# Network and Computer Security

## NotIST: Secure Note-Taking Application

Grupo A62: Duarte São José, João Maçãs, Miguel Parece

# Local Security

**Secure Document**

- [SR1] Confidentiality: Only the owner of the notes can see their content.
- [SR2] Integrity: The owner of the notes can verify they were not tampered with.
- [SR3] Integrity: The owner of the notes can verify if some note is missing.
- [SR4] Authentication: Only the owner of the notes can access them.

**Implementation**

- AES-CBC for content encryption
- HMAC-SHA256 for integrity verification
- Unique symmetric key per note

# Secure Document Format

```
{
    _id: ObjectId('67770560239543ab8405e278'),
    id: '04a34e14-2f3d-4d8b-b631-a30b0712f048',
    iv: '33f5c845e79510b1016860e5cea4cc43',
    hmac: '11eb278f3cae1ba4bc7ffffd6d0a72d27409d53bc35d65761a04c1f0592f7efd',
    title: '78882c834cd90b65516caa4cd16ef1ec',
    note: '4037afed5bec821498ee59ffe1c10e4b',
    date_created: ISODate('2025-01-02T21:30:08.098Z'),
    date_modified: ISODate('2025-01-02T21:30:08.098Z'),
    last_modified_by: 'a21b1134-62e9-4533-8ac8-152f69fa7a07',
    version: 1,
    owner: { id: 'a21b1134-62e9-4533-8ac8-152f69fa7a07', username: 'joao' },
    editors: [
      {
        id: '48ad0cc5-1eb4-42af-9b19-217b0dd3dfdd',
        username: 'miguel'
      },
    ],
    viewers: [
      {
        id: '48ad0cc5-1eb4-42af-9b19-217b0dd3dfdd',
        username: 'miguel'
      },
    ]
  }
```

Encrypted fields: title, note

Integrity protection: HMAC covers all encrypted content

Locally a note is a directory with:

- The note key encrypted in a file
- All encrypted versions of the note in separate files

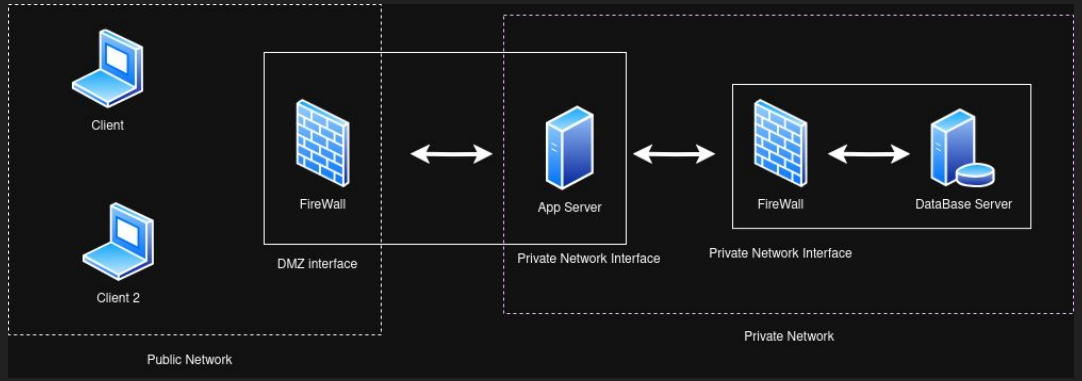
# Infrastructure Setup

## Components

- Client Application (Python TUI)
- Application Server (192.168.1.228:5000) (192.168.56.15)
- Database Server (192.168.56.17:27017)

## Security Measures

- TLS for client-server communication
- Mutual TLS between servers
- Firewall rules on both servers
- Network segregation for database




Vagrant to provision and easily deploy

# Security Architecture

## Keys

- Master Key (memory only): Derived from user password (PBKDF2)
- Note Keys: Unique symmetric key per note
- User Keys: Public/private key pair for sharing

## Security

- Digital signatures on all requests
- Timestamp verification for replay protection

# Security Challenge Implementation

**Requirements**

- [SRA1] Authentication: Only authenticated and authorized users can see the content of the notes.
- [SRA2] Integrity: Anyone that has access to the note can verify its integrity.
- [SRA3] Integrity: It is possible to verify the integrity of the notes throughout their versions.

**Solution**

- End-to-end encryption with public key
- Secure key distribution for collaboration
- Version control on the server-side

# Live demo…

- MasterKey KDF, KeyPair, Local File
- Create and edit note -> push explain the request
- Key sharing

# Conclusions

- **Main Results**
  - **Successfully achieved all the security goals outlined in the requirements.**
  - **Identified additional vulnerabilities that could be addressed in future iterations for enhanced security.**

- **Key Learnings**
  - **Gained in-depth knowledge of encryption techniques and their implementation in Python.**
  - **Explored the principles and challenges of building local-first applications, emphasizing user control and offline functionality.**