Group Digital Signature on a Mobile Cloud With Signcryption and EdDSA

SIIT SINCE 1992

TABLE OF CONTENTS

01 Introduction

02 Key Concept

03 Our Propose Scheme

05 Experiment

06 Conclusion

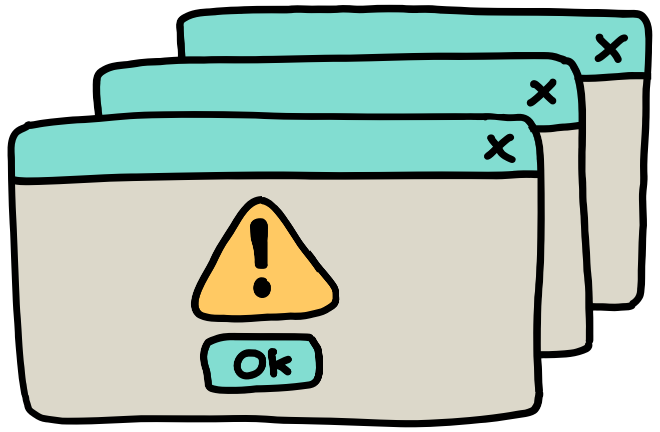
PROBLEM STATEMENT

LIMITATIONS OF MOBILE DEVICES

Mobile devices are limited in battery life, processing speed, and storage space, making them inefficient for complex cryptographic tasks on their own.

SECURITY CONCERN

Offloading to the cloud helps performance, but it risks data leaks and unauthorized access if the data isn't securely encrypted or signed.



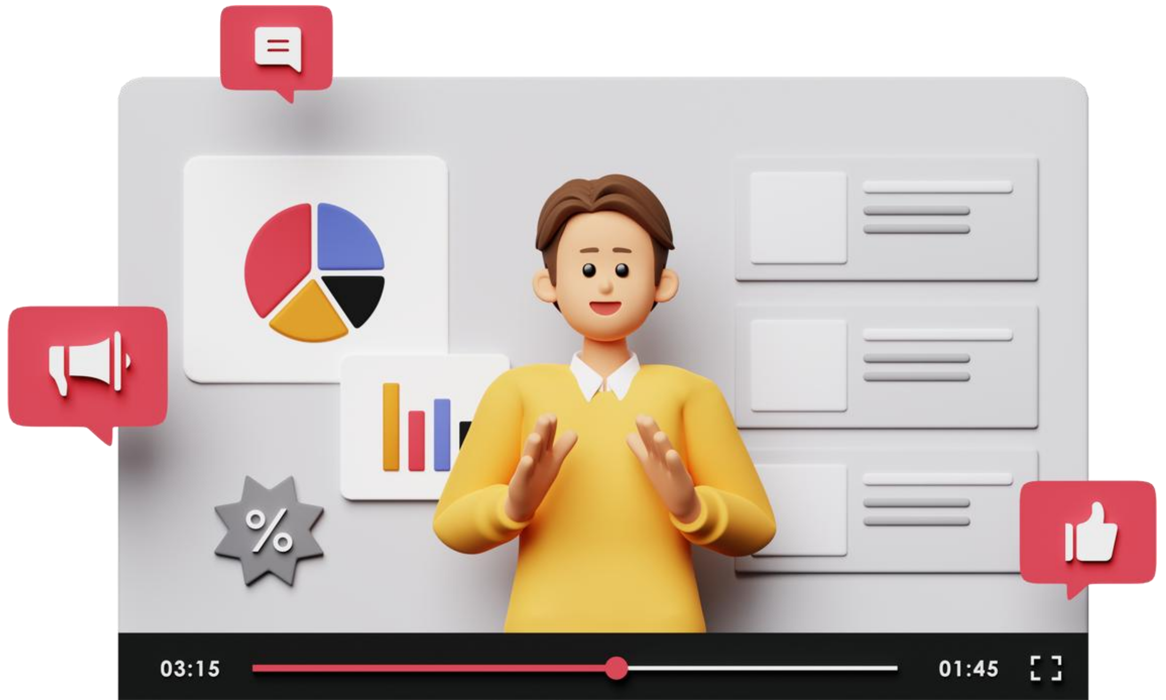
SOLUTION AND OUR APPROACH

WHY WE NEED BETTER SOLUTION

There's a need for a lightweight, secure and efficient method for digital signing that mobiles devices can handle without draining their resource

OUR APPROACH

We propose using signcryption with EdDSA to combine encryption and signing efficiently.



KEY CONCEPT

Mobile Cloud Computing

Mobile devices send heavy tasks to the cloud to save battery and processing power.
MCC improves performance but also raises privacy and security risks

Signcryption

A hybrid cryptographic technique that signs and encrypts a message in one efficient step. It ensures

- Confidentiality
- Integrity
- Authenticity

KEY CONCEPT

EdDSA

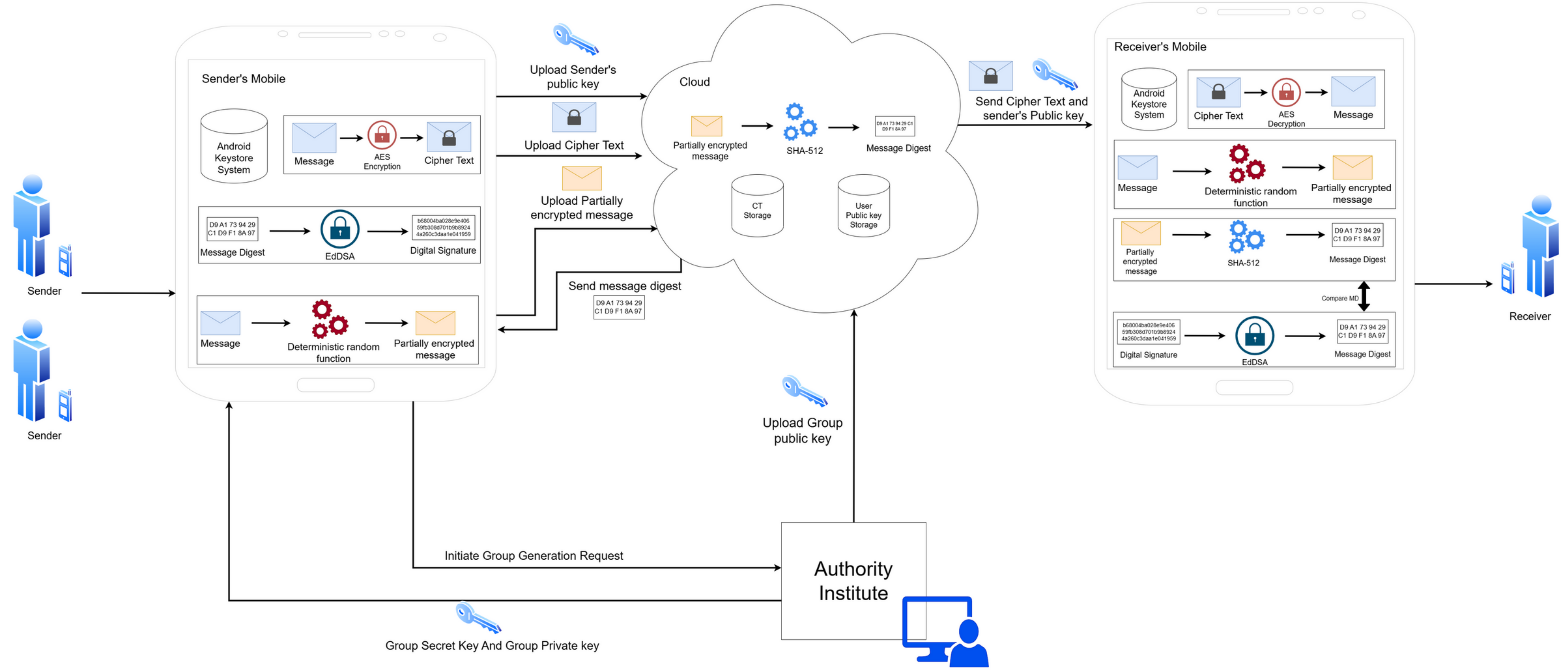
EdDSA is a fast, secure signature method with small keys, low computation cost, and strong resistance to side-channel attacks ideal for mobile use.

Group Signature

Allows multiple users to sign as a group, ensuring message authenticity while keeping individual identities private supporting shared accountability.

OUR PROPOSED SYSTEM

Lightweight Group Signature with EdDSA and Cloud Integration



KEY ENTITIES

Sender → The person who use their mobile phones to encrypt and digitally sign the message before sending it to the cloud

Receiver → The person who receives the message, use their mobile phone to decrypt the message and verify that it came from the real sender

Mobile → device that sender and receiver used to send and receive messages

Android Key System → used to store the private key and public key of the user.

Cipher Text Storage → A secure place in the cloud that keeps the locked encrypted messages. Only right key can unlock and read them

Mobile Cloud Computing → A cloud service that helps mobile phones by storing data and creating a code(hash) to check if the message is safe and unchanged.

User Pubic Key Storage → This part of the cloud keeps each user's public key. It's used by receivers to check if the message and digital signature are real and unmodified.

THE PROCESS OF OUR PROPOSE SCHEME

01

Key Generation

Each key generation by using key generation algorithm

02

Signing

Sender signs the message by using their key

03

Encryption

The plaintext of message is encrypted into ciphertext

04

Decryption

Receiver decrypt the ciphertext

PHASE 1
"KEY GENERATION"

1

2

ED25519 KEY PAIR GERENATION

Role : Signing and Verifying
Ed25519 Public Key : Verifies signature
Ed25519 Private Key : Sign a message

$$KeyPairGen_{Ed}(KeyGen_{Ed}, Key_{alisa}) \rightarrow (PubK_{Ed}, PrivK_{Ed})$$

X25519 KEY PAIR GENERATION

Role : Key exchange
X25519 Public Key : share key
X25519 Private Key : Diffie-Hellman key agreement

$$KeyPairGen_{X}(KeyGen_{X}, Key_{alisa}) \rightarrow (PubK_{X}, PrivK_{X})$$

PHASE 1
“KEY GENERATION”

3

DIFFIE-HELLMAN SHARED KEY GENERATION

a cryptographic method that allows two parties to securely generate a shared secret over a communication channel

$$\boxed{\textit{Diffie-Hellman}(PrivK_X,\ PubK_X) \rightarrow AES\_key}$$

PHASE 1
“KEY GENERATION”

4.GROUP DIGITAL SIGNATURE GENERATION

GROUP SERCERT KEY

signing by using private key from every sender to generate the key.

$$SecretKeyGen_{Group}(SecretKey_{Member}, K_a) \rightarrow GSK$$

GROUP PUBLIC KEY

verifying the digital signature which uses group secret key

$$PublicKeyGen_{Group}(SecretKey_{Group}, K_a) \rightarrow GPK$$

PHASE 2

"SIGNING"

1.COMPUTE HASH

1

2

Sender compute intermediate cipher text (Int_CT) by using deterministic random generator (F) with a seed.Both sender and receiver producing Int_CT to protected the confidentiality of the plaintext (PT).

$$F(PT, seed) \rightarrow Int\_CT$$

Int_CT will be hashed with SHA512 to produce a message digest (MD)

$$HASH(Int\_CT) \rightarrow MD$$

PHASE 2

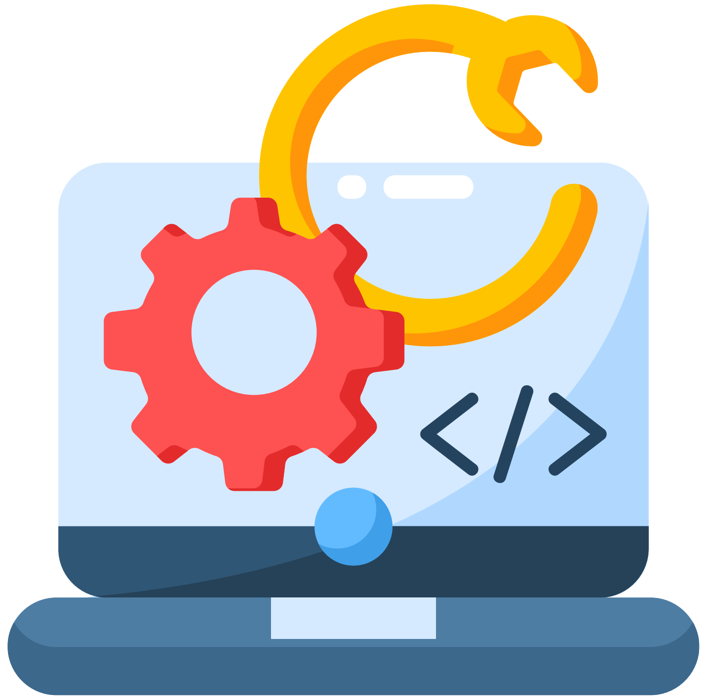
“SIGNING”

2

SIGN THE MESSAGE

The message digest (MD) is sent back to the sender to sign the message but in this case using Group Secret Key (GSK) to the message in case of group digital signature (DS).

$$\boxed{ENC_{Ed}(GSK, MD) \equiv DS}$$



PHASE 3
“ENCRYPTION”

1.ENCRYPT MESSAGE M

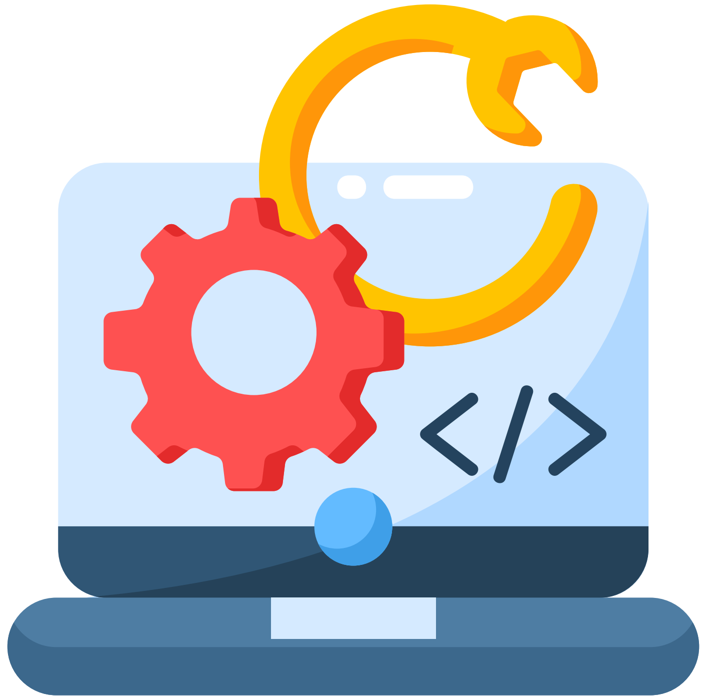
1

Sender constructs a shared secret key using sender’s private key and receiver’s public key that generate by using X25519 key generation

2

Message is encrypted into ciphertext by using AES

$$M = DS + PT$$
$$Diffie\text{-}Hellman(sPrivK_X, rPubK_X) \rightarrow AES\_key$$
$$ENC_{AES}(AES\_Key, M) \equiv CT$$



PHASE 4

“DECRYPTION”

1

DECRYPT CIPHERTEXT (CT)

Role : To recover the original message

$$Diffie\text{-}Hellman(rPrivK_X, sPubK_X) \rightarrow AES\_key$$
$$DEC_{AES}(AES\_Key, CT) \equiv M$$

2

VERIFY DIGITAL SIGNATURE

Role : Check authenticity and integrity

$$verify(DS, GPK) \equiv VerifyGroupDS$$

EVALUATION

Points for discussion

In this section, we break down how our model compare against others through three key areas of analysis.

Functional Analysis

Our scheme uses Diffie-Hellman to generate an AESKey which encrypts the message.Our cloud-less model performs pre-hasing and encryption on-device,reducing communication cost.

- Model [2] uses CP-ABE but lacks digital signatures.
- Model [17] uses Diffie-Hellman with ECC and signs messages using ECDSA.

	Crypto Operation on mobile			Outsource Cloud off-loading
	Diffie Hellman key exchange	AES	CP-ABE	pre-hashing
Our model without cloud off-loading	✓	✓	✗	✗
Ours	✓	✓	✗	✓
[2]	✗	✓	✓	✗
[17]	✓	✓	✗	✗

Computational Analysis

Our model uses partial AES encryption and offloads hashing to the cloud, then signs the digest. The AES key is derived from Diffie-Hellman (X25519).

- [2] uses AES, CP-ABE, and XOR, but lacks a signature scheme.
- [17] uses AES, Diffie-Hellman, and ECDSA for signing.

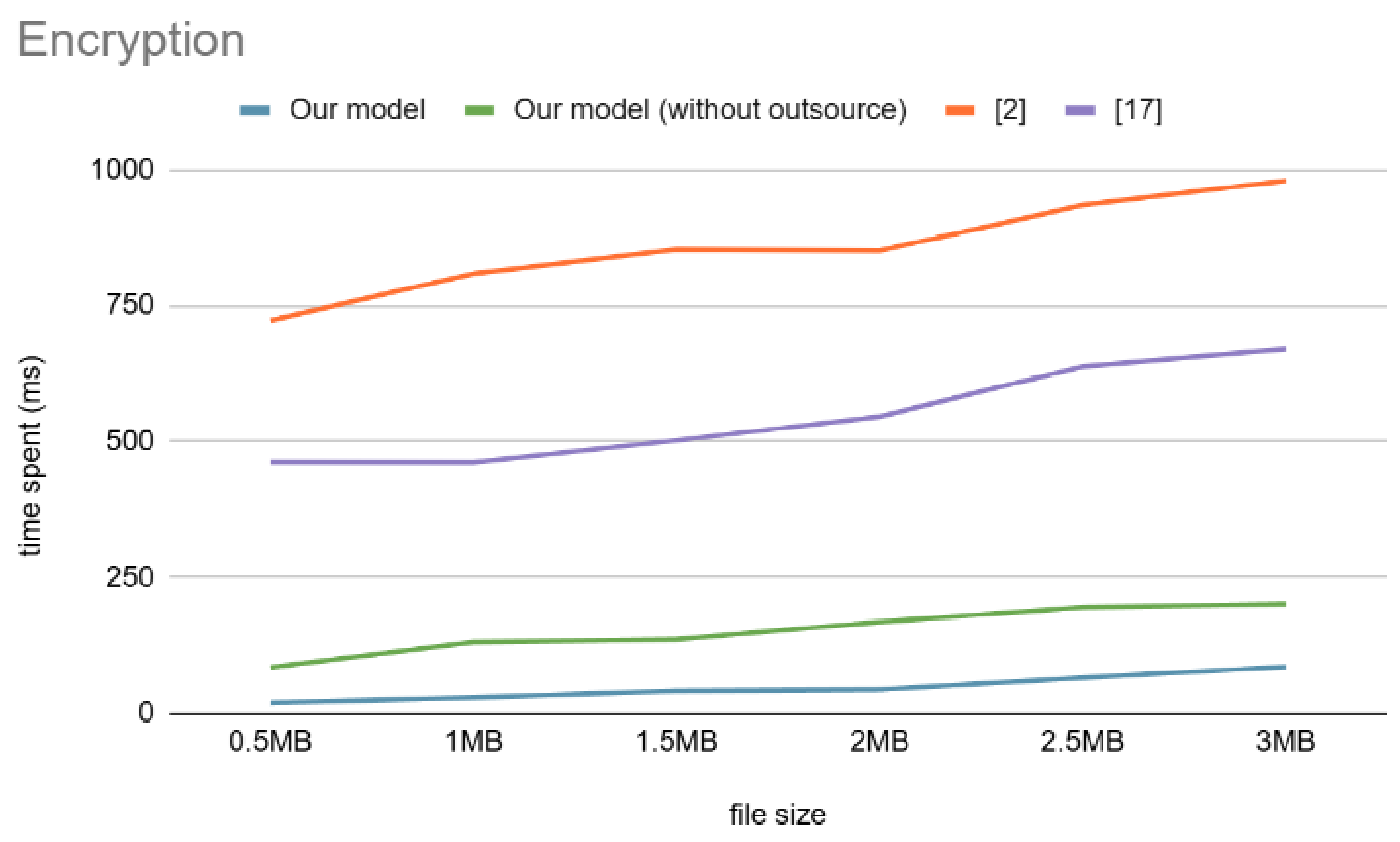
Our model's X25519 key exchange is shown (in our test) to be 10× faster than ECDSA used in [17].

	Encryption cost		Decryption Cost	
	Sender	Cloud	Receiver	Cloud
Ours	Dif + AES + Ed+ Par	Ha	Dif+ AES+Par	Ed
Ours without cloud	Dif + AES + Ha+ Ed+ Par	-	Dif+ AES+ Ed+Par	-
[2]	CP-ABE + AES + XOR	-	AES + XOR	CP-ABE
[17]	AES + Dif + ECD	-	AES+ Dif+ ECD	-

Experimental Analysis

We measure the time spent by conducted a test with Xiaomi Redmi Note 13 5G equipped with MediaTek Dimensity 6080 and 8GB of RAM running the Android 15 OS as a sender and receiver and using Google Cloud for deploy the application

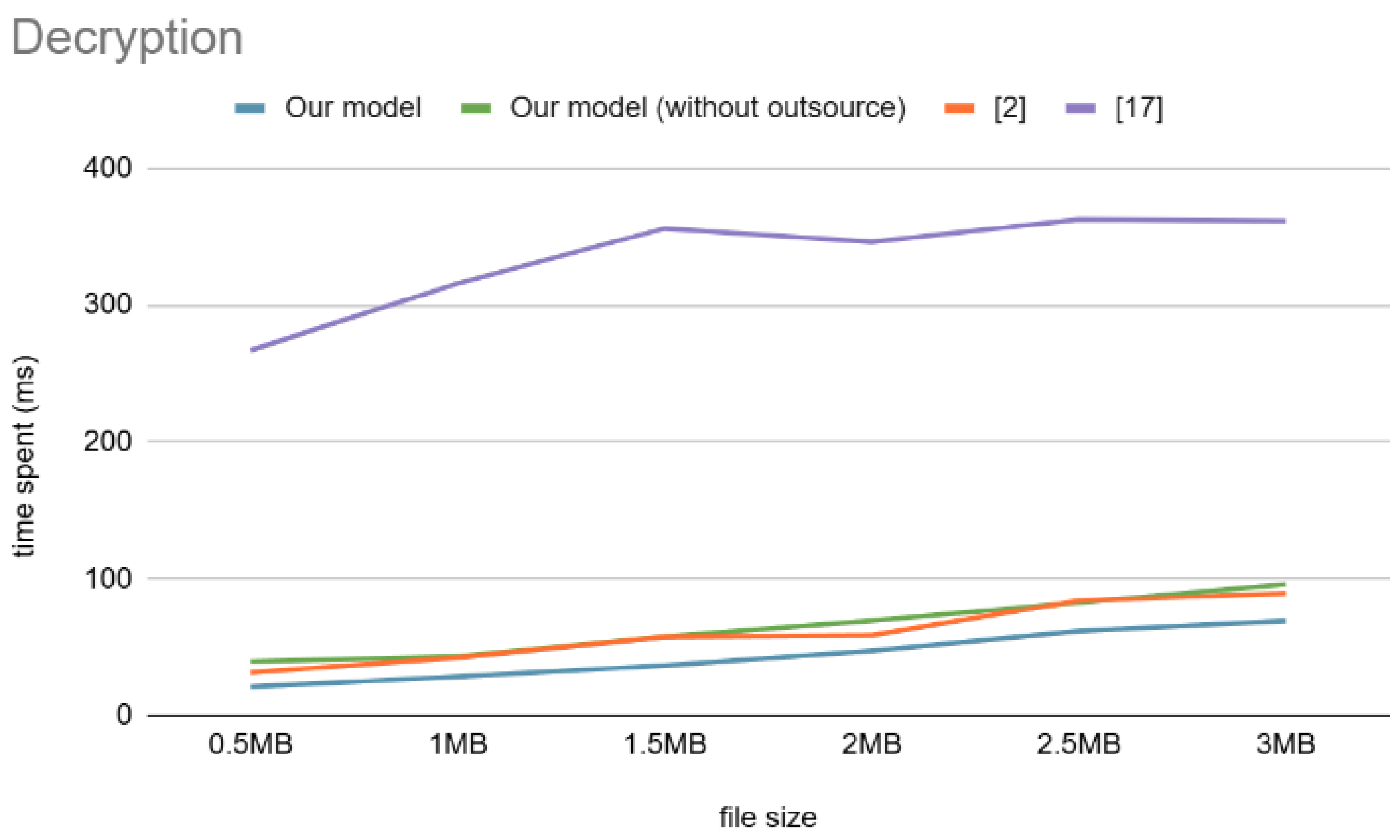
The graph shows our model has the fastest encryption time, outperforming non-cloud models.
• [2] uses AES + RSA with no cloud offloading.
• [17] uses ECC-based DH + AES, similar to us, but our use of X25519 makes key exchange faster.



Experimental Analysis

For decryption, all models use AES, but ours is the fastest. As we use EdDSA for signature verification.

- [2] does XOR operation, increasing time slightly.
- [17] spends more time on Diffie Hellman key exchange than our model.



Conclusion & Future work

In this proposed scheme we introduced a secure encryption model using X25519 and EdDSA, with cloud-based hashing to ease the load on sender devices.

So what’s next for our project?

- We will offload more tasks to the cloud to boost efficiency
- We will support multiple receivers with just one-time encryption
- We will improve our program to be faster, more reliable performance

Thank you for your attention

Siwanon - Conceptual Design, Methodology, Literature Review, Report Writing : Formatting, Experiment
Jirachaya - Conceptual Design, Literature Review, Report Writing : Detail, Presentation
Chitipat - Conceptual Design, Methodology, Literature Review, Report Writing : Detail, Presentation
Somchart Fugkeaw - Advisor