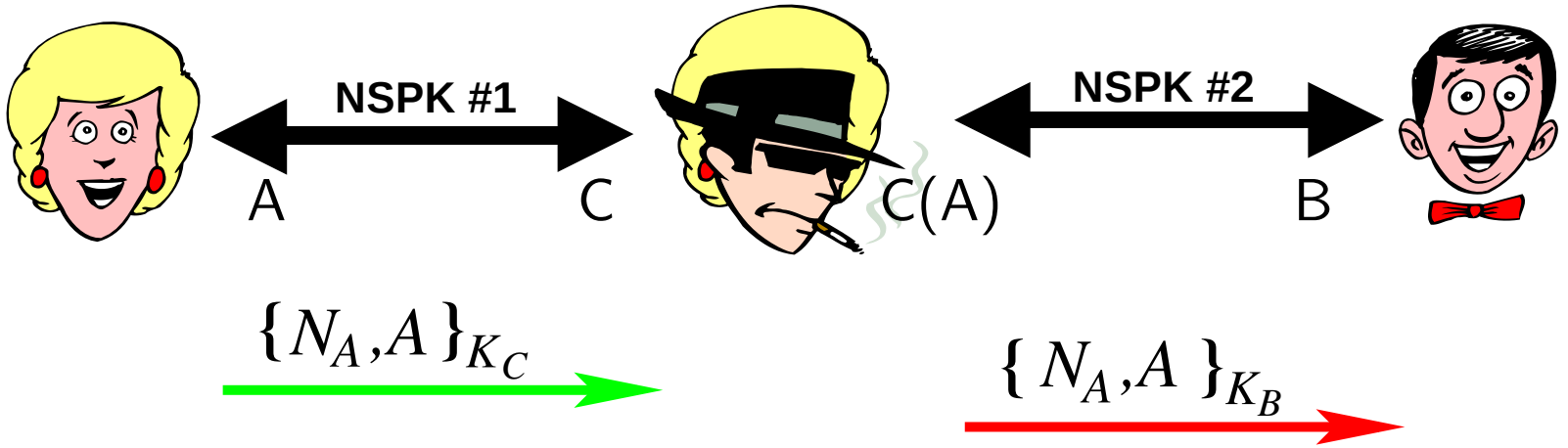
Attaque sur Needham-Schroeder

1. $A \rightarrow B : \quad \{N_A, A\}_{PK_B}$
2. $B \rightarrow A : \quad \{N_A, N_B\}_{PK_A}$
3. $A \rightarrow B : \quad \{N_B\}_{PK_B}$

$\{N_A, A\}_{K_C}$

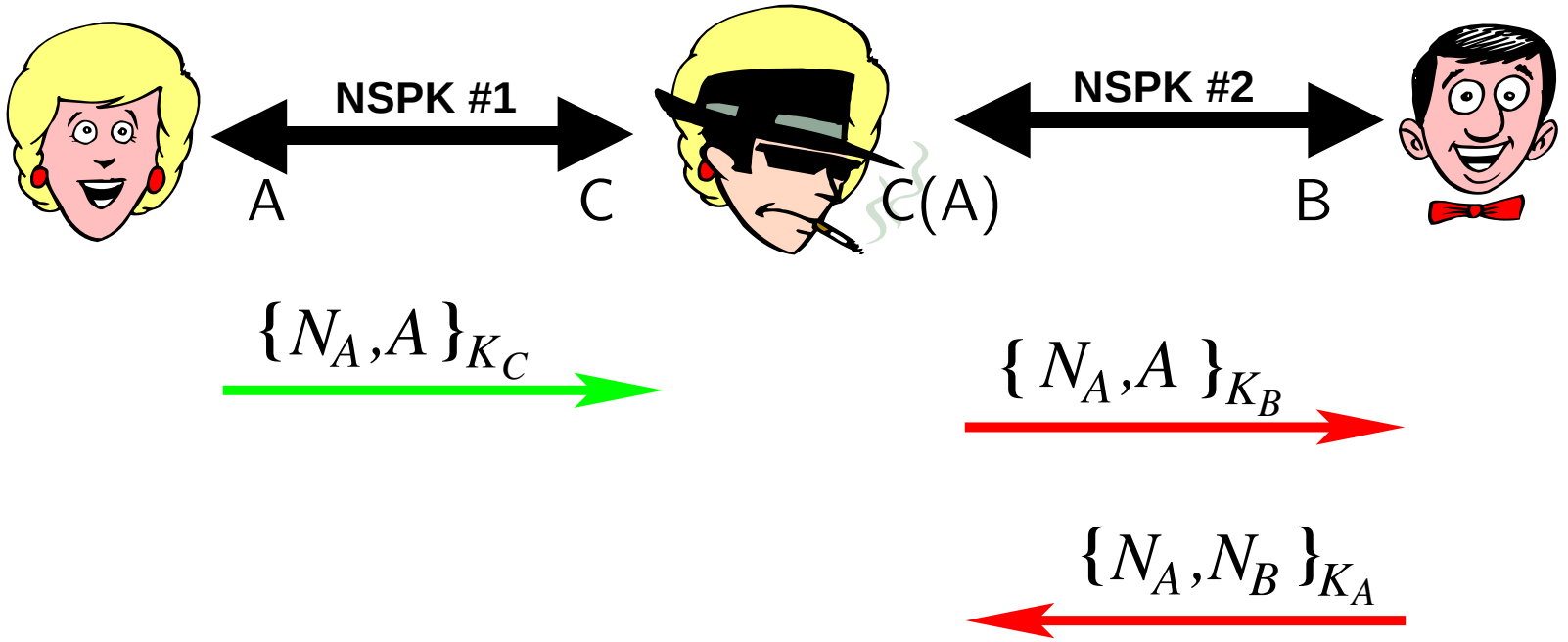
Attaque sur Needham-Schroeder

1. $A \rightarrow B : \quad \{N_A, A\}_{PK_B}$
2. $B \rightarrow A : \quad \{N_A, N_B\}_{PK_A}$
3. $A \rightarrow B : \quad \{N_B\}_{PK_B}$



Attaque sur Needham-Schroeder

1. $A \rightarrow B : \quad \{N_A, A\}_{PK_B}$
2. $B \rightarrow A : \quad \{N_A, N_B\}_{PK_A}$
3. $A \rightarrow B : \quad \{N_B\}_{PK_B}$



Attaque sur Needham-Schroeder

1. $A \rightarrow B : \quad \{N_A, A\}_{PK_B}$
2. $B \rightarrow A : \quad \{N_A, N_B\}_{PK_A}$
3. $A \rightarrow B : \quad \{N_B\}_{PK_B}$

$\{N_A, A\}_{K_C}$

$\{N_A, A\}_{K_B}$

$\{N_A, N_B\}_{K_A}$

$\{N_A, N_B\}_{K_A}$

Attaque sur Needham-Schroeder

1. $A \rightarrow B : \quad \{N_A, A\}_{PK_B}$
2. $B \rightarrow A : \quad \{N_A, N_B\}_{PK_A}$
3. $A \rightarrow B : \quad \{N_B\}_{PK_B}$

$\{N_A, A\}_{K_C}$

$\{N_A, A\}_{K_B}$

$\{N_A, N_B\}_{K_A}$

$\{N_A, N_B\}_{K_A}$

$\{N_B\}_{K_C}$

Attaque sur Needham-Schroeder

1. $A \rightarrow B : \quad \{N_A, A\}_{PK_B}$
2. $B \rightarrow A : \quad \{N_A, N_B\}_{PK_A}$
3. $A \rightarrow B : \quad \{N_B\}_{PK_B}$

$\{N_A, A\}_{K_C}$

$\{N_A, A\}_{K_B}$

$\{N_A, N_B\}_{K_A}$

$\{N_A, N_B\}_{K_A}$

$\{N_B\}_{K_C}$

$\{N_B\}_{K_B}$

Bob croit qu'il parle avec Alice !

Autre difficulté : répétitions

$$\text{Alice} \rightarrow \text{Banque} : \quad \{ \text{transférer 1000 € à Charlie} \}_{SK_{Alice}}$$

Autre difficulté : répétitions

$$\text{Alice} \rightarrow \text{Banque} : \quad \{ \text{transférer 1000 € à Charlie} \}_{SK_{Alice}}$$

$$\text{Charlie} \rightarrow \text{Banque} : \quad \{ \text{transférer 1000 € à Charlie} \}_{SK_{Alice}}$$

$$\ldots$$

Autre difficulté : confusion de types

$$\text{Alice} \rightarrow \text{Bob} : \quad \boxed{\text{Alice,Bob}}, \{ N_A, \boxed{\text{Alice,Bob}} \}_{K_{AS}}$$

$$\text{Bob} \rightarrow \text{Server} : \quad \{ N_A, \text{Alice,Bob} \}_{K_{AS}}, \{ N_B, \text{Alice,Bob} \}_{K_{BS}}$$

$$\text{Server} \rightarrow \text{Bob} : \quad \{ N_A, \text{key} \}_{K_{AS}}, \{ N_B, \text{key} \}_{K_{BS}}$$

$$\text{Bob} \rightarrow \text{Alice} : \quad \{ N_A, \boxed{\text{key}} \}_{K_{AS}}$$

Point route

Modèles de l'intrus

Différents comportements possibles

- Écouter : noter tous les messages circulant.
- Bloquer : intercepter des messages et les empêcher d'atteindre leur destinataire.
- Poster : envoyer des messages sous son propre nom.
- Imiter : envoyer des messages sous une autre identité.

Modèles de l'intrus

Conséquences

- Passif : écouter, juste pour essayer d'apprendre des informations utiles.
- Destructeur : bloquer tous les messages ; mais peu d'intérêt.
- Monopolisateur : poster énormément de messages vers un serveur pour le surcharger et l'empêcher de faire son travail normal.
- Actif : écouter, bloquer, poster, imiter,... pour intervenir dans des échanges sans que cela ne soit détecté.

Modèle de l'intrus de Dolev-Yao

Il peut

- Espionner, enregistrer, modifier, rejouer
- Se masquer en changeant l'adresse source
- Initialiser des sessions parallèles
- Provoquer des confusions de type

Il ne peut pas

- Chiffrer, déchiffrer sans la clef : *cryptographie parfaite*
- par exemple, $\{M\}_K, \{M'\}_K \nvdash \{M.M'\}_K$

Point route

Difficultés/approximations

- Nombre de sessions : fini /infini
- Taille des messages : finie / infinie
- Cryptographie : idéale / réelle
- Clefs : atomiques / composées
- Nonces : oui / non
- Conditionnelles : oui / non

Remarque : la confidentialité est indécidable pour un nombre non borné de sessions et des messages de taille non bornée.

Objectif

Utiliser des outils pour...

- vérifier qu'un protocole est implantable ;
- détecter des attaques ;
- valider un protocole.

et ceci de manière automatique.

Exemple de protocole non implantable :

1. $A \rightarrow B : \quad h(K), \{Msg\}_{PK_a}$
2. $B \rightarrow A : \quad \{B, Msg\}_K$

Point route

Point route

Spécification d'un protocole

Données de base

- Qui sont les participants du protocole ? Quel est le rôle de chacun ?
- Quels sont les messages échangés ?
- Quelles sont les connaissances initiales des participants ?
- Quelles sont les connaissances publiques ?
- Comment évoluent les connaissances de chacun ?
- Quelles sont les propriétés que doit satisfaire ce protocole ?

Spécification d'un protocole

Données plus techniques

- Quel niveau de spécification adopter ? proche de l'implantation ou assez abstrait ?
- Certaines fonctionnalités ont-elles des propriétés spécifiques (xor, exp, etc.), en liaison avec les mécanismes de chiffrement ?
- Peut-il y avoir confusion de types ?

Exemple de spécification

Needham-Schroeder Public Key

Protocole d'authentification.

- Messages :
  1. $A \rightarrow B : \{Na, A\}_{PK_b}$
  2. $B \rightarrow A : \{Na, Nb\}_{PK_a}$
  3. $A \rightarrow B : \{Nb\}_{PK_b}$
- Participants : A et B, normaux.
- Données :
  - Na, Nb : nonces
  - PK_a, PK_b : clefs publiques

Exemple de spécification

Needham-Schroeder Public Key

- Messages :
  1. $A \rightarrow B : \{Na, A\}_{PK_b}$
  2. $B \rightarrow A : \{Na, Nb\}_{PK_a}$
  3. $A \rightarrow B : \{Nb\}_{PK_b}$
- Connaissances initiales de A : B, PK_a, SK_a, PK_b
- Connaissances initiales de B : (A), PK_b, SK_b, PK_a
- Connaissances publiques : A, B, PK_a, PK_b

Exemple de spécification

Needham-Schroeder Public Key

- Messages :
  1. $A \rightarrow B : \{Na, A\}_{PK_b}$
  2. $B \rightarrow A : \{Na, Nb\}_{PK_a}$
  3. $A \rightarrow B : \{Nb\}_{PK_b}$
- Evolution des connaissances de A : ...
- Evolution des connaissances de B : ...
- Evolution des connaissances publiques : ...

Point route

Vérifier la spécification

À partir d'une spécification,

- vérifier l'exécutabilité du protocole :

  Pour chaque participant,
  à partir de ses connaissances initiales et acquises,
  - peut-il composer les messages qu'il doit envoyer ?
  - que peut-il vérifier dans les messages reçus ?

- identifier comment vérifier les propriétés : à quelles étapes ? dans quels rôles ?

Composer un message

L'agent U compose un message M à l'étape i

$$\begin{aligned} compose(U, M, i) &= \\ compose(U, \{M_1, M_2\}, i) &= \\ compose(U, \{M\}_K, i) &= \\ compose(U, M, i) &= \\ compose(U, f(M), i) &= \\ compose(U, M, i) &= \end{aligned}$$

Permet de vérifier que le message peut effectivement être composé et émis par U.

Composer un message

L'agent U compose un message M à l'étape i

$$
\begin{aligned}
compose(U, M, i) &= t && \text{si } M : t \text{ est connu par } U \text{ à l'étape } i \\
compose(U, \{M_1, M_2\}, i) &= \{compose(U, M_1, i), compose(U, M_2, i)\} \\
compose(U, \{M\}_K, i) &= \{compose(U, M, i)\}_{compose(U,K,i)} \\
compose(U, M, i) &= nonce(x_{time}) && \text{si } M \text{ est créé à l'étape } i \\
compose(U, f(M), i) &= compose(U, f, i)(compose(U, M, i)) \\
compose(U, M, i) &= \texttt{Échec} && \text{sinon}
\end{aligned}
$$

Permet de vérifier que le message peut effectivement être composé et émis par U.

Vérifier un message attendu

L'agent U attend le message M à l'étape i

$$
\begin{aligned}
expect(U, M, i) &= \\
expect(U, \{M_1, M_2\}, i) &= \\
expect(U, \{M\}_{PK}, i) &= \\
expect(U, \{M\}_{SK}, i) &= \\
expect(U, \{M\}_{K}, i) &= \\
expect(U, M, i) &=
\end{aligned}
$$

Vérifie que le protocole est exécutable.

Vérifier un message attendu

L'agent U attend le message M à l'étape i

$$
\begin{aligned}
expect(U, M, i) &= compose(U, M, i) \\
expect(U, \{M_1, M_2\}, i) &= \{expect(U, M_1, i), expect(U, M_2, i)\} \\
expect(U, \{M\}_{PK}, i) &= \{expect(U, M, i)\}_{inv(compose(U,SK,i))} \\
expect(U, \{M\}_{SK}, i) &= \{expect(U, M, i)\}_{inv(compose(U,PK,i))} \\
expect(U, \{M\}_{K}, i) &= \{expect(U, M, i)\}_{compose(U,K,i)} \\
expect(U, M, i) &= x_{U,M,i} \quad \text{nouvelle variable} \qquad \text{sinon}
\end{aligned}
$$

Vérifie que le protocole est exécutable.

Traduction simplifiée de NSPK

1. $A \rightarrow B : \quad \{N_A, A\}_{PK_b}$
2. $B \rightarrow A : \quad \{N_A, N_B\}_{PK_a}$
3. $A \rightarrow B : \quad \{N_B\}_{PK_b}$

étape i	message attendu R_i	$\Rightarrow$ réponse composée S_i
(A,1),	*Init*	$\Rightarrow \{N_A,\ A\}_{PK_b}$
(B,1),	$\{x_{B,N_A,1},\ x_{B,A,1}\}_{PK_b}$	$\Rightarrow \{x_{B,N_A,1},\ N_B\}_{PK_{x_{B,A,1}}}$
(A,2),	$\{N_A,\ x_{A,N_B,2}\}_{PK_a}$	$\Rightarrow \{x_{A,N_B,2}\}_{PK_b}$
(B,2),	$\{N_B\}_{PK_b}$	$\Rightarrow$ *End*

Point route

Règles de l'intrus

$\{a, b\}$	$\rightarrow$	a, b	a, b	$\rightarrow$	$\{a, b\}$
$SK, \{a\}_{PK}$	$\rightarrow$	a	a, K	$\rightarrow$	$\{a\}_K$
$S, \{a\}_S$	$\rightarrow$	a	a, S	$\rightarrow$	$\{a\}_S$

$t \in \mathit{forge}(E)$ ssi $E \rightarrow^* t$

Exemple :

$$\{secret\}_{\{k1\}_{k2}}\ ,\ \{k1, k2\}_{k3}\ ,\ k3$$

Exemple :

$$\{secret\}_{\{k1\}_{k2}},\ \{k1, k2\}_{k3},\ k3$$

$$\rightarrow$$

$$\{secret\}_{\{k1\}_{k2}},\ \{k1, k2\},\ k3 \ldots$$

Exemple :

$$\{secret\}_{\{k1\}_{k2}},\ \{k1, k2\}_{k3},\ k3$$

$$\rightarrow$$

$$\{secret\}_{\{k1\}_{k2}},\ \{k1, k2\},\ k3\ldots$$

$$\rightarrow$$

$$\{secret\}_{\{k1\}_{k2}},\ k1,\ k2,\ k3\ldots$$

Exemple :

$$\{secret\}_{\{k1\}_{k2}},\ \{k1,k2\}_{k3},\ k3$$

$$\rightarrow$$

$$\{secret\}_{\{k1\}_{k2}},\ \{k1,k2\},\ k3\ldots$$

$$\rightarrow$$

$$\{secret\}_{\{k1\}_{k2}},\ k1,\ k2,\ k3\ldots$$

$$\rightarrow$$

$$\{secret\}_{\{k1\}_{k2}},\ \{k1\}_{k2},\ k1,\ k2,\ k3\ldots$$

Exemple :

$$\{secret\}_{\{k1\}_{k2}},\ \{k1,k2\}_{k3},\ k3$$

$$\rightarrow$$

$$\{secret\}_{\{k1\}_{k2}},\ \{k1,k2\},\ k3\ldots$$

$$\rightarrow$$

$$\{secret\}_{\{k1\}_{k2}},\ k1,\ k2,\ k3\ldots$$

$$\rightarrow$$

$$\{secret\}_{\{k1\}_{k2}},\ \{k1\}_{k2},\ k1,\ k2,\ k3\ldots$$

$$\rightarrow$$

$$secret\ldots$$

Intrus force brute	Intrus intelligent
L'intrus compose *tous* les messages qu'il peut.	L'intrus ne considère que les messages attendus.
Pour chaque message acceptable par un agent, l'intrus essaie de l'utiliser pour trouver une faille.	Il n'est pas nécessaire de composer les parties des messages que l'agent ne peut pas encore vérifier ! Il suffit de se souvenir des connaissance pouvant être utilisées pour composer ces parties.
⇒ nombre infini de possibilités	⇒ nombre fini de possibilités, *mais* il faut gérer des contraintes

L'intrus est *paresseux* : il n'essaie de composer que les parties des messages qui sont attendues par les agents.

Détection d'attaque par résolution de contraintes

Une attaque est possible ssi il existe une solution à :
$\text{Réponse}_i \in \mathit{forge}(\text{Message}_1, \ldots, \text{Message}_{i-1})$ pour $i = 1, \ldots, k$
et
$\text{Secret} \in \mathit{forge}(\text{Message}_1, \ldots, \text{Message}_k)$

Borne sur les scénarios d'attaques

Théorème : la taille des messages de l'intrus dans la plus petite attaque d'un protocole cryptographique est bornée par la taille du protocole (dans le modèle à cryptographie parfaite).
Conséquence : procédure efficace de recherche d'attaques.

Point route

Point route

Quels protocoles ?

- Où trouver de vrais protocoles ?
- Comment identifier leurs propriétés ?
- Comment les décrire ?

Quels protocoles ?

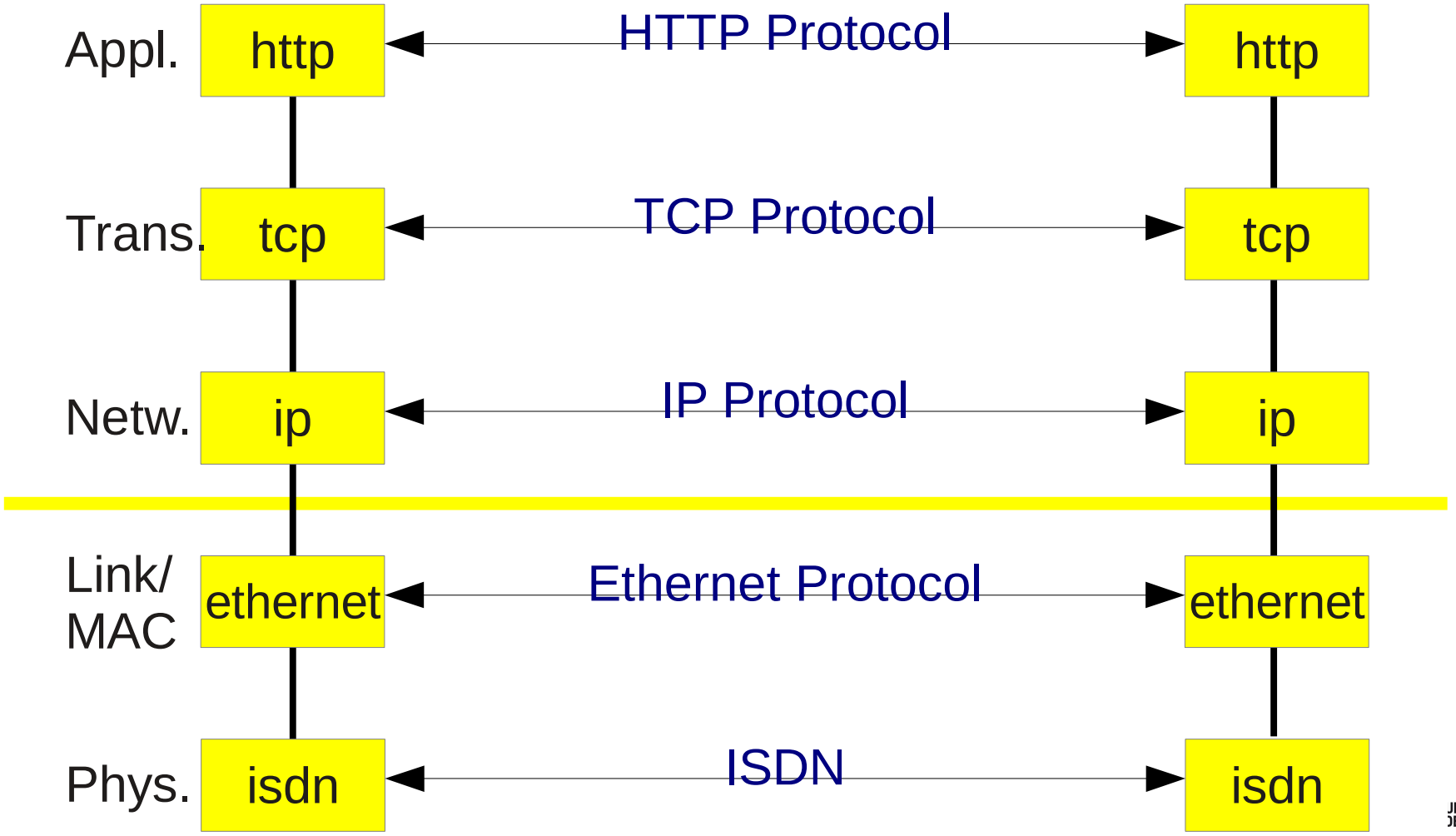
- Où trouver de vrais protocoles ?

  ⇒ collaboration avec des sociétés membres de l'IETF
- Comment identifier leurs propriétés ?

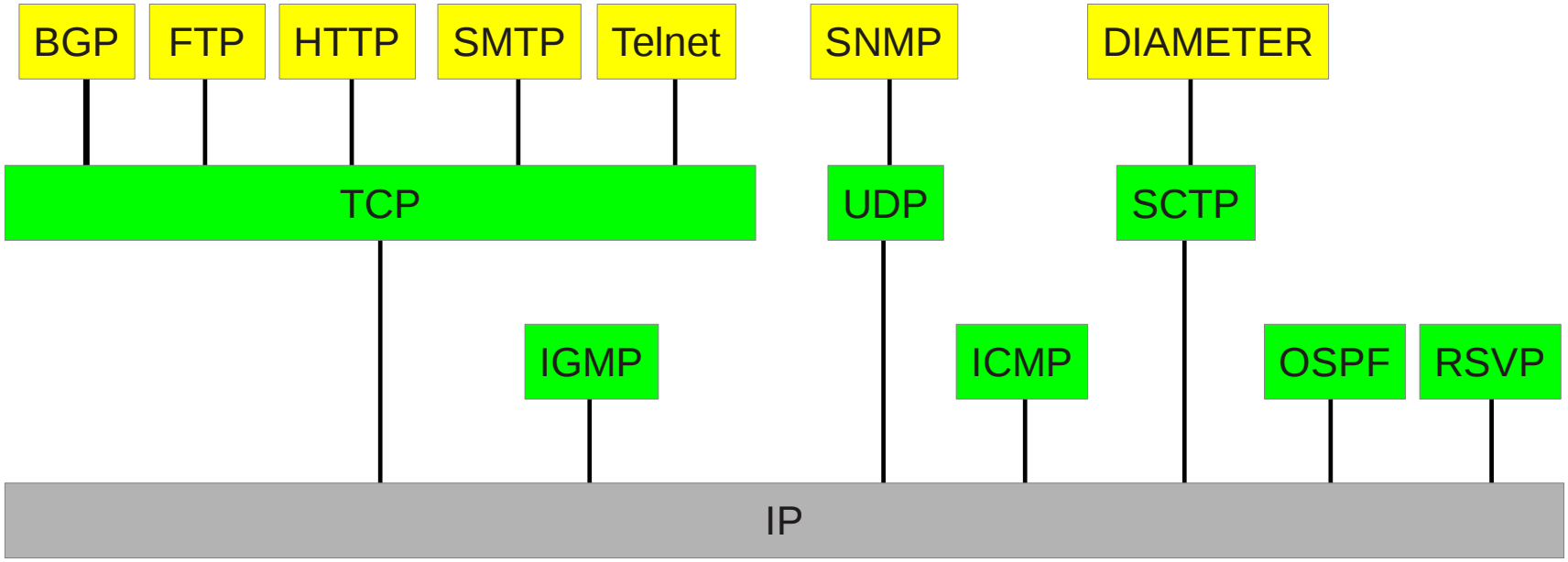
  ⇒ étude des documents produits par l'IETF (`http://www.ietf.org`)
- Comment les décrire ?

  ⇒ utiliser un langage de spécification évolué

Niveaux de protocoles



Protocoles applicatifs



BGP = Border Gateway Protocol
DIAMETER = (2xRADIUS) = New AAA Proto.
FTP = File Transfer Protocol
HTTP = Hypertext Transfer Protocol
ICMP = Internet Control Message Protocol
IGMP = Internet Group Management Protocol
IP = Internet Protocol
OSPF = Open Shortest Path First
RVSP = Resource Reservation Protocol
SMTP = Simple Mail Transfer Protocol
SNMP = Simple Network Management Protocol
TCP = Transmission Control Protocol
UDP = User Datagram Protocol
MIME = Multi-purpose Internet Mail Extension

Protocoles relevant de la sécurité

- Infrastructure (DHCP, DNS, BGP, stime)
- Accès réseaux (WLAN, pana)
- Mobilité (Mobile IP, UMTS-AKA, seamoby)
- VoIP, messaging, presence (SIP, ITU-T H530, impp, simple)
- Sécurité Internet (IKE (IPsec Key agreement), TLS, Kerberos, EAP, OTP, Sacred, ssh, telnet,...)
- Intimité (Geopriv)
- Sécurité pour QoS (NSIS)
- Authentification broadcast/multicast (TESLA)
- E-commerce (Payment)
- Téléchargement sécurisé, protection de contenu (DRM)

Catégories de protocoles

Protocoles distincts par les mécanismes qu'ils mettent en œuvre.

Authentification sur Internet :

- Systèmes à mot de passe jetable : *S/Key*, *OTP*, *SecureID*
- Systèmes de certification de serveurs : *SSL/TLS*, *IPsec*, *S/MIME*
- Systèmes d'authentification de serveurs : *Kerberos*
- Systèmes avec échange de clefs par mot de passe authentifiés : *EKE*, *SPEKE*, *A-EKE*
- Systèmes d'authentification générique : *GSS-API*, *SASL*, *EAP*

Ne pas considérer les systèmes avec mot de passe en clair (*TELNET*, *HTTP*, *RLOGIN*, *POP*, *IMAP*).

Propriétés des protocoles

- *Authentification :* entité ; message ; protection contre le re-jeu.
- *Authentification dans un environnement multicast :* destination implicite ; source.
- *Autorisation par un tiers*.
- *Propriétés d'accord sur clefs :* authentification de clef ; confirmation de clef ; clefs fraîches ; secret parfait « éternel ».
- *Confidentialité*.
- *Anonymat :* en cas d'interception, ou vis-à-vis d'un port.
- *Résistance au déni de service*.
- *Non répudiation* : comptabilisation ; preuve d'émission ; preuve de réception.
- *Propriétés de sécurité temporelle*.

Point route

AAA Mobile IP

Protocole de routage IP pour un hôte sans fil ou mobile.

- Utilisé par un nœud mobile pour s'authentifier dans un réseau visité, afin de recevoir des services de fournisseurs étrangers.
- Utilise une infrastructure AAA (authentication, authorization and accounting).
- Utilisation de chiffrements symétriques.
- Échange de trois clefs de session qui vont servir aux acteurs à s'authentifier.

AAA Mobile IP

Les acteurs

- MN : mobile node
- HA : home agent
- AAAH : home authority
- FA : foreign agent
- AAAL : local authority

Clefs partagées entre...

- MN et AAAH
- HA et AAAH
- AAAH et AAAL
- FA et AAAL

AAA Mobile IP

1. $FA \rightarrow MN : \quad FA, N_{FA}$
2. $MN \rightarrow FA : \quad N_{FA}, MN, AAAH, \{N_{FA}, MN, AAAH\}_{K_{MnAaah}}$
3. $FA \rightarrow AAAL : \quad N_{FA}, MN, AAAH, \{N_{FA}, MN, AAAH\}_{K_{MnAaah}}$
4. $AAAL \rightarrow AAAH : \quad N_{FA}, MN, AAAH, \{N_{FA}, MN, AAAH\}_{K_{MnAaah}}$
5. $AAAH \rightarrow HA : \quad MN, \{K_{MnHa}, K_{FaHa}\}_{K_{AaahHa}}, \{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}},$
   $\{MN, \{K_{MnHa}, K_{FaHa}\}_{K_{AaahHa}}, \{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}}\}_{K_{AaahHa}}$
6. $HA \rightarrow AAAH : \quad \{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}}, \{\{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}}\}_{K_{MnHa}},$
   $\{\{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}}, \{\{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}}\}_{K_{MnHa}}\}_{K_{AaahHa}}$
7. $AAAH \rightarrow AAAL : \quad N_{FA}, \{K_{MnFa}, K_{FaHa}\}_{K_{AaahAaal}}, \{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}},$
   $\{\{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}}\}_{K_{MnHa}}, \{N_{FA}, \{K_{MnFa}, K_{FaHa}\}_{K_{AaahAaal}},$
   $\{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}}, \{\{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}}\}_{K_{MnHa}}\}_{K_{AaahAaal}}$
8. $AAAL \rightarrow FA : \quad N_{FA}, \{K_{MnFa}, K_{FaHa}\}_{K_{FaAaal}}, \{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}},$
   $\{\{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}}\}_{K_{MnHa}}, \{N_{FA}, \{K_{MnFa}, K_{FaHa}\}_{K_{FaAaal}},$
   $\{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}}, \{\{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}}\}_{K_{MnHa}}\}_{K_{FaAaal}}$
9. $FA \rightarrow MN : \quad \{K_{MnFa}, K_{FaHa}\}_{K_{FaAaal}}, \{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}},$
   $\{\{K_{MnFa}, K_{MnHa}\}_{K_{MnAaah}}\}_{K_{MnHa}}$

Protocole pour Porte-monnaie électronique

- Protocole défini par France Telecom R&D.
- Permet de réaliser une transaction entre un porte-monnaie électronique et un serveur.
- But : garantir un bon niveau de sécurité.

Existe en deux versions :

- Approche symétrique.
- Approche asymétrique.

La première est plus performante (chiffrement symétrique moins coûteux).

Protocole pour Porte-monnaie électronique

Les acteurs :

- EP : porte-monnaie électronique
- SAM : serveur
- TTP : tiers de confiance

Principe (approche symétrique) :

- Échange de challenges (nonces)
- SAM envoie le montant en clair à EP
- EP modifie sa balance et transmet un message spécial à SAM
- SAM modifie sa balance et stocke une preuve de la transaction (S6)

F_{ep_isskey} : donnée secrète entre EP et TTP.

Protocole pour Porte-monnaie électronique

Propriétés :

- Non création de fausse monnaie : balances équilibrées entre EP et SAM.
- Non création de faux litiges : SAM ne stocke pas de fausses preuves.

En cas de réclamation de EP, SAM doit prouver sa bonne foi en exhibant une preuve (message S6) engendrée lors de la session.

EP		**SAM**
ep_acqkey, ep_isskey		acqkey
	$\longrightarrow^{Id\_EP, chall\_EP}$	
		$ep\_acqkey = F_{ep\_acqkey}(Id\_EP)$ $S2 = F_{ep\_acqkey}(Id\_EP, chall\_EP)$
Verify $S2$	$\longleftarrow^{Id\_SAM, chall\_SAM, S2}$	
	$\longleftarrow^{Amount}$	
$bal = bal - Amount$ $S6 = F_{ep\_isskey}(Id\_SAM, chall\_SAM, chall\_EP, Amount)$ $S3 = F_{ep\_acqkey}(Id\_SAM, chall\_SAM, chall\_EP, S6, Amount)$		
	$\longrightarrow^{S3, S6, Amount}$	Verify $S3$ $bal = bal + Amount$ Store $S6$, $chall\_SAM$, $chall\_EP$, $Amount$

Protocole de non-répudiation : FairZG

Deux agents (A et B) veulent échanger un message, mais en obtenant un preuve de la participation de l'autre à l'échange.

Problématique

- Chacun doit arriver à suivre les actions de l'autre.
- Essayer de synchroniser la découverte du message par B et l'obtention de la preuve par A que B peut effectivement le lire.

Protocole de non-répudiation : FairZG

1. $A \rightarrow B$: $fNRO.B.L.\{M\}_K.NRO$
   où $NRO = \{fNRO.B.L.\{M\}_K\}_{SK_a}$
2. $B \rightarrow A$: $fNRR.A.L.NRR$
   où $NRR = \{fNRR.A.L.\{M\}_K\}_{SK_b}$
3. $A \rightarrow TTP$: $fSUB.B.L.K.SubK$
   où $SubK = \{fSUB.B.L.K\}_{SK_a}$

4a. $B \leftrightarrow TTP$: $fCON.A.B.L.K.ConK$
   où $ConK = \{fCON.A.B.L.K\}_{SK_{ttp}}$

4b. $A \leftrightarrow TTP$: $fCON.A.B.L.K.ConK$

A la fin : A et B connaissent M, et peuvent prouver la participation de l'autre à la communication.

Protocole FairZG : Propriétés

Non-répudiation d'origine avec les évidences suivantes pour B :

$$\{NRO, ConK\} = \{\{fNRO.B.L.\{M\}_K\}_{SK_a}, \{fCON.A.B.L.K\}_{SK_{ttp}}\}$$

Non-répudiation de réception avec les évidences suivantes pour A :

$$\{NRR, ConK\} = \{\{fNRR.A.L.\{M\}_K\}_{SK_b}, \{fCON.A.B.L.K\}_{SK_{ttp}}\}$$

Equité :
A la fin d'une exécution du protocole, soit A et B ont tous les deux leurs évidences, soit aucun ne les a.

Protocole de groupe

Protocole A-GDH.2 :

- Objectif : construire une clef de groupe (partagée par tous lesmembres d'un groupe).
- Moyen : utilisation des propriétés de l'exponentielle (principe de Diffie-Hellman).

$a_1 \longrightarrow a_2 : \alpha, \alpha^{r_1}$
$a_2 \longrightarrow a_3 : \alpha^{r_2}, \alpha^{r_1}, \alpha^{r_1 r_2}$
$a_3 \longrightarrow a_4 : \alpha^{r_2 r_3}, \alpha^{r_1 r_3}, \alpha^{r_1 r_2}, \alpha^{r_1 r_2 r_3}$
$a_4 \longrightarrow a_1, a_2, a_3 : \alpha^{r_2 r_3 r_4 k_{14}}, \alpha^{r_1 r_3 r_4 k_{24}}, \alpha^{r_1 r_2 r_4 k_{34}}$

où

r_i est un nombre aléatoire engendré par a_i, et α est le générateur.

Au final, chaque participant peut calculer : $\alpha^{r_1 r_2 r_3 r_4}$.

Point route

Point route

Langage de spécification

À quoi servent les spécifications de protocoles ?

- une description rédigeable par des industriels,
- un guide pour l'implantation,
- une description formelle connectable à des outils de vérification.

Langage de spécification

Bonnes propriétés des langages de spécification :

- simplicité, compréhension,
- flexibilité,
- non ambiguïté,
- modularité,
- expressivité : contrôle de flux, connaissances, primitives cryptographiques, types de messages, propriétés algébriques.

Problème : certaines propriétés peuvent sembler contradictoires.

Langage de spécification

Solution de l'outil AVISPA : deux langages.

1. Un langage de spécification haut-niveau (HLPSL) : pour les utilisateurs non experts.
2. Un langage de spécification bas niveau (IF) : pour l'implantation ou les outils de vérification.

. . . avec une traduction automatique du HLPSL vers le IF.

Spécifications : exemple

Doit être simple et facile à comprendre,
mais suffisamment expressif pour considérer des protocoles sérieux.

Exemple : NSPK

$$A \to B : \quad \{N_A, A\}_{PK_B}$$

$$B \to A : \quad \{N_A, N_B\}_{PK_A}$$
$$A \to B : \quad \{N_B\}_{PK_B}$$

Spécifications : exemple

Doit être simple et facile à comprendre,
mais suffisamment expressif pour considérer des protocoles sérieux.

Exemple : NSPK Key Server

si A ne connaît pas PK_B,

$$A \to S : \quad A, B$$
$$S \to A : \quad \{B, PK_B\}_{SK_S}$$
$$A \to B : \quad \{N_A, A\}_{PK_B}$$

si B ne connaît pas PK_A,

$$B \to S : \quad B, A$$
$$S \to B : \quad \{A, PK_A\}_{SK_S}$$
$$B \to A : \quad \{N_A, N_B\}_{PK_A}$$
$$A \to B : \quad \{N_B\}_{PK_B}$$

Spécification modulaire à base de rôles

Deux types de rôles :

- Rôles principaux : alice (initiateur), bob (correspondant), serveur,
- Rôles composés :
  - définition d'une session : une alice et un bob,
  - instantiations : un serveur, plusieurs sessions.

→ Chaque rôle a un environnement local.

Contrôle de flux : transitions gardées.
Sémantique : explicite.
⇒ un langage basé sur TLA (Temporal Logic of Actions)

Spécification : un rôle principal

```
role bob(A,B: agent, Kb,Ks: public_key,
         KeyRing: (agent.public_key) set,
         SND,RCV: channel(dy)) played_by B def=
   local State: nat, Na,Nb: text, Ka: public_key
   init State:=0
   transition
      1a. State=0 /\ RCV({Na'.A}_Kb) /\ in(A.Ka',KeyRing)
          =|> State':=2 /\ Nb':=new()
             /\ SND({Na'.Nb'}_Ka')
      ...
end role
```

→ Définition d'information locale

Spécification : un rôle principal

```
role bob(A,B: agent, Kb,Ks: public_key,
         KeyRing: (agent.public_key) set,
         SND,RCV: channel(dy)) played_by B def=
   local State: nat, Na,Nb: text, Ka: public_key
   init State:=0
   transition
      1a. State=0 /\ RCV({Na'.A}_Kb) /\ in(A.Ka',KeyRing)
          =|> State':=2 /\ Nb':=new()
             /\ SND({Na'.Nb'}_Ka')
      ...
end role
```

→ Définition de transitions (à la TLA)

Spécification : un rôle principal

```
role bob(A,B: agent, Kb,Ks: public_key,
         KeyRing: (agent.public_key) set,
         SND,RCV: channel(dy)) played_by B def=
   local State: nat, Na,Nb: text, Ka: public_key
   init State:=0
   transition
      1a. State=0 /\ RCV({Na'.A}_Kb) /\ in(A.Ka',KeyRing)
          =|> State':=2 /\ Nb':=new()
             /\ SND({Na'.Nb'}_Ka')
      ...
end role
```

→ Utilisation de types composés et de canaux.

Spécification : composer les rôles

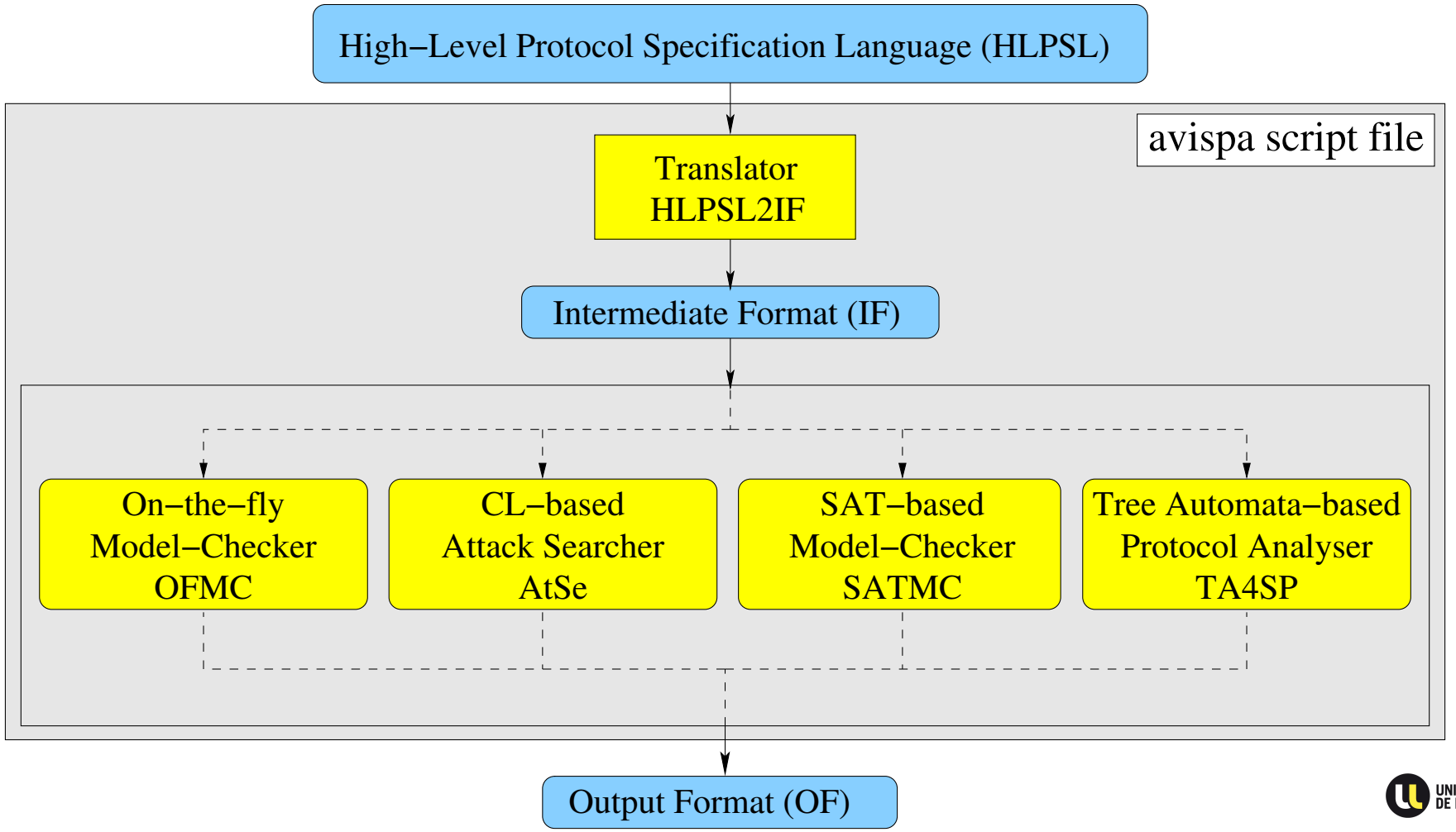
```
role session(A,B: agent,
             KeySet: agent -> (agent.public_key) set),
             Ka,Kb,Ks: public_key def=
   local SND,RCV: channel(dy)
   composition
      alice(A,B,Ka,Ks,KeySet(A),SND,RCV)
      /\ bob(A,B,Kb,Ks,KeySet(B),SND,RCV)
end role
```

Spécification : composer les rôles

```
role environment() def=
   local KeySet: agent -> (agent.public_key) set,
      KeyRing: (agent.public_key) set,
      Snd,Rcv: channel(dy)
   const a,b,s,i: agent,   ka,kb,ks,ki: public_key
   init KeySet:={ a.{}, b.{a.ka}, i.{a.ka,b.kb} }
        /\ KeyRing:={a.ka,b.kb,s.ks,i.ki}
   intruder_knowledge={i,a,b,ks,ki,inv(ki)}
   composition
      server(s,ks,KeyRing,Snd,Rcv)
      /\ session(a,b,KeySet,ka,kb,ks)
      /\ session(a,i,KeySet,ka,ki,ks)
end role
```

Point route

Architecture de l'outil



Compilateur vers un format intermédiaire

A pour but de :

- vérifier la syntaxe,
- vérifier la sémantique,

avec en plus :

- typage intégral des constantes et variables,
- possibilité de décomposer les buts,
- possibilité d'engendrer automatiquement des instances de sessions,
- modèle de l'intrus défini indépendamment du protocole.

Spécification : IF

Exemple de règle engendrée par le traducteur de HLPSL en IF :

```
step step_0 (S,Ks,KeyMap,Dummy_A,Dummy_B,Dummy_Kb,A,B,Kb,SID) :=
        state_Server(S,Ks,KeyMap,Dummy_A,Dummy_B,Dummy_Kb,SID).
        iknows(pair(A,B)).
        contains(pair(B,Kb),KeyMap)
    =>
        state_Server(S,Ks,KeyMap,A,B,Kb,SID).
        iknows(crypt(inv(Ks),pair(B,Kb))).
        contains(pair(B,Kb),KeyMap)
```

Point route

Point route

Mécanismes cryptographiques

Considérer les mécanismes cryptographiques :

- algorithmes de chiffrement/déchiffrement,
- algorithmes de calcul de clefs,
- ...

Par exemple :

services	SSL					
mécanismes	signatures		chiffrement		hachage	
algorithmes	DSA	RSA	RSA	DES	SHA1	MD5

Ou exclusif — Masque jetable

- *Rappel :* $0 \oplus 0 = 1 \oplus 1 = 0$ et $0 \oplus 1 = 1 \oplus 0 = 1$
- Alice et Bob partagent une clef aléatoire $k = 010001\ldots$ de longueur supérieure à celle du message m et jamais utilisée avant. Alice envoie m à Bob :
  $A \rightarrow B : c = m \oplus k$
- Bob déchiffre par : $(m \oplus k) \oplus k = m$
- $\Rightarrow$ Garantit le secret, mais le masque ne peut être utilisé 2 fois : si l'intrus récupère 2 messages codés avec le même masque, il obtient de l'information :
  $(m1 \oplus k) \oplus (m2 \oplus k) = m1 \oplus m2$

Clefs symétriques

- Chiffrement par bloc par l'algorithme DES (Data Encryption Standard, 1977).
- Blocs de 64 bits
- Clefs de 56 bits (16 sous-clefs k_i engendrées)
- Un bloc est scindé en deux parties L_0 et R_0, chacune de 32 bits.
- 16 cycles de l'algorithme suivant :
  $L_i = R_{i-1}$
  $R_i = L_{i-1} \oplus f(R_{i-1}, k_i)$
  (f combine permutations et substitutions)

Clefs publiques

- Algorithme RSA (Rivest, Shamir, Adleman).
- Basé sur la difficulté de factoriser les entiers : facile de trouver 2 grands nombres premiers, difficile de factoriser le produit de 2 nombres premiers.
- Déterministe : pour chaque texte en clair et chaque clef publique, un seul texte chiffré.
- Petit théorème de Fermat :
  $m^{p-1} \equiv 1 \pmod{p}$ si p premier et $\text{pgcd}(m, p) = 1$
  $m^{\phi(n)} \equiv 1 \pmod{n}$ si $\text{pgcd}(m, n) = 1$, $n = p_1 p_2 \dots p_k$ où p_i premiers, et $\phi(n) = (p_1 - 1)(p_2 - 1) \dots (p_k - 1)$

Échange de clefs par Diffie-Hellman(-Merkle)

- Basé sur la difficulté de calculer un logarithme discret : étant donnés a et b, trouver x tel que $a^x = b$.
- Alice et Bob choisissent g et n, entiers tels que n premier et g est primitif par rapport à n (pour chaque $1 \leq b < n$, il existe a tel que $g^a = b \mod n$).
- Alice choisit un grand nombre aléatoire a et envoie à Bob : $A = g^a \mod n$.
  Bob choisit un grand nombre aléatoire b et envoie à Alice : $B = g^b \mod n$.
- Alice calcule $B^a \mod n$ $(= g^{ab} \mod n)$.
  Bob calcule $A^b \mod n$ $(= g^{ab} \mod n)$.
- $\Rightarrow$ $g^{ab} \mod n$ est une clef secrète (si le log discret est difficile à calculer).

Pourquoi s'intéresser à tout ça ?

Il existe des attaques basées sur les propriétés de ces algorithmes.

Exemple 1 :

Encore une attaque sur NSPK corrigé !

1. $A \rightarrow B : \quad \{N_A, A\}_{PK_B}$
2. $B \rightarrow A : \quad \{N_B, N_A \oplus B\}_{PK_A}$
3. $A \rightarrow B : \quad \{N_B\}_{PK_B}$

Pourquoi s'intéresser à tout ça ?

Il existe des attaques basées sur les propriétés de ces algorithmes.

Exemple 1 :

Encore une attaque sur NSPK corrigé !

1. $A \to B : \quad \{N_A, A\}_{PK_B}$
2. $B \to A : \quad \{N_B, N_A \oplus B\}_{PK_A}$
3. $A \to B : \quad \{N_B\}_{PK_B}$

1. $A \to I : \quad \{N_A, A\}_{PK_I}$
1'. $I(A) \to B : \quad \{N_A \oplus B \oplus I, A\}_{PK_B}$
2'. $B \to I(A) : \quad \{N_B, N_A \oplus B \oplus I \oplus B\}_{PK_A}$
2. $I \to A : \quad \{N_B, N_A \oplus B \oplus I \oplus B\}_{PK_A}$
3. $A \to I : \quad \{N_B\}_{PK_I}$
3. $I(A) \to B : \quad \{N_B\}_{PK_B}$

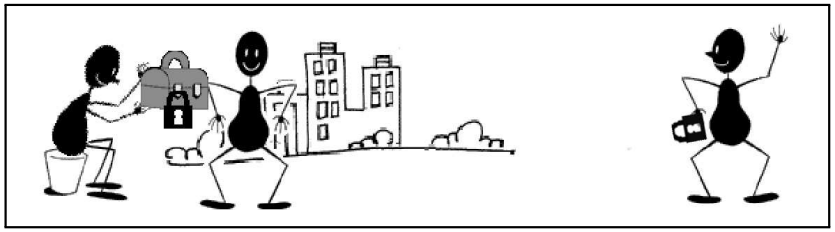
Attaque si : $N_A \oplus B \oplus I \oplus B = N_A \oplus I$

Pourquoi s'intéresser à tout ça ?

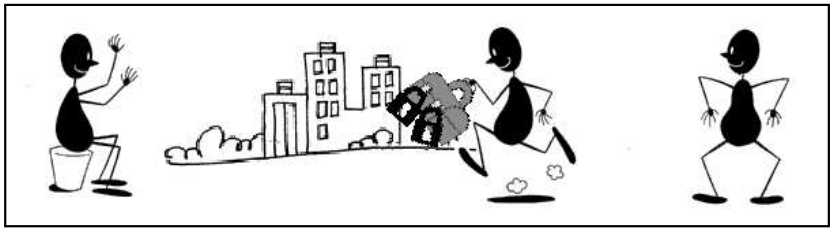
Exemple 2 :

Protocole à la RSA, $\{\{L\}_{K_a}\}_{K_b} = \{\{L\}_{K_b}\}_{K_a}$

1. $A \rightarrow B : \{L\}_{K_a}$

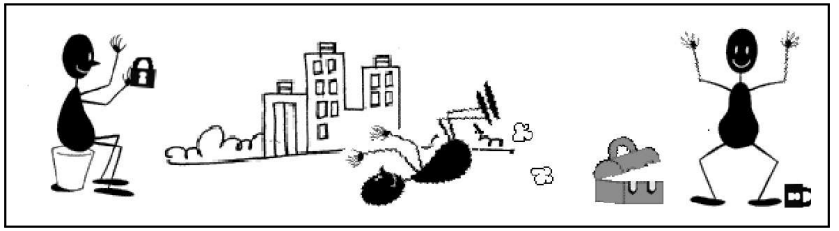


2. $B \rightarrow A : \{\{L\}_{K_a}\}_{K_b}$



3. $A \rightarrow B : \{L\}_{K_b}$

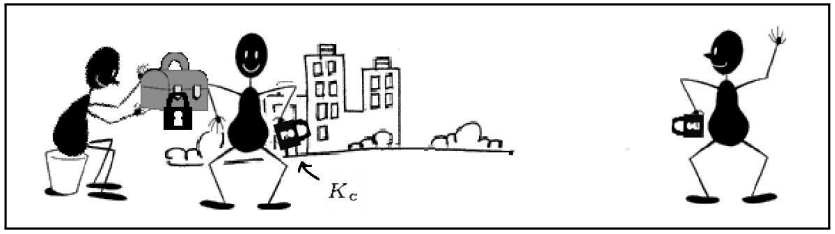
4.



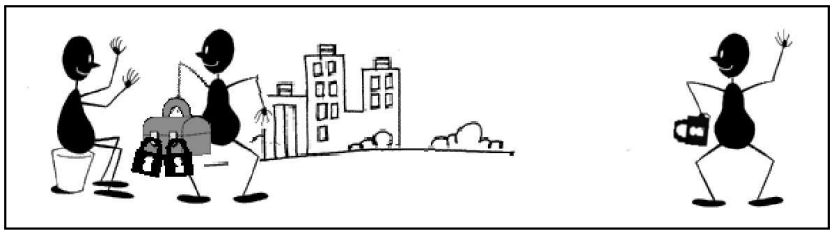
Pourquoi s'intéresser à tout ça ?

Attaque de confidentialité :

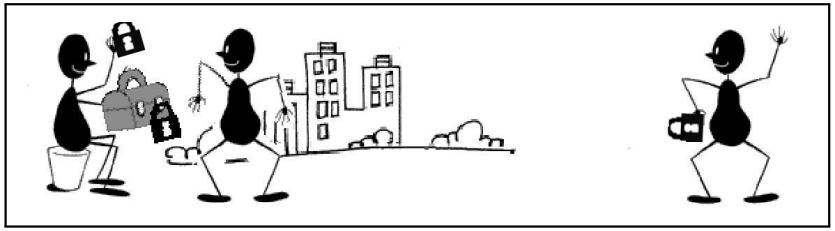
1. $A \rightarrow I(B) : \{L\}_{K_a}$



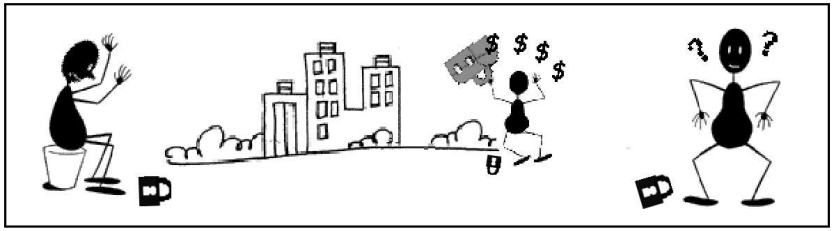
2. $I(B) \rightarrow A : \{\{L\}_{K_a}\}_{K_i}$



3. $A \rightarrow I(B) : \{L\}_{K_i}$



4.



Pourquoi s'intéresser à tout ça ?

Pour décrire et analyser l'implantation des protocoles.

Exemple :

$$\begin{cases} A \to B : & exp(g, k_1), \langle A, N_A \rangle \oplus exp(exp(g, b), k_1) \\ B \to A : & exp(g, k_2), \langle N_A, N_B \rangle \oplus exp(exp(g, a), k_2) \\ A \to B : & exp(g, k_3), N_B \oplus exp(exp(g, b), k_3) \end{cases}$$

Implantation de NSPK avec le chiffrement de El Gamal.

Trois opérateurs spécifiques utilisés : $\oplus$, concaténation, exponentiation.

Point route

Validation de protocoles

- Trouver une attaque est beaucoup plus facile que de valider un protocole ! (vite indécidable)
- « valider » $\Rightarrow$ nombre non borné de sessions !
- Il faut donc se focaliser sur des propriétés bien précises.
- Il faut aussi utiliser des abstractions : sous-/sur-approximations des connaissances de l'intrus.

Conclusion

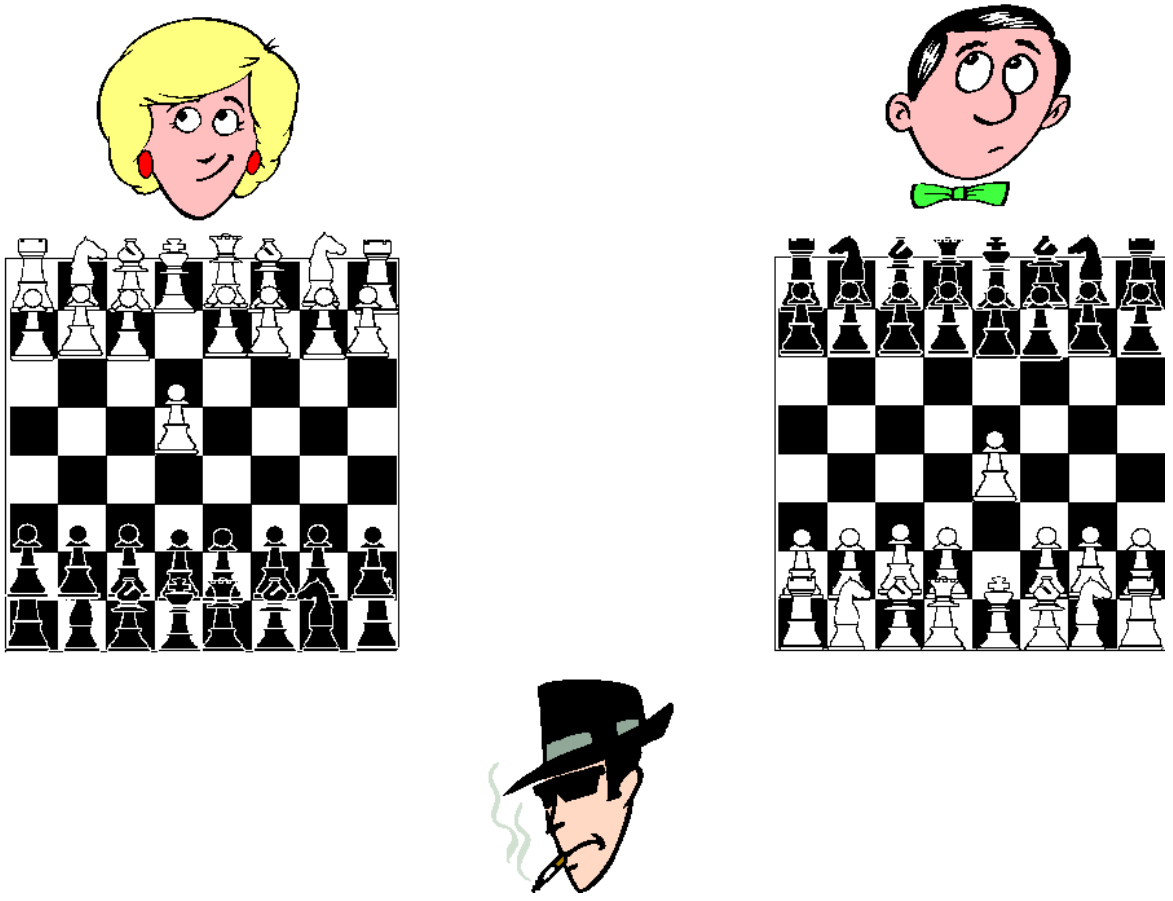
La sécurité des systèmes d'informations et de toutes les communications à distance se base sur :

- la cryptographie,
- les protocoles cryptographiques.

Cependant,

- les algos de chiffrement ne sont pas toujours suffisamment robustes (taille des clefs, MD5),
- les protocoles sont rarement scientifiquement vérifiés, car souvent développés « à la main », et même faillibles ils ne sont pas toujours corrigés (yescard).

$\Rightarrow$ pas très rassurant, mais faut pas être parano...



Crédits

Outre les travaux de l'équipe Pesto du LORIA, ce cours s'est inspiré d'informations provenant de :

Florent Jacquemard ENS Cachan

Anas Abou El Kamal ENSI Bourges

CLUSIF `https://www.clusif.asso.fr/`