УДК 004.421

Василь ПОМАЗИБІДА, Сергій КУЛИНА

Західноукраїнський національний університет

АЛГОРИТМИ ГОМОМОРФНОГО ШИФРУВАННЯ ДЛЯ БЕЗПЕЧНИХ ХМАРНИХ ОБЧИСЛЕНЬ

Вступ. У сучасному цифровому світі хмарні обчислення стали фундаментальною технологією для зберігання та обробки величезних масивів даних. Однак передача чутливої інформації - як-от медичні записи, фінансові дані чи комерційна таємниця - стороннім провайдерам створює значні ризики для безпеки та конфіденційності. Традиційне шифрування захищає дані під час зберігання та передачі, але вимагає їх розшифрування для будь-якої обробки, роблячи їх вразливими у хмарному середовищі.

Саме тому дослідження алгоритмів гомоморфного шифрування є надзвичайно важливим, оскільки вони дозволяють виконувати обчислення (наприклад, аналіз чи машинне навчання) безпосередньо над зашифрованими даними. Це дає змогу використовувати потужні ресурси хмари для обробки інформації, не розкриваючи при цьому самі дані, та є ключем до побудови справді безпечних хмарних сервісів, що гарантують повну конфіденційність.

Метою дослідження є підвищення рівня безпеки та ефективності обробки конфіденційних даних шляхом розробки або вдосконалення алгоритмів гомоморфного шифрування, що дозволяють збалансувати високий рівень захисту інформації з прийнятними обчислювальними витратами.

1. Дослідження існуючих алгоритмів гомоморфного шифрування

Дослідження існуючих алгоритмів гомоморфного шифрування зазвичай починається з їх класифікації за підтримуваними операціями та рівнем складності. Історично першими з'явилися частково гомоморфні системи (PHE), такі як Paillier (дозволяє необмежену кількість операцій додавання) або "чистий" RSA (дозволяє необмежену кількість операцій множення). Вони є обчислювально ефективними, але їхня функціональність обмежена вузькоспеціалізованими завданнями, як-от безпечне голосування чи агрегація статистичних даних. Наступним кроком стали дещо гомоморфні схеми (SHE), які підтримують обмежену кількість і додавань, і множень. Їхня головна проблема — неконтрольоване "зашумлення" даних: кожна операція збільшує "шум" у шифротексті, і після досягнення певного, заздалегідь визначеного порогу, дані стають неможливими для коректного розшифрування.

Сучасні дослідження зосереджені переважно на повністю гомоморфних (FHE) схемах, які вирішують проблему "шуму" за допомогою ресурсоємної операції "перешифрування" (bootstrapping). На практиці домінують кілька ключових "родин" алгоритмів, що базуються на проблемі складності (Ring) LWE. Схеми, як-от BGV та BFV, чудово підходять для точних обчислень над цілими числами (арифметичні схеми), що корисно для запитів до баз даних. З іншого боку, схема CKKS стала де-факто стандартом для прикладного машинного навчання та аналізу даних, оскільки вона працює з приблизними (дійсними) числами. Окремо стоять схеми TFHE/FHEW, оптимізовані для надшвидкого

перешифрування, що робить їх ідеальними для оцінки булевих схем. Таким чином, фундаментальна проблема, яку аналізує дослідження, — це компроміс: не існує єдиного "найкращого" алгоритму, і вибір (BGV, BFV, CKKS чи TFHE) залежить від конкретного хмарного завдання, при цьому всі вони все ще мають значні накладні витрати на продуктивність та розмір даних.

2. Розробка алгоритму гомоморфного шифрування

Варто зазначити, що гомоморфне шифрування — це не один конкретний алгоритм, а радше криптографічний протокол, що описує взаємодію між власником даних та обчислювальним середовищем. Розглянемо загальні кроки цього процесу на прикладі однієї з сучасних FHE-схем (як-от BFV або CKKS). Цей процес завжди включає щонайменше двох учасників: Клієнта (який володіє даними та секретним ключем) і Сервера (який виконує обчислення, маючи лише публічний ключ). Етапи роботи гомоморфного протоколу:

Крок 1. Ініціалізація та генерація ключів. На цьому етапі клієнт готує криптографічну систему та обирає набір криптографічних параметрів (наприклад, ступінь поліноміального кільця N, розмір модулів q і t). Від цих параметрів залежить рівень безпеки та "глибина" обчислень (скільки операцій можна виконати до того, як "шум" стане занадто великим).

Клієнт генерує три типи ключів:

– Секретний ключ (SK) - це приватний ключ, який клієнт нікому ніколи не передає. Він єдиний, що здатен розшифрувати дані.

– Публічний ключ (PK), який використовується для шифрування даних.

– Ключі оцінки (Evaluation Keys) - це спеціальний набір публічних даних (наприклад, "ключі перелінеаризації"), які необхідні Серверу для виконання складних операцій, найчастіше — множення шифротекстів.

Після чого клієнт надсилає Публічний ключ та Ключі оцінки на хмарний Сервер. Секретний ключ залишається у Клієнта.

Крок 2. Шифрування та передача даних.

Клієнт бере свої конфіденційні дані. Залежно від схеми (наприклад, BFV чи CKKS), дані кодуються у спеціальний формат (наприклад, у поліноми). Використовуючи свій Публічний ключ, Клієнт шифрує підготовлені дані та отримує шифротекст.

Клієнт надсилає зашифровані дані на сервер для зберігання та обробки.

Крок 3. Гомоморфна оцінка.

Сервер не має Секретного ключа і не може бачити дані. Сервер отримує від Клієнта (або авторизованого користувача) програму, яку потрібно виконати над даними. Ця програма має бути представлена у вигляді арифметичної схеми (послідовності операцій додавання та множення).

Сервер використовує властивості схеми для виконання обчислень. Кожна операція (особливо множення) додає до шифротексту "шум". Якщо "шум" перевищить певний поріг, дані буде неможливо розшифрувати. Сервер використовує Ключі оцінки для керування цим шумом (наприклад, перелінеаризація після множення).

Якщо програма дуже складна і "шум" стає критичним (у FHE-схемах), Сервер може виконати процедуру bootstrapping. Це операція, яка гомоморфно

"розшифровує" і "зашифровує" дані заново, використовуючи надані Клієнтом ключі. Вона "очищує" шифротекст від шуму, дозволяючи виконувати необмежену кількість операцій.

Виконавши всю програму, Сервер отримує кінцевий шифротекст.

Крок 4. Розшифрування результату.

Сервер надсилає зашифрований результат назад Клієнту.

Клієнт використовує свій Секретний ключ (SK) для розшифрування.

В результаті Клієнт отримує фінальний відкритий текст, який дорівнює результату обчислень, так, ніби вони виконувались на його власних незашифрованих даних.

Ефективність алгоритмів гомоморфного шифрування є головним компромісом та ключовим викликом для їхнього практичного впровадження. Хоча вони пропонують теоретично ідеальний рівень безпеки, дозволяючи обчислення на зашифрованих даних, їхня практична ефективність наразі залишається низькою порівняно з традиційною обробкою у відкритому вигляді. Це виражається у трьох основних аспектах: обчислювальна складність, роздуття даних та складність реалізації.

Висновок. На сучасному етапі розвитку, повністю гомоморфне шифрування не є ефективним для загальноцільових хмарних обчислень. Воно залишається нішевою технологією, ефективність якої є прийнятною лише для вузькоспеціалізованих завдань, де вимоги до конфіденційності є абсолютними і переважають будь-які витрати на продуктивність. Це можуть бути, наприклад, прості статистичні запити до медичних баз даних, приватне об'єднання фінансових наборів даних або виконання простих моделей машинного навчання.

Таким чином, головний напрямок досліджень сьогодні — це не стільки розробка нових криптографічних схем, скільки оптимізація існуючих: створення спеціалізованого апаратного забезпечення (FPGA, ASIC) для прискорення гомоморфних операцій та розробка кращих компіляторів і бібліотек, які б автоматизували складний процес оптимізації програм під FHE.

Перелік використаних джерел.

1. Dunuwila R. M. T. R., Senevirathne T., Weerasinghe P., Kankanamge C. A Comprehensive Survey on Fully Homomorphic Encryption for Cloud Computing: A Practical Perspective. IEEE Access. 2022. Vol. 10. P. 111494–111521.

2. Al-Khafaji S. A. G. G. N., Al-Naji A., Mohammed A. H. A survey of privacy-preserving SQL queries using homomorphic encryption in cloud databases. Journal of Cloud Computing. 2023. Vol. 12. Art. 48. DOI: 10.1186/s13677-023-00478-4.

3. Chillotti I., Gama N., Gkoulalas-Divanis A., Poly F. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. Journal of Cryptology. 2021. Vol. 34. № 4. Art. 28. DOI: 10.1007/s00145-021-09404-8.

4. Ghani M. K. A., Yusoff Z., Yunus M. A. M., Ariffin A. Homomorphic encryption in cloud computing: challenges and future directions. Multimedia Tools and Applications. 2023. Vol. 82. P. 3673–3697. DOI: 10.1007/s11042-022-13237-y.

5. Chen H., Ma M., Cui T., Han M., Wang Y. Efficient privacy-preserving machine learning framework in cloud computing using homomorphic encryption. Information Sciences. 2022. Vol. 609. P. 1007–1020. DOI: 10.1016/j.ins.2022.07.126.