République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université A/Mira de Béjaia
Faculté des Sciences Exactes
Département d'Informatique

MÉMOIRE DE MASTER

En Informatique

Option : ***Réseau et Sécurité Avancés***

Thème

Etude des vulnérabilités des protocoles d'accord de clés.

Présenté par : *M.* **BELIERDOUH Ahmed**
M. **TERRAH Zakaria**

Soutenu le : 01 juillet 2025

Devant le jury composé de :

Présidente	**Mme. S. MAMERI**	**MCB**	**U. A/Mira Béjaia.**
Examinatrice	**Mme. D. ZAMOUCHE**	**MCB**	**U. A/Mira Béjaia.**
Examinatrice	**Mme. M. YAICI**	**MCB**	**U. A/Mira Béjaia.**
Examinatrice	**Mme. S. ALOUI**	**MCA**	**U. A/Mira Béjaia.**
Encadrant	**M. M. SADI**	**MCB**	**U. A/Mira Béjaia.**

Béjaia, Juillet 2025.

⁕ *Remerciements* ⁕

Quoi que nous avancions et que nos portes s'ouvrent,

que nous atteignons tout ce que nous rêvons, il faut se rappeler ceux qui ont été la cause de notre succès, ceux qui nous ont soutenus et ont tenu notre main Pour continuer,

ceux dont la générosité nous a encouragés et motivés

Avant tout, nous remercions Dieu, le Tout-Puissant, pour sa guidance et ses bénédictions tout au long de ce projet.C'est grâce à Sa miséricorde que nous avons trouvé la force et la persévérance nécessaires pour mener à bien ce travail.

Nous adressons nos sincères remerciements à notre encadrant M.SADI MUSTAPHA, pour son accueille, ses conseils avisés et sa disponibilité, ainsi que son soutien indéfectible qui étaient précieux afin de mener notre travail sur la bonne voie. :

Nous tenons à exprimer notre profonde gratitude à nos familles, notamment à nos parents, pour leur amour, leur soutien inconditionnel et leur patience. Leurs sacrifices et leur foi inébranlable en nos capacités ont été essentiels à notre réussite, nous permettant de poursuivre nos études et de mener à bien ce projet.

Nous souhaitons également exprimer notre gratitude aux membres du jury pour avoir accepté de présider le jury et d'examiner ce mémoire.

Enfin, nous souhaitons remercier toutes les personnes qui, de près ou de loin, ont contribué à l'accomplissement de ce mémoire.

AHMED & ZAKARIA

⁕ *Dédicaces* ⁕

Alhamdulillah, toute la gloire et la gratitude reviennent à Allah pour Ses innombrables bienfaits. Cette réussite, je la dédie avec tout mon cœur à ceux qui m'ont soutenu et accompagné tout au long de ce chemin.

Un immense merci à mes chers parents, Belierdouh Mohammed et Menouche Nacera, pour leur amour inconditionnel, leurs sacrifices et leurs prières qui m'ont porté jusqu'ici. Vous êtes ma force et mon inspiration.

À mes adorables sœurs, Kouka, Foulla et Lapo, merci pour votre soutien sans faille, vos encouragements et vos sourires qui illuminent mes journées.

À mes frères, Seif-eddine, Adel et Ibrahim, merci pour vos conseils, votre présence et votre fierté qui me poussent à donner le meilleur de moi-même.

Un grand respect et une profonde gratitude à mon encadrant, M.Mustapha Sadi, pour sa guidance, sa patience et son expertise qui ont été précieux pour mener à bien ce projet.

À mes amis, chacun portant un nom spécial dans mon cœur, merci pour les moments partagés, les rires, les discussions et votre soutien indéfectible. Vous savez qui vous êtes, et je vous suis reconnaissant.

Enfin, à mon binôme Zaki, merci pour ta collaboration, ton engagement et les efforts qu'on a fournis ensemble. WE DID IT !

Cette dédicace est pour vous tous, ma famille, mes amis, mes guides. Vous êtes ma richesse et ma motivation. Alhamdulillah pour vous

M. AHMED BELIERDOUH

⁕ *Dédicaces* ⁕

C'est avec fierté que je dédie ce modeste travail à

À celui dont je porte le nom avec fierté, à celui qui a consacré toute sa vie à être meilleur pour les siens, à celui qui m'a soutenu sans compensation et m'a donné sans limites, mon cher père «Terrah MHAMMED ».

ma chère mère « A. FATMA ». *À celle qui m'a toujours soutenue et qui m'a inspiré de poursuivre mon chemin et qui a allégé mes difficultés par ses prières. À la lumière qui m'a éclairé le chemin,*
ma chère mère « A. FATMA ».

A mes agréables frères « Marouane », « sofiane », « abdellah », et mes chère sœurs « Samia », « Meriem », « Hamida », « Djazia,.

À toute ma famille pour leur soutien.

Je tiens également à remercier mes chers amis : Chems edddin, Wail, Mhammed, Islem, Raouf, Slimane, Akram, Aissam, Ilyes, Zaki, Abderrahime , Salah, Aghiles, Alilou.

À mon binôme, « Ahmed ». À que je lui souhaite beaucoup de réussite dans sa vie.

À tous ceux qui ont participé à ma réussite.

M. TERRAH ZAKARIA

Table des matières

Table des figures

Liste des tableaux

Liste des abréviations

KAP	KAP Key Agreement Protocol	Protocole d'accord de clé
DH	Diffie-Hellman	Diffie-Hellman
GDH.2	Group Diffie-Hellman version 2	Protocole Diffie-Hellman de groupe version 2
A-GDH2	Authenticated Group Diffie-Hellman version 2	Diffie-Hellman de groupe authentifié, version 2
TLS	Transport Layer Security	Sécurité de la couche de transport
IMITM	Man-in-the-Middle	Attaque de l'homme du milieu
PFS	Perfect Forward Secrecy	Confidentialité persistante
CLIQUES	Contributory Logical Key Establishment System	Système logique d'établissement de clé contributif
RFC	Request for Comments	Demande de commentaires (normes IETF)
IKEv2	Internet Key Exchange version 2	Échange de clés Internet version 2
ECDHE	Elliptic Curve Diffie-Hellman Ephemeral	Diffie-Hellman éphémère sur courbes elliptiques
DHE	Diffie-Hellman Ephemeral	Diffie-Hellman éphémère
CVE	Common Vulnerabilities and Exposures	Référentiel des vulnérabilités connues
PKI	Public Key Infrastructure	Infrastructure à clé publique

PAKE	Password Authenticated Key Exchange	Échange de clé authentifié par mot de passe
DoS	DoS Denial of Service	Déni de service
Zp*	Multiplicative group modulo p	Groupe multiplicatif modulo p

Introduction générale

À l'heure où les échanges numériques prennent une place toujours plus importante dans nos sociétés, garantir la confidentialité, l'intégrité et l'authenticité des communications est devenu un impératif majeur pour assurer la sécurité des systèmes d'information. Qu'il s'agisse d'interactions entre individus, de transmissions industrielles sensibles ou de communications automatisées entre machines, la sécurité des données repose sur un pilier essentiel de la cryptographie : l'accord de clé.

Les protocoles d'accord de clé (Key Agreement Protocols – KAP) ont pour objectif de permettre à plusieurs parties, souvent connectées par des canaux non sécurisés, de générer ensemble une clé secrète commune sans jamais avoir à l'échanger directement. Cette clé partagée est ensuite utilisée comme socle pour des opérations de chiffrement assurant la protection des données transmises.

Depuis l'apparition du célèbre protocole Diffie-Hellman en 1976, de nombreuses variantes ont vu le jour, visant à répondre à des contextes toujours plus complexes, notamment ceux impliquant plusieurs participants. À mesure que se sont développées les applications collaboratives, les visioconférences sécurisées ou encore les réseaux de capteurs intelligents, les protocoles de groupe sont devenus un élément fondamental dans la conception des architectures de sécurité contemporaines. Pourtant, leur sécurité ne dépend pas uniquement de la robustesse des algorithmes cryptographiques employés, mais aussi – et surtout – de la logique de conception du protocole, de la qualité de son implémentation et de sa résistance structurelle face à des menaces, qu'elles soient passives ou actives.

Or, plusieurs études récentes mettent en lumière une réalité préoccupante : certains protocoles, bien que reposant sur des bases mathématiques solides, présentent des vulnérabilités structurelles exploitables sans même casser la cryptographie sous-jacente. Ces failles ne sont pas liées à une faiblesse des primitives cryptographiques, mais plutôt à la manière dont les messages sont structurés, combinés ou exposés au sein du protocole. Face à cette situation, il devient essentiel de compléter les méthodes classiques d'analyse par une étude rigoureuse de la structure interne des protocoles, afin d'identifier des vulnérabilités qui échapperaient à une simple vérification algorithmique.

Dans quelle mesure les protocoles d'accord de clé multiparticipants, comme A-GDH2, sont-ils exposés à des attaques structurelles passives, et comment une analyse formelle de leur organisation interne peut-elle permettre de détecter – voire prévenir – ces failles, même en l'absence d'action directe de l'attaquant ?

Cette problématique interroge la sécurité effective de protocoles considérés comme sûrs, en appelant à dépasser les approches cryptographiques traditionnelles pour s'intéresser à la manière dont les protocoles sont structurés, organisés et comment les messages circulent au sein du système. Elle oriente ainsi notre recherche vers une démarche en deux temps. D'abord, démontrer qu'une attaque passive est réalisable sur un protocole existant et bien documenté. Puis, explorer les possibilités de généralisation de cette approche, en évaluant sa validité ou ses limites sur un autre protocole du même type.

Chapitre 1

Généralités, Définitions et Contexte de la Recherche

1.1 Introduction Générale et Contexte de la Recherche

Ce premier chapitre établit les bases pour comprendre l'étude des failles dans les protocoles d'accord de clés. Il commence par expliquer le rôle des communications modernes et l'importance de sécuriser les échanges d'informations. Ensuite, il clarifie les notions essentielles et le fonctionnement de ces protocoles. Une analyse des types de vulnérabilités et des attaques possibles est présentée, en montrant leurs conséquences potentielles. Enfin, ce chapitre explique pourquoi cette recherche est cruciale pour la cybersécurité d'aujourd'hui et de demain, en s'appuyant sur les standards et recommandations actuels.

1.1.1 Contexte des communications modernes et enjeux de la sécurité des échanges

Aujourd'hui, le monde numérique est ultra-connecté, et les échanges sécurisés sont au cœur de nombreuses activités, qu'il s'agisse de transactions bancaires, de discussions personnelles ou de communications officielles. Cette dépendance aux technologies numériques fait de la cybersécurité une priorité majeure, non seulement pour les entreprises, mais aussi pour les particuliers [1].

Cependant, cette connectivité accrue entraîne des risques importants. Les cyberattaques deviennent de plus en plus complexes, avec une hausse notable des vols de données. Ces incidents peuvent causer des pertes financières importantes, nuire gravement à la réputation, entraîner des litiges juridiques, des amendes ou encore perturber des opérations essentielles. Pour limiter ces dangers et leurs coûts, il est crucial de repérer et de gérer les failles de manière proactive [2] [3].

La cybersécurité ne se limite plus à une simple question technique, elle est devenue un enjeu stratégique et économique clé, qui impacte directement la pérennité des organisations.

Face à des menaces toujours plus évoluées, les défenseurs doivent constamment innover, adapter leurs stratégies et mettre à jour leurs systèmes de protection pour contrer les attaquants [3] .

1.1.2 Rôle central de la cryptographie

La cryptographie, c'est l'art de coder les informations pour les protéger. Imaginez que vous envoyez une lettre : la cryptographie la transforme en un message secret que seul le destinataire peut lire. Cela permet de garder les échanges privés, de s'assurer que personne ne modifie le message et de vérifier que l'expéditeur est bien celui qu'il prétend être.

Aujourd'hui, la cryptographie ne sert pas seulement à cacher des messages, Elle garantit aussi qu'ils ne sont pas altérés et qu'ils viennent de la bonne personne. Si l'un de ces aspects (confidentialité, intégrité ou authenticité) est mal conçu, même un code très fort peut être vulnérable. C'est pourquoi il faut créer des systèmes de cryptographie très soigneusement pour qu'ils soient vraiment sécurisés.

1.1.3 Aperçu sur les protocoles d'accord de clés et problèmes de leurs vulnérabilités

Les protocoles d'accord de clés (Key Agreement Protocols - KAP) sont des outils essentiels pour sécuriser les communications numériques. Ils permettent à deux personnes ou machines de créer ensemble une clé secrète, même sur un réseau non sécurisé, sans jamais envoyer cette clé directement. Cette clé sert ensuite à coder les messages pour qu'ils restent privés et intacts. Ces protocoles sont utilisés dans des technologies courantes, comme les réseaux privés virtuels (VPN), les protocoles Secure Sockets Layer (SSL) et Transport Layer Security (TLS), et Secure Shell (SSH) [4] [5].

Si ces protocoles ont des failles, que ce soit dans leur conception ou leur utilisation, les conséquences peuvent être sérieuses. Une faiblesse peut permettre à un attaquant de découvrir la clé secrète, de lire les messages, de les modifier ou même de se faire passer pour quelqu'un d'autre. Par exemple, dans une attaque dite "de l'homme du milieu (MitM) ", un pirate peut intercepter la clé et espionner ou manipuler toutes les communications qui suivent. Cela rend ces protocoles cruciaux : s'ils échouent, toute la sécurité des échanges s'effondre. Étudier leurs vulnérabilités est donc vital pour protéger les bases de la communication sécurisée [4] [6] [7].

1.1.4 Objectifs et structure du chapitre

Ce chapitre pose les bases pour comprendre l'étude des failles des protocoles d'accord de clés. Il explique les notions essentielles de la cryptographie et des protocoles d'accord de clés, présente les principaux protocoles utilisés aujourd'hui, décrit les types de failles et d'attaques qui peuvent les viser, et montre pourquoi ce sujet est important pour la cybersécurité actuelle et future. Ces explications serviront de point de départ pour les chapitres suivants, qui exploreront en détail des vulnérabilités précises, comment les détecter et comment les corriger.

1.2 Fondements de la cryptographie

1.2.1 Définition et objectifs fondamentaux de la cryptographie

La cryptographie, c'est l'art de protéger les communications pour qu'elles restent sécurisées, même sur un réseau non protégé. Elle empêche les intrus de lire, modifier ou falsifier les messages. La cryptographie moderne repose sur trois grands objectifs [1] [8] :

- **Confidentialité :** Seule la personne destinée à recevoir le message peut le comprendre. Cela passe par le chiffrement, qui transforme un message clair en un code secret que seule une clé peut déchiffrer.

- **Intégrité :** Le message ne doit pas être modifié pendant son envoi. Des outils comme les codes d'authentification (MAC) vérifient que tout est intact.

- **Authenticité :** On s'assure que le message vient bien de la bonne personne. Cela peut se faire avec une clé partagée ou une signature numérique.

En plus de ces trois piliers, d'autres aspects renforcent la sécurité :

- **Non-répudiation :** Personne ne peut nier avoir envoyé ou reçu un message, grâce à des signatures numériques.

- **Fraîcheur :** On vérifie que le message est nouveau, pour éviter qu'un attaquant réutilise un vieux message intercepté.

Le tableau suivant récapitule les objectifs fondamentaux de la cryptographie, qui sont essentiels pour comprendre les propriétés visées par les protocoles de sécurité et, par extension, les vulnérabilités qui peuvent les compromettre.

Objectif de Sécurité	**Définition**	**Moyen Cryptographique Principal**
Confidentialité	Le message n'est compréhensible que pour son destinataire autorisé.	Chiffrement (symétrique ou asymétrique)
Intégrité	Le message est protégé contre toute modification non autorisée ; toute altération est détectable.	Codes d'Authentification de Message (MAC), Fonctions de hachage
Authenticité	Le destinataire peut vérifier l'origine du message et l'identité de l'émetteur.	Signatures numériques, MAC, Certificats numériques
Non-répudiation	Un agent ne peut pas nier avoir envoyé ou reçu un message ou une transaction.	Signatures numériques
Fraîcheur	Le message est récent et n'est pas une retransmission (rejeu) d'un message ancien.	Nonces, Horodatages

TABLEAU 1.1 – Objectifs de sécurité cryptographique.

La cryptographie ne se limite pas à coder un message, Elle combine ces éléments pour créer un système sécurisé. Si l'un d'eux, comme l'authenticité, est faible, un pirate pourrait intercepter les messages sans être repéré, même si le chiffrement est solide. C'est pourquoi concevoir un bon système cryptographique demande une attention particulière à tous ces aspects.

1.2.2 Distinction entre cryptographie symétrique et asymétrique

La cryptographie se divise en deux grandes catégories selon la manière dont elle utilise les clés : la cryptographie symétrique et la cryptographie asymétrique.

1. **Cryptographie symétrique :** Elle utilise une seule clé secrète, partagée entre l'expéditeur et le destinataire, pour coder et décoder les messages. Cette méthode est rapide et idéale pour protéger de grandes quantités de données. Mais le gros défi, c'est de partager cette clé en toute sécurité : si quelqu'un l'intercepte, il peut tout déchiffrer [1] [5] [6] [7] [8] .

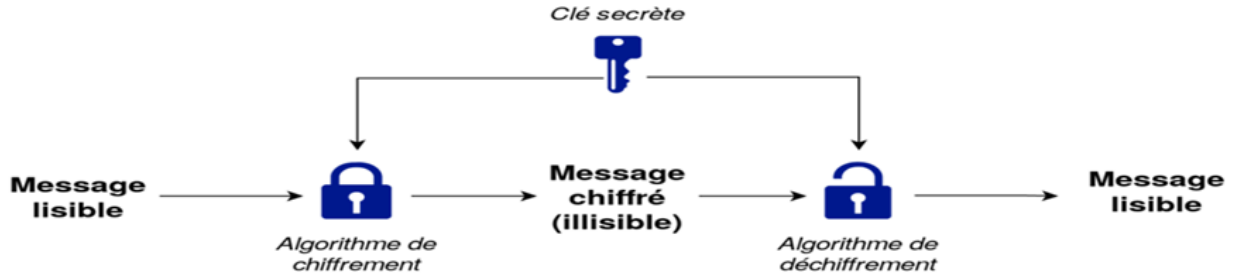

FIGURE 1.1 – La cryptographie symétrique.

2. **Cryptographie asymétrique :** Aussi appelée cryptographie à clé publique, elle fonctionne avec deux clés liées : une clé publique, que tout le monde peut connaître, et une clé privée, qui reste secrète. Un message codé avec la clé publique ne peut être décodé qu'avec la clé privée, et vice-versa. Cette méthode est parfaite pour vérifier l'identité de quelqu'un ou partager une clé sans risque, comme avec les algorithmes RSA ou ECC [1] [6] [9] [10] .

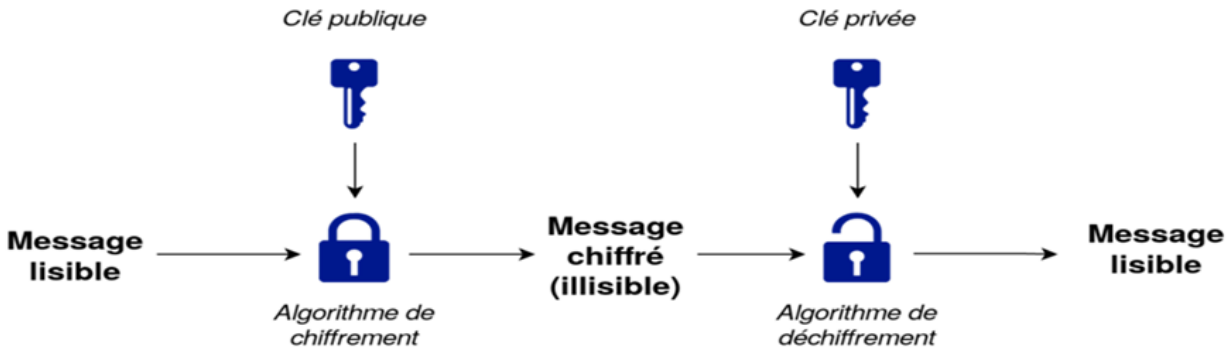

FIGURE 1.2 – Cryptographie asymétrique.

Le problème avec la cryptographie symétrique, c'est qu'il faut trouver un moyen sûr d'envoyer la clé secrète. La cryptographie asymétrique résout ça en permettant d'échanger des informations sécurisées sur un réseau non protégé. Mais elle est plus lente pour coder de gros volumes de données. C'est pourquoi on utilise souvent un système hybride : la cryptographie asymétrique sert à partager une clé, puis la cryptographie symétrique prend le relais pour coder les messages rapidement. C'est ce que font des technologies comme HTTPS pour sécuriser les sites web [1] [5] [7].

1.2.3 Comment fonctionnent les paires de clés publiques et privées

La cryptographie asymétrique utilise un duo de clés : une clé publique et une clé privée, qui fonctionnent ensemble pour sécuriser les communications [6] [10].

- **Clé publique :** Elle peut être partagée avec tout le monde sans risque. Elle sert à coder un message destiné à une personne précise ou à vérifier une signature numérique. Seule la personne avec la clé privée correspondante peut décoder le message [1] [6] [9] [10].

- **Clé privée :** Elle doit rester secrète et être gardée par son propriétaire. Elle permet de décoder les messages codés avec la clé publique associée ou de créer une signature numérique pour prouver son identité [6].

La sécurité de ce système repose sur des problèmes mathématiques très difficiles à résoudre. Par exemple, pour l'algorithme RSA, la clé publique est basée sur la multiplication de deux grands nombres premiers (comme 23 x 199 = 4577). Trouver ces deux nombres à partir du résultat est presque impossible avec les ordinateurs actuels, car il y a trop de combinaisons possibles. De même, pour des protocoles comme Diffie-Hellman ou ECDH, la sécurité dépend d'un autre problème mathématique complexe, appelé logarithme discret. Ces difficultés protègent la clé privée, rendant très difficile pour un pirate de la deviner à partir de la clé publique [10] [6] [11].

1.3 Protocoles d'Accord de Clés : Définitions et Principes de fonctionnement

1.3.1 Définition d'un protocole d'accord de clés (Key Agreement Protocol - KAP) et sa différenciation d'un protocole d'échange de clés

En cryptographie, créer une clé secrète partagée est crucial pour sécuriser les communications. Deux méthodes existent : l'accord de clés (KAP) et l'échange de clés, mais elles fonctionnent différemment. Un protocole d'accord de clés permet à deux personnes (ou plus) de créer ensemble une clé secrète, même sur un réseau non sécurisé. Chaque participant contribue à la clé, ce qui la rend imprévisible et unique. Personne ne peut décider seul de la clé finale [4] [7] [12].

En revanche, dans un protocole d'échange de clés, une seule personne crée la clé et l'envoie à l'autre de manière sécurisée. Ici, une seule partie contrôle la clé, ce qui est moins collaboratif [7] [4] [12].

L'accord de clés est plus sûr, car tout le monde participe, rendant la clé plus difficile à deviner pour un pirate. Cela protège mieux contre des attaques comme celle de l'"homme du milieu", où un attaquant essaie d'intercepter la communication. De plus, les protocoles d'accord de clés renforcent la sécurité des échanges futurs, grâce à une propriété appelée confidentialité persistante. Cette approche collaborative est donc essentielle pour des systèmes cryptographiques solides [7].

1.3.2 Propriétés de sécurité attendues des protocoles accord de clés

Un protocole d'accord de clés ne se contente pas de créer une clé secrète partagée. Il doit aussi garantir plusieurs protections pour être vraiment sécurisé, surtout face à des attaquants de plus en plus malins. Voici les principales propriétés qu'un bon KAP doit offrir [8] [13] :

- **Confidentialité :** La clé secrète reste cachée pour qu'aucun intrus ne puisse la découvrir [8].

- **Authentification de clé :** Chaque participant est sûr que seule la bonne personne (ou un système de confiance) peut connaître la clé secrète.

- **Confirmation de clé :** Tout le monde sait que les autres participants ont bien reçu la même clé secrète.

- **Authentification explicite :** Une version renforcée qui combine les deux points précédents, garantissant que seules les bonnes personnes ont la clé.

- **Fraîcheur :** La clé est nouvelle, pas une vieille clé réutilisée, ce qui évite les attaques où un pirate renvoie un ancien message [8] [13].

- **Résistance aux attaques de l'homme du milieu (MitM) :** Le protocole empêche un attaquant de se faire passer pour un participant légitime en interceptant et manipulant les messages [4] [14] [15].

- **Confidentialité persistante (PFS) :** Si une clé importante est volée plus tard, les communications passées restent protégées, car elles utilisaient des clés temporaires [7] [16].

Créer une clé partagée ne suffit pas. Un bon KAP doit assurer que les participants sont bien qui ils prétendent être et que la clé est bien reçue, tout en résistant aux attaques sophistiquées. La confidentialité persistante, par exemple, protège contre les pirates qui récupèrent des clés longtemps après les échanges. Ces protections sont essentielles face aux menaces toujours plus complexes [4] [7] [16].

1.3.3 Présentation des principaux protocoles d'accord de clés couramment utilisées

Plusieurs protocoles d'accord de clés existent pour créer des clés secrètes partagées, même sur des réseaux non sécurisés. Ils ont évolué pour devenir plus sûrs face aux attaques toujours plus complexes. Voici une présentation du protocole Diffie-Hellman, l'un des plus connus :

a. Diffie-Hellman (DH) :

Le protocole de Diffie-Hellman [5] s'appuie sur le problème du logarithme discret, un principe mathématique selon lequel il est simple de multiplier un grand nombre plusieurs fois et de calculer son reste par division avec un nombre premier, mais extrêmement difficile de retrouver le nombre initial à partir du résultat.

- Introduit en 1976 par Whitfield Diffie et Martin Hellman, c'est le premier protocole d'accord de clés public.

- Il permet à deux personnes (par exemple, Alice et Bob) de créer une clé secrète commune sans jamais l'envoyer directement, même sur un réseau non protégé.

• Ce protocole repose sur un problème mathématique complexe, appelé logarithme discret, difficile à résoudre.

• Voici comment ça marche simplement : Alice et Bob choisissent deux nombres publics (un grand nombre premier et un générateur). Chacun choisit aussi un nombre secret privé, calcule une valeur publique à partir de ce secret, et ils s'échangent ces valeurs. Ensuite, chacun utilise la valeur reçue et son propre secret pour calculer la même clé secrète partagée.

• Il existe une version améliorée, appelée Diffie-Hellman éphémère (DHE), qui utilise des clés temporaires pour chaque connexion, renforçant la sécurité des échanges passés (confidentialité persistante) [5] [4] [14].

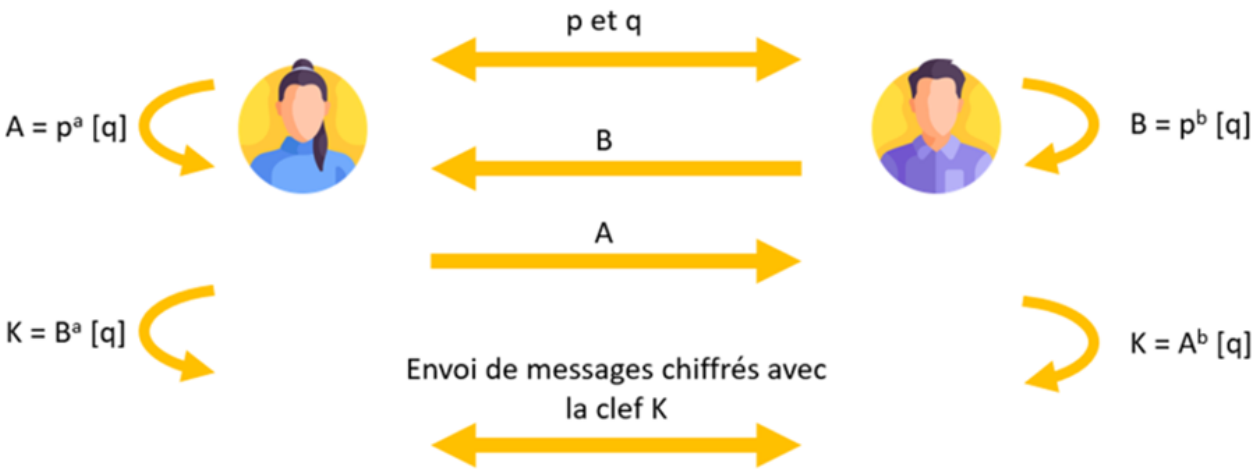

FIGURE 1.3 – Principe de l'échange de clé Diffie-Hellman.

b. Elliptic Curve Diffie-Hellman (ECDH) :

• ECDH est une version améliorée du protocole Diffie-Hellman, basée sur la cryptographie des courbes elliptiques (ECC).

• Son grand avantage est d'offrir une sécurité élevée avec des clés plus petites. Par exemple, une clé ECDH de 256 bits est aussi sûre qu'une clé RSA beaucoup plus grande (3072 bits), ce qui la rend plus rapide et moins gourmande en ressources.

• Comme Diffie-Hellman, ECDH permet à deux personnes de créer une clé secrète commune sur un réseau non sécurisé, en utilisant une paire de clés publique-privée.

• On le trouve dans des applications modernes comme WhatsApp ou Signal pour sécuriser les messages de bout en bout [1] [17] [18] [19] [20].

La figure ci dessus montre le fonctionnement du protocole ECDH

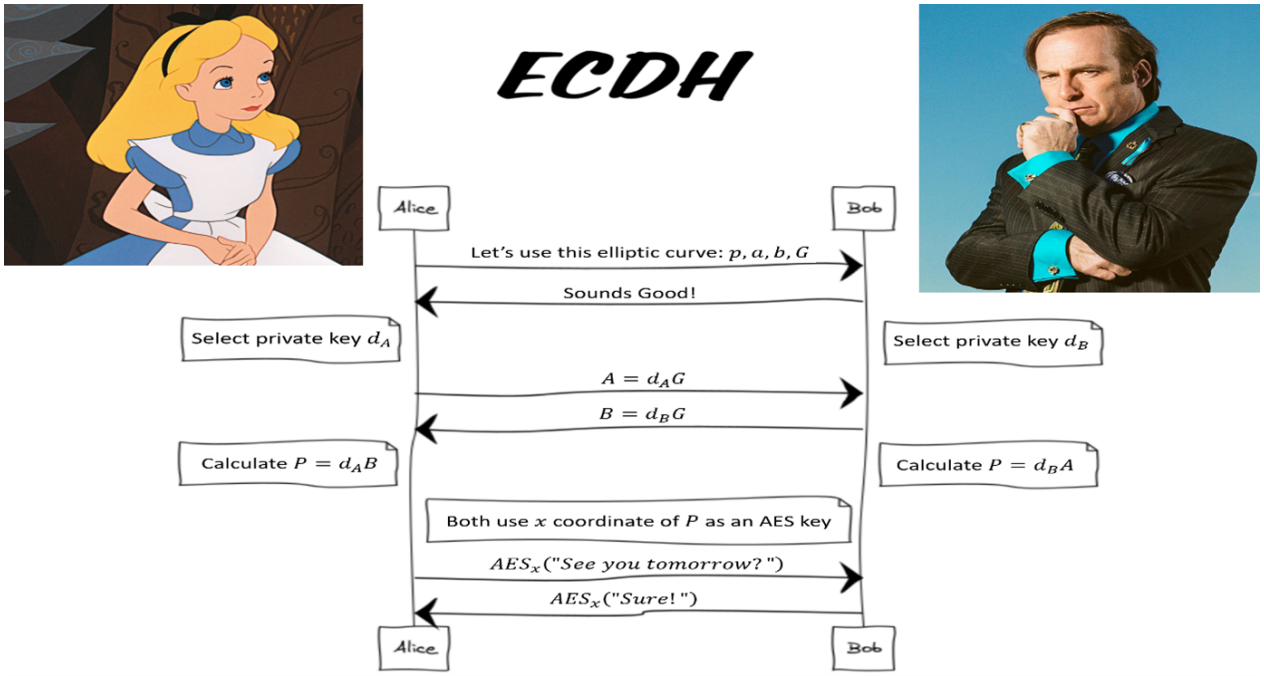

FIGURE 1.4 – Principe de l'échange de clé ECDH [11].

c. Station-to-Station (STS) Protocol :

• Le protocole STS est une version avancée de Diffie-Hellman qui ajoute une vérification d'identité.

• Il combine Diffie-Hellman avec des signatures numériques pour s'assurer que les participants sont bien qui ils prétendent être, ce qui protège contre les attaques de l'"homme du milieu" (où un pirate se fait passer pour un participant).

• STS garantit non seulement une clé secrète partagée, mais aussi que les bonnes personnes l'utilisent [5] [17] [19] [21] .

Ces protocoles montrent une évolution : le Diffie-Hellman original manquait de vérification d'identité, ce qui le rendait vulnérable. ECDH améliore l'efficacité avec des clés plus petites, et STS renforce la sécurité en ajoutant l'authentification. Ces progrès répondent aux besoins d'une sécurité plus robuste face aux menaces modernes.

Le tableau ci-dessous offre une comparaison concise des principaux protocoles d'accord de clés discutées, mettant en évidence leurs principes, avantages, inconvénients et applications courantes.

Protocole	**Principe de base**	**Avantages**	**Inconvénients/ Vulnérabilités inhérentes**	**Applications courantesl**
Diffie-Hellman (DH)	Basé sur le problème du logarithme discret dans des groupes finis.	Simplicité conceptuelle ; premier protocole d'accord de clé publique.	Aucune authentification inhérente, vulnérable aux attaques de l'homme du milieu (MitM) si non combiné avec d'autres mécanismes.	VPN, SSH, TLS (comme base pour DHE)
Elliptic Curve Diffie-Hellman (ECDH) [17]	Basé sur le problème du logarithme discret sur les courbes elliptiques.	Haute sécurité avec des clés plus courtes (efficacité cryptographique) ; performance améliorée par rapport au DH classique.	Complexité mathématique plus élevée ; vulnérable aux attaques sur courbes invalides ou paramètres de courbe mal choisis si l'implémentation est défectueuse.	Signal Protocol, WhatsApp, TLS (comme base pour ECDHE)
STS (Station to-Station) [61]	Combinaison de Diffie-Hellman avec des signatures numériques.	Fournit une authentification mutuelle des parties et de la clé, résistant aux attaques MitM.	Peut être vulnérable aux attaques de rétrogradation TLS ou à des attaques par réflexion dans des configurations spécifiques.	Protocoles d'authentification sécurisée, certaines implémentations VPN

Tableau 1.2 – Comparaison des protocoles d'accords de clés majeurs.

1.4 Classification des Vulnérabilités des Protocoles Cryptographiques

1.4.1 Vue d'ensemble des faiblesses de conception et d'implémentation

Les failles dans les systèmes cryptographiques sont une menace majeure en cybersécurité, souvent responsables de fuites de données sensibles. Elles peuvent venir des outils utilisés, de la manière dont ils sont mis en place ou de leur gestion au quotidien. Voici les causes principales expliquées simplement [22] :

- **Algorithmes dépassés ou faibles :** Certains algorithmes, comme MD5, SHA-1 ou DES, sont trop vieux ou fragiles et ne résistent pas aux attaques modernes. Par exemple, les anciennes versions de SSL (2.0 et 3.0) ont été abandonnées car elles étaient vulnérables, notamment à des attaques où un pirate peut se faire passer pour un utilisateur légitime.

- **Mauvaise gestion des clés :** Les clés cryptographiques qui protègent les données, doivent être bien créées, stockées et protégées. Si elles sont mal gérées (par exemple, stockées avec les données ou jamais renouvelées), un pirate qui les trouve peut accéder à tout.

- **Erreurs dans le code :** Même un bon algorithme peut être vulnérable si le programme qui l'utilise contient des erreurs. Par exemple, la faille Heartbleed dans OpenSSL a exposé des données sensibles à cause d'un problème de programmation.

- **Nombres aléatoires peu fiables :** Les clés et autres éléments cryptographiques reposent sur des nombres aléatoires. S'ils sont prévisibles, un attaquant peut facilement les deviner, compromettant la sécurité.

- **Protocoles ou configurations non sécurisées :** Utiliser des versions obsolètes de protocoles (comme TLS 1.0) ou des réglages par défaut mal sécurisés peut ouvrir la porte aux attaques.

- **Erreurs humaines :** Des erreurs simples, comme laisser une clé visible dans un fichier ou un journal, peuvent causer des fuites de données accidentelles.

Ces failles montrent qu'un système cryptographique doit être bien conçu, bien programmé et bien géré pour rester sécurisé face aux menaces.

1.4.2 Types d'attaques génériques

Le tableau suivant résume les catégories d'attaques cryptographiques courantes.

Catégorie d'Attaque	Principe	Exemples/Mécanismes	Conséquences
Attaque par canal auxiliaire	Exploitation des informations involontairement divulguées par le système (temps, puissance, émissions) lors d'opérations cryptographiques.	Analyse du temps de calcul, mesure de la consommation électrique, analyse des émissions électromagnétiques.	Déduction de clés privées, informations sensibles sans accès direct au logiciel.
Attaque par rejeu	Interception et retransmission malveillante de transmissions de données valides pour initier des actions non autorisées.	Rejeu de jetons d'authentification, de requêtes de transaction financière, de signaux d'entrée sans clé.	Accès non autorisé à des systèmes, transactions financières répétées, contrôle d'appareils IoT.
Attaque de l'homme du milieu (MitM)	Un attaquant intercepte et relaie la communication entre deux parties, se faisant passer pour l'interlocuteur légitime de chacune.	Interception des clés publiques Diffie-Hellman, manipulation des messages SSL/TLS.	Compromission de la confidentialité et de l'intégrité des communications, vol d'informations sensibles, usurpation d'identité.

Tableau 1.3 – Catégories d'attaques cryptographiques courantes.

Ces attaques montrent que la sécurité d'un protocole ne dépend pas seulement de ses algorithmes, mais aussi de sa conception et de son utilisation. Les failles peuvent venir du matériel, de la logique du système ou d'un manque de vérification d'identité. Un bon protocole doit donc être protégé à tous ces niveaux pour être vraiment sécurisé [14] [15] [23] [24] [25] [26] [27].

1.5 Analyse des Vulnérabilités Spécifiques aux Protocoles d'Accord de Clés

1.5.1 Détail des attaques de l'homme du milieu (MitM) sur les KAP non authentifiés

Le protocole Diffie-Hellman (DH) permet à deux personnes de créer une clé secrète commune, même sur un réseau non sécurisé. Mais sans vérification d'identité, il est vulnérable à une attaque appelée "de l'homme du milieu" (MitM) [4] [5] [14] [25].

Comment fonctionne une attaque MitM ? Prenons Alice et Bob, qui veulent partager une clé. Un pirate, Carol, s'interpose entre eux. Quand Alice envoie sa clé publique à Bob, Carol l'intercepte et envoie sa propre clé à Bob. Elle fait pareil avec la clé publique de Bob en l'envoyant à Alice. Résultat : Alice partage une clé avec Carol, et Bob partage une autre clé avec Carol, sans le savoir. Carol peut alors lire ou modifier leurs messages avant de les transmettre, trompant Alice et Bob qui pensent communiquer directement [25].

Comment éviter cette attaque ? Pour sécuriser DH, on ajoute une vérification d'identité. Par exemple, des signatures numériques ou des certificats émis par une autorité de confiance permettent à Alice et Bob de confirmer qu'ils parlent bien à la bonne personne. Sans cela, un pirate peut facilement s'infiltrer [4][14] [15] [25].

Cette faille montre que créer une clé secrète ne suffit pas : il faut aussi s'assurer que les participants sont bien ceux qu'ils prétendent être. Sans vérification d'identité, un attaquant peut détourner la communication, même sans casser le chiffrement. C'est pourquoi des outils comme les certificats sont indispensables pour rendre DH vraiment sécurisé.

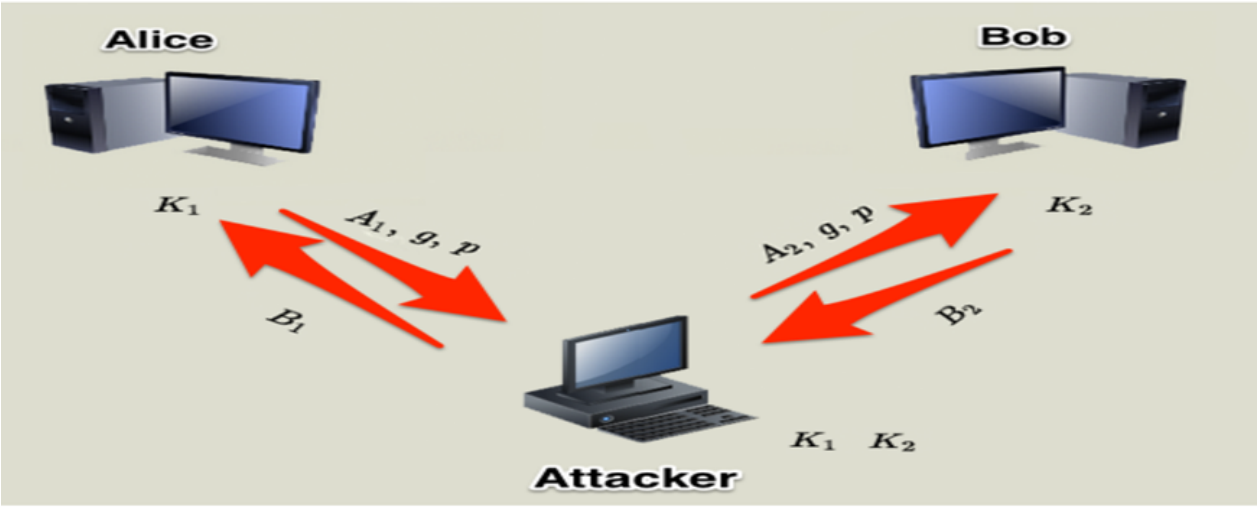

FIGURE 1.5 – Principe de l'attaque de l'homme de milieu sur Diffie-Hellman [25].

1.5.2 Vulnérabilités spécifiques au protocole Diffie-Hellman

• **Attaque Logjam :** L'attaque Logjam est une attaque cryptographique découverte en 2015 qui exploite une faiblesse dans l'utilisation de groupes Diffie-Hellman avec des paramètres faibles (notamment des clés de 512 bits). Elle permet à un attaquant en position d'interception (man-in-the- middle) de forcer les serveurs à utiliser des clés plus petites, puis de casser l'échange de clé et de déchiffrer les communications chiffrées. Elle exploite des systèmes qui acceptent des paramètres anciens pour rester compatibles. Utiliser des clés plus grandes (2048 bits ou plus) évite ce problème. [14].

• **Attaques D(HE)at (déni de service) :** Les attaques D(HE)at sont un type d'attaque par déni de service (DoS) ciblant les protocoles d'échange de clé Diffie-Hellman. Elles consistent à surcharger un serveur en lui envoyant un grand nombre de requêtes de négociation de clés, ce qui consomme excessivement ses ressources (CPU, mémoire), entraînant une dégradation ou un arrêt du service. Un exemple récent est la faille CVE-2024-41996, où une mauvaise vérification des clés cause une surcharge [28].

• **Attaques par force brute :** Une attaque par force brute est une méthode d'attaque qui consiste à essayer systématiquement toutes les combinaisons possibles d'un mot de passe, d'une clé cryptographique ou d'un identifiant jusqu'à trouver la bonne. Elle repose sur la puissance de calcul plutôt que sur une faille du système, et devient inefficace lorsque la longueur ou la complexité de la clé est suffisante. Le problème mathématique derrière DH (le logarithme discret) est difficile à résoudre, mais des méthodes comme Baby-step Giant-step ou l'algorithme de Pollard peuvent parfois l'attaquer si les paramètres sont mal choisis [29].

1.5.3 Vulnérabilités spécifiques à ECDH

Le protocole Elliptic Curve Diffie-Hellman (ECDH) est très sécurisé grâce à ses clés courtes mais puissantes. Cependant, il peut être vulnérable si ses paramètres ne sont pas bien vérifiés :

• Attaques sur courbes invalides : Si un pirate envoie un point qui ne correspond pas à la courbe elliptique convenue et que le système ne le vérifie pas, il peut deviner la clé privée à partir des informations reçues. Certaines bibliothèques TLS ont eu ce problème.

• Problèmes avec le générateur : Si le générateur de la courbe est mal choisi (par exemple, avec un ordre trop petit ou facile à factoriser), un attaquant peut plus facilement casser la clé.

• Courbes à risque : Utiliser des courbes elliptiques spéciales (singulières ou supersingulières) peut rendre le système plus facile à attaquer.

• Manque de vérification des points : Si le système ne vérifie pas que les points échangés appartiennent bien à la courbe, un pirate peut exploiter cette faille.

Ces problèmes ne viennent pas de la théorie d'ECDH, mais d'erreurs dans sa mise en œuvre. Une vérification stricte des paramètres et un codage soigné sont donc essentiels pour éviter ces failles [11] [17] [19].

1.5.4 Vulnérabilités spécifiques au protocole STS

Le protocole Station-to-Station (STS) améliore Diffie-Hellman en ajoutant une vérification d'identité. Mais il peut être vulnérable dans certains cas :

• Attaques de rétrogradation TLS : Un pirate peut forcer le système à utiliser une vieille version moins sécurisée de TLS (comme SSL 3.0), plus facile à attaquer. Des failles connues comme DROWN, POODLE ou FREAK exploitent ce problème.

• Attaques par réflexion : Si plusieurs systèmes STS utilisent la même clé pour différentes identités, un pirate peut réutiliser des messages d'une session dans une autre pour compromettre la sécurité.

• Attaques physiques : Si un pirate accède physiquement à un appareil (comme une station de base ou un serveur), il peut voler les clés, contournant les protections du protocole.

Même un protocole bien conçu comme STS peut être fragilisé par de mauvaises configurations, des versions obsolètes ou un mauvais usage des clés. Cela montre que la sécurité dépend non seulement du protocole lui-même, mais aussi de son environnement et de son utilisation [30] [31].

1.6 Pertinence de l'étude des vulnérabilités des KAP dans la Cybersécurité actuelle

1.6.1 L'importance de repérer les failles pour protéger les données et respecter les règles

Étudier les failles des protocoles d'accord de clés est crucial pour sécuriser les données sensibles et respecter les lois. Cette analyse aide à vérifier si les protections fonctionnent, à repérer les risques avant qu'ils ne causent des problèmes et à renforcer les défenses contre les attaques. Des réglementations comme le GDPR ou des normes comme ISO 27001 et NIST exigent une sécurité cryptographique solide et une gestion attentive des failles.

En repérant les problèmes à l'avance, les entreprises renforcent la confiance de leurs clients et partenaires, ce qui les rend plus compétitives. Cela permet aussi de réagir vite en cas d'incident, de limiter les coûts des cyberattaques et d'améliorer la sécurité en continu. Ce

n'est pas juste une question de corriger des erreurs après coup, mais de prévenir les problèmes pour protéger la réputation et la stabilité financière des organisations [32].

1.6.2 Les nouvelles menaces, comme l'informatique quantique

Le monde de la cybersécurité change rapidement avec des menaces nouvelles, notamment les ordinateurs quantiques. D'ici 2030-2035, ces machines pourraient casser les systèmes de chiffrement actuels, un événement appelé "Q-Day". Cela mettrait en danger les communications numériques, des transactions bancaires aux échanges officiels. Pour répondre à cette menace, la cryptographie post-quantique (PQC) développe des algorithmes résistants aux ordinateurs quantiques. Le NIST a déjà standardisé des solutions comme ML-KEM pour le chiffrement et CRYSTALS-Dilithium pour les signatures. En France, l'ANSSI pousse à intégrer ces protections dans les systèmes. Les entreprises doivent agir dès maintenant pour éviter que des données collectées aujourd'hui ne soient déchiffrées plus tard par des ordinateurs quantiques.

1.6.3 L'importance des normes et bonnes pratiques

La création et l'évaluation des KAP s'appuient sur des normes et recommandations d'organismes comme :

- **NIST :** Fournit des guides sur la gestion des clés et les algorithmes, comme des clés RSA d'au moins 2048 bits, et standardise la cryptographie post-quantique.

- **ISO :** Propose des normes, comme ISO 27001, pour gérer la sécurité et les clés.

- **ANSSI :** En France, donne des conseils sur les architectures sécurisées et la transition vers la cryptographie post-quantique.

- **IETF :** Publie des règles pour des protocoles comme TLS 1.3, avec des pratiques pour une mise en œuvre sécurisée.

Ces normes montrent que la cybersécurité est un domaine mature, mais en constante évolution. Les recommandations sont régulièrement mises à jour pour contrer les nouvelles menaces, comme l'informatique quantique. Cela prouve qu'il faut rester vigilant et suivre ces règles pour éviter de développer des systèmes cryptographiques risqués [33] [34].

1.7 Conclusion

Ce chapitre a jeté les bases pour comprendre les failles des protocoles d'accord de clés (KAP). Il a montré que la cryptographie est essentielle pour protéger les communications numériques, en assurant des objectifs comme garder les messages secrets (confidentialité), intacts (intégrité), vérifiables (authenticité), non contestables (non-répudiation) et récents (fraîcheur).

Nous avons vu la différence entre la cryptographie symétrique (une seule clé partagée) et asymétrique (une clé publique et une privée). L'asymétrique facilite le partage sécurisé des clés, mais les systèmes hybrides combinent les deux pour allier rapidité et sécurité.

Les KAP, comme Diffie-Hellman (DH), Elliptic Curve Diffie-Hellman (ECDH) et Station-to-Station (STS), permettent à deux parties de créer une clé secrète commune, contrairement aux protocoles d'échange de clés où une seule partie décide. Ces protocoles doivent garantir des protections comme la vérification d'identité et la sécurité des clés passées (confidentialité persistante).

Nous avons aussi exploré les failles des systèmes cryptographiques, qu'elles viennent de la conception, du codage ou de la gestion. Les attaques courantes incluent celles par canal auxiliaire (exploiter des fuites comme le temps de calcul), par rejeu (réutiliser un vieux message) et de l'homme du milieu (se faire passer pour un participant). Des failles spécifiques touchent DH (comme Logjam ou D(HE)at), ECDH (courbes mal choisies) et STS (rétrogradation TLS).

Enfin, étudier ces failles est crucial face aux menaces actuelles et futures, comme les ordinateurs quantiques qui pourraient casser les systèmes actuels d'ici 2030-2035 ("Q-Day"). La cryptographie post-quantique se développe pour y répondre, et des normes (NIST, ISO, ANSSI, IETF) guident la création de protocoles sécurisés.

Ce chapitre offre une base solide pour la suite du mémoire. Les prochains chapitres vont approfondir les failles spécifiques des KAP, examiner comment les détecter et proposer des solutions concrètes pour les corriger. Cette approche passera d'une vue générale à des analyses pratiques et des idées nouvelles pour renforcer la sécurité.

Chapitre 2

Menaces et Vulnérabilités dans les protocoles d'accord de clé

La robustesse des protocoles d'accord de clé ne se limite pas à la solidité de leurs bases cryptographiques. Ces mécanismes doivent également faire preuve d'une résilience opérationnelle face à un spectre étendu de menaces, depuis les attaques passives d'interception jusqu'aux manipulations actives sophistiquées. Notre analyse dans ce chapitre s'articule autour de trois axes principaux : l'examen des paradigmes d'attaque conventionnels, l'étude des faiblesses intrinsèques affectant des protocoles fondamentaux tels que Diffie-Hellman, et les défis particuliers posés par l'administration des groupes dynamiques.

Le processus de rekeying fait l'objet d'une attention particulière dans notre investigation. Bien que fréquemment sous-estimé dans les analyses théoriques, ce mécanisme joue pourtant un rôle pivot dans la préservation de la confidentialité au sein des ensembles évolutifs de participants. Son importance stratégique égale celle des algorithmes cryptographiques sous-jacents lorsqu'il s'agit de maintenir la sécurité dans des contextes dynamiques.

En point d'orgue de ce chapitre, l'étude approfondie de la vulnérabilité Heartbleed sert de démonstration éloquente. Ce cas pratique révèle avec force comment une imperfection d'implémentation, pourtant anodine en apparence, peut anéantir la sécurité d'un système cryptographique dont les fondements théoriques apparaissaient inattaquables. Cet épisode marquant vient confirmer une vérité essentielle : la cryptographie la plus rigoureuse sur le plan conceptuel demeure exposée aux risques sans une mise en œuvre méticuleuse et systématiquement vérifiée.

La leçon fondamentale qui émerge de cette analyse est double. D'une part, la sécurité effective résulte de l'articulation harmonieuse entre théorie et pratique. D'autre part, tout protocole, aussi sophistiqué soit-il, doit être envisagé comme un système global où chaque composante - y compris celles paraissant secondaires - peut devenir le maillon faible compromettant l'ensemble de l'édifice sécuritaire. Cette perspective holistique guide notre approche tout au long de ce chapitre

2.1 Introduction aux modèles d'attaques sur les protocoles d'accord de clé

2.1.1 Définition générale des attaques sur les protocoles

Dans le domaine sécuritaire des communications, une attaque se définit comme toute action malveillante entreprise par un adversaire visant à subvertir les propriétés fondamentales d'un protocole. Ces propriétés cardinales comprennent principalement : la confidentialité des échanges, garantissant que seules les parties autorisées accèdent au contenu ; l'authenticité des participants, assurant l'identité vérifiée des interlocuteurs ; l'intégrité des données, protégeant contre toute altération non détectée ; et enfin la disponibilité du service, préservant l'accès légitime aux ressources .

Concernant spécifiquement les protocoles d'accord de clé, les vecteurs d'attaque se concentrent particulièrement sur trois axes majeurs : l'interception illicite des échanges, la reconstruction non autorisée de la clé secrète, ou la manipulation frauduleuse du processus d'établissement de cette clé partagée entre les participants légitimes [35].

Évaluation de la robustesse protocolaire

L'analyse approfondie des modèles d'attaque représente une étape indispensable pour apprécier la résilience effective d'un protocole cryptographique. Cette nécessité découle d'un principe fondamental : chaque mécanisme de sécurité repose sur un ensemble d'hypothèses explicites ou implicites concernant les capacités attribuées à l'adversaire potentiel. Ces postulats conditionnent directement le niveau de sécurité apparent - un même protocole pouvant apparaître absolument inviolable sous certaines hypothèses, tout en se révélant totalement vulnérable lorsque ces dernières ne correspondent pas à la réalité opérationnelle .

C'est pourquoi la classification primaire entre attaques passives (simple observation des échanges) et actives (modification interventionniste des communications) constitue le socle initial de toute évaluation sécuritaire sérieuse. Cette dichotomie fondamentale structure l'analyse des vulnérabilités potentielles et guide la conception des contre-mesures appropriées, établissant ainsi le cadre méthodologique essentiel à l'étude des protocoles cryptographiques modernes [36].

2.1.2 Attaque passive : l'observation silencieuse

Une attaque passive se caractérise par un adversaire qui agit comme un simple observateur des communications. Cet acteur malveillant se limite à enregistrer les échanges protocolaires sans aucune interférence, dans l'objectif d'extraire des informations sensibles telles que les paramètres publics, des composantes partielles de clés ou des métadonnées temporelles .

La dangerosité principale de cette attaque réside dans son caractère totalement furtif. Comme l'attaquant n'altère en rien le déroulement normal du protocole, son activité reste imperceptible, rendant ainsi inefficaces les mécanismes classiques de détection. Toutefois, son succès est généralement conditionné par l'existence de vulnérabilités inhérentes au système, notamment des paramètres cryptographiques insuffisamment robustes ou une gestion déficiente des éléments confidentiels [37].

Exemple typique :

Considérons le protocole Diffie-Hellman. Un attaquant passif peut capturer les valeurs publiques g^x et g^y échangées entre les participants. Son but ultime consiste à dériver la clé secrète partagée g^{xy} en résolvant le problème computationnel du logarithme discret. Cette opération, théoriquement complexe, devient réalisable en pratique lorsque les paramètres initiaux - particulièrement le groupe multiplicatif Zp* utilisé - présentent des faiblesses, permettant alors l'application de techniques d'optimisation comme les tables de précalcul ou les algorithmes de calcul d'indices.

2.1.3 Attaque active : l'intrusion malveillante

Contrairement à l'attaque passive, l'attaque active implique une intervention directe de l'adversaire dans les échanges. Ce dernier ne se contente pas d'observer, mais agit délibérément sur les communications en modifiant, supprimant, répétant ou injectant des messages falsifiés. Son objectif principal est de tromper un ou plusieurs participants, de compromettre l'établissement de la clé secrète ou d'usurper l'identité d'un acteur légitime du protocole.

Bien que théoriquement plus détectables que les attaques passives, ces intrusions restent particulièrement dangereuses lorsque le protocole ne dispose pas de mécanismes d'authentification robustes ou de procédures de validation strictes. Leur efficacité est maximale dans les systèmes où les identités des participants ne sont pas rigoureusement vérifiées [37].

Exemple classique : l'attaque Man-in-the-Middle (MITM)

Dans ce scénario classique, l'attaquant s'insère subtilement entre deux entités communicantes (typiquement Alice et Bob). Après avoir intercepté leurs échanges initiaux, il substitue astucieusement ses propres paramètres aux leurs. Le résultat est pernicieux : chaque correspondant croit établir une connexion sécurisée avec l'autre, alors qu'en réalité, ils créent chacun une session distincte avec l'attaquant. Ce dernier acquiert ainsi un contrôle total sur les communications, pouvant librement consulter, altérer et relayer les messages à son insu, tout en maintenant l'illusion d'un échange direct entre les victimes.

Cette attaque met en lumière l'importance cruciale des mécanismes d'authentification mutuelle dans tout protocole d'échange de clés. Sans ces garde-fous, même les schémas cryptographiques les plus robustes sur le plan théorique deviennent vulnérables à ce type de manipulation [25].

2.1.4 Objectifs stratégiques des attaques

Les attaques ciblant les protocoles d'échange de clés poursuivent plusieurs buts distincts mais souvent complémentaires. La finalité première reste l'obtention illicite de la clé secrète partagée, soit par interception directe des paramètres échangés, soit par déduction cryptographique à partir des éléments publics. Un autre objectif majeur concerne la compromission de l'authenticité, où l'attaquant cherche à usurper l'identité d'un participant légitime pour s'insérer frauduleusement dans les échanges.

Certaines attaques plus sophistiquées visent à induire une désynchronisation des états internes entre les participants, créant ainsi des incohérences dans les communications pouvant mener à leur interruption complète. D'autres exploitent astucieusement la répétition de messages valides (attaques par rejeu) pour réactiver des sessions expirées ou contourner des mécanismes de protection. Enfin, les attaquants peuvent chercher à affaiblir la sécurité à long terme en exploitant l'absence de renouvellement périodique des clés, compromettant ainsi la propriété de confidentialité persistante (perfect forward secrecy) qui protège les communications passées contre de futures compromissions.

Chacun de ces objectifs correspond à une vulnérabilité spécifique des protocoles et nécessite des contre-mesures adaptées, soulignant l'importance d'une conception globale intégrant l'ensemble de ces aspects sécuritaires. La diversité de ces attaques démontre que la sécurité d'un protocole d'échange de clés ne peut se réduire à la seule protection contre un type particulier de menace [37].

2.1.5 Importance cruciale du modèle de sécurité

Tout protocole cryptographique repose fondamentalement sur un modèle de sécurité précis, qu'il soit explicitement défini ou implicitement supposé. Ces modèles varient considérablement dans leur portée : certains se limitent à contrer les attaques passives les plus basiques, tandis que d'autres intègrent des protections avancées contre les attaques actives, voire même contre les menaces internes émanant de participants légitimes mais malveillants.

Cette diversité de modèles soulève des questions essentielles lors de l'analyse de tout protocole d'accord de clé. Premièrement, il convient d'identifier précisément le profil de l'attaquant que le protocole prétend contrer. Deuxièmement, les hypothèses sous-jacentes concernant la sécurité du canal de communication doivent être clairement établies. Troisièmement,

la garantie de confidentialité persistante (forward secrecy) constitue un critère d'évaluation majeur. Enfin, la résistance du protocole face à des adversaires sophistiqués capables d'observer ou de manipuler simultanément plusieurs sessions doit être rigoureusement examinée.

Ces interrogations fondamentales permettent d'évaluer si le modèle de sécurité adopté correspond effectivement aux menaces réelles auxquelles le protocole sera exposé en situation opérationnelle. Une inadéquation entre le modèle théorique et le contexte pratique peut rendre le système vulnérable, même si ses fondements cryptographiques apparaissent solides. Cette analyse critique constitue donc une étape indispensable dans l'évaluation et la sélection de tout protocole d'accord de clé [38].

2.1.6 Transition vers l'analyse approfondie

Cette présentation des modèles d'attaque établit le cadre conceptuel nécessaire pour les investigations à venir. Elle sert de fondement théorique à l'examen détaillé que nous mènerons dans les prochaines sections, consacrées d'une part aux vulnérabilités spécifiques des protocoles classiques comme Diffie-Hellman, et d'autre part aux particularités des environnements dynamiques impliquant de multiples participants. Notre analyse ultérieure intégrera notamment l'étude d'attaques emblématiques telles que Logjam, qui illustre les risques liés à la réutilisation de paramètres cryptographiques. Nous examinerons également les attaques par rejeu et les vulnérabilités associées aux mécanismes de renouvellement de clés (rekeying), en démontrant comment ces différentes menaces s'articulent avec les modèles théoriques présentés ici. Chaque cas concret sera analysé en tenant compte de son contexte d'application spécifique, mettant ainsi en lumière les interactions complexes entre théorie cryptographique et réalité opérationnelle. Cette approche progressive permettra d'appréhender de manière exhaustive les défis sécuritaires posés par les protocoles d'accord de clé, tout en établissant des ponts entre les concepts fondamentaux et leurs implications pratiques.

2.2 Vulnérabilités dans Diffie-Hellman et ses dérivés

2.2.1 Principes fondamentaux du protocole Diffie-Hellman

Le protocole Diffie-Hellman (DH), révolutionnaire lors de son introduction en 1976, a établi un nouveau paradigme en cryptographie en permettant à deux parties de générer une clé secrète commune sans jamais avoir à l'échanger directement. Son mécanisme s'appuie sur la difficulté computationnelle du problème du logarithme discret au sein d'un groupe cyclique : connaissant les éléments g, g^a et g^b, il est mathématiquement complexe de déterminer g^{ab} sans disposer des exposants secrets a ou b.

Dans son déploiement standard, le protocole suit une séquence claire :

1. Alice et Bob sélectionnent indépendamment leurs secrets respectifs a et b.
2. Ils échangent leurs valeurs publiques g^a et g^b.
3. Chacun calcule localement la clé partagée g^{ab}.

Bien que d'une élégance conceptuelle remarquable, ce protocole présente néanmoins plusieurs faiblesses structurelles significatives. Ces vulnérabilités proviennent soit de limitations inhérentes à sa conception originale, soit de choix d'implémentation inadéquats qui en compromettent la sécurité pratique. Ces failles seront analysées en détail dans les sections suivantes, mettant en lumière l'écart parfois important entre la sécurité théorique et la sécurité opérationnelle des systèmes cryptographiques [14] [39].

2.2.2 Vulnérabilité à l'attaque de l'homme du milieu (MITM)

La version originale du protocole Diffie-Hellman présente une faille majeure par l'absence de tout mécanisme d'authentification intégré. Cette lacune permet à un attaquant actif d'intercepter et de manipuler les échanges de manière totalement transparente pour les participants légitimes.

1. Interception : L'attaquant (Eve) s'interpose dans la communication entre Alice et Bob.

2. Substitution :

- o Eve remplace g^b destiné à Alice par sa propre valeur g^e.
- o Elle substitue également g^a destiné à Bob par la même valeur g^e.

3. Établissement des clés :

- o Alice calcule une clé partagée avec Eve (g^{ae}).
- o Bob établit une clé distincte avec Eve (g^{be}).

4. Contrôle total :

- o Eve peut désormais intercepter tous les messages.
- o Elle les déchiffre, les modifie éventuellement, puis les rechiffre.
- o L'ensemble du processus reste indétectable sans authentification.

Cette vulnérabilité fondamentale met en évidence l'importance critique des mécanismes d'authentification dans tout protocole d'échange de clés. Les versions modernes de Diffie-Hellman intègrent donc des extensions d'authentification pour contrer cette menace [25] [40].

Exemple concret d'une attaque MITM sur Diffie-Hellman :

Scénario :

- **Participants légitimes :** Alice (A) et Bob (B) veulent échanger une clé secrète.
- **Attaquant :** Ève (E) s'interpose sans qu'ils le sachent

Étapes de l'attaque :

1. Échange légitime initial :

o Alice choisit a=3 et envoie g^3 à Bob.

o Bob choisit b=4 et envoie g^4 à Alice.

Mais Ève intercepte ces messages.

2. Intervention malveillante :

o Ève choisit e=2 (son propre exposant secret).

o Elle envoie g^2 à Alice en se faisant passer pour Bob.

o Elle envoie g^2 à Bob en se faisant passer pour Alice

3. Calcul des clés corrompues :

o Alice calcule : $(g^2)^3 = g^6$ (clé Alice-Ève).

o Bob calcule : $(g^2)^4 = g^8$ (clé Bob-Ève).

o Ève calcule les deux clés :

- Avec Alice : $(g^3)^2 = g^6$
- Avec Bob : $(g^4)^2 = g^8$

4. Résultat :

o Alice pense partager g^6 avec Bob.

o Bob pense partager g^8 avec Alice.

o En réalité :

- Alice communique avec Ève (g^6).
- Bob communique avec Ève (g^8).

Conséquences :

- Ève peut :

1. Lire tous les messages en les déchiffrant.
2. Modifier les messages avant retransmission.

3. Injecter de faux messages.

• Les victimes ne détectent rien, car le protocole semble normal.

Cet exemple montre comment l'absence d'authentification permet à un attaquant de prendre complètement le contrôle de la communication sans être détecté.

2.2.3 Risques liés aux groupes cryptographiques inadéquats

La sécurité du protocole Diffie-Hellman dépend fondamentalement de la difficulté à résoudre le problème du logarithme discret au sein du groupe mathématique choisi. Cette sécurité peut être gravement compromise lorsque le groupe sélectionné présente des caractéristiques faibles ou inappropriées [41] [42].

Problématiques des groupes vulnérables :

• **Taille insuffisante :** Les groupes de petite taille (notamment ceux de 512 ou 768 bits) permettent aujourd'hui un calcul réalisable du logarithme discret avec des moyens computationnelles modernes.

• **Paramètres prédéfinis :** L'utilisation répétée des mêmes groupes standards dans différentes implémentations permet à un attaquant d'amortir le coût initial de précalcules.

Mécanisme d'attaque facilité :

Un attaquant peut pré-calculer une table des correspondances entre puissances et logarithmes discrets pour un groupe donné. Ce travail, bien que coûteux initialement, devient extrêmement rentable lorsque le même groupe est utilisé par de nombreuses installations. Une fois cette table établie, le décryptage des échanges devient quasi instantané.

Exemple concret :

Considérons un groupe Zp* où p est un nombre premier de 512 bits. Avec des ressources computationnelles modernes :

1. Précalcule des logarithmes discrets : ˜100 jours sur un cluster.

2. Attaque ultérieure sur chaque session : quelques secondes.

3. Coût amorti sur des milliers de systèmes utilisant le même groupe.

Définition clé - Groupe cyclique :

En cryptographie, un groupe cyclique est une structure algébrique composée d'éléments pouvant tous s'exprimer comme puissance d'un élément générateur g. Dans Zp*, l'ensemble des entiers modulo p premier forme un tel groupe, avec :

- g générateur du groupe.
- Opération : exponentiation modulaire.
- Propriété : difficulté du logarithme discret.

Conséquences pratiques :

Les implémentations modernes doivent impérativement :

- Utiliser des groupes de taille suffisante ($\geq$2048 bits actuellement).
- Générer des paramètres uniques pour chaque installation.
- Éviter les groupes standardisés et trop fréquemment utilisés.
- Privilégier les groupes dont l'ordre contient de grands facteurs premiers.

Cette vulnérabilité a été exploitée de manière spectaculaire par l'attaque Logjam (2015), qui démontra la faisabilité pratique de casser des échanges DH 512 bits en quelques minutes seulement.

2.2.4 L'attaque Logjam et ses implications

L'attaque Logjam, révélée en 2015, exploite une vulnérabilité structurelle des implémentations utilisant des groupes Diffie-Hellman standardisés et réutilisés. Cette attaque sophistiquée démontre comment la réutilisation de paramètres cryptographiques peut compromettre la sécurité d'échanges pourtant théoriquement robustes. Le mécanisme repose sur trois phases distinctes : un précalcul initial coûteux mais unique des logarithmes discrets pour un groupe donné, l'interception des sessions utilisant ce groupe, puis le calcul quasi instantané des clés grâce aux données précalculées.

Cette approche permet à un attaquant bien équipé de casser des clés de 512 bits en quelques minutes seulement, rendant obsolètes de nombreuses implémentations historiques. Les groupes de 1024 bits, bien que plus résistants, sont désormais considérés comme vulnérables lorsqu'ils sont massivement réutilisés. Logjam a particulièrement affecté les protocoles TLS, où la pratique courante d'utiliser des groupes prédéfinis a créé une vulnérabilité systémique. L'attaque exploite souvent une combinaison de faiblesses : la disponibilité de groupes standards largement déployés, la possibilité de forcer un downgrade vers des tailles de clés plus faibles, et l'absence de mécanismes de vérification des paramètres.

Les implications de Logjam ont été profondes dans la communauté cryptographique. L'attaque a mis en évidence les dangers des économies de calcul réalisées par la réutilisation de paramètres, démontrant comment un investissement initial en ressources computationnelles pouvait permettre de compromettre un grand nombre de sessions ultérieures. Elle a également souligné l'importance des tailles de clés suffisantes, conduisant à l'abandon progressif des groupes de moins de 2048 bits dans les implémentations modernes. Cette vulnérabilité historique continue d'informer les meilleures pratiques actuelles en matière de configuration des paramètres cryptographiques et de rotation des clés [43] [44].

2.2.5 Risques liés à la persistance des clés et absence de confidentialité persistante

Certaines implémentations dérivées du protocole Diffie-Hellman ont adopté une pratique dangereuse en réutilisant les mêmes clés privées sur plusieurs sessions. Cette approche, bien que permettant des gains en performance, annule complètement l'avantage crucial de la Perfect Forward Secrecy (PFS). La PFS représente une propriété essentielle en cryptographie moderne garantissant que la compromission future d'une clé privée ne permet pas de déchiffrer rétroactivement les communications passées. Ce mécanisme de protection repose sur l'utilisation systématique de paramètres éphémères uniques pour chaque session [39] [45].

Lorsque cette propriété fait défaut, les conséquences pour la sécurité sont graves. Un attaquant qui parvient à obtenir une clé privée (par exemple grâce à une fuite de données ou une attaque ultérieure) peut alors déchiffrer l'ensemble des échanges précédemment enregistrés. Cette vulnérabilité transforme une brèche ponctuelle en compromission globale de l'historique des communications. Dans des contextes sensibles où la confidentialité des données doit être préservée à long terme (communications diplomatiques, échanges médicaux, secrets industriels), l'absence de PFS peut avoir des conséquences désastreuses, rendant potentiellement caduques des années de protections.

Les implémentations modernes doivent donc impérativement utiliser des variantes éphémères de Diffie-Hellman (DHE ou ECDHE) et proscrire toute réutilisation des paramètres privés entre différentes sessions. Cette bonne pratique, combinée à une rotation régulière des clés, constitue aujourd'hui un standard incontournable pour les protocoles sécurisés [39].

2.2.6 Évolutions et bonnes pratiques issues des vulnérabilités

Les faiblesses identifiées dans les implémentations traditionnelles de Diffie-Hellman ont conduit à des améliorations majeures dans les protocoles modernes. Les variantes contemporaines intègrent désormais systématiquement des mécanismes d'authentification robustes, comme ceux présents dans IKEv2 ou TLS 1.3, éliminant ainsi le risque d'attaques de type "man-in-the-middle". Le passage aux courbes elliptiques (ECDH) a permis d'obtenir une sécurité équivalente avec des paramètres plus courts et des calculs plus efficaces, tout en résistant mieux aux attaques par précalcul.

L'adoption généralisée de clés éphémères (DHE, ECDHE) a rétabli la propriété essentielle de Perfect Forward Secrecy, protégeant les communications passées même en cas de compromission future des clés. Ces évolutions techniques s'accompagnent de meilleures pratiques opérationnelles : utilisation de groupes de grande taille ($\geq$2048 bits), génération de paramètres uniques pour chaque installation, et rotation régulière des clés.

Cependant, la persistance d'implémentations vulnérables rappelle que la sécurité effective dépend autant du déploiement correct que de la robustesse théorique du protocole. L'analyse rétrospective de ces vulnérabilités historiques offre des critères d'évaluation précieux : elle enseigne à distinguer la sécurité théorique d'un algorithme de sa résilience pratique face aux attaques réelles, et souligne l'importance des mécanismes auxiliaires (authentification, gestion des clés, paramétrage) dans la protection globale d'un système cryptographique [46].

2.3 Menaces subtiles dans les protocoles d'échange de clés

Les protocoles d'accord de clé doivent faire face à des menaces souvent négligées qui dépassent les vulnérabilités algorithmiques classiques. Ces attaques exploitent principalement des faiblesses dans l'implémentation ou l'environnement d'exécution. Parmi elles figurent les attaques par canaux auxiliaires qui analysent les fuites d'informations physiques, les failles de gestion de session permettant de réactiver frauduleusement des connexions, et les erreurs d'implémentation contournant les protections théoriques. Ces vulnérabilités sont d'autant plus dangereuses qu'elles échappent souvent aux analyses de sécurité traditionnelles. Leur prévention nécessite une approche globale combinant audits rigoureux, sécurisation de l'environnement d'exécution et gestion stricte du cycle de vie des sessions. Ces menaces rappellent que la sécurité effective dépend autant des détails d'implémentation que de la robustesse théorique du protocole [**?**].

2.3.1 Attaques par rejeu et désynchronisation : une menace insidieuse

Les attaques par rejeu exploitent une vulnérabilité subtile mais potentiellement dévastatrice dans les protocoles d'échange de clés. Ces attaques se produisent lorsqu'un adversaire capture et réutilise des messages valides provenant de sessions antérieures. Le danger réside dans le fait que ces messages, bien que légitimes à l'origine, sont réintroduits dans un contexte inapproprié.

Le mécanisme de ces attaques repose sur l'absence de vérification de la fraîcheur ou de l'unicité des messages. Sans protections adéquates comme des nonces (nombres aléatoires à usage unique) ou des horodatages contrôlés, trois scénarios critiques peuvent se produire :

1. Un ancien message de négociation de clé peut être réinjecté, permettant à un attaquant de rétablir une session expirée avec des paramètres potentiellement compromis.

2. La réutilisation forcée d'une clé précédente peut annuler les bénéfices d'un rekeying récent, réactivant ainsi des vulnérabilités déjà corrigées.

Le protocole peut subir une désynchronisation où les participants perdent leur alignement sur l'état courant de la session, rendant les communications incohérentes ou impossibles.

Les conséquences opérationnelles sont particulièrement graves dans les environnements sensibles. La désynchronisation peut non seulement interrompre les communications, mais aussi créer des opportunités pour des attaques plus sophistiquées en exploitant l'état incohérent des machines. Dans certains cas, elle peut même conduire à une perte complète de l'authenticité des messages, où les participants ne peuvent plus distinguer les communications légitimes des messages falsifiés.

La protection contre ces attaques nécessite une conception minutieuse du protocole, intégrant des mécanismes garantissant à la fois l'unicité et l'actualité de chaque échange. Les implémentations modernes combinent généralement plusieurs couches de protection, incluant souvent des nonces cryptographiques, des horodatages sécurisés et des séquences de message vérifiées, créant ainsi une défense robuste contre ces attaques subtiles mais potentiellement catastrophiques [47].

2.3.2 Analyse de trafic et fuites d'information par métadonnées

Même avec un chiffrement robuste du contenu des messages, l'analyse des métadonnées de communication représente une menace sérieuse pour les protocoles d'échange de clés. Cette attaque passive exploite quatre caractéristiques observables :

1. La fréquence et la périodicité des échanges.
2. La taille des paquets transmis.
3. Le moment d'initiation des connexions.
4. Les motifs et séquences des transmissions.

Ces éléments apparemment anodins permettent à un adversaire persistant de reconstituer des informations sensibles. L'analyse temporelle peut révéler quel participant agit comme initiateur, tandis que les variations de taille des messages trahissent souvent les phases de négociation de clés ou de rekeying. Des schémas répétitifs dans les séquences d'échange peuvent même permettre de deviner la structure sous-jacente du protocole utilisé.

Dans les protocoles de groupe, ces techniques permettent d'identifier les membres ayant des rôles particuliers (comme le coordinateur). Elles facilitent également le repérage des moments critiques où une nouvelle clé est en cours de négociation - instant privilégié pour lancer des attaques ciblées. Plus insidieusement, l'analyse statistique de la taille des messages peut parfois révéler des informations sur la longueur des clés utilisées.

Ces fuites d'information indirectes démontrent que le chiffrement du contenu ne suffit pas à garantir une confidentialité complète. Les protections efficaces contre cette menace incluent le padding des messages (pour uniformiser leur taille), l'ajout de trafic factice, et des mécanismes masquant les motifs temporels des échanges. Ces techniques, bien que coûteuses en bande passante, sont essentielles dans les contextes nécessitant une confidentialité absolue [48].

2.3.3 Risques liés à la persistance des clés et confidentialité compromise

L'utilisation prolongée d'une même clé cryptographique entraîne une vulnérabilité majeure : la perte de confidentialité rétroactive. Ce risque se matérialise lorsqu'un adversaire parvient à compromettre une clé à posteriori, lui donnant accès à l'ensemble des communications passées qui l'utilisaient. Ce défaut, connu sous le terme d'absence de Perfect Forward Secrecy (PFS), représente une faille critique pour les secteurs exigeant une protection durable des données, comme les communications militaires, les échanges industriels stratégiques ou les discussions diplomatiques.

Dans ces contextes sensibles, où la capture ultérieure d'un dispositif constitue un scénario plausible, la persistance des clés peut avoir des conséquences désastreuses. Les protocoles contemporains pallient ce risque en intégrant des mécanismes de renouvellement régulier des clés (rekeying). Cette approche permet de générer systématiquement des clés de session temporaires et uniques, limitant ainsi considérablement l'impact potentiel d'une future compromission.

Cette pratique de rotation des clés, combinée à l'utilisation de paramètres éphémères, forme aujourd'hui un standard incontournable pour les systèmes nécessitant une protection durable de la confidentialité. Elle illustre l'importance d'une gestion dynamique et rigoureuse du cycle de vie des clés cryptographiques dans les architectures sécurisées modernes [49].

2.3.4 Menaces internes : le danger venant de l'intérieur

Les protocoles multiparticipants doivent composer avec une vulnérabilité souvent sous-estimée : la menace interne. Contrairement aux modèles traditionnels qui se concentrent sur les attaquants externes, ces systèmes sont exposés aux risques provenant de leurs propres membres. Un participant légitime peut devenir une menace, que ce soit délibérément (comportement malveillant) ou accidentellement (via un terminal compromis).

Ces acteurs internes disposent d'un avantage considérable : leur position légitime au sein du groupe. Ils peuvent ainsi enregistrer frauduleusement des clés et messages confidentiels, perturber subtilement les phases de négociation, ou exfiltrer des secrets à des entités externes. Le danger est particulièrement aigu dans les environnements où la confiance entre membres est présupposée, comme c'est souvent le cas dans les groupes dynamiques sans mécanisme d'authentification renforcée.

Cette vulnérabilité remet en question le principe de confiance implicite et souligne la nécessité d'architectures résilientes capables de limiter les dégâts causés par un membre compromis, tout en maintenant la fonctionnalité du groupe. Les solutions modernes intègrent des mécanismes de vérification distribuée et de secret partagé pour atténuer ces risques [50] [51].

2.3.5 Faiblesses dans l'authentification initiale

La phase d'authentification initiale constitue souvent le maillon faible des protocoles d'échange de clés, même lorsque leurs fondements cryptographiques sont solides. Trois vulnérabilités principales émergent régulièrement : la validation insuffisante des clés publiques, l'acceptation de certificats non vérifiés, et la possibilité d'injecter des paramètres falsifiés. Ces failles permettent à un attaquant de compromettre complètement l'établissement de la clé en usurpant l'identité d'un participant légitime. Les protocoles modernes pallient ces risques

grâce à des signatures numériques, des certificats X.509, ou des mécanismes PAKE pour les environnements sans infrastructure à clés publiques.

L'analyse de ces différentes menaces révèle une vérité fondamentale : la sécurité réelle d'un protocole ne se réduit pas à son élégance mathématique, mais dépend crucialement de la qualité de son implémentation, de la rigueur de sa gestion et de son adaptation au contexte opérationnel. Une évaluation complète doit ainsi considérer l'ensemble du cycle de vie du protocole - depuis l'initiation des sessions jusqu'à la révocation des secrets - ainsi que les interactions complexes entre ses différents composants. Cette approche holistique est essentielle pour concevoir des systèmes véritablement résilients face à la diversité des menaces actuelles [52] [53].

2.4 Gestion de la sécurité des secrets dans un groupe dynamique

Les protocoles d'échange de clés entre deux parties sont bien établis et relativement simples à analyser. Cependant, leur extension à des groupes dont la composition varie dans le temps présente des défis spécifiques. Dans ce contexte, les mécanismes cryptographiques doivent assurer la sécurité des secrets partagés tout en permettant une gestion flexible des membres.

La particularité des groupes dynamiques réside dans la nécessité de maintenir la confidentialité, l'intégrité et l'authenticité des communications malgré les changements fréquents de participants. Contrairement aux échanges biparticipants, la sécurité ne repose plus seulement sur la robustesse des algorithmes de chiffrement, mais également sur la capacité à gérer efficacement les entrées et sorties des membres.

Cette gestion dynamique implique des mécanismes spécifiques pour :

- Renouveler les clés lors des changements de composition.
- Authentifier les nouveaux participants.
- Révoquer les accès des membres sortants.
- Maintenir la cohérence des échanges.

Les protocoles modernes doivent donc concilier sécurité cryptographique et flexibilité opérationnelle, un équilibre délicat particulièrement important pour les applications collaboratives nécessitant à la fois adaptabilité et protection des données.

2.4.1 Gestion des départs volontaires

Lorsqu'un membre quitte un groupe, il conserve potentiellement l'accès aux communications passées grâce aux clés précédemment partagées. Pour prévenir cette vulnérabilité, les protocoles modernes implémentent un processus de rekeying automatique qui combine trois actions essentielles : invalidation cryptographique du membre sortant, génération d'une nouvelle clé de groupe, et verrouillage de son accès aux échanges futurs. Ce mécanisme doit opérer sans nécessiter de communication individuelle avec chaque membre restant.

2.4.2 Départ du coordinateur

Les protocoles centralisés (comme CLIQUES) présentent une vulnérabilité particulière lors du changement de leader. Ce rôle critique, disposant souvent de secrets intermédiaires, nécessite une procédure de remplacement rigoureuse comprenant :

1. Une élection sécurisée du nouveau leader.
2. Un rekeying complet avec de nouvelles contributions.
3. L'invalidation systématique des sessions existantes.

Cette transition doit garantir la continuité sécuritaire tout en maintenant les fonctionnalités du groupe.

2.4.3 Intégration de nouveaux membres

L'arrivée de nouveaux participants exige des mesures spécifiques pour préserver la backward secrecy. Le protocole doit :

- Générer une nouvelle clé de groupe.
- Initialiser une session cryptographique fraîche.
- Rendre irréversiblement obsolète la clé précédente.

Ces étapes empêchent tout accès rétroactif aux communications antérieures à l'adhésion.

2.4.4 Exclusion des membres compromis

Le cas le plus complexe concerne l'exclusion proactive de membres suspects. La procédure doit :

- Isoler immédiatement le membre malveillant.
- Générer une nouvelle clé excluant sa contribution.
- Notifier les autres participants de la menace.

Cette opération doit résister aux tentatives de sabotage tout en maintenant la cohésion du groupe, notamment face aux attaques visant à perturber la synchronisation des mises à jour de sécurité.

2.4.5 Défis supplémentaires dans les groupes dynamiques

Au-delà des entrées et sorties de membres, les groupes dynamiques rencontrent des problématiques complexes nécessitant des solutions innovantes :

1. Optimisation du rekeying

Les changements fréquents de composition entraînent une charge computationnelle importante. Les protocoles doivent trouver un équilibre délicat entre :

- La fréquence nécessaire des mises à jour de clés.
- La préservation des performances globales du système.
- Le maintien des garanties de sécurité.

2. Adaptation aux changements structurels

Les architectures complexes (hiérarchiques, arborescentes) posent des défis spécifiques lors :

- Des fusions entre sous-groupes.
- Des scissions organisationnelles.
- Des réorganisations de la topologie.

3. Gestion de la scalabilité

L'augmentation de la taille du groupe impacte :

- Les temps de calcul cryptographique.
- La bande passante nécessaire aux mises à jour.
- La complexité de la coordination des membres .

2.4.6 Synthèse et perspectives

La sécurisation des groupes dynamiques représente un enjeu majeur en cryptographie moderne, nécessitant :

- Des mécanismes robustes de confidentialité persistante (forward/backward secrecy).
- Des protocoles résilients aux changements de membres.
- Des architectures adaptables aux différentes topologies.

Cette complexité a motivé le développement de solutions spécialisées comme les pro-

tocoles CLIQUES, qui seront analysés dans le prochain chapitre. Ces approches innovantes tentent de concilier sécurité rigoureuse et flexibilité opérationnelle dans des environnements collaboratifs en constante évolution [54].

2.5 Le rekeying : renouvellement sécurisé des clés partagées

2.5.1 Définition et enjeux du rekeying

Le rekeying constitue le mécanisme fondamental permettant d'assurer la sécurité continue dans les groupes dynamiques. Ce processus de renouvellement des clés répond à trois impératifs critiques : maintenir la confidentialité des échanges après le départ des membres (forward secrecy), protéger les communications passées contre les nouveaux entrants (backward secrecy), et limiter la durée de vie des clés pour réduire l'impact d'une éventuelle compromission. Ces mécanismes sont essentiels pour préserver l'intégrité des communications dans des environnements où la composition du groupe évolue constamment [38].

2.5.2 Déclencheurs du rekeying

Plusieurs situations critiques nécessitent l'activation immédiate du rekeying. Le départ d'un membre, qu'il soit volontaire ou forcé, exige une invalidation rapide de ses accès. L'arrivée de nouveaux participants impose quant à elle des protections contre l'accès rétroactif aux échanges. Les politiques de sécurité préventives implémentent également des rekeyings périodiques, tandis que les détections d'anomalies peuvent déclencher des renouvellements d'urgence. Chaque scénario présente des contraintes opérationnelles spécifiques en termes de rapidité et de synchronisation [38].

2.5.3 Approches techniques de rekeying

Les protocoles modernes ont développé différentes stratégies adaptées aux divers besoins opérationnels. Le rekeying réactif offre une réponse précise aux changements de composition mais demande une coordination complexe. Le rekeying proactif, planifié à intervalles fixes, simplifie la gestion au prix d'opérations parfois inutiles. Dans les architectures hiérarchiques, le rekeying localisé permet d'optimiser les ressources en ne mettant à jour que les sous-ensembles concernés, réduisant ainsi significativement la surcharge computationnelle [38].

2.5.4 Défis d'implémentation

La mise en œuvre efficace du rekeying se heurte à plusieurs obstacles techniques majeurs. La synchronisation précise des membres pendant la transition reste problématique, particulièrement dans les grands groupes où les délais de propagation deviennent significatifs. La gestion des pannes partielles et des pertes de messages critiques exige des mécanismes de récupération robustes. Par ailleurs, la période de transition elle-même crée une fenêtre de vulnérabilité que des attaquants pourraient exploiter pour injecter de fausses informations ou perturber le processus.

2.5.5 Solutions contemporaines

Face à ces défis, les protocoles modernes ont développé des approches innovantes. Les structures arborescentes de clés, comme dans le Tree-based Diffie-Hellman, permettent de réduire considérablement la complexité des mises à jour. Les systèmes de messagerie sécurisée ont popularisé l'usage de clés éphémères générées à chaque session. Des mécanismes sophistiqués de rollback et de chiffrement en cascade assurent une transition fluide entre les anciennes et nouvelles clés. Ces solutions combinées permettent aujourd'hui d'implémenter des rekeyings à la fois sécurisés et performants, même dans des environnements à grande échelle [38].

2.6 Conclusion

Ce chapitre a permis de mener une exploration approfondie des enjeux de sécurité liés aux protocoles d'accord de clé, notamment dans les systèmes multiparticipants. En examinant aussi bien les attaques traditionnelles – qu'elles soient passives ou actives – que les vulnérabilités structurelles plus discrètes, nous avons mis en évidence la diversité et la complexité des risques pouvant affecter ces protocoles, pourtant essentiels à la protection des communications.

L'analyse a montré que la sécurité ne repose pas uniquement sur la robustesse des primitives cryptographiques employées. Elle dépend également de la cohérence architecturale du protocole, de la manière dont les messages sont échangés, de l'authentification des entités, ainsi que des mécanismes de gestion et de renouvellement des clés tout au long du cycle de vie du groupe. Des vecteurs souvent sous-estimés, comme les attaques par rejeu, les désynchronisations, ou encore les compromissions internes, peuvent suffire à xposer des failles critiques. Dans les contextes dynamiques, où la composition des groupes évolue, la question du rekeying est apparue comme un enjeu fondamental. Garantir la confidentialité persistante tout en assurant l'exclusion effective des membres compromis et l'intégration sécurisée des nouveaux participants suppose des protocoles capables de conjuguer résilience, évolutivité et efficacité opérationnelle, y compris dans des environnements asymétriques ou distribués.

En définitive, ce chapitre a posé les fondations conceptuelles nécessaires à l'évaluation rigoureuse des protocoles d'accord de clé. Il souligne l'importance d'une approche globale, qui tienne compte non seulement des aspects cryptographiques, mais aussi de la structure du protocole, de ses flux d'interactions, et de sa comportementalisation face aux menaces réelles.

Fort de ces éléments théoriques, le chapitre suivant adopte une démarche empirique en s'appuyant sur une étude de cas ciblée, celle du protocole A-GDH2, appartenant à la famille CLIQUES. À travers une analyse structurée, nous chercherons à démontrer comment une faille passive d'origine structurelle peut être exploitée par un adversaire silencieux, en dépit de bases cryptographiques réputées sûres. Cette étude visera à illustrer concrètement les limites de certaines conceptions protocolaires, tout en posant les jalons d'une méthode formelle d'audit de sécurité.

Chapitre 3

Etude de cas : évaluation des failles structurelles et de la sécurité du protocole A-GDH2

Ce chapitre se focalise sur l'analyse de la sécurité du protocole A-GDH2 [62], qui fait partie de la famille des protocoles CLIQUES [62]. Ce protocole a été conçu pour établir une clé commune de façon collaborative et sécurisée entre plusieurs participants. Ici, nous examinons une faille structurelle récemment mise en évidence dans des travaux scientifiques. Notre analyse s'appuie sur l'article « Cryptanalyses of CLIQUES using external passive attacks » d'András Steiner et al., qui décrit une attaque passive capable de compromettre la clé finale sans interférer avec les communications ni révéler les secrets individuels des participants. À travers cette étude de cas, nous mettrons en lumière les points faibles de protocoles pourtant considérés comme robustes, tout en posant les bases pour une méthode d'analyse plus générale, développée dans le chapitre suivant.

3.1 Introduction au protocole A-GDH2

3.1.1 Problématique des protocoles de groupe

Les protocoles d'accord de clés traditionnels sont généralement conçus pour des échanges entre deux parties uniquement. Cependant, de nombreux systèmes modernes nécessitent une clé secrète partagée entre plusieurs entités, comme dans les conférences virtuelles sécurisées, les applications de messagerie de groupe ou les environnements collaboratifs.

Ces contextes imposent des exigences supplémentaires : chaque membre doit contribuer à la clé finale, l'identité des participants doit être vérifiée, et le protocole doit s'adapter aux changements dynamiques, tels que l'arrivée ou le départ de membres. Dans ces cas, les approches classiques, comme Diffie-Hellman, atteignent leurs limites. Il devient donc essentiel de développer des protocoles spécifiquement adaptés aux groupes, capables de garantir des niveaux de sécurité équivalents dans un cadre impliquant plusieurs participants.

3.1.2 Présentation de la famille CLIQUES

La famille de protocoles CLIQUES (Contributory Logical Key Establishment System) répond précisément aux besoins des accords de clé dans un groupe. Elle repose sur trois principes fondamentaux : une participation décentralisée, une vérification des échanges et une adaptation aux changements dans le groupe. Chaque membre contribue activement à la création de la clé, limitant ainsi une centralisation excessive et renforçant la transparence cryptographique. Le processus se déroule en trois phases : la contribution, où chaque participant produit et envoie une valeur issue de son secret ; l'accumulation, qui regroupe et combine ces valeurs ; et la distribution, où chaque membre reçoit les informations nécessaires pour obtenir la clé finale. Cette approche organise le protocole selon une structure logique, souvent représentée sous forme d'une chaîne ou d'un arbre de calcul.

3.1.3 Positionnement et fonctionnement général d'A-GDH2

A-GDH2 (Authenticated Group Diffie-Hellman version 2) est une variante du protocole Group Diffie-Hellman, enrichie par des mécanismes d'authentification renforcés. Chaque participant génère une valeur publique à partir de son secret, la signe avec sa clé privée, puis la transmet au leader du groupe. Ce dernier vérifie les signatures, combine les contributions selon une organisation précise et distribue à chaque membre les données nécessaires pour calculer la clé partagée. Les signatures numériques jouent un rôle clé, car elles confirment l'identité de l'expéditeur de chaque message, empêchant ainsi toute tentative d'usurpation. Conçu pour s'adapter aux changements dans la composition du groupe, A-GDH2 maintient sa robustesse face aux modifications tout en préservant la sécurité.

3.1.4 Justification de l'étude de cas

Malgré des bases cryptographiques apparemment robustes, A-GDH2 a récemment fait l'objet d'une analyse approfondie qui a révélé une faille structurelle. Cette vulnérabilité ne repose pas sur une faiblesse des primitives cryptographiques, mais sur une exploitation subtile de la logique interne du protocole. En examinant les relations entre les messages échangés, il est possible de reconstruire la clé sans compromettre les algorithmes utilisés. Cette découverte remet en question l'idée qu'une combinaison de signatures numériques et d'une approche collaborative suffit à garantir une sécurité complète. Ce chapitre cherche à explorer cette faille en détail, à expliquer son fonctionnement, les conditions qui la rendent possible et ses impacts sur la sécurité des protocoles de groupe.

3.2 Présentation du fonctionnement d'A-GDH2

3.2.1 Organisation du groupe et structure du protocole

A-GDH2 est conçu pour fonctionner dans un groupe réunissant plusieurs participants. L'un d'eux, appelé leader, est chargé de coordonner le déroulement du protocole. Chaque membre dispose d'une paire de clés asymétriques pour prouver son identité, et le groupe suit une organisation hiérarchique. Les échanges de messages s'effectuent sur un canal potentiellement non sécurisé, ce qui rend l'authentification indispensable.

3.2.2 Phase de contribution des membres

Chaque participant génère un secret local et calcule une valeur publique, sous la forme (g_i^x), qu'il signe numériquement. Cette contribution est ensuite envoyée au leader. Cette étape garantit que tous les membres participent activement à la création de la clé tout en confirmant leur identité de manière vérifiable.

3.2.3 Phase d'accumulation par le leader

Le leader vérifie les signatures des contributions reçues, puis les combine selon un processus ordonné pour produire une valeur globale. Cette valeur est conçue de manière à ce que chaque participant, en l'utilisant avec son propre secret, puisse obtenir la clé partagée finale.

3.2.4 Phase de distribution des données finales

Le leader prépare un message spécifique pour chaque participant, contenant les informations nécessaires pour déduire la clé, et le signe numériquement avant de l'envoyer. À la fin de cette étape, tous les membres du groupe partagent la même clé secrète, sans qu'elle ait été directement transmise sur le réseau.

3.2.5 Mécanismes d'authentification

Tous les messages clés du protocole sont signés numériquement. Chaque participant vérifie l'authenticité des messages reçus à l'aide des clés publiques correspondantes. Ce mécanisme garantit la fiabilité des échanges et confirme l'identité des expéditeurs.

3.2.6 Résumé du déroulement

Le protocole suit une série d'étapes bien définies : chaque membre contribue à la clé finale, le leader joue un rôle de coordination, et les signatures numériques assurent la sécurité

des échanges. Grâce à cette organisation structurée, A-GDH2 se présente comme une solution efficace pour sécuriser les communications de groupe.

3.3 Objectifs de sécurité visés par A-GDH2

Le protocole A-GDH2 a été conçu pour répondre aux exigences de sécurité des communications en groupe dans des environnements non sécurisés. Contrairement aux protocoles bilatéraux, qui se limitent à un échange de clés entre deux parties, les protocoles de groupe doivent garantir la sécurité d'un processus impliquant plusieurs participants, tout en s'adaptant aux possibles changements dans la composition du groupe. À cet effet, A-GDH2 cherche à offrir un ensemble de propriétés essentielles, assurées quel que soit le nombre de membres ou la structure logique choisie.

La robustesse d'un protocole désigne ici sa capacité à maintenir ses garanties de sécurité face à des perturbations, des attaques malveillantes ou des modifications dans l'organisation du groupe. Un protocole robuste doit résister aux attaques passives (comme l'interception des messages), aux attaques actives (telles que la modification ou l'injection de messages), tout en gérant les erreurs de configuration ou les évolutions naturelles du groupe, comme l'arrivée ou le départ de membres.

Pour atteindre cette robustesse, A-GDH2 intègre plusieurs mécanismes cryptographiques, notamment l'authentification numérique, une participation équitable de chaque membre et une vérification continue de l'intégrité des messages. Les sections suivantes détaillent les objectifs précis que le protocole vise à accomplir pour assurer un échange de clés sécurisé et fiable.

3.3.1 Confidentialité de la clé de groupe

L'objectif principal d'A-GDH2 est de garantir que seules les personnes autorisées à participer au protocole puissent accéder à la clé de groupe partagée. Cela signifie qu'un observateur ou un attaquant, même en interceptant les échanges, ne doit pas pouvoir découvrir cette clé. La confidentialité repose sur la difficulté du problème du logarithme discret, une base mathématique reconnue comme complexe à inverser lorsqu'elle est bien configurée. Toutefois, protéger la confidentialité ne suffit pas si d'autres propriétés de sécurité ne sont pas également assurées.

3.3.2 Authentification des participants

Dans un contexte de groupe, il est crucial de garantir que chaque message vient bien de la personne qui le prétend. Pour cela, A-GDH2 inclut un système d'authentification qui de-

mande que chaque contribution soit signée numériquement. Une signature numérique est une technique cryptographique permettant à un participant de démontrer qu'il est l'auteur d'un message en utilisant sa clé privée. Les autres membres du groupe peuvent ensuite confirmer cette signature grâce à sa clé publique. Ce mécanisme protège contre les tentatives d'usurpation d'identité ou l'ajout de messages frauduleux.

3.3.3 Contributivité

La contributivité est une caractéristique d'un protocole qui oblige chaque participant à apporter une partie secrète à la clé finale. Cela assure qu'aucun individu ne peut décider seul de la valeur de la clé sans l'implication des autres. Dans A-GDH2, cette condition est remplie grâce à un calcul organisé en chaîne ou en arbre, où chaque membre ajoute un exposant secret au processus. Ce secret, qui doit rester confidentiel, joue un rôle essentiel dans la formation de la clé finale. Ainsi, la clé résulte véritablement de la contribution de tous les membres du groupe.

3.3.4 Intégrité des messages

L'intégrité assure qu'un message reste inchangé durant son envoi. Dans A-GDH2, cette protection est renforcée par l'utilisation de signatures numériques. Toute modification, même légère, d'un message signé rendra la vérification de la signature invalide. Cela permet aux membres du groupe de repérer toute altération, qu'elle soit accidentelle ou intentionnelle, comme une substitution, une suppression ou une modification du contenu.

3.3.5 Résilience face aux changements de groupe

A-GDH2 est conçu pour s'adapter aux environnements dynamiques, où les participants peuvent entrer ou sortir du groupe à tout moment. Par résilience, on désigne la capacité du protocole à rester sécurisé malgré les changements dans la composition du groupe. Cela implique qu'il continue d'assurer la confidentialité, l'authentification et l'intégrité, peu importe les modifications dans la structure du groupe. Deux propriétés clés sont visées ici : la forward secrecy et la backward secrecy.

• La forward secrecy (confidentialité persistante) garantit qu'un membre ayant quitté le groupe ne peut pas accéder aux clés créées après son départ, ni déchiffrer les futures communications.

• La backward secrecy (confidentialité rétrospective) empêche un nouveau membre de déchiffrer les messages ou d'accéder aux clés utilisées avant son arrivée.

Pour répondre à ces exigences, A-GDH2 relance entièrement le protocole à chaque changement de composition du groupe, en intégrant de nouveaux secrets aléatoires fournis par les

participants. Ainsi, chaque session repose sur des contributions uniques, empêchant la récupération ou la réutilisation des clés précédentes. Cette solidité structurelle permet à A-GDH2 de maintenir un niveau élevé de sécurité, même lorsque les membres changent souvent.

3.3.6 Résistance aux attaques actives

A-GDH2 ne se contente pas de se protéger contre les attaquants passifs, qui se limitent à écouter ou espionner, mais s'efforce aussi de résister aux attaques actives. Ces attaques se produisent lorsqu'un adversaire intervient directement dans le protocole, par exemple en envoyant de faux messages, en altérant des données ou en perturbant certaines étapes. Pour contrer ces risques, A-GDH2 met en œuvre plusieurs mesures :

• Les signatures numériques empêchent l'insertion de messages non autorisés.

• La structure logique du protocole complique la modification des contributions sans que cela soit détecté.

• Les éléments finaux, uniques à chaque participant, rendent les tentatives de substitution inefficaces.

Grâce à ces mécanismes, A-GDH2 ambitionne de fournir un cadre solide pour établir une clé sécurisée dans les groupes. Cependant, comme nous le verrons plus loin, une faiblesse peut parfois émerger, non pas dans les outils cryptographiques eux-mêmes, mais dans la façon dont ils sont coordonnés au sein du protocole. Cela nous conduit à analyser une faille récemment identifiée dans A-GDH2, qui remet en cause certains des objectifs décrits ici.

3.4 Présentation de la vulnérabilité identifiée dans A-GDH2

Bien que le protocole A-GDH2 ait été développé pour assurer un accord de clé sécurisé entre participants authentifiés, une étude récente a mis en lumière une faille structurelle dans son fonctionnement. Cette vulnérabilité ne découle pas d'une faiblesse dans les algorithmes cryptographiques ou dans le système d'authentification, mais d'une analyse détaillée de la logique interne du protocole et des liens entre les messages échangés. Elle illustre parfaitement une faille dite structurelle.

Cette faiblesse a été découverte en observant passivement et systématiquement le comportement du protocole dans divers scénarios. Les chercheurs ont modélisé A-GDH2 dans un cadre formel, en examinant le flux des messages, les interdépendances entre les opérations et les valeurs intermédiaires calculées par les participants. Leur objectif n'était pas de compromettre les primitives cryptographiques (comme le problème du logarithme discret), mais de

vérifier si la conception du protocole permettait à un attaquant de déduire des informations sensibles sans intervenir directement.

Par structure logique, on désigne ici la séquence prévisible des opérations et la manière dont les valeurs échangées sont liées mathématiquement. C'est précisément cette organisation qui devient exploitable. Les chercheurs ont montré qu'un attaquant, en observant simplement les messages publics (comme les valeurs signées de type (g^{x_i})), peut, grâce aux propriétés de la multiplication modulaire et aux relations entre les puissances, reconstruire la clé finale ou une valeur équivalente sans accéder aux secrets privés des participants. Cette attaque, qualifiée de passive, repose uniquement sur l'analyse des informations visibles lors d'une exécution normale, sans aucune modification des messages.

Ce problème découle de la structure hiérarchique et prévisible du calcul de la clé, qui génère des redondances et des corrélations exploitables entre les messages. Ces redondances signifient que certains messages partagent des éléments communs ou sont basés sur les mêmes calculs, ce qui permet à un attaquant de les comparer et de les analyser pour établir des liens. Par exemple, si les contributions s'accumulent de manière linéaire ou si des valeurs publiques sont réutilisées sans suffisamment de diversification, un attaquant peut établir des équations reliant les puissances reçues et ainsi déduire la clé globale. Cette faille compromet un objectif central du protocole : empêcher un acteur non autorisé de reconstruire la clé de groupe.

Les impacts de cette vulnérabilité sont importants et affectent plusieurs aspects de la sécurité :

• La confidentialité est directement menacée, car un attaquant peut parfois retrouver la clé sans connaître les secrets privés.

• La contributivité est fragilisée, car la clé peut être reconstituée sans la participation effective de tous les membres.

• La robustesse globale du protocole est remise en question, car sa sécurité repose sur des aspects structurels qui ne sont pas suffisamment validés dans ses hypothèses de sécurité.

Cette faille est particulièrement préoccupante, car elle ne provient ni d'une mauvaise configuration ni d'une erreur d'implémentation, mais de la conception même du protocole. Elle montre qu'un système basé sur des primitives cryptographiques fiables peut être vulnérable si les relations logiques entre les messages créent des opportunités exploitables.

Cette découverte souligne une leçon clé : un protocole ne doit pas seulement être évalué sur la robustesse de ses algorithmes, mais aussi sur sa structure, son modèle de calcul et la manière dont il gère les contributions des participants. Elle met en évidence l'importance des attaques structurelles, une catégorie de menaces qui doit être systématiquement considérée lors de la conception et de l'analyse formelle des protocoles cryptographiques.

3.5 Analyse de la vulnérabilité dans le protocole A-GDH2

Cette sous-section examine une faille structurelle critique du protocole A-GDH2. Elle analyse son exploitation et ses implications sur la sécurité.

La vulnérabilité d'A-GDH2 repose sur une faiblesse dans la structure des échanges, permettant à un attaquant passif de compromettre la clé de session. Cette analyse détaille le mécanisme de l'attaque, ses facteurs, et les leçons pour la conception des protocoles.

3.5.1 Modèle d'attaque et hypothèses de départ

Cette sous-section décrit le modèle d'attaque passif ciblant A-GDH2. Elle précise les conditions permettant son exploitation.

L'attaquant, extérieur au groupe, peut intercepter tous les messages circulants entre les participants. Ces messages incluent :

- Les valeurs publiques envoyées par chaque membre, comme (g^{x_i}),
- Les messages intermédiaires transmis par le leader, combinant les contributions,
- Les valeurs finales distribuées pour permettre aux membres de calculer la clé.

Le protocole fonctionne par accumulation exponentielle de secrets, sous forme de puissances imbriquées, comme ($g^{x_1x_2}$), puis ($g^{x_1x_2x_3}$), et ainsi de suite. Ces calculs suivent un ordre précis et prévisible, généralement défini par le rôle du leader.

3.5.2 Étude d'un cas simple : 3 membres (Alice, Bob, Charlie)

Imaginons un groupe avec trois membres : Alice, Bob et Charlie. Chacun génère un secret respectif (x_A), (x_B), (x_C), et produit une contribution publique : (g^{x_A}), (g^{x_B}), (g^{x_C}). Ces valeurs, signées, sont envoyées au leader (par exemple, Alice). Le leader procède à l'accumulation :

1. Il combine (g^{x_B}) avec son propre secret pour obtenir ($g^{x_Ax_B}$),

2. Puis il intègre (x_C) pour produire ($g^{x_Ax_Bx_C}$),

3. Enfin, il partage avec chaque membre les informations nécessaires pour retrouver cette valeur.

L'attaquant, observant ces échanges, collecte toutes les valeurs publiques. Il peut repérer des motifs mathématiques, par exemple :

- Si l'ordre des calculs reste constant,
- Si les valeurs publiques sont liées de manière prévisible,
- Si certaines contributions sont répétées ou redondantes.

Dans certains cas, cela lui permet de retracer les étapes du calcul du leader, même sans connaître les secrets (x_A), (x_B), (x_C). Par exemple, s'il voit ($g^{x_A x_B}$) et (g^{x_C}), il peut supposer que la valeur finale est ($(g^{x_A x_B})^{x_C}$), soit ($g^{x_A x_B x_C}$). Puisque (g^{x_C}) est connu, il peut vérifier si la puissance appliquée à ($g^{x_A x_B}$) correspond à la valeur finale observée, obtenant ainsi indirectement la clé. Cette méthode n'est pas universelle, mais elle est réalisable dans de nombreux cas avec des paramètres standards.

3.5.3 Extension à un groupe de 4 membres

L'attaque s'applique aussi à des groupes plus larges, comme un groupe de quatre membres : Alice, Bob, Charlie et David. Le protocole suit le même principe :

1. Chaque membre publie sa contribution : (g^{x_A}), (g^{x_B}), (g^{x_C}), (g^{x_D}),
2. Le leader combine ces valeurs étape par étape :
   - o D'abord ($g^{x_A x_B}$),
   - o Puis ($g^{x_A x_B x_C}$),
   - o Et enfin ($g^{x_A x_B x_C x_D}$).

Ces valeurs, visibles et signées, permettent à l'attaquant de retracer le processus en comparant les puissances intermédiaires. Par exemple, il peut :

- Identifier que ($g^{x_A x_B x_C x_D}$) est la clé finale,
- Utiliser les valeurs précédentes pour recomposer les étapes.

Même sans connaître les secrets, l'attaquant peut déduire les puissances appliquées et tester mathématiquement s'il peut reproduire les calculs. Bien que cela ne fonctionne pas dans tous les cas, des études ont prouvé sa faisabilité dans des scénarios concrets, comme indiqué dans l'article scientifique à l'origine de cette découverte.

3.5.4 Résultat et implications

À l'issue de cette analyse, l'attaquant parvient à recalculer la clé finale ou une valeur permettant de déchiffrer les communications, sans jamais résoudre de logarithme discret, falsifier de signature ou modifier un message. Il se contente d'exploiter les motifs répétitifs et les relations entre les puissances.

Cette attaque remet en question plusieurs aspects de la sécurité :

- La confidentialité, car un observateur non autorisé peut accéder à la clé,
- La contributivité, car le processus ne garantit plus que seuls les membres légitimes détiennent la clé,
- La robustesse, car l'attaque réussit même lorsque tout semble sécurisé.

Elle montre que la structure d'un protocole est un élément clé de sa sécurité. Une accumulation logique trop prévisible peut être exploitée, même sans faille cryptographique évidente, soulignant l'importance d'analyser la conception globale des protocoles.

3.6 Conditions d'apparition et limites de l'attaque

L'attaque structurelle décrite précédemment, bien qu'importante, ne fonctionne pas dans tous les cas. Sa réussite dépend de conditions spécifiques liées à la configuration et au déploiement du protocole A-GDH2. Cette section détaille les circonstances qui rendent l'attaque possible, ainsi que les situations où elle devient difficile, voire impossible, à exploiter.

3.6.1 Conditions favorisant l'attaque

L'attaque repose sur plusieurs facteurs techniques qui, lorsqu'ils sont réunis, permettent à un attaquant passif de reconstruire la clé de groupe :

• **Structure linéaire et prévisible du protocole :** A-GDH2 impose un ordre fixe pour combiner les contributions des membres. Lorsque le leader traite les secrets un par un dans une séquence prévisible, cela crée des liens mathématiques entre les puissances, exploitables par l'attaquant.

• **Visibilité des messages publics :** Les contributions publiques, comme (g^{x_1}), (g^{x_2}), etc., sont transmises en clair, même si elles sont signées. Cela permet à l'attaquant de les collecter pour modéliser la clé.

• **Réutilisation des puissances intermédiaires :** Si des valeurs intermédiaires, comme ($g^{x_1 x_2}$) ou ($g^{x_1 x_2 x_3}$), sont visibles à plusieurs reprises ou réutilisées dans les calculs distribués, elles créent des corrélations qui facilitent la reconstitution de la clé.

• **Petite taille du groupe ou structure simple :** Dans des groupes réduits, le nombre de combinaisons possibles est limité, ce qui simplifie l'analyse. L'attaque est aussi plus efficace lorsque les échanges sont centralisés par un seul leader.

Ces conditions sont fréquentes dans les implémentations standard d'A-GDH2, surtout lorsque l'efficacité est privilégiée au détriment de la diversification ou du masquage des données.

3.6.2 Facteurs limitant l'attaque

Certaines mesures peuvent réduire, voire bloquer, cette vulnérabilité sans modifier fondamentalement le protocole. Ces choix de conception compliquent le travail de l'attaquant ou rendent son attaque irréalisable :

• **Diversification des bases :** Si chaque participant utilise une base différente ((g_i)) au lieu d'une base commune ((g)), les relations entre les puissances deviennent beaucoup plus difficiles à exploiter.

• **Chiffrement ou masquage des contributions :** En chiffrant les valeurs publiques ou en les envoyant via un canal sécurisé, on empêche l'attaquant d'accéder aux données nécessaires à son analyse.

• **Ajout de nonces aléatoires :** L'intégration de valeurs aléatoires (nonces) dans chaque contribution brise la régularité des calculs, rendant les dépendances imprévisibles.

• **Variation dynamique des structures :** Si les membres changent souvent ou si l'ordre de traitement des secrets varie d'une session à l'autre, l'attaque devient moins fiable.

Ces contre-mesures n'éliminent pas la faille dans la conception théorique du protocole, mais elles en réduisent l'exploitabilité pratique, transformant une attaque passive en une attaque active ou beaucoup plus complexe.

3.6.3 Portée et impact de la faille

Cette vulnérabilité ne découle ni d'une mauvaise configuration ni d'une erreur d'implémentation, mais de la structure logique même du protocole. Cela la rend particulièrement préoccupante, car elle est inhérente à la conception d'A-GDH2.

Dans des conditions d'utilisation courantes, notamment pour des groupes de taille modérée, l'attaque est réaliste, discrète et reproductible. Elle compromet gravement la robustesse du protocole dans des environnements sensibles ou ouverts.

Cependant, la faille n'est pas inévitable. En adoptant des mécanismes de différenciation, de masquage ou de randomisation, il est possible de renforcer la sécurité ou de nécessiter une refonte partielle du protocole.

3.7 Conclusion

Ce chapitre a exploré en détail le protocole A-GDH2, membre de la famille CLIQUES, en examinant ses objectifs de sécurité, sa structure et sa vulnérabilité face à une attaque passive. Après avoir décrit les principes de base de ce protocole, nous avons analysé une faille récemment découverte, qui met en doute sa robustesse logique, malgré l'utilisation de primitives cryptographiques fiables.

L'analyse a révélé une faiblesse structurelle dans A-GDH2 : la façon dont les contributions des membres sont combinées, étape par étape, dans un ordre prédéfini, génère des relations mathématiques exploitables. Un attaquant passif, qui se contente d'observer les puissances échangées (comme (g^{x_i}), ($g^{x_i x_j}$), ($g^{x_i x_j x_k}$)), peut, dans certains cas, reconstruire la clé de groupe sans accéder aux secrets privés des participants. Cette attaque ne repose pas sur une faille cryptographique classique, mais sur la logique d'organisation du protocole lui-même.

L'étude scientifique à l'origine de cette analyse adopte une approche formelle rigoureuse. Les chercheurs montrent que, dans des scénarios courants, cette attaque est réalisable, discrète et compromet plusieurs propriétés essentielles : la confidentialité, la contributivité et la robustesse globale du protocole. Ils notent également que ces failles échappent souvent aux méthodes de vérification traditionnelles, car elles ne touchent pas directement les mécanismes de sécurité sous-jacents.

Nous avons aussi identifié les conditions qui favorisent cette attaque (structure linéaire, visibilité des messages, absence de masquage) et les moyens de limiter son impact (diversification, ajout d'aléa, chiffrement des contributions). Cela souligne qu'un protocole, même bien conçu, peut être vulnérable si sa structure interne est trop prévisible ou manque de diversité.

Cette étude met en évidence une catégorie de menaces souvent sous-estimée dans la conception des protocoles : les vulnérabilités structurelles, qui exploitent la logique du système sans attaquer directement la cryptographie. Elle insiste sur l'importance d'analyser non seulement les algorithmes utilisés, mais aussi la manière dont les messages sont organisés et liés.

Dans le chapitre suivant, nous appliquerons cette même méthode d'analyse à un autre protocole pour déterminer si elle peut révéler des failles similaires, confirmant ainsi la portée générale de cette approche.

Chapitre 4

Analyse comparative par étude de cas sur le protocole GDH.2

Après avoir examiné en profondeur le protocole A-GDH2 dans le chapitre précédent et mis en lumière une faille structurelle exploitable par un adversaire passif, nous poursuivons ici notre réflexion en l'appliquant à un autre protocole appartenant à la même famille : GDH.2 (Group Diffie-Hellman version 2). L'objectif est d'évaluer si ce protocole, bien que reposant sur des fondements similaires, présente une meilleure résistance structurelle face à une attaque passive, en suivant une démarche d'analyse identique : observation des messages visibles et tentative de reconstruction de la clé de groupe par un attaquant silencieux.

Contrairement à A-GDH2, GDH.2 adopte une architecture plus centralisée, dans laquelle un leader joue un rôle actif dans la collecte et la diffusion des données nécessaires à la génération de la clé. Ce choix de conception soulève une question essentielle : cette organisation hiérarchique renforce-t-elle la sécurité du protocole face à des attaques menées par simple observation externe ?

Ce chapitre a pour ambition d'appliquer la même méthodologie que celle utilisée pour A-GDH2 afin de déterminer si GDH.2 présente une faille similaire ou, au contraire, s'il offre une meilleure protection contre ce type de menace. Nous chercherons à vérifier dans quelle mesure un attaquant passif pourrait déduire la clé commune à partir des seules informations rendues publiques. L'analyse permettra également d'établir une comparaison directe avec les résultats obtenus dans l'étude précédente, et d'en tirer des conclusions sur la résilience de GDH.2 face aux attaques structurelles non-interactives.

Enfin, cette seconde étude constituera un test de portabilité de notre méthode d'analyse : elle nous permettra de mesurer son efficacité lorsqu'elle est appliquée à un protocole ayant une structure différente, et d'en identifier les limites ou les conditions de validité selon l'architecture du protocole étudié.

4.1 Présentation du protocole GDH.2

Le protocole GDH.2 (Group Diffie-Hellman version 2) est une adaptation du célèbre protocole de Diffie-Hellman, pensée pour les environnements de communication en groupe. Il s'inscrit dans la famille des protocoles CLIQUES, conçue pour répondre aux défis de l'accord de clé dans des contextes multiparticipants, tout en préservant des propriétés cryptographiques fondamentales telles que la contributivité, la confidentialité, et la compatibilité avec les primitives asymétriques déjà établies.

À la différence d'un protocole d'accord de clé bilatéral, où deux entités échangent leurs secrets de manière équilibrée, GDH.2 reposes sur une architecture asymétrique. L'un des membres du groupe y est désigné comme leader, et occupe un rôle central dans le processus de génération de la clé. C'est lui qui reçoit les contributions individuelles des autres participants, puis les transforme afin de leur retourner les éléments nécessaires à la reconstruction de la clé finale.

Le fonctionnement du protocole s'articule autour de deux phases principales. Dans la phase montante, tous les membres – à l'exception du leader – envoient à ce dernier une valeur publique calculée sous la forme g^{x_i}, où x_i représente leur secret privé et g un générateur commun. Le leader agrège ces valeurs selon un mécanisme multiplicatif propre au protocole, avant d'initier la phase descendante. Durant cette seconde phase, il renvoie à chaque membre une valeur intermédiaire personnalisée, calculée de manière à ce que chaque destinataire, en la combinant avec son propre secret, soit en mesure de reconstituer la même clé de session que tous les autres.

Mathématiquement, la clé finale partagée est de la forme :

$$K = g^{x1x2x\ldots xn}$$

où

- g est le générateur public d'un groupe fini cyclique,
- x1 est le secret privé du leader P_1,
- x2,x3,...,xn sont les secrets privés des autres membres du groupe.

Pour permettre à chaque membre Pi (avec $i \in \{2, ..., n\}$) d'obtenir cette clé, le leader calcule pour chaque Pi la valeur :

$$V_i = g^{x_1 \cdot \prod_{\substack{j=2 \\ j \neq i}}^{n} x_j}$$

où :

- le produit $(_{n,j=2,ji)}xj$ représente la multiplication de tous les secrets des autres membres.

• Vi est donc une valeur intermédiaire propre à chaque membre, transmise en clair.

Ensuite, chaque membre Pi, en possession de son propre secret xi, élève cette valeur à la puissance xi :

$$K = \left(g^{x_1 \cdot \prod_{\substack{j \neq i \\ j \neq 1}} x_j} \right)^{x_i} = g^{x_1 \cdot \left(\prod_{\substack{j \neq i \\ j \neq 1}} x_j \right) \cdot x_i}$$

Or, comme xix_ixi complète le produit partiel, on obtient bien :

$$K = g^{x1x2x3. \ldots Xn}$$

Ainsi, chaque membre reconstruit exactement la même clé de session, sans connaître les secrets des autres participants, et sans avoir besoin de retracer le calcul du leader.

Ce mécanisme permet à l'ensemble des participants d'aboutir à une clé commune identique, tout en garantissant que seuls les membres disposant de leur secret privé respectif peuvent la reconstituer.

Le protocole GDH.2 présente plusieurs avantages cryptographiques majeurs qui en font une solution pertinente pour les environnements multiparticipants :

• Il préserve la confidentialité de la clé partagée, reposant sur la difficulté computationnelle du problème du logarithme discret.

• Il assure la contributivité, chaque participant intervenant activement dans la génération de la clé via sa propre contribution secrète.

• Il offre une résistance naturelle aux attaques passives, dans la mesure où les informations échangées ne sont exploitables que si l'on possède le secret correspondant.

• Il montre également une bonne scalabilité, car l'ajout de nouveaux membres peut se faire sans nécessiter une refonte complète du protocole.

Cependant, certaines limites structurelles doivent être soulignées. Le fonctionnement du protocole repose fortement sur la présence et le bon comportement du leader, ce qui induit une centralisation du calcul et crée un point de fragilité potentiel en cas de compromission ou de départ de ce membre central. De plus, le protocole, dans sa forme de base, ne fournit ni mécanisme d'authentification intégré, ni prise en charge native des modifications dynamiques du groupe, rendant souvent nécessaire l'ajout de composants complémentaires pour répondre à ces besoins.

En définitive, GDH.2 constitue une approche mathématiquement rigoureuse et élégante pour l'établissement d'une clé de session dans un groupe. Sa conception en fait une solution solide pour des environnements où un coordinateur central, tel qu'un serveur ou un chef de session, orchestre les échanges entre plusieurs clients.

4.2 Méthode d'analyse appliquée

Pour évaluer la résistance du protocole GDH.2 face aux attaques passives, nous adoptons ici la même approche méthodologique que celle utilisée dans l'analyse du protocole A-GDH2. Il s'agit de se placer dans la perspective d'un adversaire externe passif, capable d'observer l'intégralité des échanges publics transitant sur le réseau, sans toutefois intervenir dans les communications ni compromettre les secrets privés détenus par les membres du groupe.

La démarche se décompose en plusieurs étapes. En premier lieu, nous procédons à la collecte de tous les messages échangés durant la phase d'établissement de la clé. Dans le cas de GDH.2, le processus suit une structure centralisée : les membres envoient leurs contributions individuelles au leader, qui leur renvoie ensuite des informations dérivées leur permettant de reconstituer la clé finale. Un attaquant passif peut ainsi intercepter les valeurs publiques de type g_{x_i}, transmises par les participants, ainsi que les réponses calculées par le leader.

Dans un second temps, nous examinons les valeurs cryptographiques visibles publiquement, afin de déterminer si leur structure offre une opportunité d'exploitation. L'objectif est d'évaluer si les puissances échangées présentent des motifs ou des relations mathématiques permettant à un observateur de déduire tout ou partie de la clé de session, sans connaître les secrets privés. Cela implique une analyse fine des combinaisons de puissances, ainsi qu'un raisonnement logique basé sur leur agencement et les propriétés du protocole.

Enfin, les résultats de cette tentative sont comparés à ceux obtenus pour A-GDH2, dans le but d'apprécier l'effet de la structure du protocole sur sa sécurité. Cette comparaison nous permettra de voir si la centralisation propre à GDH.2 constitue un facteur de renforcement face aux attaques passives, ou si, malgré cela, une vulnérabilité analogue à celle détectée dans A-GDH2 pourrait apparaître.

4.3 Étude de cas : simulation passive sur GDH.2

Pour analyser de manière plus concrète la résistance du protocole GDH.2 face à une attaque passive, nous mettons en œuvre une simulation complète impliquant quatre participants : P1 (jouant le rôle de leader), P2, P3 et P4. Contrairement à une analyse purement théorique, cette simulation s'appuie sur des valeurs numériques explicites, ce qui permet de suivre pas à pas l'exécution du protocole, tout en évaluant avec précision les capacités d'un adversaire passif à reconstituer la clé de groupe à partir des seuls échanges observables.

Nous utilisons un petit groupe fini :

- Groupe de travail : $\mathbb{Z}_{23}^*$ (groupe multiplicatif mod 23)
- Générateur : g = 5
- Les valeurs secrètes sont :

x1 = 6 (pour P1), x2 = 15 (pour P2), x3 = 13 (pour P3), x4 = 7 (pour P4)

Phase montante :

- P2 $\rightarrow P1 : g^{x2} = 5^{15} \mod 23 = 19$
- P3 $\rightarrow P1 : g^{x3} = 5^{13} \mod 23 = 21$
- P4 $\rightarrow P1 : g^{x4} = 5^{7} \mod 23 = 17$

Calcul de la clé par le leader :

Le leader P possède sa propre valeur secrète x1=6, ainsi que les contributions des autres membres. Il calcule une clé de groupe K en combinant les secrets :

- $K = g^{x1x2x3x4} \mod 23$
- $x1.x2.x3.x4 = 6 * 15 * 13 * 7 = 8190$
- $K = 5^{8190} \mod 23$

Phase descendante :

Le leader P envoie ensuite à chaque membre une valeur intermédiaire personnalisée, leur permettant de reconstruire la clé de session, à condition qu'ils disposent de leur propre secret. Les messages envoyés sont :

- À P : g^{x1x3x4}
- À P : g^{x1x2x4}
- À P : g^{x1x2x3}

Chaque membre peut alors combiner cette valeur reçue avec son secret pour retrouver la clé finale, selon la logique du protocole.

Point de vue de l'attaquant :

L'attaquant observe :

- Les valeurs publiques échangées pendant la phase montante :
  - $g^{x2} = 19$
  - $g^{x3} = 21$
  - $g^{x4} = 17$
- Les fragments de clé envoyés pendant la phase descendante :
  - g^{x1x3x4}

- g^{x1x2x4}
- g^{x1x2x3}

• En revanche, l'attaquant ne connaît pas :

- Le secret x1 du leader
- Les produits croisés (ex : x1x2x3)
- La valeur finale de la clé $K = g^{x1x2x3}$

Conclusion

Cette simulation montre que, malgré l'accès complet aux messages échangés publiquement, un attaquant ne dispose pas des liens mathématiques nécessaires pour reconstruire la clé finale. En effet, les puissances observées sont des produits croisés opaques, qui ne permettent pas d'inférer les secrets individuels ou la structure exacte de la clé. Le cloisonnement des calculs dans le leader, ainsi que la spécificité des messages envoyés à chaque membre, assurent une bonne résistance structurelle du protocole GDH.2 face à une attaque passive.

4.4 Analyse des résultats

L'application de notre méthode d'analyse au protocole GDH.2 a mis en évidence sa résistance face à l'attaque passive qui s'était révélée efficace contre A-GDH2. Grâce à sa structure centralisée, GDH.2 segmente les échanges de manière rigoureuse, chaque message adressé à un participant n'a de sens qu'en conjonction avec le secret privé de ce dernier. Cela empêche tout observateur externe de reconstituer la clé à partir des seules données publiques.

Contrairement à A-GDH2, aucune redondance exploitable n'est observable dans les puissances échangées. Le protocole évite les chaînes de dépendance visibles et ne révèle pas d'intermédiaires partagés entre les membres. Même en interceptant l'ensemble du trafic échangé, un attaquant passif ne peut reconstituer la clé de groupe, faute de points d'appui suffisants dans les messages collectés.

La différence essentielle entre les deux protocoles réside dans leur topologie cryptographique : alors qu'A-GDH2 expose une partie trop importante du processus de calcul sous forme de messages intermédiaires visibles, GDH.2 concentre l'ensemble du traitement au niveau du leader et dissocie les réponses individuelles, ce qui rend l'observation passive inefficace.

Cette analyse confirme que la structure interne du protocole est un facteur déterminant de sa sécurité. Lorsqu'elle est bien conçue, la centralisation du calcul, loin d'être une faiblesse, peut au contraire représenter un avantage stratégique face aux attaques silencieuses.

4.5 Conclusion

Ce chapitre a permis de valider expérimentalement la méthodologie d'analyse passive développée précédemment, en l'appliquant cette fois-ci au protocole GDH.2. À travers une simulation concrète et rigoureusement conduite, nous avons constaté que, bien qu'appartenant à la même famille que A-GDH2, GDH.2 présente une résistance nettement accrue aux attaques passives. Cette robustesse s'explique par la structure hiérarchique du protocole, qui limite considérablement la diffusion d'éléments intermédiaires exploitables.

Même en bénéficiant d'un accès complet aux messages publics échangés, un attaquant passif ne parvient pas à extraire d'informations suffisantes pour reconstruire la clé finale. L'absence de chaînes de puissances corrélées et la forte dépendance des messages aux secrets individuels des membres empêchent toute déduction externe. Cette observation met en évidence que, à partir d'un socle cryptographique commun, des choix de conception protocolaires peuvent conduire à des niveaux de sécurité très contrastés.

En définitive, cette étude confirme que l'analyse passive de la structure est un outil pertinent pour auditer la sécurité des protocoles d'accord de clé. Toutefois, son efficacité n'est pas systématique : elle dépend étroitement de la nature de l'échange, de la visibilité des messages et de la manière dont les dépendances cryptographiques sont organisées entre les participants. Ces constats alimenteront, dans la conclusion générale du mémoire, une réflexion plus large sur les approches possibles pour renforcer la sécurité dans les environnements de groupe dynamiques.

Conclusion générale

Ce mémoire a porté sur l'évaluation de la sécurité des protocoles d'accord de clé multi-participants, avec une attention particulière portée à une catégorie de menaces encore relativement méconnue, les attaques passives de nature structurelle. En adoptant une approche rigoureuse et comparative, notre objectif était de déterminer dans quelle mesure la logique interne d'un protocole, indépendamment de la solidité des primitives cryptographiques utilisées, peut constituer un point de vulnérabilité exploitable par un simple observateur.

L'analyse détaillée du protocole A-GDH2, pourtant fondé sur des principes cryptographiques éprouvés tels que le logarithme discret et les signatures numériques, a mis en lumière une faille préoccupante au niveau de sa structure. En exploitant uniquement les messages échangés publiquement et les relations mathématiques qui en découlent, un attaquant passif est capable, dans certaines conditions, de reconstituer la clé de groupe, sans n'intervenir activement ni compromettre les primitives de base. Cette observation souligne que la sécurité d'un protocole ne repose pas uniquement sur la robustesse des algorithmes, mais aussi sur la manière dont les éléments sont structurés, exposés et combinés durant le déroulement du protocole.

Afin d'évaluer la généralité de cette méthode d'analyse, nous l'avons appliquée à un second protocole de la même famille, GDH.2. Cette seconde étude a révélé une résilience nettement plus forte face aux attaques passives structurelles. Grâce à une architecture centralisée, à des échanges non répétitifs et à une organisation plus compartimentée des opérations, GDH.2 ne laisse pas apparaître de schémas exploitables par un adversaire passif. Ce contraste nous a permis de confirmer que certaines conceptions protocolaires, par leur seule organisation interne, offrent une protection structurelle naturelle, sans qu'il soit nécessaire d'ajouter des mécanismes cryptographiques supplémentaires.

Dans quelle mesure les protocoles d'accord de clé multiparticipants sont-ils vulnérables à des attaques structurelles passives, et comment une analyse formelle de leur logique permet-elle de les anticiper ?

Notre étude a permis de démontrer que certains protocoles de groupe, bien qu'ils s'appuient sur des primitives cryptographiques fiables et soient correctement implémentés, peuvent néanmoins présenter des vulnérabilités structurelles accessibles à un adversaire purement passif. C'est précisément ce que révèle le cas du protocole A-GDH2, dont la prévisibilité des calculs internes et la transparence excessive des puissances échangées ouvrent la voie à une compromission indirecte de la clé de groupe.

À l'inverse, notre analyse du protocole GDH.2 montre qu'une organisation plus rigoureuse et cloisonnée des opérations, ainsi qu'un échange de messages non redondant, permet de résister à cette classe d'attaques. Cela souligne que l'architecture du protocole elle-même joue un rôle déterminant dans sa sécurité globale.

En conclusion, l'approche formelle que nous avons suivie, centrée sur l'étude de la structure interne des protocoles plutôt que sur la seule vérification des primitives cryptographiques, s'avère indispensable. Elle permet de déceler des failles invisibles aux analyses classiques et fournit des pistes concrètes pour améliorer la conception des protocoles futurs, en particulier dans des contextes collaboratifs où les échanges sont fréquents, dynamiques et critiques.

Références

[1] *IDEMIA. Cryptographie. Web Article, n.d. URL.* https://www.idemia.com/fr/cryptographie.

[2] *AMI Gestion. Attaque MITM (Man-In-The-Middle). Web Article, n.d.* https://ami-gestion.fr/attaque-mitm/.

[3] *Segura Security. Encryption for Cybersecurity. Web Article, n.d. URL.* https://segura.security/post/encryption-for-cybersecurity.

[4] *Wikipedia. Key-agreement protocol. Encyclopedia Article, n.d. URL.* https://en.wikipedia.org/wiki/Key-agreement_protocol.

[5] *Kaspersky. Diffie-Hellman Protocol (DH). Encyclopedia Article, n.d. URL.* https://encyclopedia.kaspersky.com/glossary/diffie-hellman-protocol-dh/.

[6] *IONOS. La cryptographie asymétrique. Web Article, n.d. URL.* https://www.ionos.fr/digitalguide/serveur/securite/la-cryptographie-asymetrique/.

[7] *KZero. Key Agreement vs. Key Exchange : Definition. Web Article, n.d. URL.* https://kzero.com/resources/glossary/key-agreement-vs-key-exchange-/definition-2/.

[8] *Cortier, V.Protocoles cryptographiques : sécurité et vérification formelle. Disponible en ligne :.* https://members.loria.fr/VCortier/files/Papers/protocolesTSI.pdf/.

[9] *Thales Group. What is an Asymmetric Key or Asymmetric Key Cryptography ? Web Article,n.d. URL.* https://cpl.thalesgroup.com/faq/key-secrets-management/what-asymmetric-key-or-asymmetric-key-cryptography.

[10] *Molin, P. Cryptographie Asymétrique. Lecture Notes, n.d. URL.* https://webusers.imj-prg.fr/~pascal.molin/cours/crypto/cours_asymetrique.html.

[11] *Kaski, E. ECC Attacks. Code Repository, n.d. URL.* https://github.com/elikaski/ECC_Attacks.

[12] *Nakov, S. Key Exchange. Web Article, n.d. URL.* https://cryptobook.nakov.com/key-exchange.

[13] *Boichut, Y. Protocoles de Sécurité. Lecture Notes, 2009. URL.* https://www.univ-orleans.fr/lifo/Members/Yohan.Boichut/Enseignement/Securite/Securite_6_protocoles_securite_2008_2009.pdf.

[14] *NinjaOne. Qu'est-ce qu'un échange de clés Diffie-Hellman ? Web Article, n.d. URL.* https://www.ninjaone.com/fr/it-hub/endpoint-security/qu-est-ce-qu-un-echange-de-cles-diffie-hellman/.

[15] *Wikipedia. Man-in-the-middle attack. Encyclopedia Article, n.d. URL.* https://en.wikipedia.org/wiki/Man-in-the-middle_attack.

[16] *Ribeiro, R.V. Slides de Cryptographie. Lecture Notes, n.d. URL.* https://www.dcc.fc.up.pt/~rvr/resources/Criptografia/slides12.pdf.

[17] *Wikipedia. Elliptic-curve Diffie–Hellman. Encyclopedia Article, n.d. URL.* https://en.wikipedia.org/wiki/Elliptic-curve_Diffie%E2%80%93Hellman.

[18] *IBM. EC Diffie-Hellman (CSNDEDH). Web Article, n.d. URL.* https://www.ibm.com/docs/en/linux-on-systems?topic=keys-ec-diffie-hellman-csndedh.

[19] *Sectigo. Chiffrement RSA vs DSA vs ECC. Web Article, n.d. URL.* https://www.sectigo.com/fr/ressources/chiffrement-rsa-vs-dsa-vs-ecc.

[20] *Wikipédia. Protocole cryptographique. Encyclopedia Article, n.d. URL.* https://fr.wikipedia.org/wiki/Protocole_cryptographique.

[21] *Cárdenas, J.J., Rosero, J., amp ; Vargas, A. A Cryptographic Authentication Protocol with ECC for IoT Applications. Journal Article, 2023. URL.* https://www.oaepublish.com/articles/jsss.2023.16.

[22] *OWASP. A02 :2021 – Cryptographic Failures. Web Article, 2021. URL.* https://owasp.org/Top10/fr/A02_2021-Cryptographic_Failures/.

[23] *Tangem. Side-Channel Attack. Web Article, n.d. URL.* https://tangem.com/fr/glossary/side-channel-attack/.

[24] *Bibmath. Canal auxiliaire. Encyclopedia Article, n.d. URL.* https://www.bibmath.net/crypto/index.php?ction=affiche&quoi=chasseur/canalauxiliaire.

[25] *Stack Overflow. How Does the Man-in-the-Middle Attack Work in Diffie-Hellman ? Web Article, n.d. URL.* https://stackoverflow.com/questions/10471009/how-does-the-man-in-the-middle-attack-work-in-diffie-hellman.

[26] *Kaspersky. Replay Attack. Web Article, n.d. URL.* https://www.kaspersky.fr/resource-center/definitions/replay-attack.

[27] *Hideez. What is a Replay Attack ? Web Article, n.d. URL.* https://hideez.com/fr/blogs/news/what-is-a-replay-attack.

[28] *F5 Networks. Information about the Logjam Vulnerability (K000148343). Web Article, n.d. URL.* https://my.f5.com/manage/s/article/K000148343.

[29] *Sinha, V.B. Diffie-Hellman Attacks. Code Repository, n.d. URL.* https://github.com/vbsinha/Diffie-Hellman-Attacks.

[30] *Hostwinds. SSL vs TLS Protocols. Web Article, n.d. URL.* https://www.hostwinds.fr/blog/ssl-vs-tls-protocols.

[31] *DMARC Report. Understanding TLS Downgrade Attacks and How MTA-STS Mitigates Them. Blog Article, n.d. URL.* https://dmarcreport.com/blog/understanding-tls-downgrade-attacks-and/how-mta-sts-mitigates-them/.

[32] *Meegle. Cryptographic Vulnerabilities. Web Article, n.d. URL.* https://www.meegle.com/en_us/topics/cryptography/cryptographic-vulnerabilities.

[33] *ISEO Blue. ISO 27001 Control 8.24 : Use of Cryptography. Web Article, n.d. URL.* https://iseoblue.com/post/iso-27001-control-8-24-use-of-cryptography/.

[34] *Avanista. La cybersécurité post-quantique : une nouvelle menace à prendre en compte. Web Article, n.d. URL.* https://www.avanista.fr/actualites/106-la-cybersecurite-post-quantique/-une-nouvelle-menace-a-prendre-en-compte.

[35] *Singh, R. Analyse de la sécurité et de la performance des protocoles cryptographiques post-quantiques. Ph.D. Dissertation, Université de Limoges, 2021. URL.* https://theses.hal.science/tel-03326960v1/file/2021LIMO0037.pdf.

[36] *Barker, E. Recommendation for Key Management : Part1–General. NISTSP80057pt.1rev.5,May2020. DOI :10.6028/NIST.SP.80057pt1r5.* https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf.

[37] *Vesterås, B. Analysis of Key Agreement Protocols. Master's Thesis, NTNU, 2015.* https://ntnuopen.ntnu.no/ntnu-xmlui/bitstream/handle/11250/143863/Vesteras%20-%20Analysis%20of%20key%20agreement%20protocols.pdf.

[38] *Barker, E. Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography. Special Publication, 2023. URL.* https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Cr2.pdf.

[39] *Alrawais, A., Hu, C., amp ; Cheng, X. Fog Computing for the Internet of Things : Security and Privacy Issues. Conference Paper, 2015. URL.* https://dl.acm.org/doi/pdf/10.1145/2810103.2813707.

[40] *Kanwar, A., amp ; Singh, D. Survey on Key Agreement Protocols in Wireless Sensor Networks. Journal Article, 2019. URL.* https://theijes.com/papers/vol8-issue8/Series-2/H0808028894.pdf.

[41] *Wikipédia. Logarithme discret. Encyclopedia Article, n.d. URL.* https://fr.wikipedia.org/wiki/Logarithme_discret.

[42] *Adrian, D., Bhargavan, K., Durumeric, Z., Gaudry, P., Green, M., Halderman, J.A., Heninger, N., Springall, D., Thomé, E., Valenta, L., VanderSloot, B., Wustrow, E., Zanella-Béguelin, S., amp ; Zimmermann, P. Imperfect Forward Secrecy : How Diffie-Hellman Fails in Practice. Technical Report, 2015. URL.* https://weakdh.org/imperfect-forward-secrecy.pdf.

[43] *Hafizul Munaim, M.F., Azmi, S., amp ; Mohd Shariff, S.S. Comparative Study on Key Agreement Protocol in Wireless Sensor Network. Journal Article, 2022. URL.* https://jtec.utem.edu.my/jtec/article/view/3334/3469.

[44] *Number Analytics. Ultimate Guide to Logjam Attack. Web Article, n.d. URL.* https://www.numberanalytics.com/blog/ultimate-guide-to-logjam-attack.

[45] *Kaliski, B.S. The Diffie-Hellman Key Agreement Method. Conference Paper, 2003. URL.* https://link.springer.com/content/pdf/10.1007/3-540-39200-9_16.pdf.

[46] *Barker, E. Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography. Special Publication, 2019. URL.* https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Br2.pdf.

[47] *Wikipédia. Attaque par rejeu. Encyclopedia Article, n.d. URL.* https://fr.wikipedia.org/wiki/Attaque_par_rejeu.

[48] *IS Decisions. Fuite de données : comment se produisent-elles ? Blog Article, n.d. URL.* https://www.isdecisions.com/fr/blog/securite-des-donnees/fuite-de-donnees.

[49] *Gantz, S.D., amp ; Philpott, D.R. Guide to Protecting the Confidentiality of Personally Identifiable Information (PII). Special Publication, 2010. URL.* https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-122.pdf.

[50] *CrowdStrike. Insider Threat Indicators. Web Article, n.d. URL.* https://www.crowdstrike.com/fr-fr/cybersecurity-101/identity-protection/insider-threat-indicators.

[51] *Proofpoint. Insider Threat. Web Article, n.d. URL.* https://www.proofpoint.com/fr/threat-reference/insider-threat.

[52] *Yassin, W., Ismail, A.S., Ahmad, R.B., amp ; Nordin, M.J. Detecting Insider Threats Using Rule-Based Anomaly Detection Technique. Conference Paper, 2017. URL.* https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8245777.

[53] *Lafourcade, P., Lombard-Platet, V., amp ; Puche, R. Formal Analysis of a Group Key Exchange Protocol. Conference Paper, 2012. URL.* https://inria.hal.science/hal-01534770/file/978-3-642-31540-48_Chapter.pdf.

[54] *Rescorla, E., Oku, K., Sullivan, N., amp ; Thomson, M. The Messaging Layer Security (MLS) Protocol. RFC, 2024. URL.* https://datatracker.ietf.org/doc/html/rfc9420#section-7.3.

[55] https://tuxcare.com/blog/heartbleed-bug/.

[56] https://www.heartbleed.com/.

[57] https://safe.security/wp-content/uploads/heartbleed-attack.pdf.

[58] https://digitalcommons.onu.edu/student_research_colloquium/2025/Papers/21/.

[59] https://www.ioriver.io/terms/heartbleed-vulnerability.

[60] https://knowledge.digicert.com/quovadis/ssl-certificates/ssl-general-topics/what-is-heartbleed.html.

[61] *Wikipedia. Station-to-Station protocol. Encyclopedia Article, n.d. URL.* https://en.wikipedia.org/wiki/Station-to-Station_protocol.

[62] *Camazón Portela, D., Otero Sánchez, Á., amp ; López-Ramos, J. A. (2024). Cryptanalysis of Ateniese–Steiner–Tsudik-Authenticated Group Key Management Protocol. Applied Sciences, 14(18), 8179.*

Résumé

Ce mémoire traite de la sécurité des protocoles d'accord de clé dans les environnements multi participants, en mettant l'accent sur les vulnérabilités structurelles qui peuvent survenir indépendamment de la solidité des primitives cryptographiques. Après avoir rappelé les fondements de la cryptographie, les différents types d'attaques et les propriétés de sécurité attendues, l'étude se concentre sur l'analyse formelle de deux protocoles de la famille CLIQUES : A-GDH2 et GDH.2.

Une méthodologie d'analyse passive est développée afin de simuler la position d'un attaquant silencieux observant les messages publics échangés au cours de l'établissement de la clé. Cette approche est d'abord appliquée au protocole A-GDH2, pour lequel une vulnérabilité structurelle est identifiée, permettant à un attaquant d'en déduire la clé partagée. L'analyse est ensuite étendue au protocole GDH.2, dans lequel la structure hiérarchique centralisée se révèle plus résistante face à ce type d'attaque.

Les résultats montrent que la sécurité d'un protocole dépend autant de son organisation logique que de la robustesse de ses mécanismes cryptographiques. Le mémoire conclut en soulignant la nécessité d'intégrer l'analyse structurelle dans les processus de vérification de sécurité, notamment pour les systèmes dynamiques où la composition du groupe peut varier dans le temps.

Mot clés : cryptographie, accord de clé, protocole de groupe, attaque passive, A-GDH2, GDH.2, vulnérabilité structurelle, sécurité des communications, CLIQUES, rekeying.

Abstract

This thesis explores the security of key agreement protocols in multiparty communication environments, with a focus on structural vulnerabilities that may arise independently of cryptographic algorithm strength. After outlining the fundamentals of cryptography, attack models, and expected protocol properties, the study investigates two group key agreement protocols from the CLIQUES family : A-GDH2 and GDH.2.

A passive analysis methodology is developed, simulating the behavior of a silent external observer intercepting public messages exchanged during the key establishment phase. This method is first applied to A-GDH2, where a structural weakness is identified, enabling an attacker to reconstruct the shared group key. The same methodology is then applied to GDH.2, whose centralized structure proves more resilient to such passive attacks.

The findings demonstrate that a protocol's security depends not only on its cryptographic primitives but also on its structural design. The thesis concludes by advocating for the integration of structural analysis into protocol verification processes, particularly for dynamic systems where group membership frequently changes.

Keywords : cryptography, key agreement, group protocol, passive attack, A-GDH2, GDH.2, structural vulnerability, secure communication, CLIQUES, rekeying.