# Vérifier la sécurité de nos communications

Pascal Lafourcade

# Vérifier la sécurité de nos communications

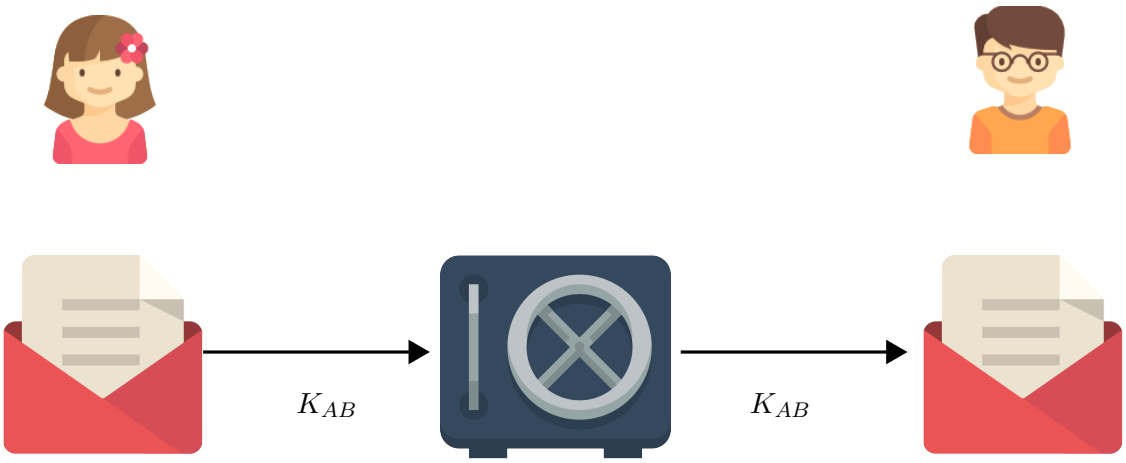
Pascal Lafourcade,
LIMOS, University Clermont Auvergne

April 6, 2018

De plus en plus de communications sont dématérialisées de par l'essor des moyens de communications numériques : emails, messagerie instantanées, commerce en ligne etc. Afin d'assurer la sécurité de ces échanges de nombreux protocoles utilisant la cryptographie ont vu le jour. Il existe deux familles d'algorithmes de chiffrement : les algorithmes symétriques comme le chiffrement de-Vigenère [4] ou AES [7] et les algorithmes assymétriques aussi appelés algorithmes à clef publique comme RSA inventé [10].

### Chiffrement symétrique.

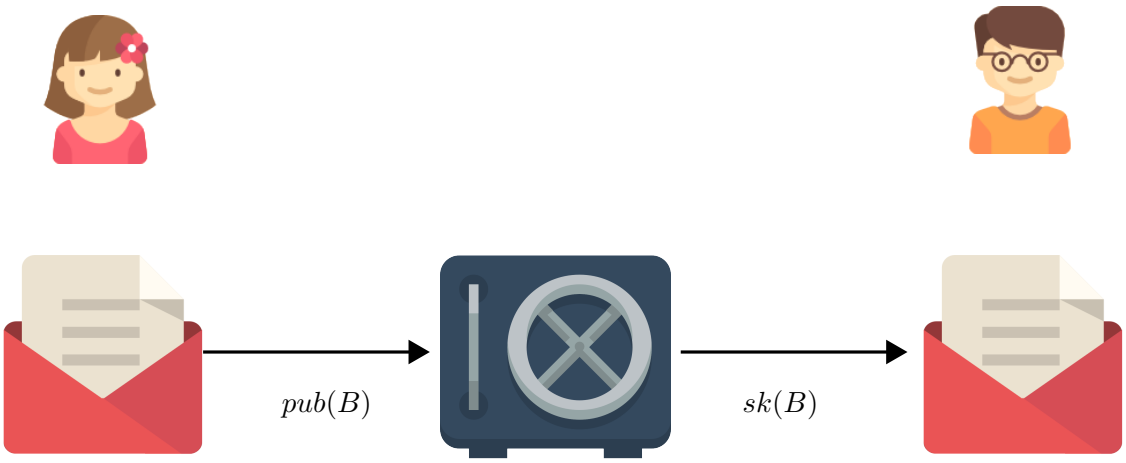
Dans un chiffrement symétrique, deux personnes souhaitant communiquer de manière chiffrée, ici Alice et Bob, possèdent la même clef symétrique $K_{AB}$. Ainsi, comme expliqué sur le dessin ci-dessou, Alice chiffre avec la clef $K_{AB}$ son message puis l'envoie à Bob. Une fois le message chiffré reçu, Bob peut le lire en le déchiffrant en utilisant la même clef $K_{AB}$. Ces chiffrements sont dits symétriqus car la même clef est utilisée pour chiffrer et déchiffrer.




### Chiffrement assymétrique.

Les chiffrements assymétriques fonctionnent différemment. Chaque participant posséde une clef publique et une clef secrète, pour Alice nous noterons $pub(A)$ sa clef publique et $sk(A)$ la clef secréte associée. Tous les participants connaissent les clefs publiques de tous les autres car elles sont publiées dans des annuaires sur Internet. Chaque participant a interêt de publié sa clef publique

car elle sert à générer des messages chiffrés pour lui. Par exemple, si Alice veut envoyer un message chiffré à Bob, il lui suffit de trouver la clef publique de Bob $pub(B)$ sur Internet. Puis de chiffrer le message avec celle-ci et de l'envoyer à Bob. Une fois le message reçu, Bob utilise la clef secréte $sk(B)$ pour déchiffrer le message.




### Le problème de l'échange de clef.

La cryptographie à clef publique permet à une personne d'envoyer une information chiffrée sans aucun échange au préalable. Malheureusement toutes les communications ne peuvent pas s'effectuer avec de tels chiffrement car par exemple il faut avec un ordinateur de bureau 30 secondes pour chiffrer un message de 10 Mo avec RSA et $0,1$ seconde avec AES, soit 100 fois moins.

Afin d'avoir le meilleur des deux mondes, nous allons utiliser la cryptographie à clef publique pour échanger une clef symmétrique entre Alice et Bob. Ainsi ils pourront communiquer en toute sécurité en utilisant cette nouvelle clef. Les protocoles cryptographiques permettant cela sont appelés protocoles déchange de clef.

### Attaques.

Pour attaquer un protocole cryptographique, il existe deux approches possibles : La première consiste à essayer d'obtenir de l'infromation sur les messages chiffrés échangés, par exemple en les déchiffrant ou en cherchant à obtenir la clef utilisée. Ces attaques sont appelées cryptanalyses et portent sur la sécurité d'algorithme cryptographique employé [6]. La seconde approche suppose que la méthode de chiffrement est incassable, c'est à dire que le seul moyen d'ouvrir un message chiffré c'est de connaître la clef secrète. Utiliser un algorithme de chiffrement incassable est un pré-requis indispensable pour assurer la sécurité mais n'est pas suffisant, en combinant plusieurs échanges de message un intrus peut obtenir de l'information si le protocole est mal conçu. Ce genre d'attaque est appelée attaque *logique*. Pour illustrer ce type d'attaque, nous présentons le plus célèbre de protocole d'échange de clef : le protocole de Needham-Schroeder. Ce protocole est célebre car 17 ans après sa publication une attaque fut découverte. Elle fit prendre conscience qu'un protocole dont toutes les informations transmises sont cryptées par un chiffrement inviolable et utilisant un canal de communication accessible à tous n'est pas nécessairement un échange de messages sûr. De

part la conception même du protocole, des informations confidentielles peuvent être découvertes par un intrus.

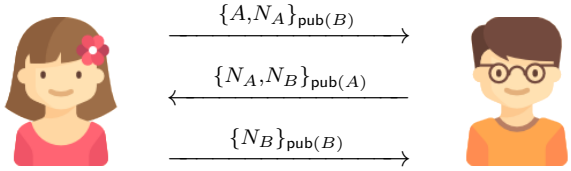
### Le protocole de Needham-Schroeder.

En 1978, R. Needham et M. Schroeder [9] proposérent un protocole d'échange de messages entre deux participants, Alice et Bob, notés respectivement par $A$ et $B$. Ces deux participants veulent échanger en toute sécurité une nouvelle clef symétrique $N_B$. Le protocole utilise un algorithme de chiffrement asymétrique. Un message $m$ chiffré par la clef publique d'Alice $pub(A)$ est noté $\{m\}_{pub(A)}$. Le protocole est fonctionne comme suit :

1. Alice envoie à Bob un message chiffré par la clef publique de Bob, $pub(B)$. Il contient son identité $A$ et un nombre aléatoire $N_A$ "fraîchement" choisi par Alice, aussi appelé *nonce*.

2. Une fois ce message reçu, Bob choisit une nouvelle clef symétrique notée $N_B$ et répond à Alice. Il envoie le nonce $N_A$ précédemment reçu, ainsi que $N_B$ dans un message chiffré par la clef publique d'Alice, $pub(A)$.

3. Alice confirme à Bob qu'elle a bien reçu le message, en lui renvoyant le $N_B$ chiffré par $pub(B)$, la clef publique de Bob.




Une fois le protocole terminé, Alice est convaincue d'avoir effectué une session du protocole avec Bob, car il lui a renvoyé son nonce $N_A$. De même, Bob pense avoir communiqué avec Alice. Ils partagent alors une information commune $N_B$ qui sera la clef publique utilisée dans leurs communications futures. Ils pensent également qu'ils sont les seuls à connaître $N_B$.

### L'attaque de l'homme au centre de Gavin Lowe.

L'attaque consiste est l'exécution de deux sessions du protocole pour permettre à l'intrus d'apprendre la clef $N_B$ supposée secrète. G. Lowe a trouvé, 17 ans après la publication du protocole, une attaque en utilisant un outil automatique de vérification de protocoles cryptographiques [8]. Cette attaque permet à un intrus, appelé Charlie et identifié par $C$, d'obtenir le nonce $N_B$ échangé par Alice et Bob.

Cette attaque est aussi connue sous le nom de "man in the middle", car Charlie est placé entre Alice et Bob. Il joue donc deux sessions en parallèle du protocole, une avec Alice et une avec Bob. Plus précisémment l'attaque fonctionne comme suit.

Alice commence une session du protocole avec Charlie. L'intrus déchiffre ce message et récupère l'identité d'Alice et son nonce $N_A$. Il peut ensuite commencer une autre session avec Bob et se faire passer pour Alice auprès de Bob en lui envoyant un message contenant l'identité d'Alice et le nonce $N_A$. D'après le message qu'il a reu, Bob pense communiquer avec Alice. Il répond donc à Alice, comme l'indique le protocole, avec le chiffré avec la clef publique d'Alice de $N_A$ et de la clef

symétrique $N_B$ qu'il vient de générer. Charlie ne peut pas ouvrir ce message chiffré pour Alice, il le transfére donc à Alice. Après avoir déchiffré le message reu, Alice renvoie à Charlie les nonces $N_A$ et $N_B$ cryptés par la clef publique $pub(C)$, comme cela est spécifié dans le protocole. Charlie conclut alors le protocole initié avec Bob en envoyant $N_B$ chiffré par la clef publique de Bob qui pense parler en toute sécurité à Alice alors qu'il communique en fait avec Charlie.


Ce protocole sert à échanger une clef symétrique $N_B$ entre Alice et Bob. L'intrus connat maintenant cette clef, il peut alors continuer sans aucun problème à écouter les messages envoyés par Bob à Alice, les déchiffrer et mme, se faire passer pour Alice. En effet, il peut répondre à Bob en chiffrant ses réponses grâce à la clef symétrique $N_B$.

### Correction.

Afin de corriger cette attaque il suffit que Bob ajoute son nom dans sa première réponse à Alice. Le message $\{N_A, N_B\}_{pub(A)}$ devient $\{B, N_A, N_B\}_{pub(A)}$ Ainsi l'attaque ne fonctionne plus car Alice s'aperçoit qu'elle discute avec deux personnes à la fois et donc ne continue pas le protocole; ce qui empêche l'intrus d'apprendre le secret.

### Vérification formelle pour les protocoles cryptographiques

La sécurité des protocoles de communication utilise des fonctions mathématiques pour chiffrer les messages. Bien que nécessaires ces techniques n'assurent pas forcémentle secret des communications, comme le montre l'attaque sur le protocole de Needham Schoreder. Construire de tels protocoles de manière sécuritaire n'est pas une tâche facile et dépasse largement les capacités humains. Sur un protocole de communication comme le protoocle de Needham-Shcoreder qui n'a que 3 messages, il a fallu 17 ans pour découvrir une attaque. La découverte de cette attaque fut le point de départ de la vérification formelle pour les protocoles cryptographiques. Depuis, les chercheurs ont développé avec succés des techniques de vérification automatique de ces protocole, afin de garantir l'absence de faille de sécurité pour un protocole donné face à un intrus ou bien de découvrir des failles sur des prtocooles pour pouvoir les corriger.

Pour vérifier qu'un protocole cryptographique est sûr il faut modéliser plusieurs choses : le protocole, la propriété de sécurité et l'attaquant contre lequel le protocole doit être sûr. Chacune de ces étapes nécessite de trouver le bon niveau d'abstraction, c'est-à-dire le bon compromis entre abstraction et efficacité de la vérification. Il existe de nombreux outils de vérification [1, 3, 2, 5], qui permettent aujourd'hui de vérifier automatiquement le secret et l'authentification.

Il reste encore de nombreuses techniques à développer pour pouvoir vérifier automatiquement les nouvelles propriétés de sécurité des protocoles des protocoles à venir.

## References


[1] Alessandro Armando, David Basin, Yohan Boichut, Yannick Chevalier, Luca Compagna, Jorge Cuellar, Paul Hankes Drielsma, Pierre-Cyrille Heám, Olga Kouchnarenko, Jacopo Mantovani, Sebastian Mödersheim, David von Oheimb, Michael Rusinowitch, Judson Santiago, Mathieu Turuani, Luca Viganò, and Laurent Vigneron. The avispa tool for the automated validation of internet security protocols and applications. In *Proceedings of CAV'2005*, LNCS 3576, pages 281–285. Springer-Verlag, 2005.

[2] B. Blanchet. An efficient cryptographic protocol verifier based on prolog rules. In *Proc. CSFW'01*, pages 82–96. IEEE Comp. Soc. Press, 2001.

[3] Bruno Blanchet. Automatic verification of correspondences for security protocols. *Journal of Computer Security*, 2009.

[4] Franois Cayre. Cryptographie, du chiffre et des lettres. `https://interstices.info/jcms/c_30225/nombres-premiers-et-cryptologie-l-algorithme-rsa`.

[5] C.J.F. Cremers. The Scyther Tool: Verification, falsification, and analysis of security protocols. In *Computer Aided Verification, 20th International Conference, CAV*, volume 5123/2008 of *LNCS*, pages 414–418. Springer, 2008.

[6] Mathieu Cunche. à lattaque des codes secrets. `https://interstices.info/jcms/i_53837/a-l-attaque-des-codes-secrets`.

[7] Joan Daemen and Vincent Rijmen. *The Design of Rijndael*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2002.

[8] G. Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In *Proc. 2nd International Workshop on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'96)*, volume 1055 of *LNCS*, pages 147–166, Berlin, Germany, 1996. Springer-Verlag.

[9] R. Needham and M. Schroeder. Using encryption for authentification in large networks of computers. *Communications of the ACM*, 21(12):993–999, 1978.

[10] Jonathan Touboul. Nombres premiers et cryptologie : lalgorithme rsa. `https://interstices.info/jcms/c_43248/cryptographie-du-chiffre-et-des-lettres`.