



Gwenn Feunteun  
gwenn@acceis.fr

# Architecture & sécurité réseau

Comment se protéger



# Se protéger

Comment se protéger contres les attaques réseaux (mais aussi un parfois applicatives) ?

- **En mettant en œuvre une architecture réseau sécurisé.** Afin de rendre difficile les compromissions ou d'en limiter les impacts.
- **En protégeant les flux de communication.** Afin de potentiellement détecter les attaques et d'assurer la confidentialité et l'intégrité des échanges.

# Pourquoi & comment ?

Détourner, intercepter ou modifier des échanges entre deux interlocuteurs peut être relativement facile.

Si l'on ne maîtrise pas le canal de communication (Internet par exemple) ou si l'on ne peut se permettre qu'il soit compromis (car l'information est sensible). Il faut assurer la sécurité au plus proche du message.

# Pourquoi & comment ?

De nombreux protocoles n'ont pas été conçus avec la sécurité en tête :

- Telnet ;
- FTP ;
- LDAP ;
- HTTP ;
- DNS ;
- Etc.

Aucun de ces protocole ne prend la peine d'assurer la confidentialité des échanges. Les mots de passe transitent donc en clair sur le réseau.

# Rappel / Intro Crypto



# Historique de la cryptographie

La cryptographie est une discipline très ancienne puisqu'on rapporte des exemples d'usages datant de l'Égypte des pharaons. Toutefois, sa mise en œuvre est restée très longtemps cantonnée aux besoins militaires et diplomatiques. Avec l'explosion des communications électroniques et la nécessité de les sécuriser, la cryptographie est apparue comme indispensable et a ainsi opéré une transition vers le domaine civil. En parallèle, elle a subi un développement rapide depuis ces dernières décennies, rythme qui ne semble pas ralentir. Voici quelques repères historiques :

- -404 av. J.C. – La scytale ou bâton de Plutarque ;
- -50 av. J.C. – le chiffre de César ;
- 1586 – Le chiffre de Vigenère ;
- 1883 – Le Principe de Kerckhoffs ;
- 1975 – D.E.S. (*Data Encryption Standard*) ;
- 1976 – Diffie-Hellman (ou D.H.) ;
- 1977 – R.S.A. (MM. Rivest, Shamir et Adleman en sont les inventeurs) ;
- 2000 – A.E.S. (*Advanced Encryption Standard*).

# Services de la cryptographie

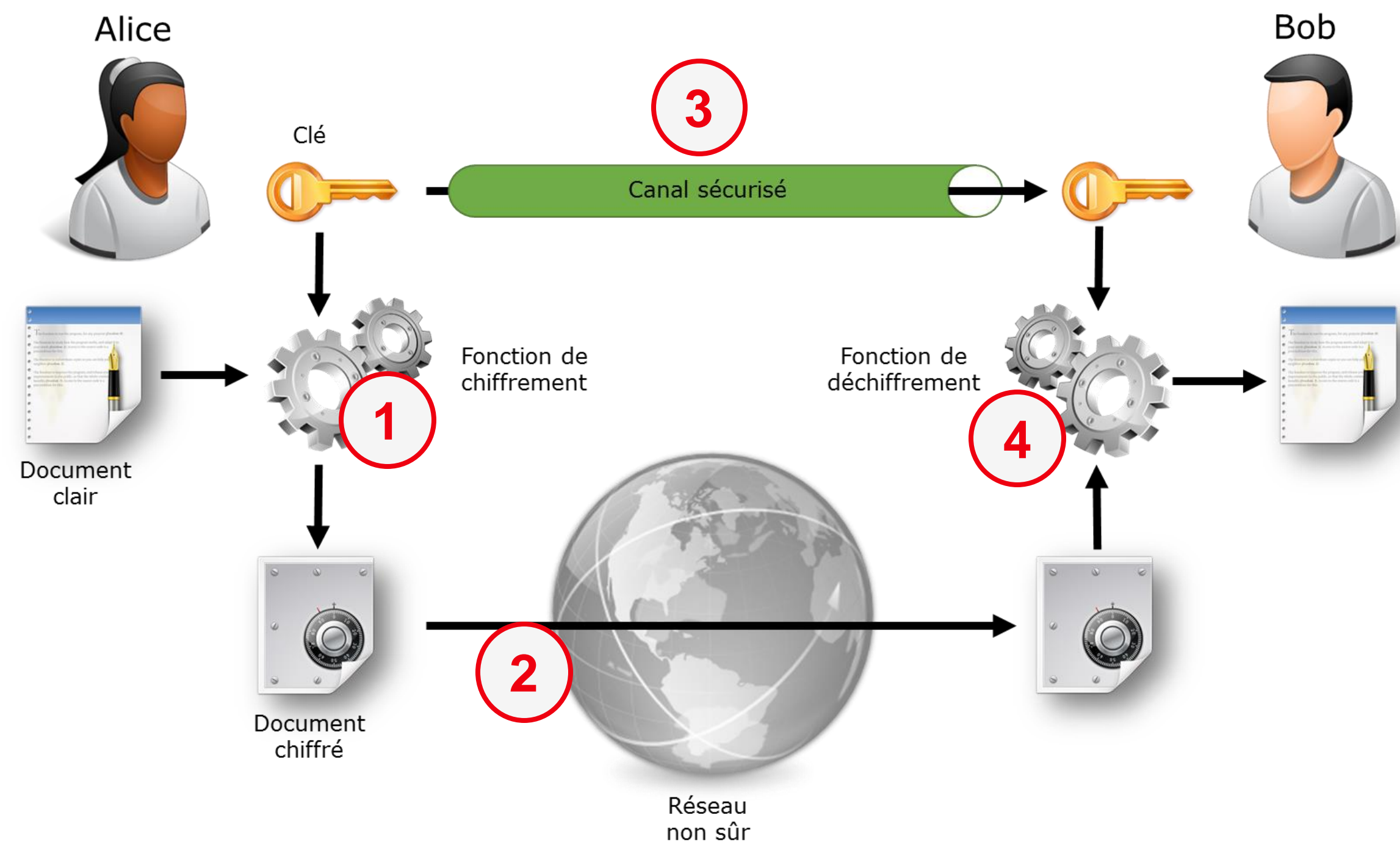
En fonction de son usage de la cryptographie permet d'offrir plusieurs services de sécurité :

- **Confidentialité**, afin de s'assurer que les informations ne seront accessibles qu'aux seules personnes autorisées ;
- **Intégrité**, pour garantir que les informations ne sont pas altérées (intentionnellement ou non) pendant leur transport ou leur stockage ;
- **Authentication**, pour prouver l'identité d'une personne ou garantir l'origine de l'information ;
- **Signature**, permettant à une entité de prendre par un contrat sans possibilité de renier ses engagements (non répudiation).



# Cryptographie symétrique

Les opérations de chiffrement et de déchiffrement sont assurées avec la même clé. C'est pourquoi on parle de cryptographie symétrique.



1. Alice chiffe le document qu'elle veut transmettre à Bob au moyen d'une clé choisie.
2. Elle transmet le document chiffré à Bob (via Internet par exemple).
3. Elle lui communique également la clé utilisée via un canal sécurisé (la clé peut avoir été définie entre eux au préalable).
4. Bob déchiffre le document chiffré en utilisant la même clé qu'Alice.

# Cryptographie symétrique

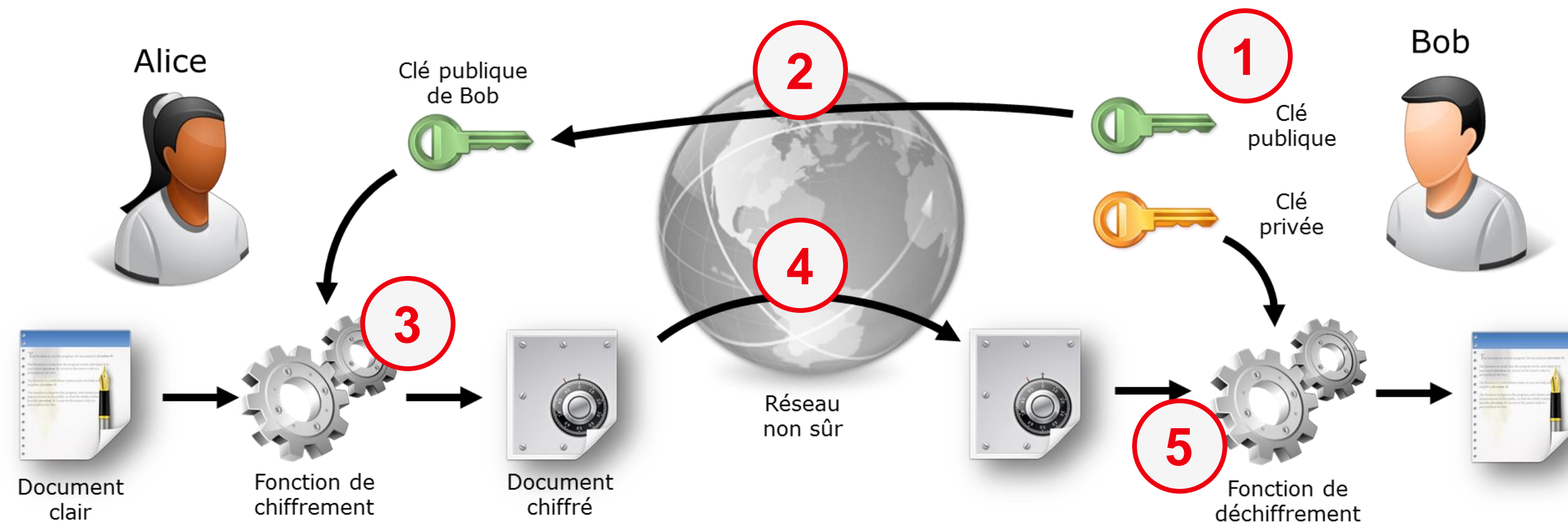
De nombreux indices indiquent que l'emploi de clé de moins de 100 bits semble risqué. Les clés de 56 bits (telles que celles utilisées pour DES) sont clairement insuffisantes et la capacité actuelle à attaquer des clés de 64 bits est aujourd'hui admise, même si un tel calcul n'est pas à la portée de n'importe qui. De telles attaques ont cependant déjà été menées concrètement dans le milieu public.



- La taille minimale des clés symétriques utilisées jusqu'en 2020 doit être de 100 bits ;
- La taille minimale des clés symétriques devant être utilisées au-delà de 2020 doit être de 128 bits ;

# Cryptographie asymétrique

Les opérations de chiffrement et de déchiffrement sont assurées avec des clés différentes. Ces clés sont liées entre-elles par une relation mathématique particulière établie lors de leur génération. On les appelle généralement un bi-clé. La principale conséquence est que ce qui est chiffré avec une de ces clés ne peut être déchiffré que par l'autre.



1. Bob génère un bi-clé.
2. Il transmet sa clé publique à Alice et conserve sa clé privée, **qu'il doit être le seul à connaître.**
3. Alice utilise la clé publique de Bob pour chiffrer le document.
4. Elle transmet le document chiffré à Bob (via Internet par exemple).
5. Bob déchiffre le document chiffré en utilisant sa clé privée.



# Cryptographie asymétrique

Le **problème de la factorisation** consiste à retrouver la décomposition en facteurs premiers d'un entier donné, obtenu de manière secrète par multiplication de deux nombres premiers, généralement de taille comparable. Un tel nombre composé est classiquement appelé « module ». Le problème de la factorisation est principalement utilisé par le cryptosystème RSA.



- La taille minimale du module doit être de 2048 bits, pour une utilisation ne devant pas dépasser 2030 ;
- Pour une utilisation au-delà de 2030, la taille minimale du module doit être de 3072 bits ;

# Fonction de hashage

Une fonction de hashage est un algorithme permettant de fournir une empreinte en principe unique d'une source de données. Cette opération est réalisée en combinant des informations de la source afin de produire une suite d'octet de taille réduite la caractérisant. Cette empreinte est aussi appelée hash, résumé ou condensat cryptographique. L'opération de hashage est à sens unique, car il y a une perte d'information durant le calcul.

En pratique, l'empreinte ne peut pas être unique car l'espace de destination (l'ensemble des empreintes possibles) est fini. Quand deux sources ont le même hash, on appelle cela **une collision**. L'important pour un algorithme de hashage est qu'on ne puisse pas produire de collision de manière arbitraire.



# Fonction de hachage

Les fonctions de hachage cryptographiques doivent avoir plusieurs propriétés telles que la résistance à la recherche de « collisions ». Celles-ci peuvent cependant être trouvées au moyen d'attaques génériques fondées sur le « paradoxe des anniversaires ». Afin de les contrer une empreinte doit être deux fois plus longue qu'une clé symétrique pour atteindre le même niveau de robustesse.

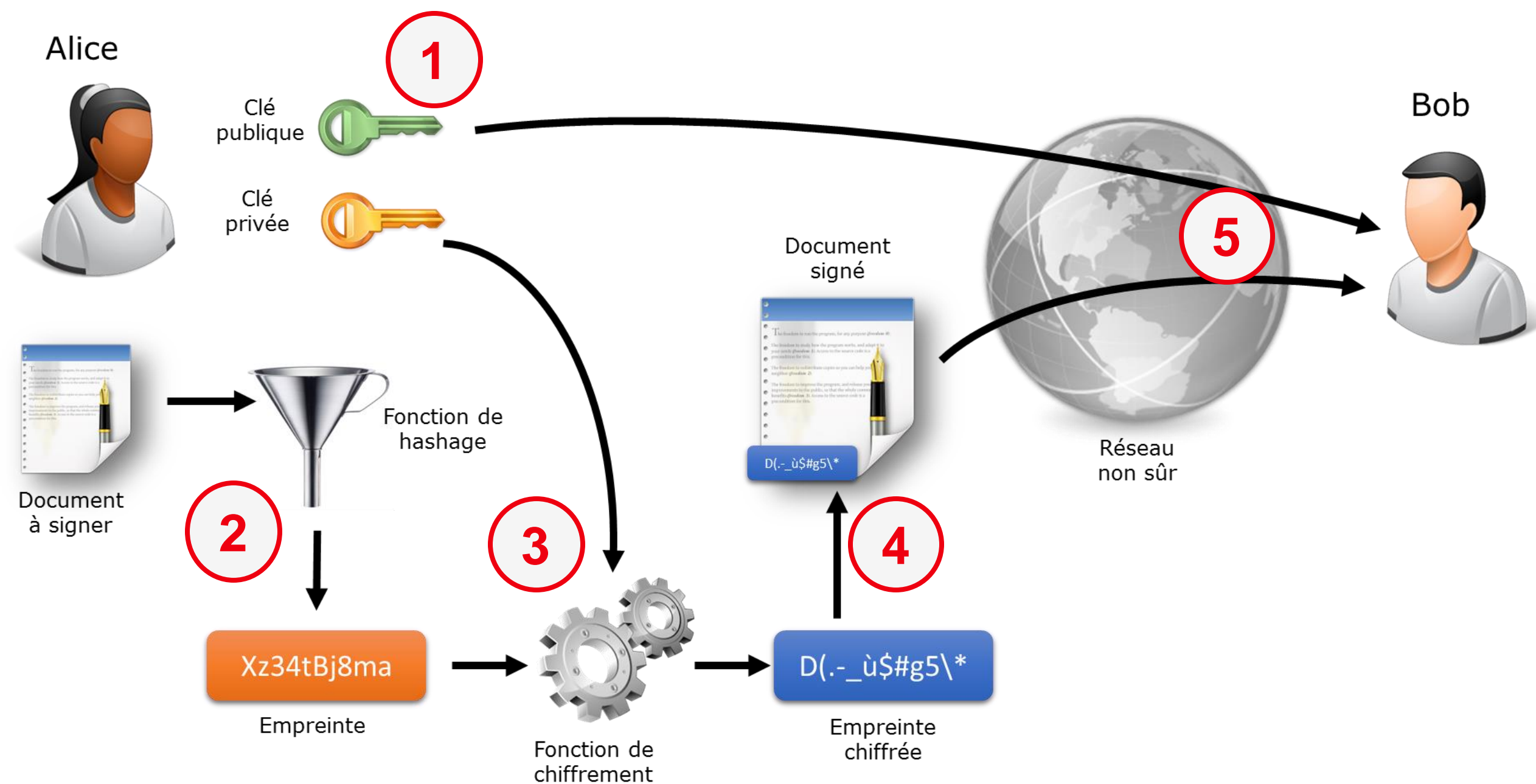


- Pour une utilisation ne devant pas dépasser 2020, la taille minimale des empreintes générées par une fonction de hachage doit être de 200 bits.
- Pour une utilisation au-delà de 2020, la taille minimale des empreintes générées par une fonction de hachage doit être de 256 bits.
- L'emploi de fonctions de hachage pour lesquelles des « attaques partielles » sont connues est déconseillé.



# Signature numérique

La signature numérique est une opération consistant à apposer une marque sur un document permettant de garantir son authenticité (validité de l'identité du signataire) et son intégrité (absence d'altération depuis la signature).



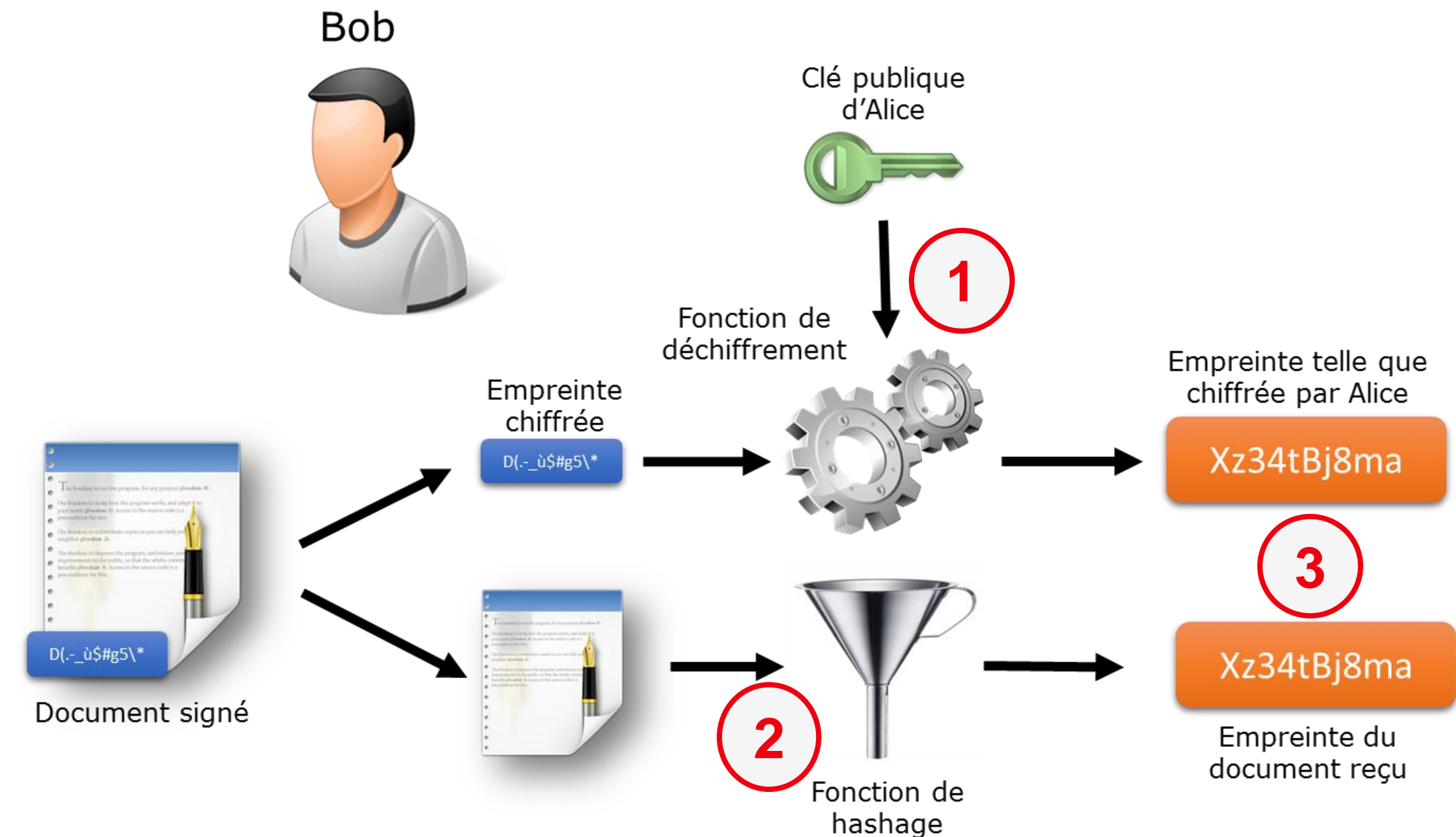
## Signature d'un document

1. Alice génère un bi-clé.
2. Elle calcul l'empreinte du document à signer.
3. Elle chiffre cette empreinte avec sa clé privée.
4. Elle ajoute l'empreinte chiffrée au document initial.
5. Elle transmet le document et sa clé publique à Bob.

# Signature numérique

## Vérification de la signature

1. Bob déchiffre l'empreinte présente sur le document avec la clé publique d'Alice.
2. Il calcul l'empreinte, hors signature, du document qu'il a reçu.
3. Si les empreintes sont identiques, c'est que le document n'a pas été altéré depuis sa signature et que c'est bien Alice qui l'a signée (puisque'elle seule dispose de la clé privée qui a permis de chiffrer l'empreinte).



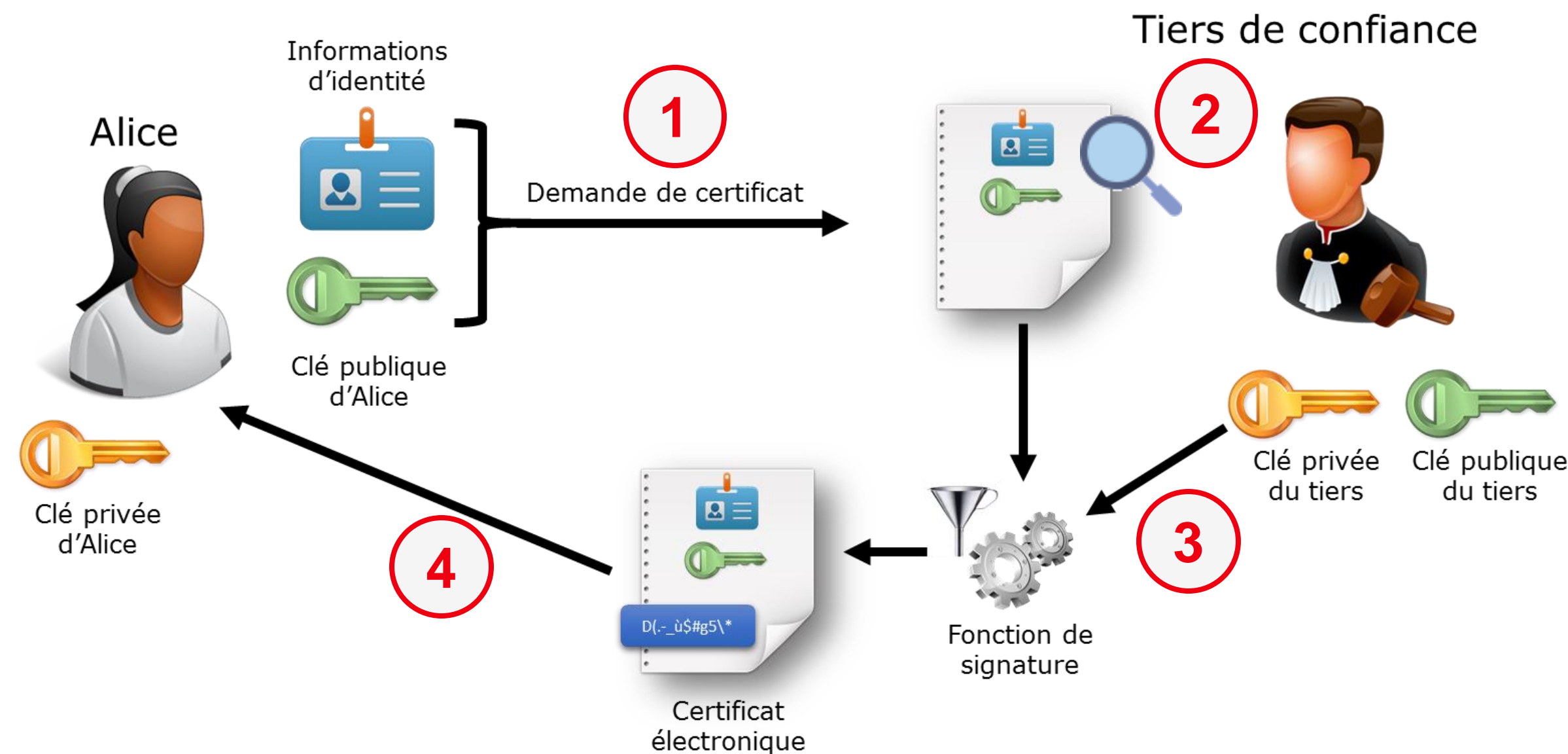
Risque

Comment Bob peut-il s'assurer que la clé publique qu'il a reçue est bien celle d'Alice et non pas celle d'un individu malveillant qui aurait intercepté des communications ?

→ En faisant appel à un tiers de confiance pour garantir l'authenticité de cette clé.

# Certificats électroniques

Les certificats électroniques sont des documents, sous forme électronique, attestant du lien entre une clé publique et l'identité de son propriétaire (personne physique ou service applicatif).



1. Alice transmet à un tiers de confiance sa clé publique et des informations d'identité au sein d'une demande de signature (*Certificate Signing Request* ou CSR).
2. Le Tiers contrôle la validité des informations transmises.
3. Il signe les informations avec sa clé privée et produit ainsi le certificat.
4. Il retourne le certificat à Alice.

Alice ne transmet plus sa clé publique, mais son certificat. Bob doit tout de même contrôler sa validité. Pour cela, il réalise une opération de vérification de la signature du certificat au moyen de la clé publique du tiers de confiance **qu'il connaît**.













# Certificats électroniques

X.509 est une norme spécifiant les formats pour les certificats à clé publique, les listes de révocation de certificat, les attributs de certificat et un algorithme de validation du chemin de certification. Elle a été créée en 1988 dans le cadre de la norme X.500. Elle repose sur un système hiérarchique d'autorités de certification.

Le certificat est composé de plusieurs champs de base, généralement accompagnés d'extensions. Celles-ci précisent les contextes d'utilisation du certificat et fournissent des moyens de valider plus précisément le certificat au sein de la chaîne de certification.

# Certificats électroniques










## Informations de base

 Version	V3	Numéro unique permettant d'identifier le certificat au sein de l'autorité de certification qui l'a émis.
 Serial number	00 ae a9 b7 4e cd a3 a2 02 62...	
 Signature algorithm	sha256RSA	Algorithmes employés par l'autorité de certification pour signer les informations du certificat.
 Signature hash algorithm	sha256	
 Issuer	Certigna Services CA, NTRFR-...	Autorité qui a délivré le certificat.
 Valid from	mercredi 5 octobre 2016 08:4...	Date à partir de laquelle le certificat est valide
 Valid to	samedi 5 octobre 2019 08:44:53	Date après laquelle le certificat n'est plus valide.
 Subject	S3142002, www.anssi.eu, NT...	Détenteur de la clé.
 Public key	RSA (2048 Bits)	Informations sur la clé publique du certificat (notamment identifiant de l'algorithme de clé utilisé et clé elle-même).
 Public key parameters	05 00	

# Certificats électroniques

## Extensions

Depuis la version 3 du standard X.509, des extensions peuvent être rajoutées aux certificats.

	Basic Constraints	Subject Type=End Entity, Pat...	Identifie si le détenteur du certificat est une autorité et dans quelle mesure ce certificat peut être utilisé pour en signer d'autres.
	Enhanced Key Usage	Server Authentication (1.3.6....	Usages possibles du certificat
	Subject Key Identifier	7a e4 12 0e 54 59 aa 51 c0 c2...	Informations pour valider le certificat au sein de la chaîne de Certification.
	Authority Key Identifier	KeyID=ac ec 86 8f 4b 37 1c b...	Possesseur de la paire de clés asymétriques à laquelle se réfère
	Subject Alternative Name	DNS Name=www.anssi.eu, DN...	le certificat (noms alternatifs).
	Authority Information Access	[1]Authority Info Access: Acc...	Lien vers la politique de certification qui s'applique au certificat
	Certificate Policies	[1]Certificate Policy:Policy Ide...	Informations servant à vérifier l'état de révocation du certificat
	CRL Distribution Points	[1]CRL Distribution Point: Distr...	Usages possibles du certificat
	Key Usage	Digital Signature, Key Encipher...	

Il existe un attribut permettant de marquer les extensions considérées critiques (c'est le cas du champ *Key Usage* de cet exemple). Si une application ne reconnaît pas une extension d'un certificat marquée comme critique, alors elle doit rejeter ce certificat.



# Infrastructure de Gestion de Clés (IGC)

Une IGC (ou PKI en anglais pour Public Key Infrastructure) se définit comme l'ensemble de composantes, fonctions et procédures dédiées à la gestion de clés cryptographiques et de leurs certificats utilisés par des services de **confiance**. Il s'agit d'une infrastructure technique permettant à une **Autorité de Certification** (AC) d'assurer la fourniture des prestations de gestion des certificats tout au long de leur cycle de vie. Une IGC peut présenter de plusieurs fonctions :

- **Autorité d'Enregistrement** (AE), qui vérifie et valide les informations d'identification du futur porteur d'un certificat (et éventuellement d'autres attributs) ;
- **Génération des certificats**, qui génère (création du format, signature électronique avec la clé privée de l'Autorité de Certification) les certificats à partir des informations transmises par l'autorité d'enregistrement et de la clé publique du porteur ;
- **Génération des éléments secrets du porteur**, génère les éléments secrets à destination du porteur, si l'AC a en charge une telle génération, et les prépare en vue de leur remise au porteur (par exemple, personnalisation de la carte à puce destinée au porteur, courrier sécurisé avec le code d'activation, etc.).

# Infrastructure de Gestion de Clés (IGC)

- **Remise au porteur**, qui sert à remettre au porteur au minimum son certificat ainsi que, le cas échéant, les autres éléments fournis par l'AC ;
- **Publication**, qui met à disposition des différentes parties concernées, les conditions générales, politiques et pratiques publiées par l'AC, les certificats d'AC et toute autre information pertinente destinée aux porteurs et/ou aux utilisateurs de certificats, hors informations d'état des certificats ;
- **Gestion des révocations**, qui traite les demandes de révocation (notamment identification et authentification du demandeur) et détermine les actions à mener. Les résultats des traitements sont diffusés via la fonction d'information sur l'état des certificats.
- **Information sur l'état des certificats**, qui fournit aux utilisateurs de certificats des informations sur l'état des certificats (révoqués, suspendus, etc.).

# SSL / TLS

# SSL / TLS

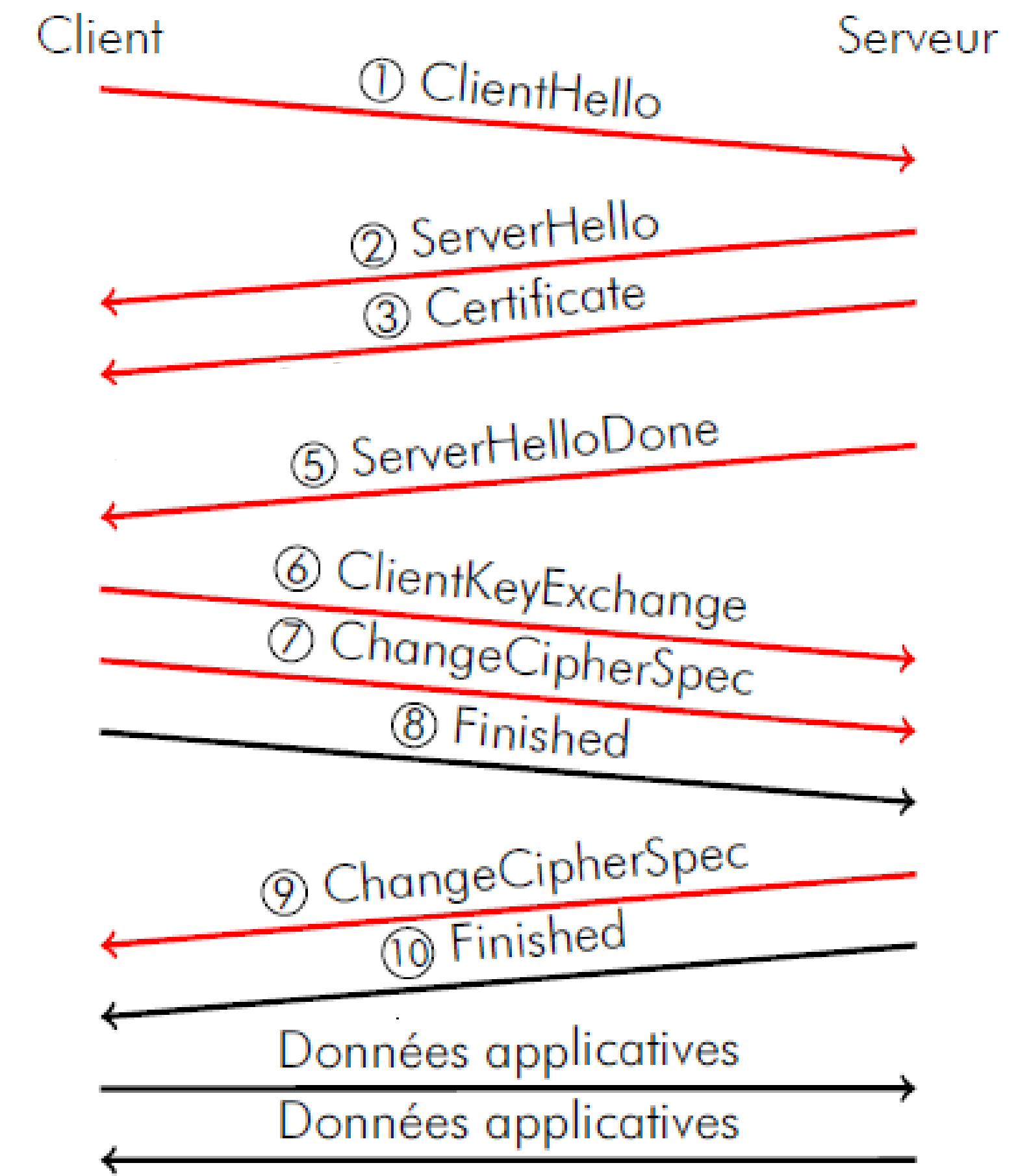
Le développement du protocole TLS a suivi plusieurs itérations depuis la conception du protocole SSL, désormais obsolète. Le processus de standardisation est à la charge de l'IETF. Les valeurs numériques des différents paramètres sont référencées par l'IANA. Les messages transmis par l'intermédiaire du protocole TLS sont appelés records. Ils sont généralement encapsulés dans des segments TCP, protocole chargé d'assurer les fonctionnalités de transport réseau telles que l'acquittement à la réception de données.

Par souci d'interopérabilité, les spécifications permettent aux deux parties impliquées de négocier la version du protocole qu'ils adopteront communément. Ce paramètre est établi au cours d'une phase *TLS handshake* qui précède la protection effective des échanges. De même, les spécifications autorisent l'utilisation de différentes combinaisons d'algorithmes cryptographiques.



# SSL / TLS

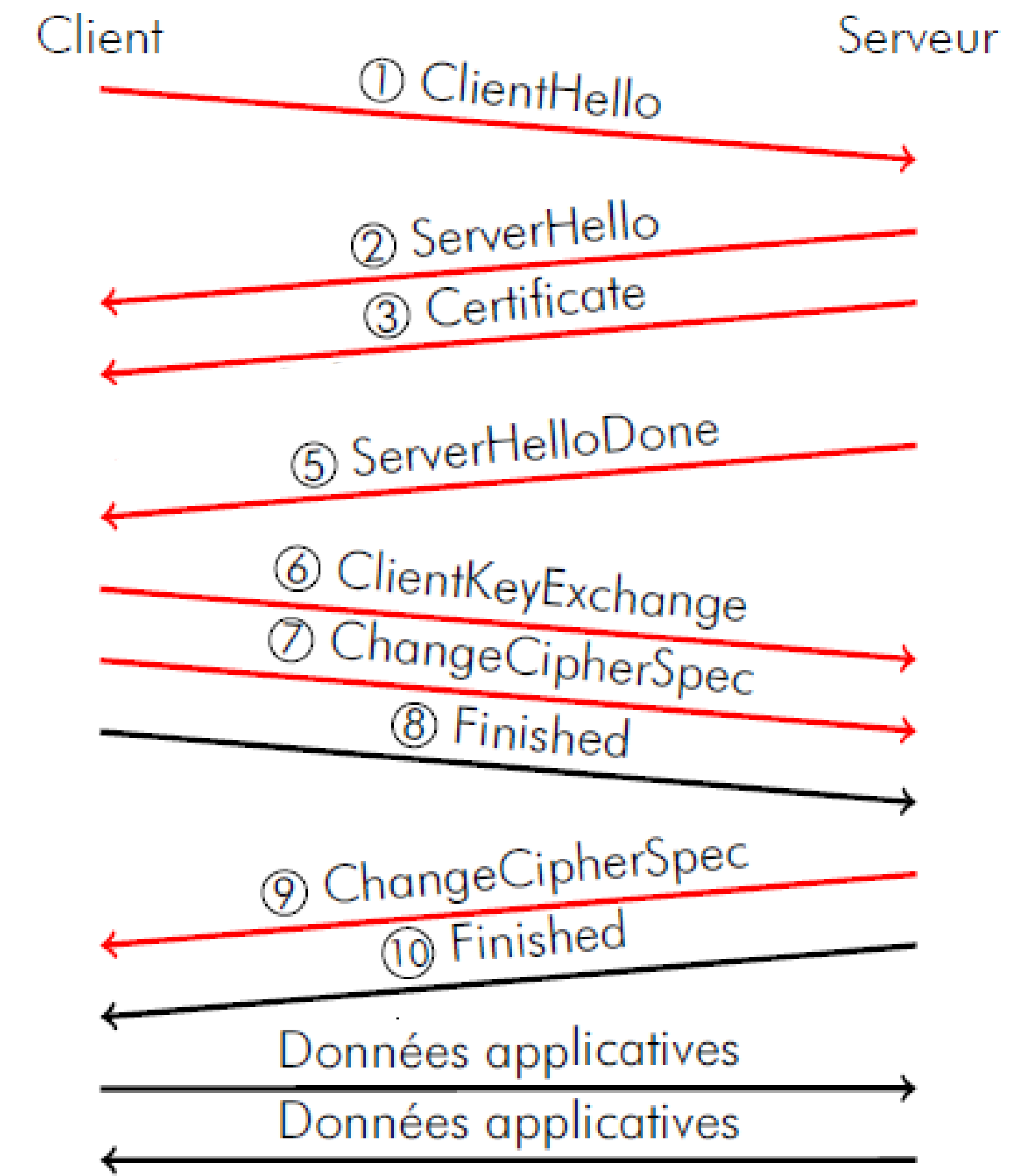
1. Le client initie une requête en envoyant un message de type *ClientHello*, contenant notamment les suites cryptographiques qu'il prend en charge ;
2. Le serveur répond par un *ServerHello* qui contient la suite retenue et une valeur aléatoire ;
3. Le serveur envoie un message *Certificate*, qui contient en particulier sa clé publique au sein d'un certificat numérique ;
5. Le serveur manifeste sa mise en attente avec un *ServerHelloDone* ;
6. Après vérification du certificat, le client choisit aussi une valeur aléatoire qu'il chiffre à l'aide de la clé publique du certificat puis transmet dans un *ClientKeyExchange*. Il constitue le *premaster secret* de son côté.
7. Le client signale l'adoption de la suite négociée avec un *ChangeCipherSpec* ;
8. Le client envoie un *Finished*, premier message protégé selon la suite cryptographique avec le secret issu de l'échange de clés éphémères précédent ;
9. Le serveur constitue *premaster secret* son côté et signale l'adoption de la même suite avec un *ChangeCipherSpec* ;
10. Le serveur envoie à son tour un *Finished*, son premier message sécurisé.



# SSL / TLS

À l'issue du handshake, le client et le serveur disposent d'un secret partagé, le ***premaster secret***, calculé grâce aux éléments contenus dans le *ServerKeyExchange* et le *ClientKeyExchange*.

Le ***premaster secret*** est dérivé de part et d'autre en un même *master secret* qui prend en compte les aléas contenus dans le *ClientHello* et le *ServerHello*. Enfin, le master secret est à son tour dérivé afin de générer les clés qui permettent de protéger les données applicatives en confidentialité et en intégrité, avant de les encapsuler dans des messages de type *application\_data*.





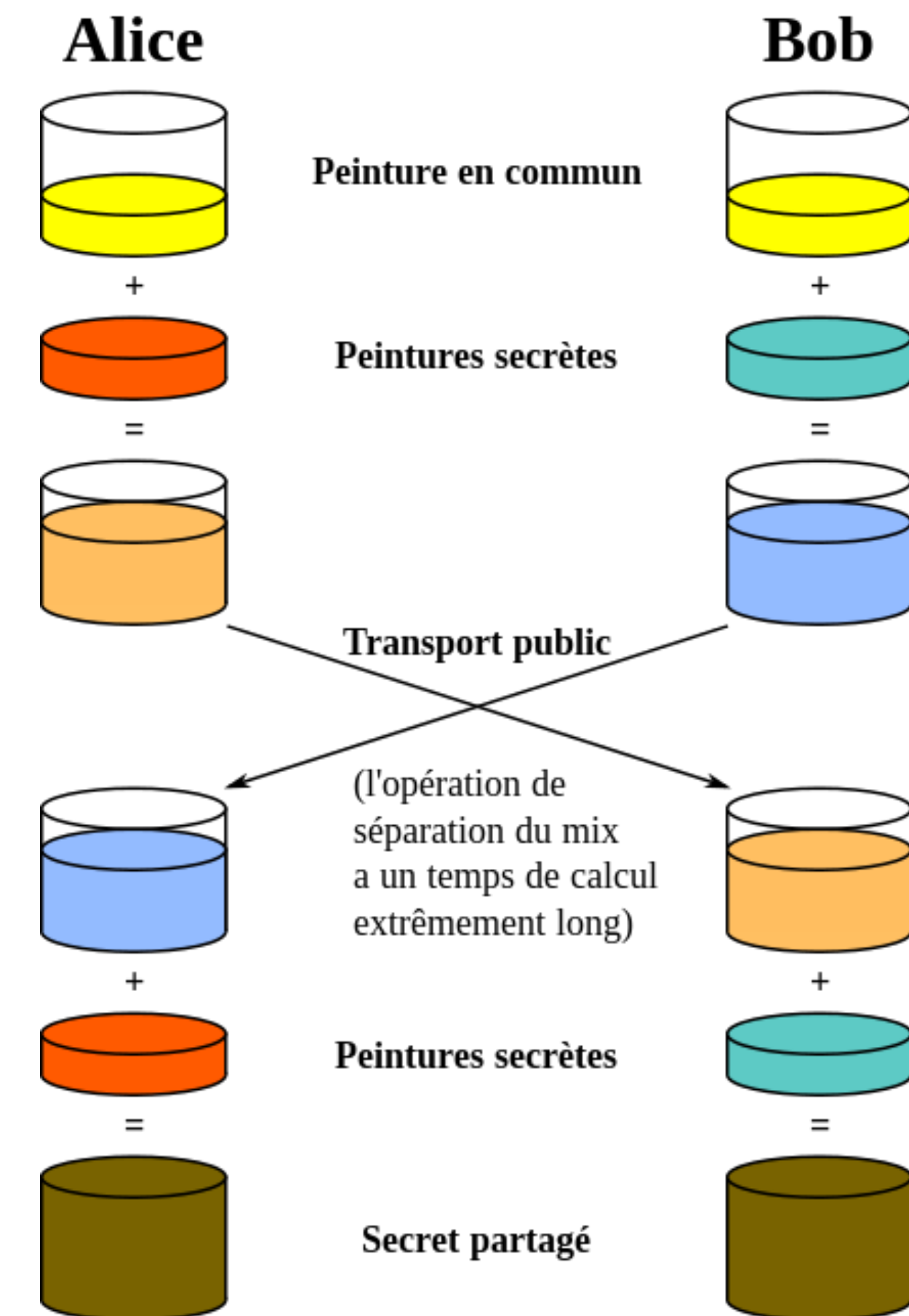
# SSL / TLS

Ce procédé est sûr à condition que seul le serveur puisse déchiffrer le premier secret (à l'aide de sa clef privée) envoyé par le client. Supposons maintenant qu'un attaquant enregistre tous les échanges entre le serveur et ses nombreux clients pendant plusieurs mois. **S'il est un jour en mesure de compromettre la clé privée du serveur, il pourra déchiffrer la communication *a posteriori*** (merci la NSA).

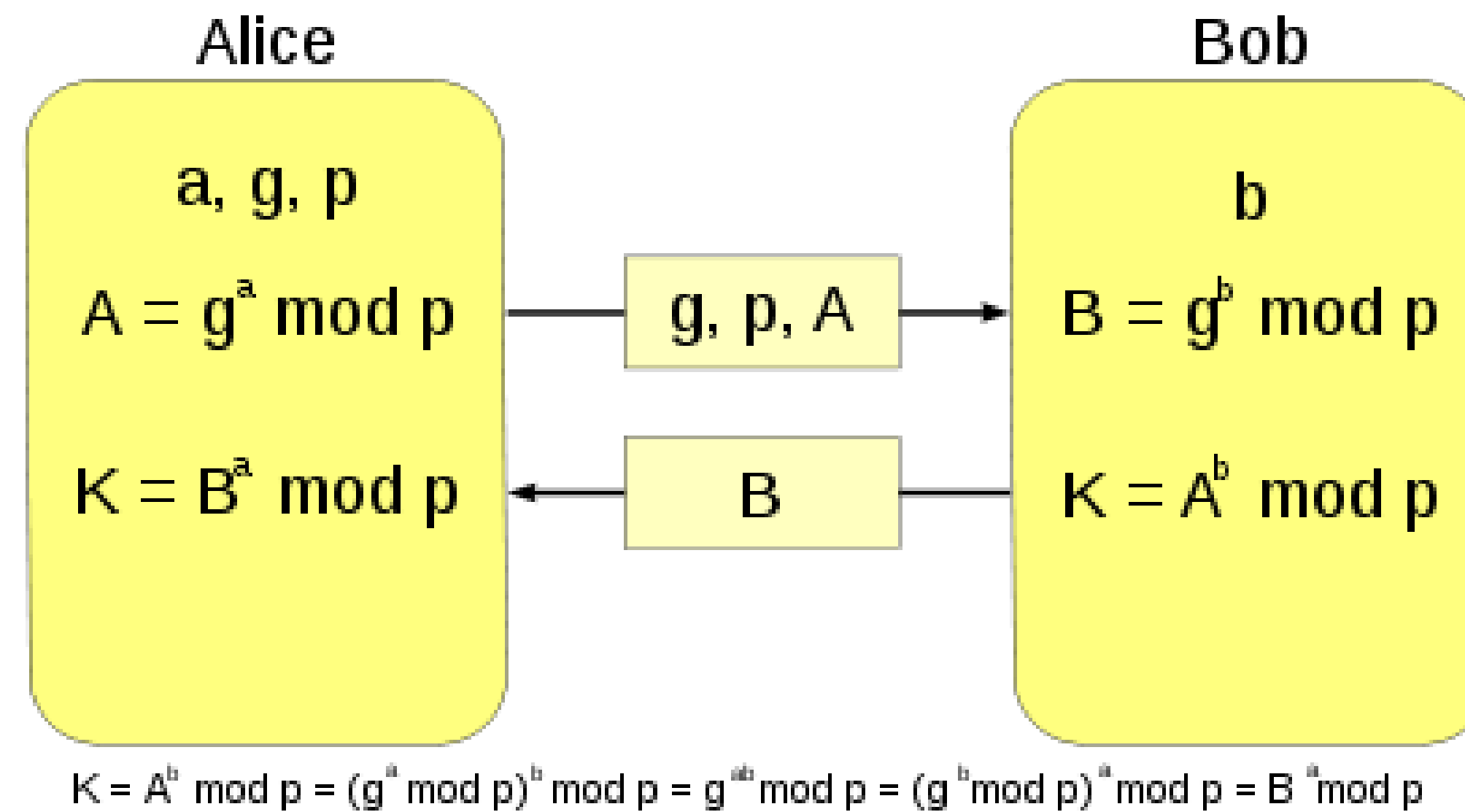


# PFS

Il est possible d'adopter une suite cryptographique qui assure la propriété de confidentialité persistante, ou **PFS** (*Perfect Forward Secrecy*). Celle-ci consiste à prévenir le déchiffrement de messages de sessions passées quand bien même la clé privée du serveur serait compromise, en négociant un secret éphémère à l'aide d'un échange **Diffie-Hellman**.



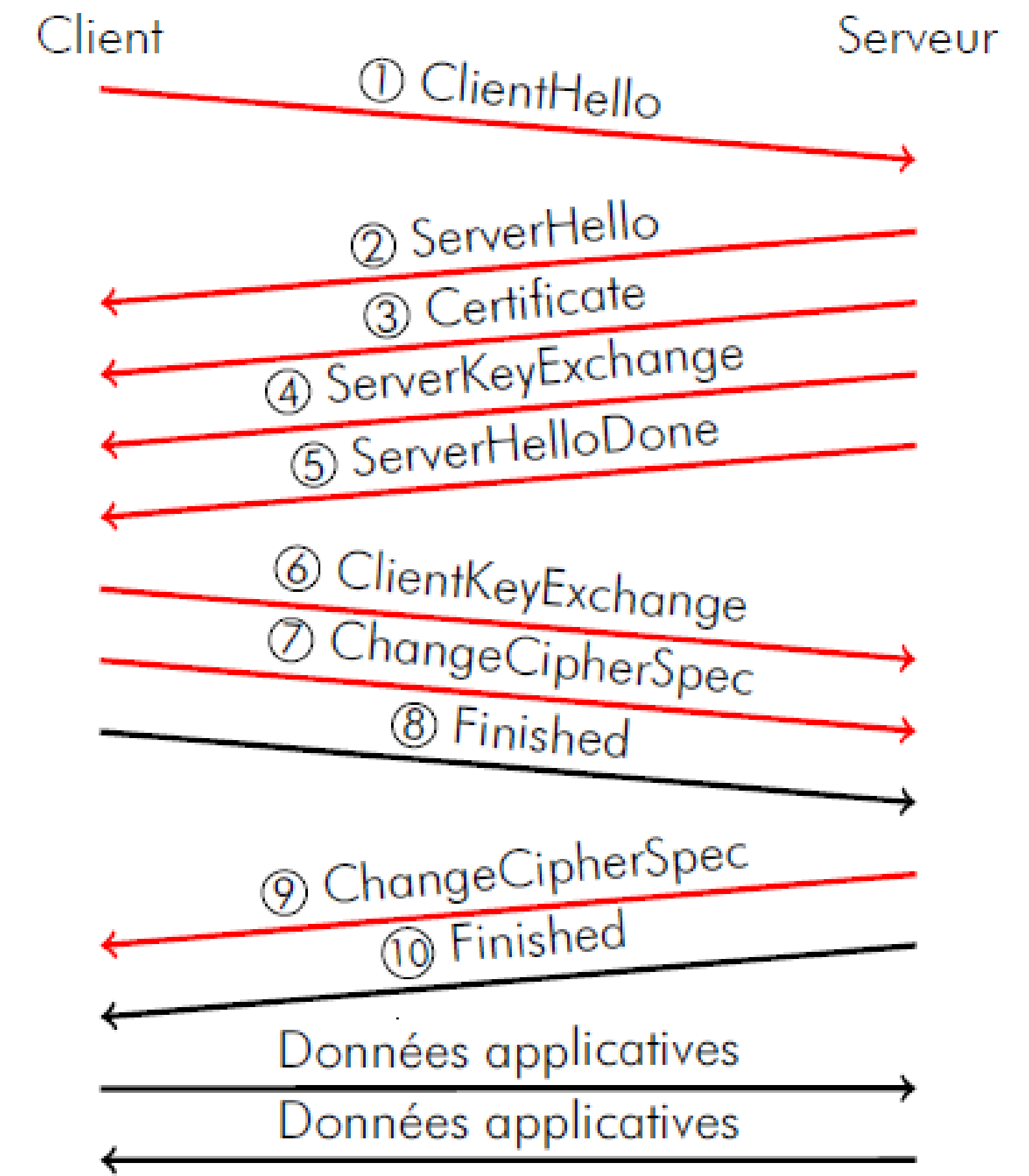
# DH



Le problème dit « du **logarithme discret** » est fondé sur la difficulté d'inverser l'opération d'exponentiation dans un groupe. C'est ce qui assure la sécurité des échanges via Diffie-Hellman.

# SSL / TLS avec PFS

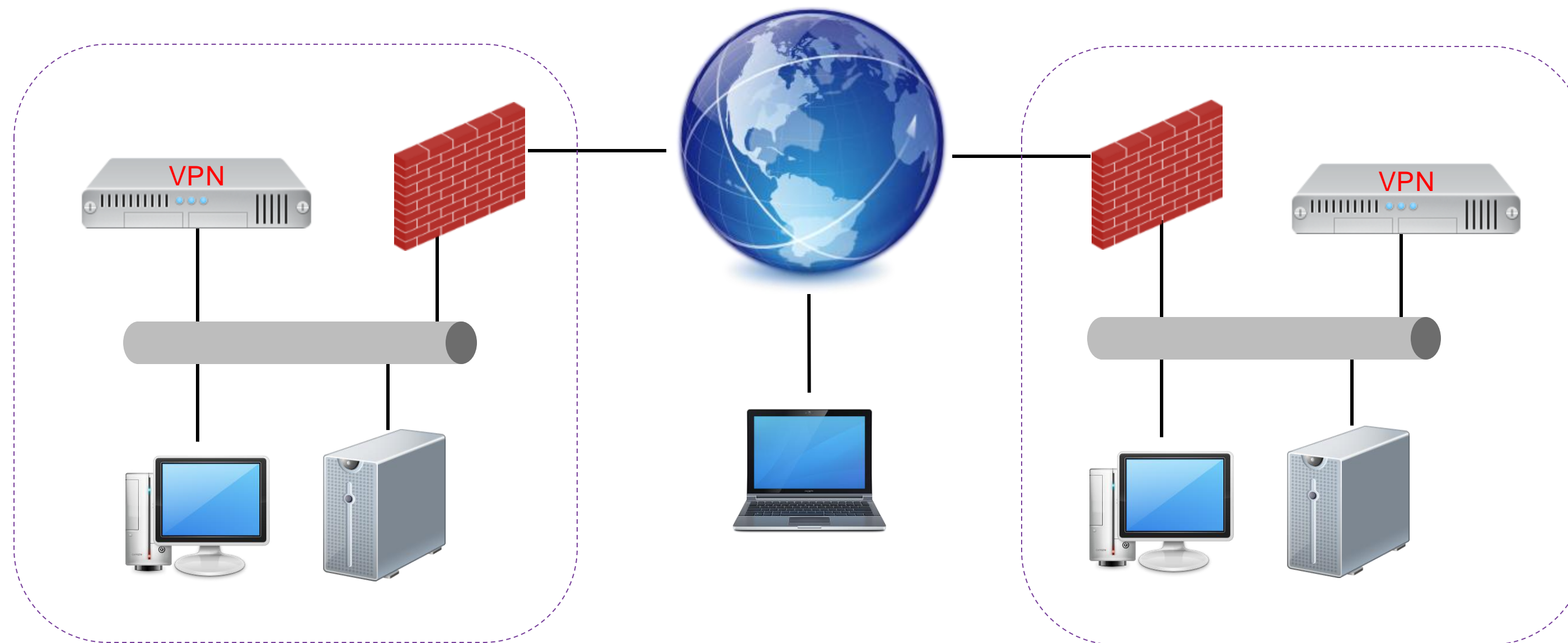
1. Le client initie une requête en envoyant un message de type ClientHello, contenant notamment les suites cryptographiques qu'il prend en charge ;
2. Le serveur répond par un ServerHello qui contient la suite retenue ;
3. Le serveur envoie un message Certificate, qui contient en particulier sa clé publique au sein d'un certificat numérique ;
4. **Le serveur transmet dans un ServerKeyExchange une valeur éphémère qu'il signe à l'aide de la clé privée associée à la clé publique précédente ;**
5. Le serveur manifeste sa mise en attente avec un ServerHelloDone ;
6. Après vérification du certificat et authentification de la valeur précédente, le client choisit à son tour une valeur éphémère qu'il chiffre à l'aide de la clé publique du certificat puis transmet dans un ClientKeyExchange ;
7. Le client signale l'adoption de la suite négociée avec un ChangeCipherSpec ;
8. Le client envoie un Finished, premier message protégé selon la suite cryptographique avec les secrets issus de l'échange de clés éphémères précédent ;
9. Le serveur signale l'adoption de la même suite avec un ChangeCipherSpec ;
10. Le serveur envoie à son tour un Finished, son premier message sécurisé.



# Les VPN

# VPN

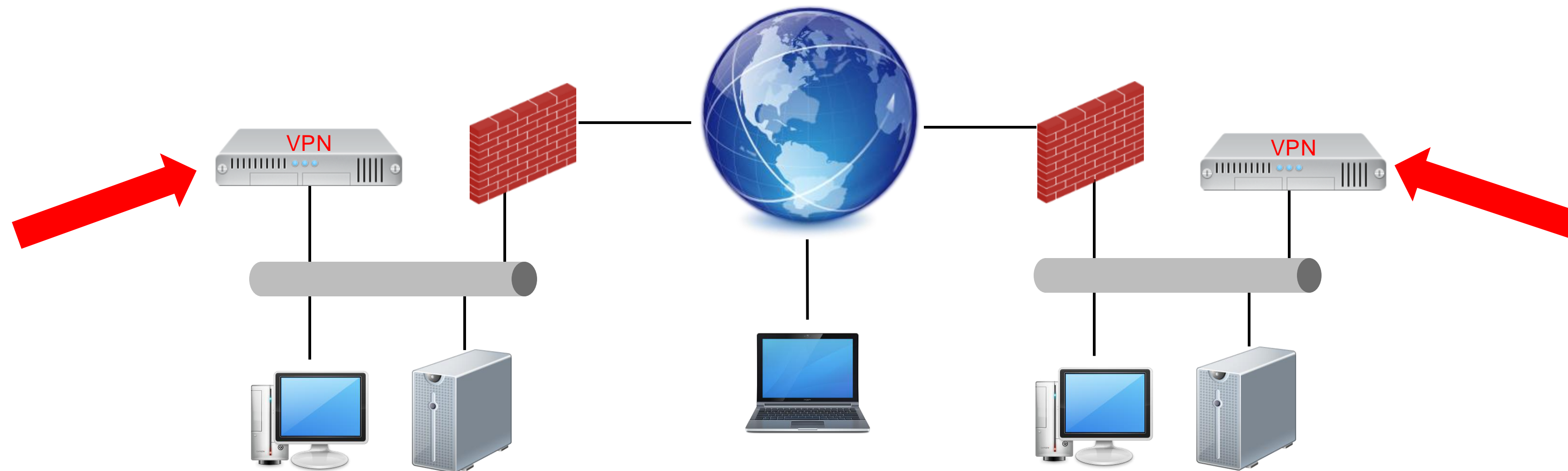
Un réseau privé virtuel (*Virtual Private Network* en anglais) est un concept visant à assurer une **interconnexion** entre des éléments constitutifs d'un ou plusieurs systèmes d'information, indépendamment du support logique les liant et des contraintes géographiques associées.





# VPN

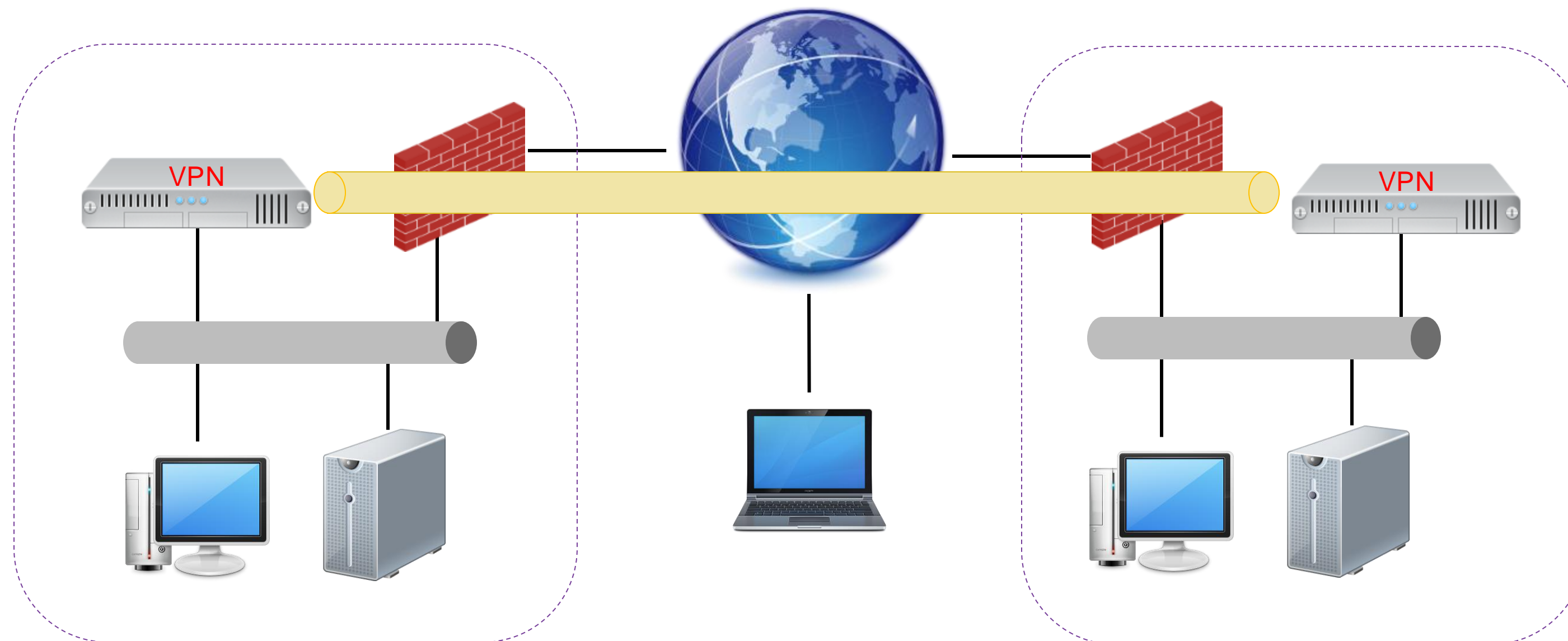
La mise en relation entre deux réseaux est effectuée par l'intermédiaire d'équipements servant de **passerelle** et chargés de gérer les communications entre eux. **La passerelle encapsule les flux de communication au sein d'un protocole de *tunneling* avant de les envoyer au réseau distant** et reçoit ceux de son homologue qu'elle retransmet sur son réseau local.



# VPN

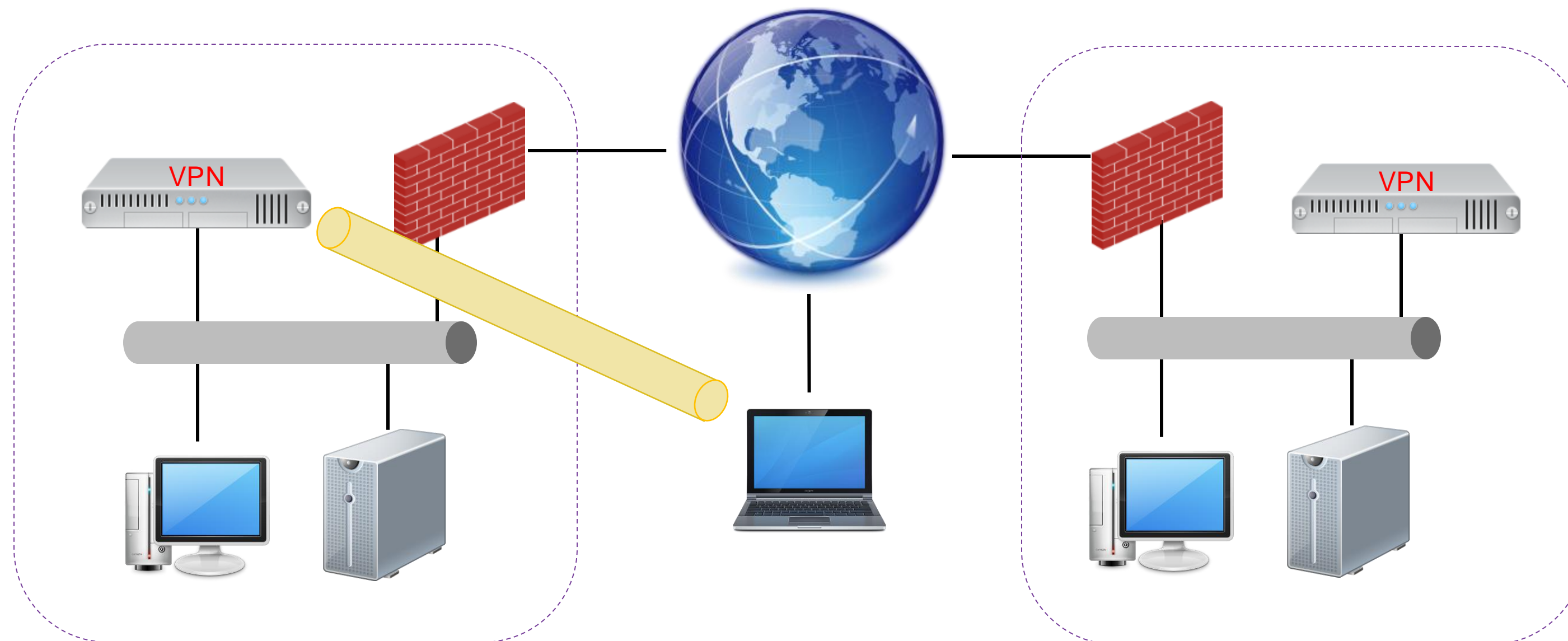
On distingue généralement deux types de liens VPN en fonction de la méthode de connexion et des accès aux ressources :

- Le VPN « **site à site** ». Il s'agit d'une connexion  $N \Leftrightarrow N$  ; souvent établie de manière permanente. Il sert à mettre en relation des réseaux entiers.



# VPN

- Le VPN « **nomade** », « d'accès distant » ou « client / serveur ». Cette fois-ci, la connexion établie est de type 1⇔N. Ce VPN est utilisé pour permettre à des machines individuelles (souvent des équipements nomades de se connecter à un Système d'Information. Le lien est établi à la demande de l'utilisateur et lui permet d'accéder à différentes ressources.



# VPN

Un réseau privé virtuel utilise un ou plusieurs protocoles parmi les suivants :

- **GRE**, souvent remplacé par L2TP, tous deux développés par Cisco.
- **PPTP** (Point-to-Point tunneling Protocol) est un protocole de niveau 2 développé par Microsoft, 3Com, Ascend, US Robotics et ECI Telematics.
- **L2TP** (Layer Two Tunneling Protocol) est l'aboutissement des travaux de l'IETF (RFC 39313) pour faire converger les fonctionnalités de PPTP et L2F. Il s'agit ainsi d'un protocole de niveau 2 s'appuyant sur PPP.
- **IPsec** est un protocole de niveau 3, issu des travaux de l'IETF, permettant de transporter des données chiffrées pour les réseaux IP.
- **SSL/TLS**, déjà utilisé pour sécuriser la navigation sur le web via HTTPS, permet également l'utilisation d'un navigateur Web comme client VPN. Ce protocole est notamment utilisé par OpenVPN.
- **SSH** permet, entre autres, d'envoyer des paquets depuis un ordinateur auquel on est connecté.



# IPSec

IPsec, de par ses subtilités, est souvent partiellement compris et peu maîtrisé. Les choix de configuration, y compris ceux par défaut, ne sont pas toujours judicieux et l'emploi d'IPsec peut alors offrir un niveau de sécurité plus faible que celui attendu.

Les services de sécurité fournis par IPsec reposent sur deux protocoles différents qui constituent le cœur de la technologie :

- **AH** : *Authentication Header*
- **ESP** : *Encapsulation Security Payload*

Ces deux protocoles peuvent être utilisés indépendamment ou, plus rarement, de manière combinée.



# AH / ESP

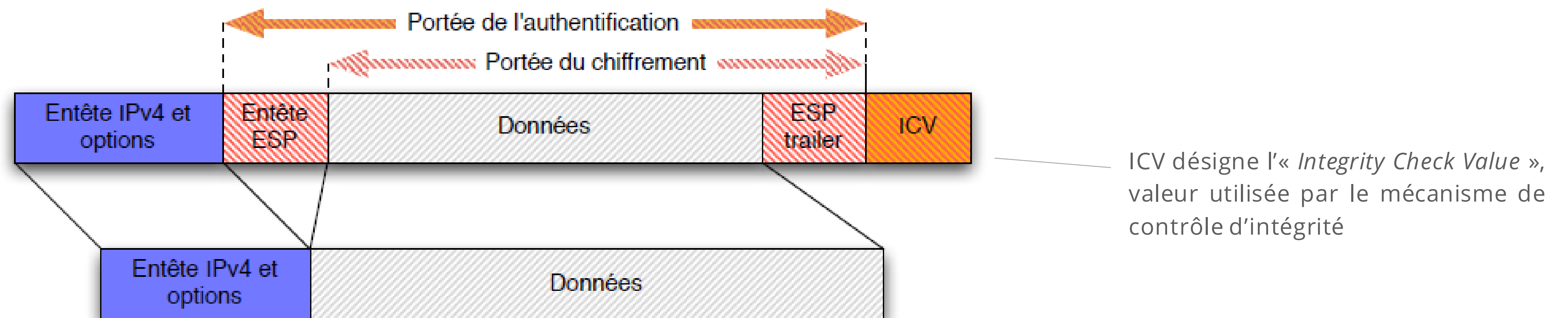
Le protocole **AH** permet d'assurer l'intégrité et, employé avec IKE, l'authentification des paquets IP. Il permet d'une part de s'assurer que les paquets échangés n'ont pas été altérés et d'autre part de garantir l'identité de l'expéditeur d'un paquet. Il garantit aussi une protection contre le rejeu. On notera **qu'AH ne protège pas la confidentialité des données échangées**. Le contrôle d'intégrité s'effectue sur l'ensemble des paquets IP y compris les en-têtes (ce qui le rend incompatible avec les mécanismes de traduction d'adresses).

Le protocole **ESP** permet quant à lui d'assurer la confidentialité, l'intégrité et, employé avec IKE, l'authentification des données échangées. Il garantit aussi une protection contre le rejeu. Il est possible d'utiliser uniquement les fonctions d'intégrité et d'authentification sans chiffrement (ce qui peut satisfaire la plupart des cas d'usage d'AH et justifie donc l'abandon de ce dernier).

# Modes transport et tunnel

Indépendamment du choix entre AH et ESP, il est possible d'utiliser IPsec dans deux modes distincts : le mode tunnel et le mode transport. Le mode tunnel rend le service attendu dans la majorité des cas.

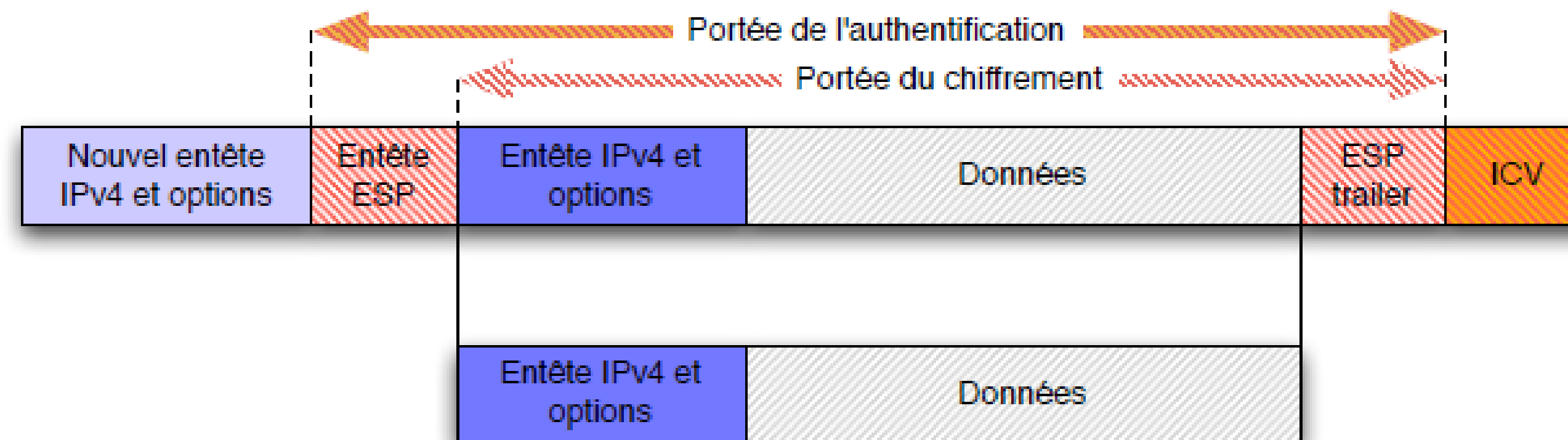
Dans le mode transport, les données associées à AH ou à ESP viennent se greffer sur le paquet IP initial (c'est à dire celui qu'on aurait envoyé en l'absence d'IPsec). Le paquet IP résultant contient un paquet AH ou ESP qui contient lui-même le contenu du paquet initial (un segment TCP par exemple).



*Utilisation d'ESP en mode transport*

# Modes transport et tunnel

Dans le mode tunnel en revanche, un nouveau paquet IP est généré pour contenir un paquet AH ou ESP qui contient lui-même le paquet IP initial sans modification. Dans ce mode, il y a donc en définitive deux en-têtes IP. L'en-tête externe sera effectivement utilisé pour le routage dès l'émission du paquet. L'en-tête interne, qui peut être chiffrée dans le cas où l'on utilise ESP avec le service de confidentialité, ne sera traitée que par le destinataire (du paquet externe). Elle sera ignorée par les équipements réseau situés entre l'émetteur et le destinataire.



*Utilisation d'ESP en mode tunnel*

# Security Policy

Le terme « Security Policy » désigne, dans le contexte IPsec, le choix pour un lien unidirectionnel donné :

- De l'utilisation obligatoire ou facultative ou de la non-utilisation d'IPsec ;
- De l'utilisation du mode tunnel ou transport ;
- De l'utilisation d'AH ou d'ESP.

L'ensemble des SP est regroupé dans une SPD : « Security Policy Database ». À l'image des règles de flux d'un pare-feu , les SP ont pour but de spécifier les flux que l'on veut autoriser et ceux que l'on veut interdire.

# Security Association

Pour chaque lien unidirectionnel, on désigne par *Security Association (SA)* les données de contexte telles que :

- Les hôtes source et destination ;
- Le mode (transport/tunnel) et les protocoles (AH/ESP) employés ;
- Les algorithmes cryptographiques employés ;
- Les clés associées à ces algorithmes.

Chaque SA est associée à une période de validité et à un nombre entier la désignant de manière univoque et appelé SPI (*Security Parameter Index*). Les en-têtes indiquent systématiquement le SPI associé à la SA utilisée. Les premiers éléments (hôtes aux extrémités, mode, protocole) sont conditionnés par les SP en vigueur : un système ne doit pas avoir de SA qui violent ses SP. Les paramètres cryptographiques peuvent être fixés manuellement ou négociés par le protocole IKE.

On définit la SAD comme étant la « Security Association Database » (base des SA).



# IKE

La négociation dynamique des algorithmes et clés d'une SA peuvent se faire grâce au protocole IKE, actuellement en version 2. Il se décompose en deux phases distinctes. Dans une première phase, un canal sécurisé (chiffré et authentifié) est créé entre les deux participants. Dans une deuxième phase, ce canal est utilisé pour négocier les divers paramètres de la SA.

L'authentification des participants à la première peut se faire soit au moyen d'un secret partagé (*Pre-Shared Key*) soit par utilisation d'un mécanisme de cryptographie asymétrique, et donc s'appuyer sur une IGC.

IKE permet aussi de négocier les SP. Dans la plupart des cas, tous les paramètres des SP sont connus à l'avance et cette négociation ne présente que peu d'intérêt. Ce mécanisme prend toutefois tout son sens pour les situations de mobilité. Dans ce cas, en effet, l'adresse IP du client nomade n'est pas connue a priori.

# IPSec et PFS

IPSec permet, comme TLS, d'utiliser la propriété de *Perfect Forward Secrecy*. C'est-à-dire la garantie qu'un attaquant ayant enregistré des échanges chiffrés à un instant donné et parvenant à obtenir les secrets cryptographiques à une date ultérieure ne puisse pas pour autant déchiffrer les enregistrements effectués. Cette propriété est obtenue (pour le cas d'IPsec) en employant un mécanisme d'échange de clé « Diffie-Hellman éphémère » ou sa variante sur courbe elliptique.

Il est possible avec certains équipements d'améliorer la granularité de cette propriété en utilisant un second échange de clé en phase 2 (plutôt que de dériver toutes les clés de celle négociée en phase 1), échange qui sera renouvelé plusieurs fois au cours de la durée de vie d'une SA.

# Le cas du WiFi

# Problématiques

L'utilisation du Wi-Fi, reposant sur la radio, pose de nouvelles problématiques de sécurité. Initialement on utilisait des câbles pour relier les équipements entre eux : a priori, seul le destinataire recevait les paquets transmis.

Avec les ondes radio, les choses sont différentes :

- Le paquet réseau est émis en radio ;
- Tous les équipements le reçoivent.

La problématique de la confidentialité des échanges est donc très prégnante.

# Sécurité

Il existe trois modes de sécurité :

- **WEP** (Wired Equivalent Privacy)
  - Premier algorithme utilisé ;
  - Plusieurs attaques existent contre ce protocole : il n'est pas sécurisé ;
  - Ironiquement rebaptisé « Weak Encryption Protocol » par certains.
- **WPA** (Wi-Fi Protected Access)
  - Protocole de transition entre WEP et WPA2 ;
  - Peu utilisé.
- **WPA2**
  - Standard actuel ;
  - Propose plusieurs modes de fonctionnement.



# Authentification

L'authentification entre la station et le point d'accès peut être effectuée selon plusieurs protocoles :

- Pre-Shared Key (PSK)
  - Fonctionne grâce à une clé pré-partagée ;
  - Sert de secret partagé entre les stations et le point d'accès ;
  - Une clé unique, diffusée à l'ensemble des stations ;
  - Des clés de session différentes.
- Extensible Authentication Protocol (EAP)
  - Protocole d'authentification générique ;
  - Plusieurs types d'authentifications prédéfinis (8) ;
  - Possibilité d'intégrer de nouveaux types d'authentification (extensible).

**Ces modes n'offrent pas tous le même niveau de sécurité**

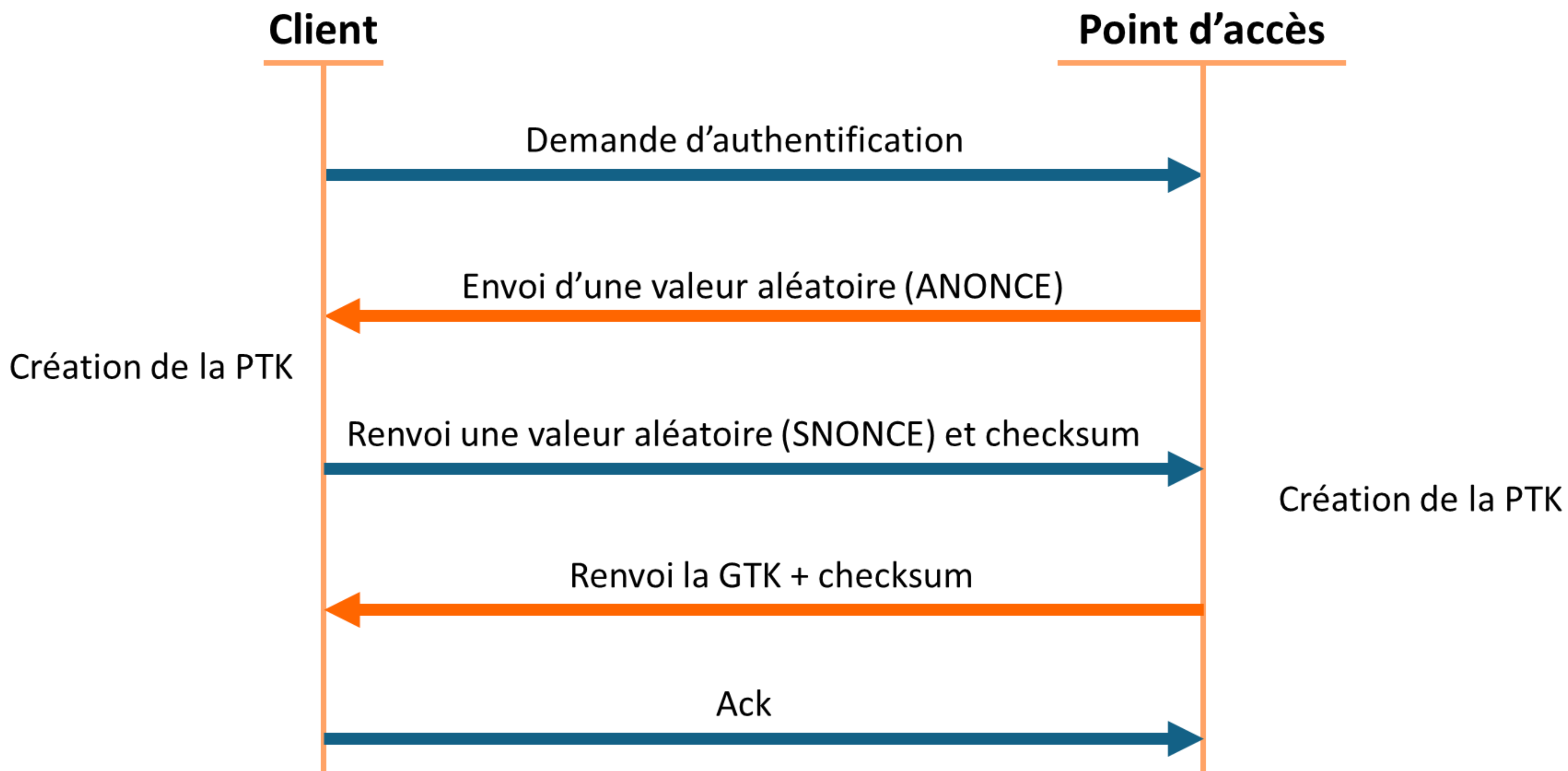
# WPA PSK

L'authentification entre la station et le point d'accès peut être effectuée selon plusieurs protocoles :

- Pre-Shared Key (PSK)
  - Fonctionne grâce à une clé pré-partagée ;
  - Sert de secret partagé entre les stations et le point d'accès ;
  - Une clé unique, diffusée à l'ensemble des stations ;
  - Des clés de session différentes.
- Extensible Authentication Protocol (EAP)
  - Protocole d'authentification générique ;
  - Plusieurs types d'authentifications prédéfinis (8) ;
  - Possibilité d'intégrer de nouveaux types d'authentification (extensible).

**Ces modes n'offrent pas tous le même niveau de sécurité**

# WPA PSK



# WPA PSK

Mais ça pose des problèmes parfois...



## Key Reinstallation Attacks

Breaking WPA2 by forcing nonce reuse

*Discovered by [Mathy Vanhoef](#) of [imec-DistriNet](#), KU Leuven*





Merci de votre attention.

**Gwenn Feunteun**

*gwenn@acceis.fr*

