Unit 8: Cryptography and Its Use in Operating Systems

Initial Post

Collaborative Discussion 2: Cryptography case study - TrueCrypt

Task:

TrueCrypt was a popular and well-respected operating system add-on that could create encrypted volumes on a Windows and/or Linux system. In addition, it was also designed to create a complete, bootable volume that could encrypt the entire operating system and data for a Windows XP system. It was discontinued in 2014.

Case Study: Read the TrueCrypt cryptanalysis by Junestam & Guigo (2014) (link is in the reading list) and then answer the following questions:

- The (anonymous) TrueCrypt authors have said “Using TrueCrypt is not secure as it may contain unfixed security issues” (TrueCrypt, 2014). Does the cryptanalysis provided above prove or disprove this assumption?
- Would you be prepared to recommend TrueCrypt to a friend as a secure storage environment? What caveats (if any) would you add?

Remember to save this to your e-portfolio.

Present an ontology design which captures the weaknesses of TrueCrypt, and organise them according to their severity. Expand the ontology design by considering the factors which will cause each weakness to become an issue from a user's perspective. For example, if a user wishes to encrypt a disk storing bank details using TrueCrypt, which weakness of the software might cause this specific user goal to be negatively impacted?

Initial Post:

The analysis conducted by Junestam & Guigo (2014) provides a detailed examination of TrueCrypt authors' claim that their software is insecure because of potential unresolved security issues. Their assessment revealed 11 vulnerabilities in TrueCrypt 7.1a, including four medium and four low-severity problems. Although no high-severity risks were identified, the study shows significant flaws that challenge TrueCrypt's reputation as a secure encryption tool.

The analysis highlights several vulnerabilities, such as an inadequate key derivation algorithm for the volume header and possible sensitive information leaks from the kernel stacks. Other issues include using outdated string-handling APIs, lack of integer overflow checks, and improper use of memset() instead of RtlSecureZeroMemory(). While no intentional backdoors or malicious code were found, these findings support the authors' warning and demonstrate that TrueCrypt falls short of the expected secure code quality standards (Junestam & Guigo, 2014).

Endorsing TrueCrypt as a secure storage option requires substantial qualification. Although it remains functional for basic encryption tasks, users should be aware of the associated risks, particularly for high-priority applications, such as storing sensitive personal or financial information. Alternatives such as VeraCrypt, which builds upon and enhances TrueCrypt's original code, have been suggested to improve security (Spero et al., 2019). Additionally, built-in encryption tools, such as BitLocker for Windows or FileVault for macOS, offer reliable and actively maintained solutions (Ciesla, 2020).

From a classification standpoint, the identified weaknesses can be grouped according to their severity:

- **Medium:** Inadequate key derivation algorithm, kernel stack paging of sensitive data, and vulnerabilities in bootloader decompressor.
- **Low:** Integer overflows in IOCTL_DISK_VERIFY, kernel pointer disclosure, and use of insecure APIs.

User choices, such as opting out of full-disk encryption or using weak passwords, can exacerbate these vulnerabilities. For instance, encrypting a disk containing sensitive banking information risks exposure because of the weak volume header key derivation, making the encrypted data vulnerable to brute-force attacks.

These discoveries emphasize the necessity for ongoing security assessments and updated development practices for cryptographic software.

Ontology

The TrueCrypt weaknesses ontology chart visually represents the primary vulnerabilities discovered in TrueCrypt encryption software. The chart divides these vulnerabilities into five principal categories: cryptography vulnerabilities, data exposure vulnerabilities, data validation vulnerabilities, error handling and reporting weaknesses, and code quality issues (Junestam & Guigo, 2014).

Within each category, the specific weaknesses are detailed. Examples include a Weak Key Derivation Algorithm that may be vulnerable to brute-force attacks, Kernel Stack Paging Sensitive Data that could potentially expose encryption keys, and a Lack of Error Handling in EncryptDataUnits(), which might lead to the writing of unencrypted data (Junestam & Guigo, 2014).

The chart also connects each vulnerability to possible risks and user-related factors that could exacerbate these issues, such as weak passwords, physical access to devices, and system instability. By organizing the information in this manner, the chart facilitates the identification of areas of concern and offers insights into potential exploitation methods for each weakness. This underscores the importance of implementing more secure practices and improving software development techniques (Spero et al., 2019).

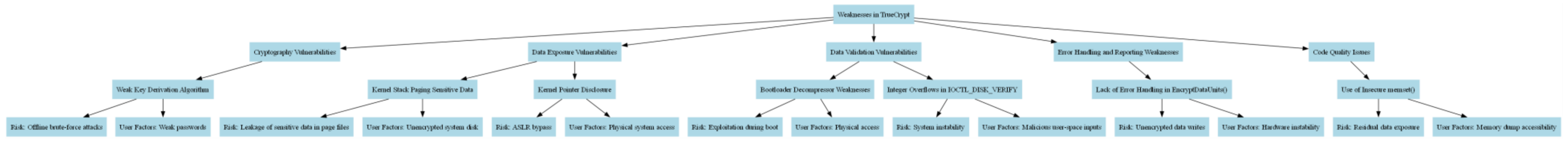
Weaknesses in TrueCrypt
Cryptography Vulnerabilities
Data Exposure Vulnerabilities
Data Validation Vulnerabilities
Error Handling and Reporting Weaknesses
Code Quality Issues
Weak Key Derivation Algorithm
Kernel Stack Paging Sensitive Data
Kernel Pointer Disclosure
Bootloader Decompressor Weaknesses
Integer Overflows in IOCTL_DISK_VERIFY
Lack of Error Handling in EncryptDataUnits()
Use of Insecure memset()
Risk: Offline brute-force attacks
User Factors: Weak passwords
Risk: Leakage of sensitive data in page files
User Factors: Unencrypted system disk
Risk: ASLR bypass
User Factors: Physical system access
Risk: Exploitation during boot
User Factors: Physical access
Risk: System instability
User Factors: Malicious user-space inputs
Risk: Unencrypted data writes
User Factors: Hardware instability
Risk: Residual data exposure
User Factors: Memory dump accessibility

Figure 1. Ontology of TrueCrypt Weaknesses

References

Ciesla, R. (2020) ‘Creating extremely secure encrypted systems’, *Encryption for Organizations and Individuals*, pp. 103–148. doi:10.1007/978-1-4842-6056-2_6.

Junestam, A., & Guigo, N. (2014). Open Crypto Audit Project: TrueCrypt Security Assessment. iSEC Partners.

Spero, E., Stojmenović, M. & Biddle, R. (2019) ‘Helping users secure their data by supporting mental models of Veracrypt’, Communications in Computer and Information Science, pp. 211–218. doi:10.1007/978-3-030-23522-2_27.