

EFFICIENT IMPLEMENTATION OF HOMOMORPHIC ENCRYPTION

by Johannes Mono

Cryptography originally had one goal in mind: encrypting messages and ensuring the confidentiality of the encrypted data. Since then, it branched out to a variety of other applicable areas. One such area was first envisioned in the late 1970s under the name privacy homomorphism, a hopeful possibility of arbitrary computations on encrypted data. Bringing it to life proved to be a tough challenge for many years, until finally, in 2009, Craig Gentry introduced the first ever encryption scheme for arbitrary privacy homomorphisms, today known as homomorphic encryption. Homomorphic encryption has never been more relevant, especially since data analysis is becoming ever more important and attracting interest from academia and industry alike. Due to great research efforts, homomorphic encryption has seen continued theoretical and practical improvements. Nevertheless, a gap between theory and practice remains. This thesis further narrows the gap between mathematical theory and practical deployment of homomorphic encryption, with a particular focus on improving the performance and reducing the complexity of implementations.

In the first part, we investigate how hardware, algorithms, parameters, and use cases interact to improve performance of homomorphic encryption. We analyze BGV-like schemes, a widely-used class of homomorphic schemes, and particularly focus on key switching, their main bottleneck. Our analysis delivers new theoretical insights, reduces the computational workload, and introduces new guidelines for parameter selection. Furthermore, we advance parameter selection with hardware-friendly approaches and implement and optimize a homomorphic use case. In the second part, we introduce several tools to reduce complexity for implementations: We propose new optimizations for high-level tools, automatically handle the intricacies of parameter selection for developers, and simplify low-level scheme interfaces via new algorithmic tools. Altogether, this thesis improves performance, reduces implementation complexity, and advances the current boundaries of homomorphic encryption in theory and practice.

EFFICIENT IMPLEMENTATION OF HOMOMORPHIC ENCRYPTION

von Johannes Mono

Ursprünglich hatte die Kryptographie nur ein Ziel: Nachrichten zu verschlüsseln und die Vertraulichkeit der verschlüsselten Daten zu schützen. Seitdem hat sie sich in viele weitere Anwendungsbereiche verbreitet. Einer dieser Bereiche wurde erstmals Ende der 1970er Jahre erwähnt unter dem Namen Privatsphäre-Homomorphismus, verbunden mit der Hoffnung mit verschlüsselten Daten beliebig rechnen zu können. Die Umsetzung erwies sich lange Zeit als große Herausforderung, bis Craig Gentry 2009 das erste Verschlüsselungsverfahren für Privatsphäre-Homomorphismen vorstellte, heute bekannt als Homomorphe Verschlüsselung. Homomorphe Verschlüsselung war noch nie so relevant wie heute, insbesondere da Datenanalyse immer wichtiger wird und in Wissenschaft und Industrie auf großes Interesse stößt. Aufgrund großer Forschungsanstrengungen hat die Homomorphe Verschlüsselung kontinuierlich theoretische und praktische Verbesserungen erfahren. Dennoch bleibt eine Lücke zwischen Theorie und Praxis bestehen. Diese Dissertation verringert die Lücke zwischen mathematischer Theorie und praktischem Einsatz der Homomorphen Verschlüsselung und fokussiert sich insbesondere darauf, die Leistung zu verbessern und die Komplexität von Implementierungen zu reduzieren.

Im ersten Teil untersuchen wir, wie Hardware, Algorithmen, Parameter und Anwendungsfälle interagieren, um die Leistung der Homomorphen Verschlüsselung zu verbessern. Wir analysieren BGV-ähnliche Schemata, eine weit verbreitete Klasse homomorpher Schemata, und konzentrieren uns insbesondere auf den Schlüsselwechsel, ihren größten Kostenfaktor. Unsere Analyse liefert neue theoretische Erkenntnisse, reduziert den Rechenaufwand und führt neue Richtlinien für die Parameterauswahl ein. Darüber hinaus verbessern wir die Parameterauswahl durch hardwarefreundliche Ansätze und implementieren sowie optimieren einen homomorphen Anwendungsfall. Im zweiten Teil stellen wir mehrere Werkzeuge vor, um die Komplexität für Implementierungen zu reduzieren: Wir schlagen neue Optimierungen für High-Level-Werkzeuge vor, automatisieren die Feinheiten der Parameterauswahl und vereinfachen Low-Level-Schema-Schnittstellen durch neue algorithmische Werkzeuge. Insgesamt verbessert diese Dissertation die Leistung, reduziert die Implementierungskomplexität und verschiebt dadurch die aktuellen Grenzen der Homomorphen Verschlüsselung in Theorie und Praxis.