Unit 8: Cryptography and Its Use in Operating Systems

Seminar:

Cryptography Programming Exercise

Task:

Read the Cryptography with Python blog at tutorialspoint.com (link is in the reading list). Select one of the methods described/ examples given and create a python program that can take a short piece of text and encrypt it.

Create a python program (you can use the Jupyter Notebooks space) that can take a text file and output an encrypted version as a file in your folder on the system. Demonstrate your program operation in this week's seminar session.

Answer the following questions in your e-portfolio:

- Why did you select the algorithm you chose?
- Would it meet the GDPR regulations? Justify your answer.

Answer:

1. **Algorithm Selection**

- The code uses Fernet encryption from the cryptography library, which is a symmetric encryption method. Fernet was selected because:
- Ease of Use: It provides a simple interface for encryption and decryption, making it accessible even for developers who are not cryptography experts (Cryptography.io, n.d.).
- Security: Fernet guarantees that the message encrypted cannot be altered or read without the key. It uses AES in CBC mode with a 128-bit key for encryption,

along with HMAC for authentication, ensuring both confidentiality and integrity (Bellare and Rogaway, 2006).

- Standard Practices: AES (Advanced Encryption Standard) is a widely recognized encryption standard, approved by the National Institute of Standards and Technology (NIST), and is commonly used in secure applications worldwide (Ferguson, Schneier, and Kohno, 2010).

2. GDPR Compliance

The General Data Protection Regulation (GDPR) requires organizations to implement technical and organizational measures to protect personal data. Here's how the use of Fernet aligns with GDPR requirements:

1. Data Protection by Design (Article 25).

   The use of AES (Fernet) ensures strong encryption, safeguarding personal data (EUR-Lex, 2016a).

   Encrypted files cannot be accessed without the key, which fulfills the principle of minimizing data exposure.

2. Secure Processing (Article 32).

   Encryption protects data against unauthorized access during processing, transfer, and storage (EUR-Lex, 2016b).

   The inclusion of HMAC ensures that the encrypted data cannot be tampered with, meeting integrity requirements (Bellare and Rogaway, 2006).

3. Key Management

   A strong key is generated and stored securely, a practice that aligns with GDPR recommendations. However, secure storage and handling of the key.key file need to be enforced organizationally to prevent breaches (ISO/IEC 27001, 2013).

4. Breach Notification (Article 34)
5. If encrypted data is leaked but the key remains secure, it is considered pseudonymized. This can reduce the severity of required reporting and mitigate risks to individuals (EUR-Lex, 2016c).

Areas for Improvement to Fully Meet GDPR

- Key Storage: The key is stored locally in a file (key.key). For GDPR compliance, keys should ideally be stored in a secure key management system (OWASP, n.d.).
- Decryption: Currently, the app does not support decryption, meaning encrypted data cannot be accessed if required for compliance (GDPR.eu, n.d.).
- Audit Logging: Implementing logging for key generation, file encryption, and access can help provide evidence of compliance (ENISA, 2016).
- Access Controls: Access to the application and the key.key file must be restricted to authorized users to ensure GDPR compliance (NIST, 2020).

Conclusion

The selected algorithm (Fernet/AES) and its implementation provide a strong foundation for GDPR compliance. However, secure key management and additional safeguards (e.g., access control and audit trails) are necessary to fully meet GDPR regulations.

References

Bellare, M. and Rogaway, P., 2006. Introduction to Modern Cryptography. Available at: https://cseweb.ucsd.edu/~mihir/cse207/classnotes.html [Accessed 10 Jan 2025].

Cryptography.io, n.d. Cryptography documentation. Available at: https://cryptography.io/en/latest/ [Accessed 10 Jan 2025].

ENISA, 2016. Guidelines for Secure Use of Cryptography. Available at: https://www.enisa.europa.eu/ [Accessed 10 Jan 2025].

EUR-Lex, 2016a. Regulation (EU) 2016/679, Article 25: Data protection by design and by default. Available at: https://eur-lex.europa.eu/ [Accessed 10 Jan 2025].

EUR-Lex, 2016b. Regulation (EU) 2016/679, Article 32: Security of processing. Available at: https://eur-lex.europa.eu/ [Accessed 10 Jan 2025].

EUR-Lex, 2016c. Regulation (EU) 2016/679, Article 34: Communication of a personal data breach. Available at: https://eur-lex.europa.eu/ [Accessed 10 Jan 2025].

Ferguson, N., Schneier, B. and Kohno, T., 2010. Cryptography Engineering: Design Principles and Practical Applications. Indianapolis: Wiley Publishing.

GDPR.eu, n.d. GDPR Compliance Guidelines. Available at: https://gdpr.eu/ [Accessed 10 Jan 2025].

ISO/IEC 27001, 2013. Information technology - Security techniques - Information security management systems - Requirements. ISO.

NIST, 2020. Special Publication 800-57 Part 1: Key Management. Available at: https://nvlpubs.nist.gov/ [Accessed 10 Jan 2025].

OWASP, n.d. Key Management Guidelines. Available at: https://owasp.org/ [Accessed 10 Jan 2025>.