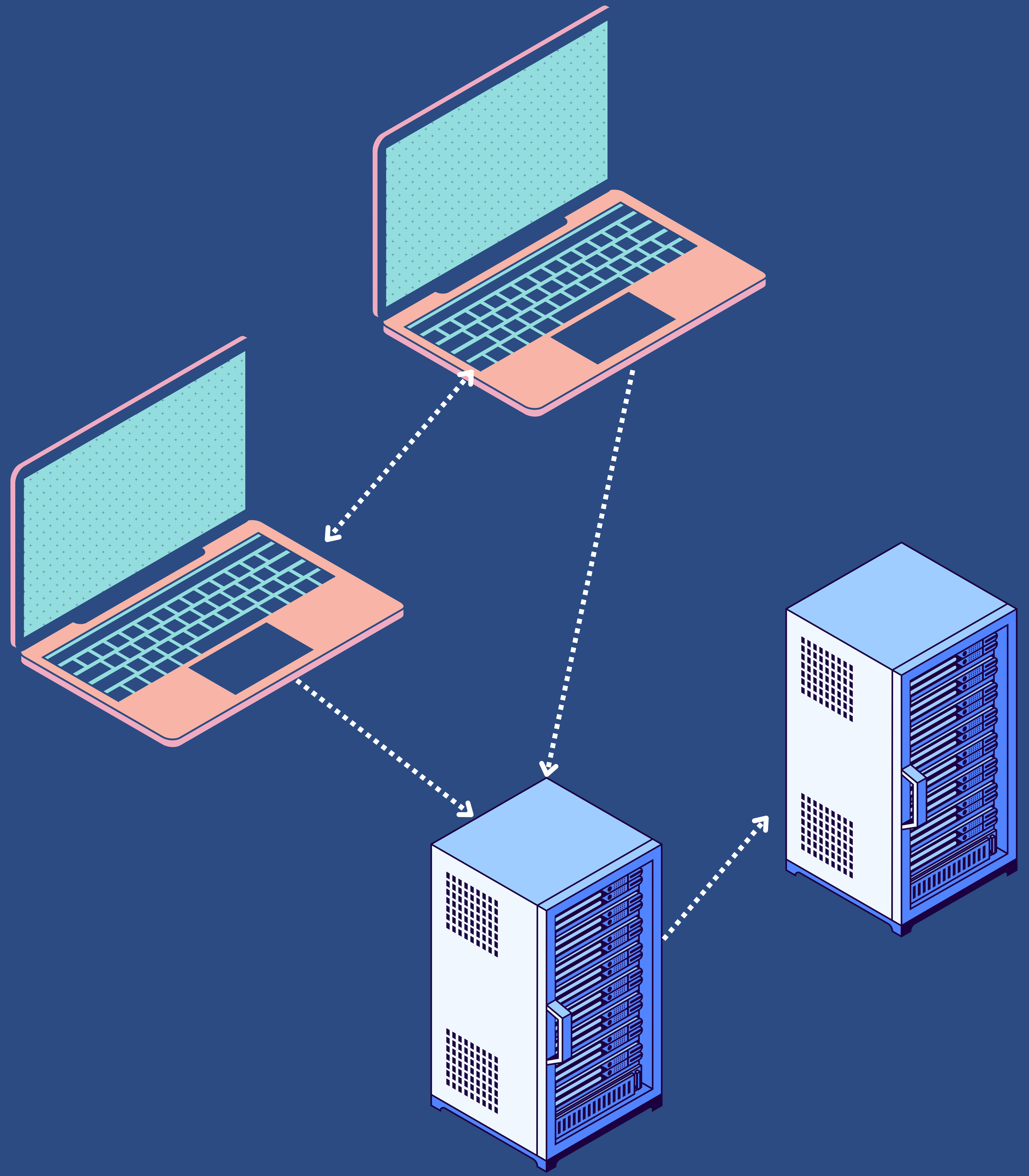


SIRS - 2024/2025

MessagIST

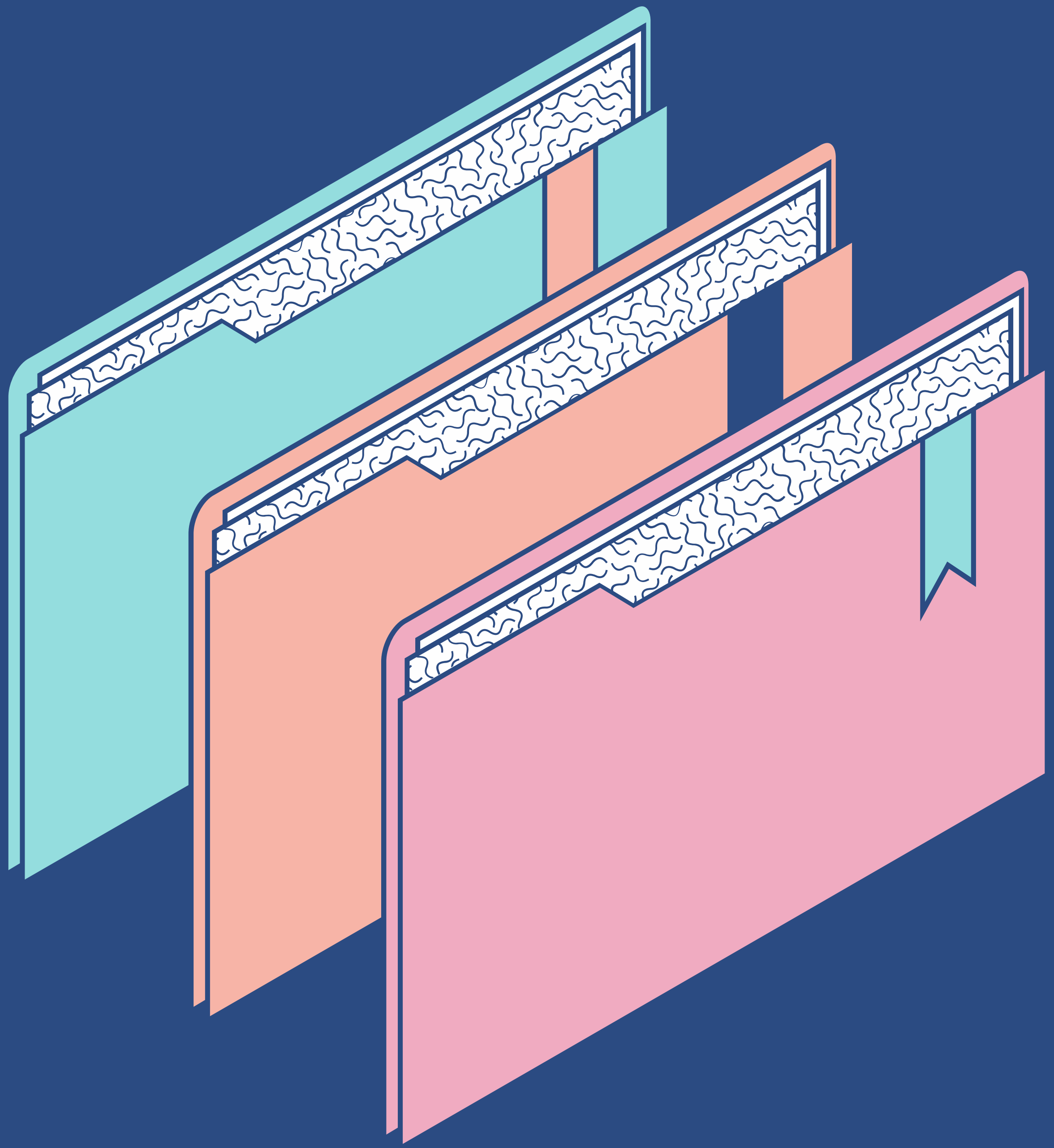
Group: A34

Daniela Camarinha ist1112265

Sofia Du ist1104195

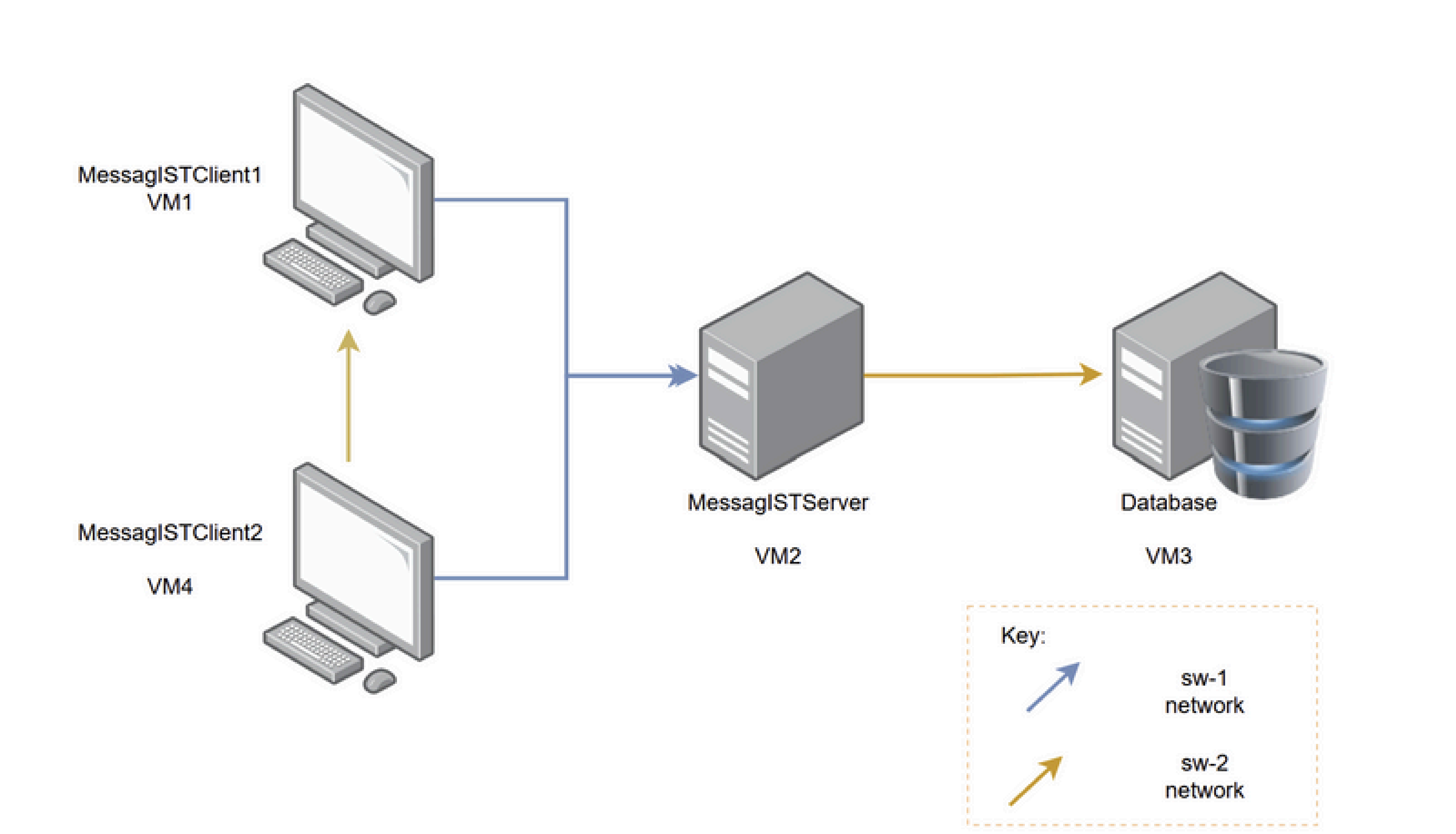
Tomás Gouveia ist1103793

Table of Contents



- 1. Build Infrastructure
- 2. Protecting Communications
- 3. Secure Document Library
- 4. Database
- 5. Security Challenge - A
- 6. Demo
- 8. Conclusion
- 9. Questions
- 10. Attacker Model

Build Infrastructure



<fig.1>

Protecting Communications

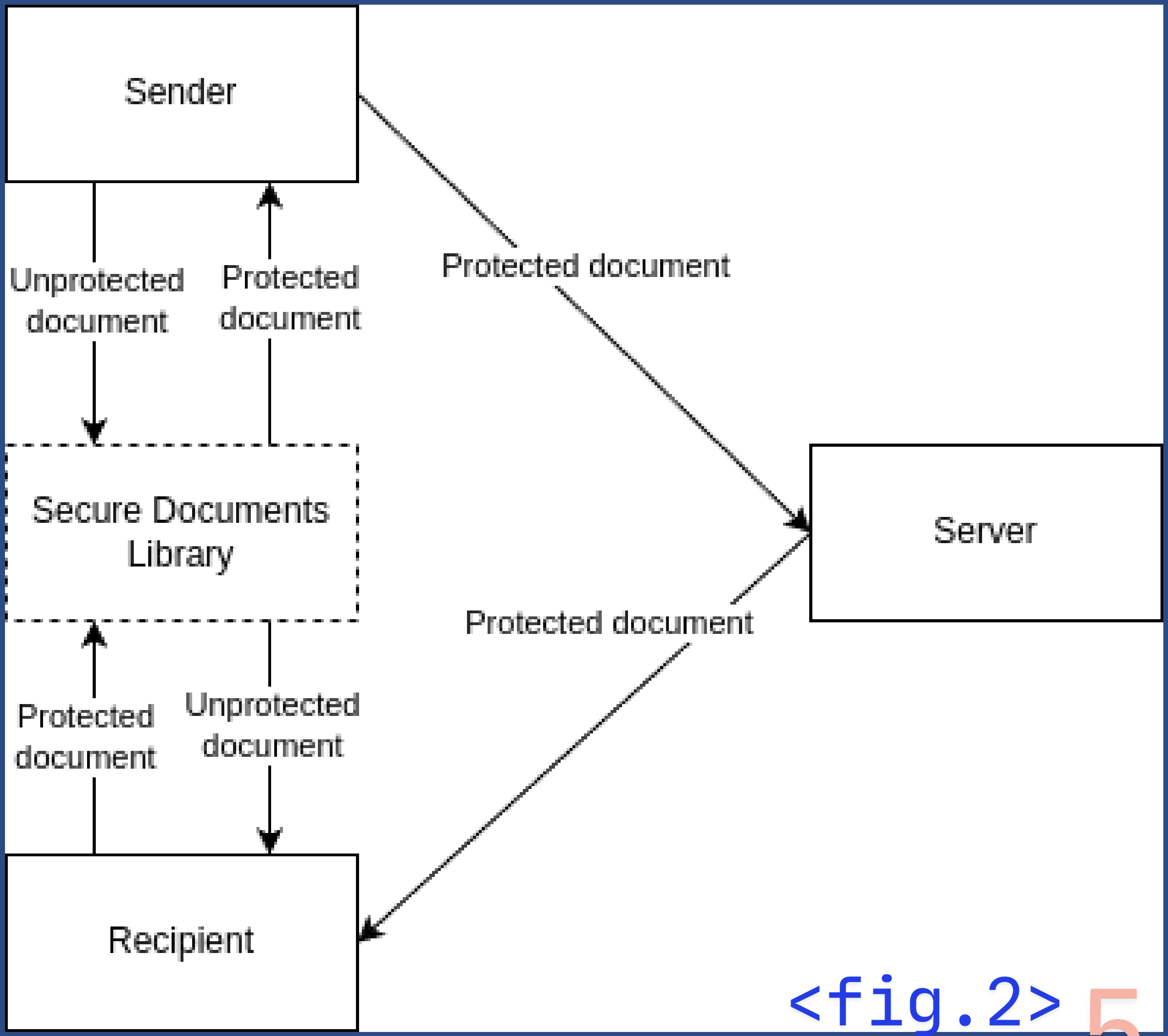
Communications are made through SSL/TLS over TCP:

- No messages can be lost in transit (TCP)
- Messages are **encrypted** (TLS)
- Each component maintains its own dedicated **KeyStore** and **TrustStore**.
- **One-way authentication** (server is authenticated, client is not - same in web browsers)

Secure Document Library

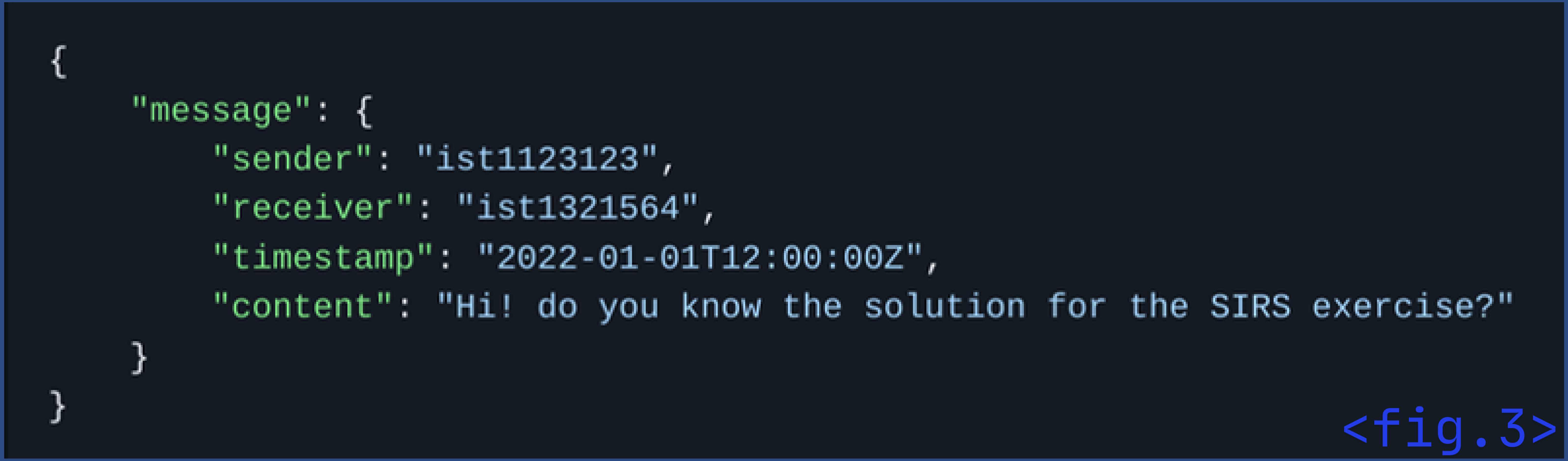
Provides these operations:

- Protect document
- Check integrity of document
- Unprotect document



<fig.2>

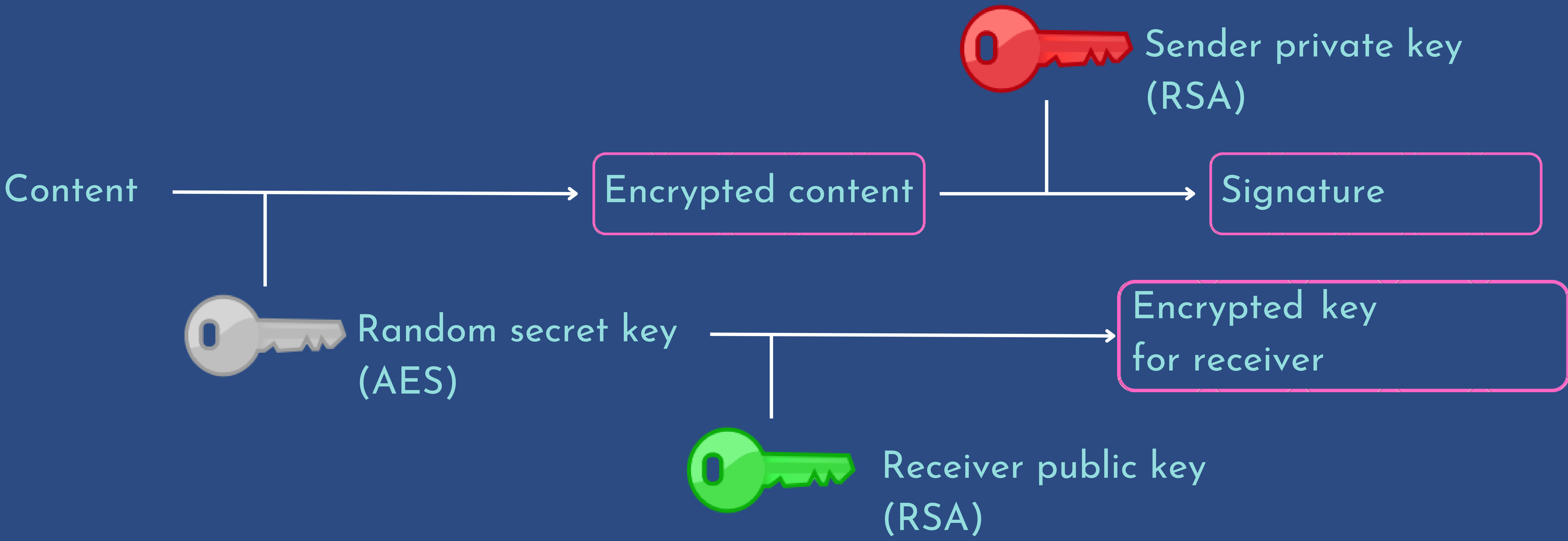
Secure Document Library



```
{
    "message": {
        "sender": "ist1123123",
        "receiver": "ist1321564",
        "timestamp": "2022-01-01T12:00:00Z",
        "content": "Hi! do you know the solution for the SIRS exercise?"
    }
}
```

<fig.3>

Secure Document Library



Content

Encrypted content

Sender private key (RSA)

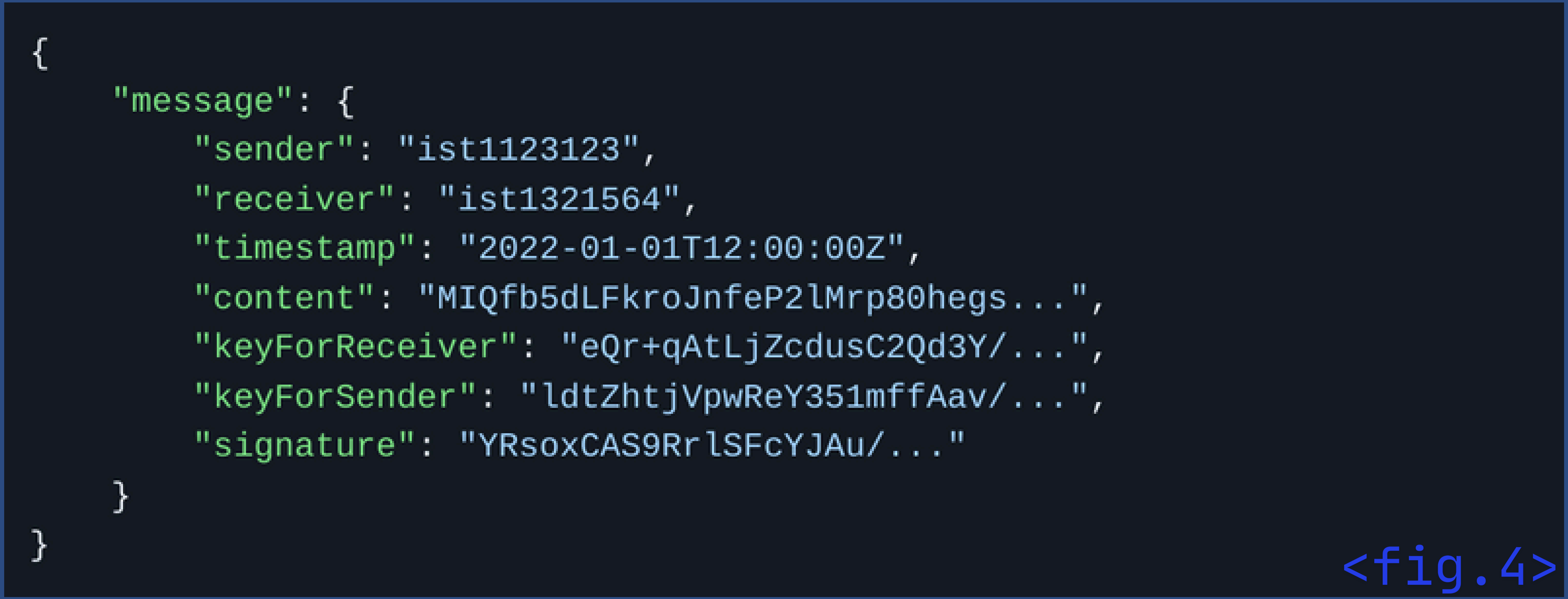
Signature

Random secret key (AES)

Encrypted key for receiver

Receiver public key (RSA)

Secure Document Library



```
{
    "message": {
        "sender": "ist1123123",
        "receiver": "ist1321564",
        "timestamp": "2022-01-01T12:00:00Z",
        "content": "MIQfb5dLFkroJnfeP2lMrp80hegs...",
        "keyForReceiver": "eQr+qAtLjZcdusC2Qd3Y/...",
        "keyForSender": "ldtZhtjVpwReY351mffAav/...",
        "signature": "YRsoxCAS9RrlSFcYJAu/..."
    }
}
```

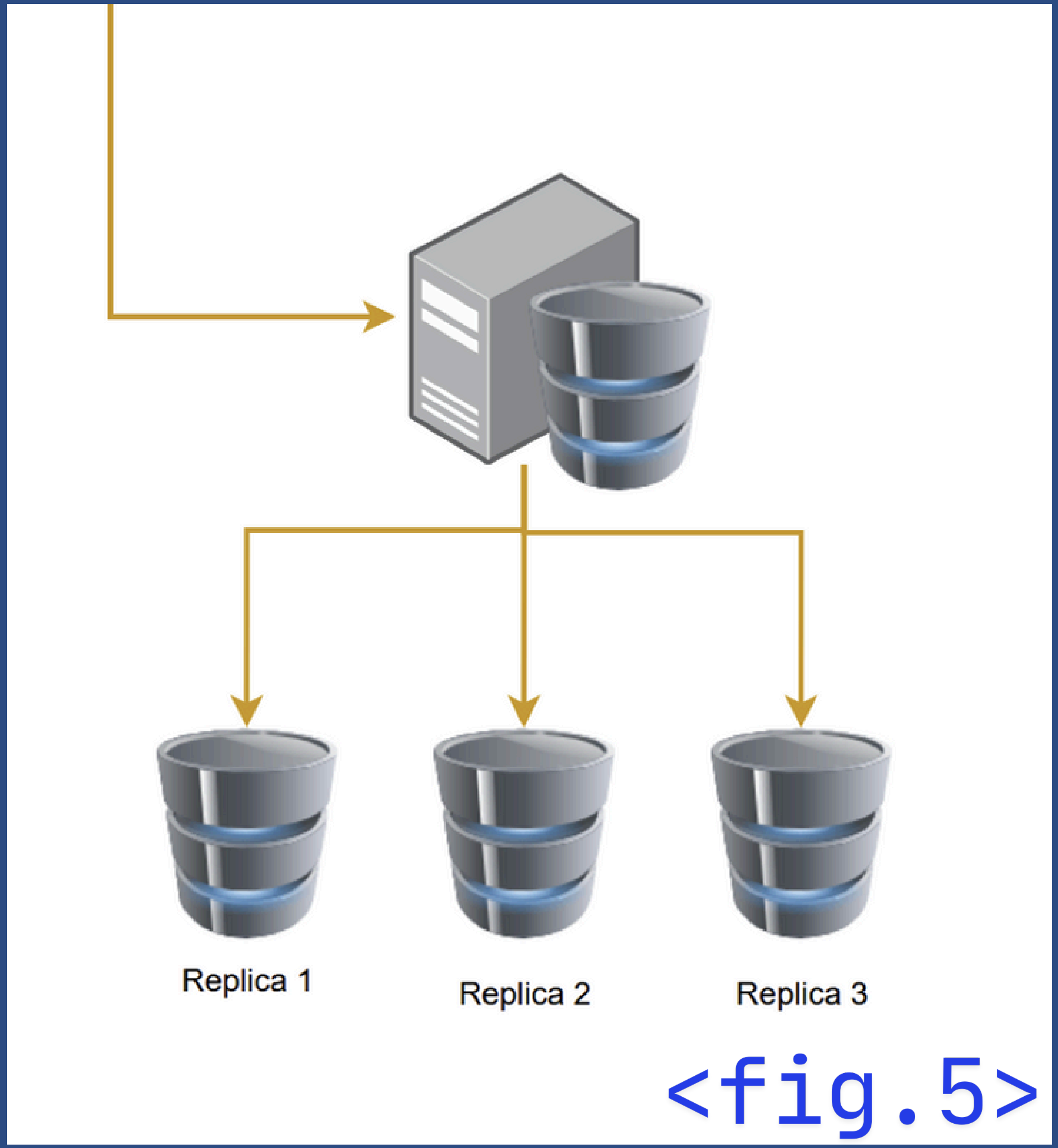
<fig.4>

Database

DuckDB

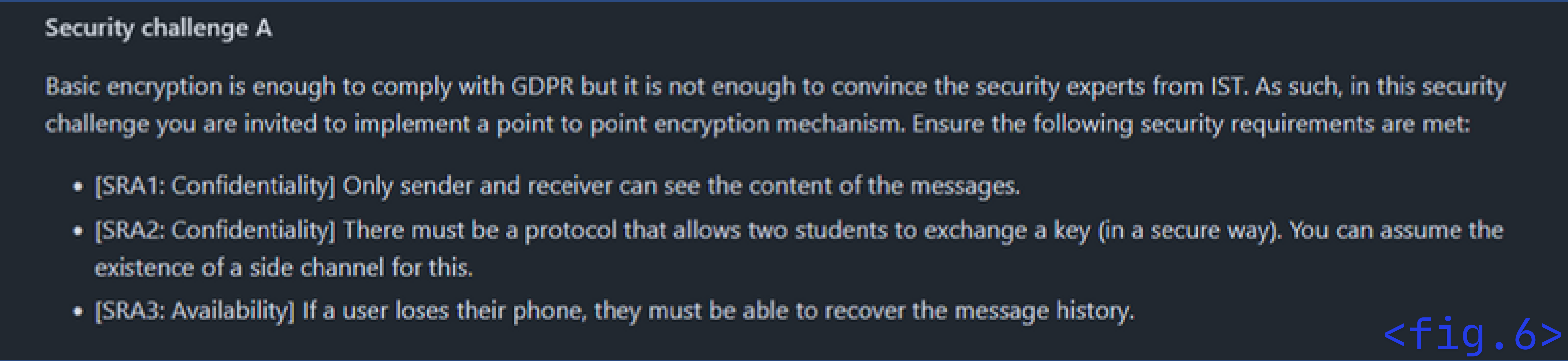
Database is trusted by the Server and:

- Is an **in-memory** database
- Should be replicated to ensure availability and reliability
- Scheduled Backup Service is executed every 14 days.
- Only responds to server-side requests, never accessed directly by the client.
- Sanitization of user input is done through the use of **PreparedStatements**



<fig.5>

Security Challenge A



Security challenge A

Basic encryption is enough to comply with GDPR but it is not enough to convince the security experts from IST. As such, in this security challenge you are invited to implement a point to point encryption mechanism. Ensure the following security requirements are met:

- [SRA1: Confidentiality] Only sender and receiver can see the content of the messages.
- [SRA2: Confidentiality] There must be a protocol that allows two students to exchange a key (in a secure way). You can assume the existence of a side channel for this.
- [SRA3: Availability] If a user loses their phone, they must be able to recover the message history.

<fig.6>

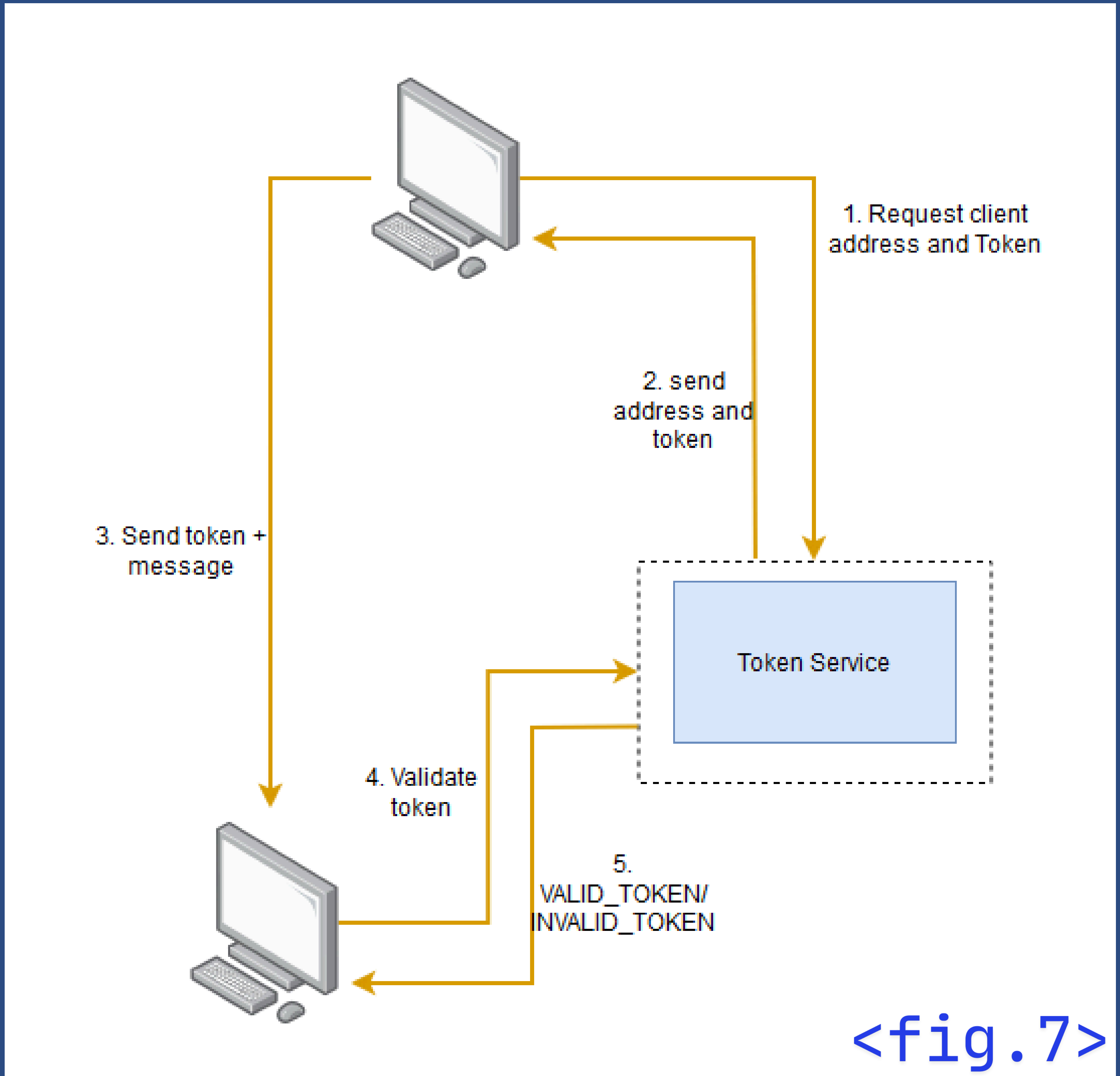
Security Challenge A

[SRA1: Confidentiality]:Solved by using the Secure Document Library ✓

[SRA2: Confidentiality]: E2E System - Token Service ✓

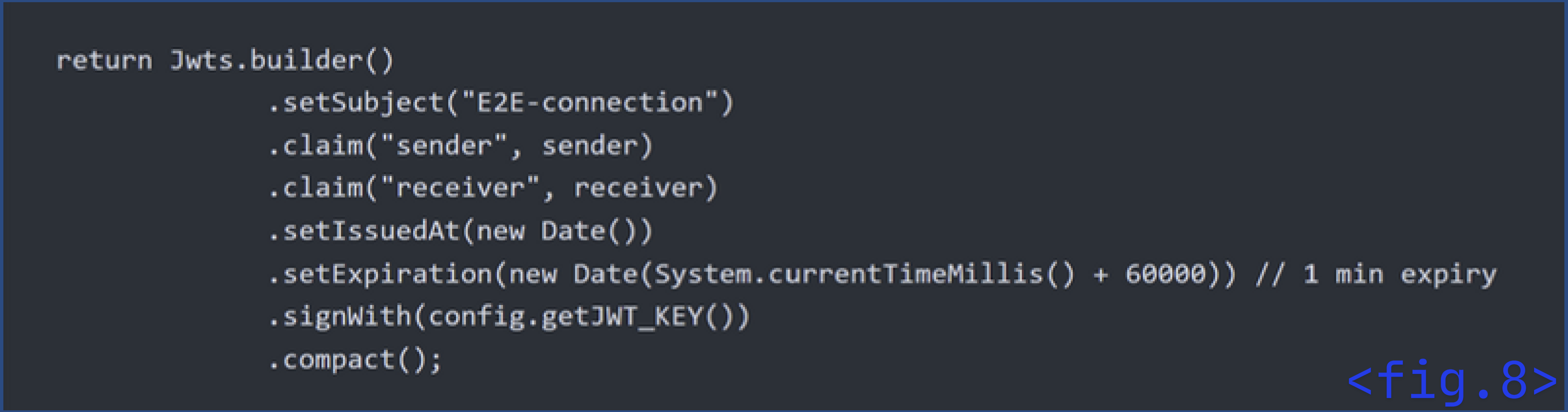
[SRA3: Availability]: PasswordDerivationService - Partial Implementation

E2E - End-to-End Communication


1. Request client address and Token
2. send address and token
3. Send token + message
Token Service
4. Validate token
5. VALID_TOKEN/ INVALID_TOKEN

<fig.7>

E2E - End-to-End Communication



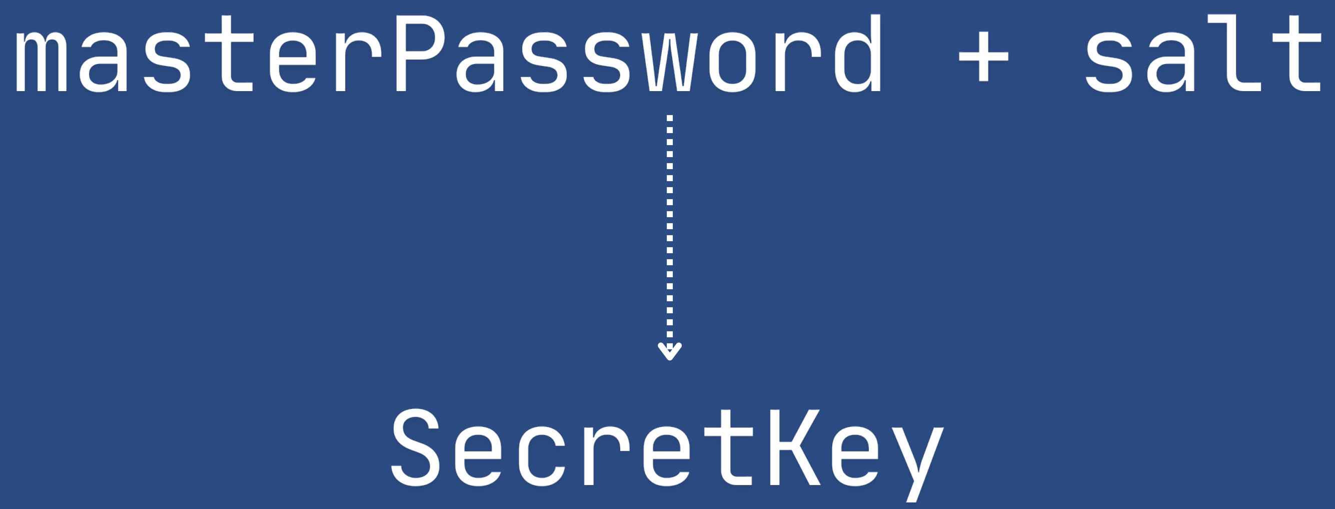
```
return Jwts.builder()
                .setSubject("E2E-connection")
                .claim("sender", sender)
                .claim("receiver", receiver)
                .setIssuedAt(new Date())
                .setExpiration(new Date(System.currentTimeMillis() + 60000)) // 1 min expiry
                .signWith(config.getJWT_KEY())
                .compact();
```

<fig.8>

Can only be validate by the server.

Expiry date secures against replay attacks

Password Derivation Service



masterPassword + salt

SecretKey

1. symmetric key

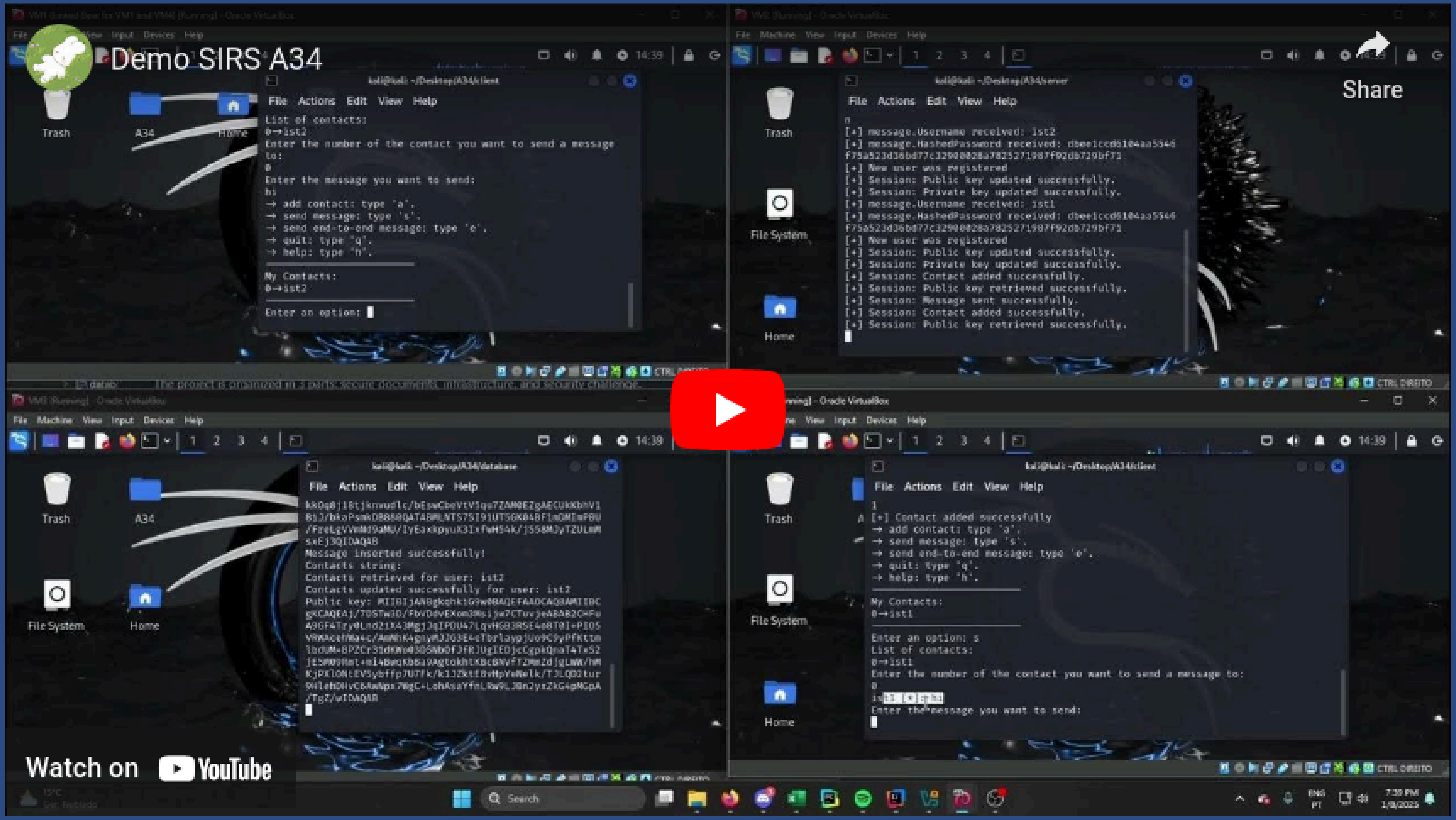
2. public key (generated keyPair)

Recovery of the Messages:

- The service regenerates the secret AES key
- Use private or symmetric key to decrypt messages

Demo

Link to video


Demo SIRS A34
Share
Watch on YouTube
kali@kali: ~/Desktop/A34/client
kali@kali: ~/Desktop/A34/server
kali@kali: ~/Desktop/A34/database
File Actions Edit View Help
Trash
A34
Home
File System
14:39

```
List of contacts:
0→ist2
Enter the number of the contact you want to send a message
to:
0
Enter the message you want to send:
hi
→ add contact: type 'a'.
→ send message: type 's'.
→ send end-to-end message: type 'e'.
→ quit: type 'q'.
→ help: type 'h'.
────────────────────
My Contacts:
0→ist2
────────────────────
Enter an option: █
```

```
n
[+] message.Username received: ist2
[+] message.HashedPassword received: dbee1ccd6104aa5546
f75a523d36bd77c32900028a7825271987f92db729bf71
[+] New user was registered
[+] Session: Public key updated successfully.
[+] Session: Private key updated successfully.
[+] message.Username received: ist1
[+] message.HashedPassword received: dbee1ccd6104aa5546
f75a523d36bd77c32900028a7825271987f92db729bf71
[+] New user was registered
[+] Session: Public key updated successfully.
[+] Session: Private key updated successfully.
[+] Session: Contact added successfully.
[+] Session: Public key retrieved successfully.
[+] Session: Message sent successfully.
[+] Session: Contact added successfully.
[+] Session: Public key retrieved successfully.
█
```

```
kkOq0j18tjknvudlc/bEswCbeVtV5qs7ZAM0EZgAECUkKbhV1
B1J/bkoPsmkD8B80QATABMLNTS7SI91UTS6K0A8F1mDMImPBU
/FreLgVVmNd9aMU/IyExxkpyuX31xfwH54k/jS58MJyTZULmM
sxEj3QIDAQAB
Message inserted successfully!
Contacts string:
Contacts retrieved for user: ist2
Contacts updated successfully for user: ist2
Public key: MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBC
gKCAQEAj/7DSTw3D/FbvDdvEXom3Msijw7CTuvjeA8A82CHFu
A9GF4Try0Lnd21X43MgjJgIPDU47LqwHGB3RSE4o8T0I+PIOS
VRMAcehMa4c/AmNhK4gnyMJJG3E4eTbrlaygjUo9C9yPfKttm
1bdUM+BPZCr31dKMo03DSNbOfJFRJUgIEDjcCgpkQnaT4TxS2
jE5MM9Hmt+mi4BwqKbba9AgtokhtKBcBNVfT2MmZdjgLWW/hM
KjPXlONtEV5ybffp7U7Fk/k1JZktE8vHpYeNelk/TJLQD2tur
9HlehDHvCbAwNpx7WgC+LohAsaYfnLRw9LJBn2yzZkG4pMGpA
/TgZ/wIDAQAB
█
```

```
1
[+] Contact added successfully
→ add contact: type 'a'.
→ send message: type 's'.
→ send end-to-end message: type 'e'.
→ quit: type 'q'.
→ help: type 'h'.
────────────────────
My Contacts:
0→ist1
────────────────────
Enter an option: s
List of contacts:
0→ist1
Enter the number of the contact you want to send a message to:
0
ist1 x hi
Enter the message you want to send:
█
```

Conclusion

Possible Enhancements:

- Local Persistent Storage for all message instances
- Change how token + message is sent to receiver

This project allowed us to develop our critical analysis of systems regarding how secure they are, and broaden our knowledge of the different possibilities to make a system secure.

Questions

Attacker Model

Assumptions:

What attacker **can** do:

- Setup virtual machine with own IP on the network to:
  - Use wireshark and try to sniff packets
  - Connect to the Server and try to send malicious playload

What attacker **cannot** do:

- Have knowledge of Runtime information of any of the Server machines, hence not being able to check the data stored by the Database.