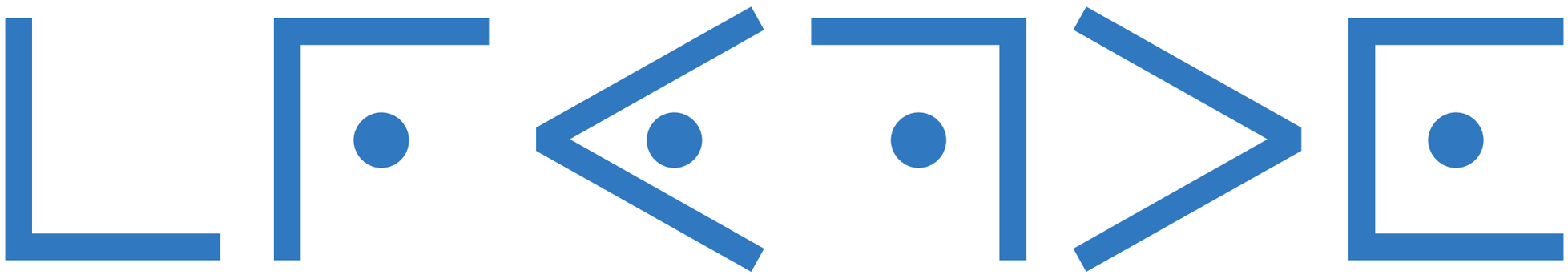

cryptographie à clé publique I

Nicolas Ollinger

M1 informatique — 2025/2026

Au-delà du chiffrement symétrique

L’établissement d’un canal sûr à l’aide du chiffrement symétrique nécessite un **secret partagé** a priori entre les deux parties.

Cette contrainte pose des difficultés :

- dans un environnement fermé avec beaucoup d’utilisateurs ;
- dans un environnement ouvert où des interlocuteurs ne se sont jamais rencontrés auparavant.

Protocole de distribution de clés

Distribution de clés centralisée

Utiliser une **autorité centrale**, le centre de distribution de clés (KDC), pour partager les clés de session entre les paires d’utilisateurs.

Tous les utilisateurs doivent faire confiance au KDC qui est chargé de la gestion des clés.

Un utilisateur n’a besoin que d’un **secret partagé** avec le KDC. Le KDC peut gérer les droits à communiquer entre les paires.

Protocole de sécurité

Ensemble de règles régissant le comportement d'individus pour répondre aux besoins d'une application.

Quelques notations :

$M, M', \ldots$	des messages
$K, K_A, \ldots$	des clés
$N, N', \ldots$	des nombres à usage unique
$A, B, \ldots$	des agents
$M.M'$	concaténation de M et M'
$\{M\}_K$	chiffrement de M avec la clé K
$A \rightarrow B : M$	A envoie le message M à B

Using Encryption for Authentication in Large Networks of Computers

Roger M. Needham and
Michael D. Schroeder
Xerox Palo Alto Research Center

Use of encryption to achieve authenticated communication in computer networks is discussed. Example protocols are presented for the establishment of authenticated connections, for the management of authenticated mail, and for signature verification and document integrity guarantee. Both conventional and public-key encryption algorithms are considered as the basis for protocols.

Key Words and Phrases: encryption, security, authentication, networks, protocols, public-key cryptosystems, data encryption standard
CR Categories: 3.81, 4.31, 4.35

itative descriptions of the connected computers, of the purposes for which they are used, or of the individuals who use them. We present protocols for decentralized authentication in such a network that are integrated with the allied subject of naming. There is minimal reliance on network-wide services; in particular there is no reliance on a single network clock or a single network name management authority.

Three functions are discussed:

(1) Establishment of authenticated interactive communication between two principals on different machines. By interactive communication we mean a series of messages in either direction, typically each in response to a previous one.

(2) Authenticated one-way communication, such as is found in mail systems, where it is impossible to require protocol exchanges between the sender and the recipient while sending an item, since there can be no guarantee that sender and recipient are simultaneously available.

(3) Signed communication, in which the origin of a communication and the integrity of the content can be authenticated to a third party.

Secure communication in physically vulnerable networks depends upon encryption of material passed between machines. We assume that it is feasible for each computer in the network to encrypt and decrypt material efficiently with arbitrary keys, and that these keys are not readily discoverable by exhaustive search or cryptanalysis. We consider both conventional encryption algorithms and public-key encryption algorithms as a basis

Protocole de Needham-Schroeder

$$A \rightarrow S : A \,.\, B \,.\, N_{AS}$$

$$S \rightarrow A : \left\{ N_{AS} \,.\, B \,.\, K_{AB} \,.\, \left\{ K_{AB} \,.\, A \right\}_{K_{BS}} \right\}_{K_{AS}}$$

$$A \rightarrow B : \left\{ K_{AB} \,.\, A \right\}_{K_{BS}}$$

$$B \rightarrow A : \left\{ N_{AB} \right\}_{K_{AB}}$$

$$A \rightarrow B : \left\{ N_{AB} - 1 \right\}_{K_{AB}}$$

Technical Note
Operating Systems

R. Stockton Gaines*
Editor

Timestamps in Key Distribution Protocols

Dorothy E. Denning and Giovanni Maria Sacco
Purdue University

The distribution of keys in a computer network using single key or public key encryption is discussed. We consider the possibility that communication keys may be compromised, and show that key distribution protocols with timestamps prevent replays of compromised keys. The timestamps have the additional benefit of replacing a two-step handshake.

Key Words and Phrases: encryption, encryption keys, key distribution, communications, security, timestamps.
CR Categories: 3.81, 4.39

I. Introduction

Secure communication between two users on a computer network is possible using either single key (conventional) encryption or public key encryption. In both

protocol is secure (i.e., can be used to establish a secure channel).

We will show that the protocol is not secure when communication keys are compromised, and propose a solution using timestamps. Although the likelihood of such a compromise may be small, the timestamps are useful for another reason: they can replace a two-step handshake designed to prevent replays of (noncompromised) keys.

We also show that timestamps can replace the handshake in the Needham and Schroeder protocol for public key systems. Here the AS is responsible for distributing users' public keys; it does not require access to their private keys. Because there are no secret communication keys, their compromise is not an issue. However, timestamps are valuable in public key systems to ensure the integrity of keys.

Public key systems also provide an alternate method of exchanging communication keys for single-key data encryption. As before, timestamps protect against replays of previously compromised communication keys.

No protocol is secure if users' private keys are compromised. We conclude with a brief discussion of the threats to private keys in both types of systems.

II. Single Key Systems

A. Distribution of Communication Keys

Needham and Schroeder assume that each user A has a private (secret) key K_A which is known only to A

$$A \rightarrow B : A$$

$$B \rightarrow A : \left\{ A, N_{BA} \right\}_{K_{BS}}$$

$$A \rightarrow S : A \,.\, B \,.\, N_{AS} \,.\, \left\{ A \,.\, N_{BA} \right\}_{K_{BS}}$$

$$S \rightarrow A : \left\{ N_{AS} \,.\, B \,.\, K_{AB} \,.\, \left\{ K_{AB} \,.\, A \,.\, N_{BA} \right\}_{K_{BS}} \right\}_{K_{AS}}$$

$$A \rightarrow B : \left\{ K_{AB} \,.\, A \,.\, N_{BA} \right\}_{K_{BS}}$$

$$B \rightarrow A : \left\{ N_{AB} \right\}_{K_{AB}}$$

$$A \rightarrow B : \left\{ N_{AB} - 1 \right\}_{K_{AB}}$$

En pratique

Ce type d’algorithme permet de mettre en œuvre des systèmes à authentification unique.

Le protocole **Kerberos** est un protocole d’authentification réseau normalisé reposant sur le protocole de Needham-Scroeder.

Vers la cryptographie à clé publique

New Directions in Cryptography

Invited Paper

WHITFIELD DIFFIE AND MARTIN E. HELLMAN, MEMBER, IEEE

We stand today on the brink of a revolution in cryptography

I. Introduction

WE STAND TODAY on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade cryptographic devices down to where they can be used in such commercial applications as remote cash dispensers and computer terminals. In turn, such applications create a need for new types of cryptographic systems which minimize the necessity of secure key distribution channels

mon occurrence in business, however, and it is unrealistic to expect initial business contacts to be postponed long enough for keys to be transmitted by some physical means. The cost and delay imposed by this key distribution problem is a major barrier to the transfer of business communications to large teleprocessing networks.

Section III proposes two approaches to transmitting keying information over public (i.e., insecure) channels without compromising the security of the system. In a *public key cryptosystem* enciphering and deciphering are governed by distinct keys, E and D, such that computing D from E is computationally infeasible (e.g., requiring

Cryptographie à clé publique

Idée : briser la symétrie !

Dans la vie, il existe de nombreux phénomènes asymétriques, pour lesquels l'opération inverse est plus difficile que l'opération initiale.

Une **clé publique** distribuée librement (annuaire ?) permet de chiffrer les messages.

Une **clé privée**, gardée secrète, permet de déchiffrer les messages.

Encore plus fort que Kerckhoff !

Fonction à sens unique avec trappe

Une fonction f est une fonction à sens unique **à trappe** si le calcul de *f(x)* est facile et si retrouver *x* à partir de *f(x)* est calculatoirement impossible sans connaître la trappe, une information secrète *k*. L'inverse *g* de *f* se calcule facilement à partir de *k*.

Construire des couples *(f,g)* doit être facile.

Communiquer *f* ne doit rien révéler sur *g*.

On ne sait pas si de telles fonctions existent !

Formellement

Un schéma de chiffrement à clé publique est défini par 3 algorithmes PPT :

$\text{Gen}(1^k) = (pk, sk)$ algo probabiliste de génération de clés

$\text{Enc}_{pk}(m) = c$ algo probabiliste de chiffrement

$\text{Dec}_{sk}(c) = m$ algo déterministe de déchiffrement

$$\forall (pk, sk) = Gen(1^n) \quad \forall m \quad \text{Dec}_{sk}(\text{Enc}_{pk}(m)) = m$$

Sécurité prouvée

Partir d'un problème algorithmique Π **réputé difficile** (c'est une hypothèse de travail).

Par réduction, montrer que si un adversaire PPT casse le chiffre alors cet adversaire sait aussi résoudre Π en temps polynomial.

Une pincée de théorie des nombres et de recettes de cuisine…

« No one has yet discovered any warlike purpose to be served by the theory of numbers or relativity, and it seems unlikely that anyone will do so for many years »

G.H. Hardy,
A Mathematicians Apology, 1940.

Exponentiation rapide

Calculez 5^{21} modulo 17 !

$$(x^n \mod p) = \begin{cases} \left(x^2\right)^m \mod p & \text{si } n = 2m \\ x \times \left(x^2\right)^m \mod p & \text{si } n = 2m+1 \end{cases}$$

carrés successif modulo p	moitiés de n successives	éléments à multiplier mod p
5	**21**	5
8	10	
13	**5**	13
16	2	
1	**1**	1

5×13×1 ≡ 14 (mod 17)

Calcul du PGCD

Comment calculer le plus grand diviseur de deux entiers positifs *a* et *b* ?

Utiliser l'algorithme d'Euclide pour calculer le pgcd de 42 et 30.

Exercice calculez le pgcd de 4712 et 816 !

Inverses multiplicatifs modulo

Comment calculer l'inverse multiplicatif de *a* modulo *n* ? Et d'ailleurs, quand est-ce qu'il existe ?

Utiliser l'algorithme d'Euclide étendu pour calculer l'inverse de 5 modulo 13.

Exercice calculez l'inverse de 9 modulo 50 !

x = au + bv	u	v	
a=50	1	0	
b=9	0	1	soustraire 5×
5	1	-5	soustraire 1×
4	-1	6	soustraire 1×
1	2	-11	

2a - 11b = 1 donc 9×-11 = 9×39 = 1 (mod 50)

Théorème d'Euler

L'indicatrice d'Euler $\varphi(n)$ est le nombre d'entiers de 1 à n premiers avec n.

$$\varphi(p) = p - 1 \text{ si } p \text{ est premier}$$

$$\varphi\left(\prod_i p_i^{\alpha_i}\right) = \prod_i \left(p^{\alpha_i} - p^{\alpha_i - 1}\right)$$

Théorème de Fermat-Euler

$$a^{\varphi(n)} = 1 \pmod{n} \quad \text{si } \operatorname{pgcd}(a, n) = 1$$

Protocole d’échange de clés DH

Protocole d’échange de clés

Le protocole de Diffie-Hellman permet d’échanger une clé secrète sur un canal non sûr. Il est utilisé dans de nombreux logiciels.

Repose sur la difficulté à calculer des logarithmes discrets.

Attention ce protocole ne gère pas l’authentification, seul il est sensible à des attaques type *homme du milieu*.

Un peu de maths

Pour tout entier $n \geq 1$, l'ensemble des nombres premiers avec n munis de la multiplication forment un groupe noté $(\mathbb{Z}/n\mathbb{Z})^\times$ *ou* $\mathbb{Z}_n^*$.

Ce groupe est cyclique si $n=p^k$ ou $n=2p^k$ avec p un nombre premier.

Une racine primitive modulo n est un entier g tel que pour tout m, il existe un unique $1 \leq k \leq n-1$ tel que $m \bmod n = g^k \bmod n$.

k est le logarithme discret de m pour la base g modulo n.

Protocole DH

Le calcul des logarithmes discret est calculatoirement difficile.

Données du protocole : n premier et g non nul, ils peuvent être publiquement connus

(1) Alice choisit a et transmet $g^a \pmod n$ à Bob.

(2) Bob choisit b et transmet $g^b \pmod n$ à Alice.

(3) Alice et Bob calculent $k=(g^a)^b=(g^b)^a \pmod n$.

Illustration

$n = 941 \quad g = 627 \quad a = 137 \quad b = 513$

Calculer g^a, g^b et g^{ab} !

Suites de carrés modulo n :

627, 732, 395, 760, 767, 164, 548, 125, 569, 57, 426, ...

641, 605, 917, 576, 544, 462, 778, 221, 850, 753, 527, ...

922, 361, 463, 762, 47, 327, 596, 459, 838, 258, 694, ...

Quelques produits modulo n :

57×627=922, 125×374=641, 459×578=881,
627×760=374, 641×753=881, 762×922=578, ...

Problèmes associés

Problèmes de décision associés au protocole de Diffie-Hellman :

DLP étant donnés n, g et h, déterminer x tel que $g^x = h \pmod{n}$.

CDH étant donnés n et g, calculer $g^{ab} \pmod{n}$ à partir de $g^a \pmod{n}$ et $g^b \pmod{n}$.

DDH étant donnés n et g, distinguer modulo n les triplets (g^a, g^b, g^{ab}) des triplets (g^a, g^b, g^c) où a, b et c sont aléatoires et indépendants.

Sécurité prouvée

Si **DLP** est facile alors **CDH** l'est aussi.

Si **CDH** est facile alors **DDH** l'est aussi.

Il existe des groupes pour lesquels **DDH** est facile sans qu'on sache résoudre **DLP** et **CDH**.

Si **DDH** est difficile alors DH est sûr.

Problème de l'intrus

Pause exercice Expliquer comment un intrus Isidore, capable d'écouter et d'émettre des messages en se faisant passer pour Alice et Bob, peut s'arranger pour écouter la conversation entre Alice et Bob si ceux-ci utilisent le protocole DH pour échanger leur clé de session.

https://weakdh.org

Weak Diffie-Hellman and the Logjam Attack

Good News! Your browser is safe against the Logjam attack.

Diffie-Hellman key exchange is a popular cryptographic algorithm that allows Internet protocols to agree on a shared key and negotiate a secure connection. It is fundamental to many protocols including HTTPS, SSH, IPsec, SMTPS, and protocols that rely on TLS.

We have uncovered several weaknesses in how Diffie-Hellman key exchange has been deployed:

1. **Logjam attack against the TLS protocol.** The Logjam attack allows a man-in-the-middle attacker to downgrade vulnerable TLS connections to 512-bit export-grade cryptography. This allows the attacker to read and modify any data passed over the connection. The attack is reminiscent of the FREAK attack, but is due to a flaw in the TLS protocol rather than an implementation vulnerability, and attacks a Diffie-Hellman key exchange rather than an RSA key exchange. The attack affects any server that supports `DHE_EXPORT` ciphers, and affects all modern web browsers. 8.4% of the Top 1 Million domains were initially vulnerable.

2. **Threats from state-level adversaries.** Millions of HTTPS, SSH, and VPN servers all use the same prime numbers for Diffie-Hellman key exchange. Practitioners believed this was safe as long as new key exchange messages were generated for every connection. However, the first step in the number field sieve—the most efficient algorithm for breaking a Diffie-Hellman connection—is dependent only on this prime. After this first step, an attacker can quickly break individual connections.

Who is Affected?

Websites, mail servers, and other TLS-dependent services that support `DHE_EXPORT` ciphers are at risk for the Logjam attack. We use Internet-wide scanning to measure who is vulnerable.

Protocol	Vulnerable to Logjam
HTTPS — Top 1 Million Domains	8.4%
HTTPS — Browser Trusted Sites	3.4%
SMTP+StartTLS — IPv4 Address Space	14.8%
POP3S — IPv4 Address Space	8.9%
IMAPS — IPv4 Address Space	8.4%

Websites that use one of a few commonly shared 1024-bit Diffie-Hellman groups may be susceptible to passive eavesdropping from an attacker with nation-state resources. Here, we show how various protocols would be affected if a single 1024-bit group were broken in each protocol, assuming a typical up-to-date client (e.g., most recent version of OpenSSH or up-to-date installation of Chrome).

	Vulnerable if most common 1024-bit group is broken
HTTPS — Top 1 Million Domains	17.9%
HTTPS — Browser Trusted Sites	6.6%
SSH — IPv4 Address Space	25.7%
IKEv1 (IPsec VPNs) — IPv4 Address Space	66.1%

What Should I Do?

If you run a server…

If you have a web or mail server, you should disable support for export cipher suites and use a 2048-bit Diffie-Hellman group. We have published a **Guide to Deploying Diffie-Hellman for TLS** with step-by-step instructions. If you use SSH, you should upgrade both your server and client installations to the most recent version of OpenSSH, which prefers Elliptic-Curve Diffie-Hellman Key Exchange.

If you use a browser…

Make sure you have the most recent version of your browser installed, and check for updates frequently. Google Chrome (including Android Browser), Mozilla Firefox, Microsoft Internet Explorer, and Apple Safari are all deploying fixes for the Logjam attack.

If you're a sysadmin or developer …

Make sure any TLS libraries you use are up-to-date, that servers you maintain use 2048-bit or larger primes, and that clients you maintain reject Diffie-Hellman primes smaller than 1024-bit.

These results were first made public on May 20, 2015; peer-reviewed conference paper published October 13, 2015.

Cryptographie à clé publique

Le protocole de DH n’est pas un schéma de cryptographie à clé publique !

Le premier schéma est RSA, publié en 1978 par Rivest Shamir et Adleman, en cherchant à montrer qu’il n’en existait pas.

Intéressons nous d’abord à un schéma inspiré par le protocole de Diffie-Hellman et inventé par ElGamal en 1985.

Chiffrement ElGamal

ElGamal

Données du schéma : n premier et g non nul, ils peuvent être publiquement connus

Alice choisit une clé secrète s et calcule sa clé publique $y=g^s \pmod{n}$.

Pour chiffrer un message $2 \leq m \leq n-1$, Bob choisit aléatoirement k et et transmet la paire

$$c_1 = g^k \pmod{n} \qquad c_2 = my^k \pmod{n}$$

Alice déchiffre le message en calculant

$$\left(c_1^s\right)^{-1} c_2 = \left(g^{sk}\right)^{-1} my^k = m \pmod{n}$$

Illustration

$n = 467$

$g = 2$

$s = 153$

$m = 331$

$k = 197$

Calculer y, c_1, c_2 pour ensuite retrouver m !

Sécurité prouvée

Pause exercice Démontrer que savoir déchiffrer efficacement ElGamal est équivalent à résoudre efficacement **CDH**.

On pourra pour cela considérer l’existence d’un oracle qui résout un problème et l’utiliser pour résoudre l’autre.

En pratique

La cryptographie à clé publique est généralement bien plus lente à chiffrer que la cryptographie symétrique.

On utilise alors des systèmes hybrides : une clé de session est chiffrée avec une primitive à clé publique, qui permet de déchiffrer les données chiffrées avec une primitive symétrique (avec contrôle d’intégrité !)