CSCE 4050/5050

Project 1 “Code Signing”

Group 21: Shajith Vignesh Ananthaneni(11691714) & Taha Al Obaidi(11502216)

Abstract

This project demonstrates code signing using the Elliptic Curve Digital Signature Algorithm (ECDSA) in Python to ensure software integrity and authenticity. We have set up the system in a way that it takes 3 main components before it goes ahead with execution. The 3 main components are a key generator, a product signer, and a validator. All the components play an important role in ensuring integrity and authentication. The key generator’s role is to generate a private and public key, the product signer’s role is to create a digital signature, and the validator's task is to check if the signature is valid before execution. The simulated attack scenario provides an excellent and simple understanding of what the project aims to do, it demonstrates the system’s ability to detect any tampering. Hence, if any tampering has been detected, the system will not go ahead with the execution which helps in preventing any malicious code from spreading. By running and testing our simulated attack scenario, we have been able to present the significance of code signing in software security, and with the results concluded, we can confirm the effectiveness of the system in detecting if a code has been tampered with.

Introduction

Code signing is an essential security mechanism that includes the usage of digital signatures to ensure the authenticity and integrity of software by making sure that the specific software has not been tampered with since the developer signed it. This cryptographic technique is important in preventing the spread of malicious software that could be harmful to user systems and data. In other words, it is like a certificate of authenticity you get when buying gold bars but, in the code, signing scenario, you ensure security and authenticity at the same time.

The main goal of this project is to create a system that implements code signing to verify authenticity and integrity. This is where the 3 main components come in handy, key generator, product signer, and validator. With each of the components contributing their roles to the system, we have managed to develop a simulated attack scenario that checks if the signature is valid, if so then the software is safe to run otherwise the execution is blocked to prevent potential harm.

For the implementation of this project, we have decided to use python due to its simplicity and the availability of ECDSA library that carried out the generation of cryptographic keys, created digital signatures, and verified them. The project presents the importance of code signing and how it can ensure security with authenticity and integrity.

Related Work:

There are many articles and studies out there explaining and emphasizing the importance of software security by code signing and how it can provide software integrity and the preventing unauthorized modifications.

1. "Security Considerations for Code Signing" Security Considerations for Code Signing

This NIST Cybersecurity White Paper gives us a better understanding of how code signing provides data integrity and that the code has not been tampered with. This article outlines the architecture and characteristics of commonly used code signing methods. The document covers code signing use cases and identifies potential security issues when using these solutions. The report concludes with recommendations for avoiding challenges and additional resources.

2. "The Importance of Code Signing Best Practices in the Software Development Lifecycle" The Importance of Code Signing Best Practices in the Software Development Lifecycle

This article posted by Sreeram Raju in 2025 presents us with a new idea that code signing is not only beneficial for protecting software but also improves operational efficiency by streamlining certificate management and increasing accountability throughout the development lifecycle.

3. "What is a Code Signing Certificate? Best and Effective Practices to Manage It" Code Signing Certificate and Best Practices around it

This is a slightly older article compared to the rest, posted in 2023, mainly talked about the same points that were presented in our report while also presenting best practices and vulnerabilities associated with code signing. It highly emphasizes the importance of restricting access to private keys.

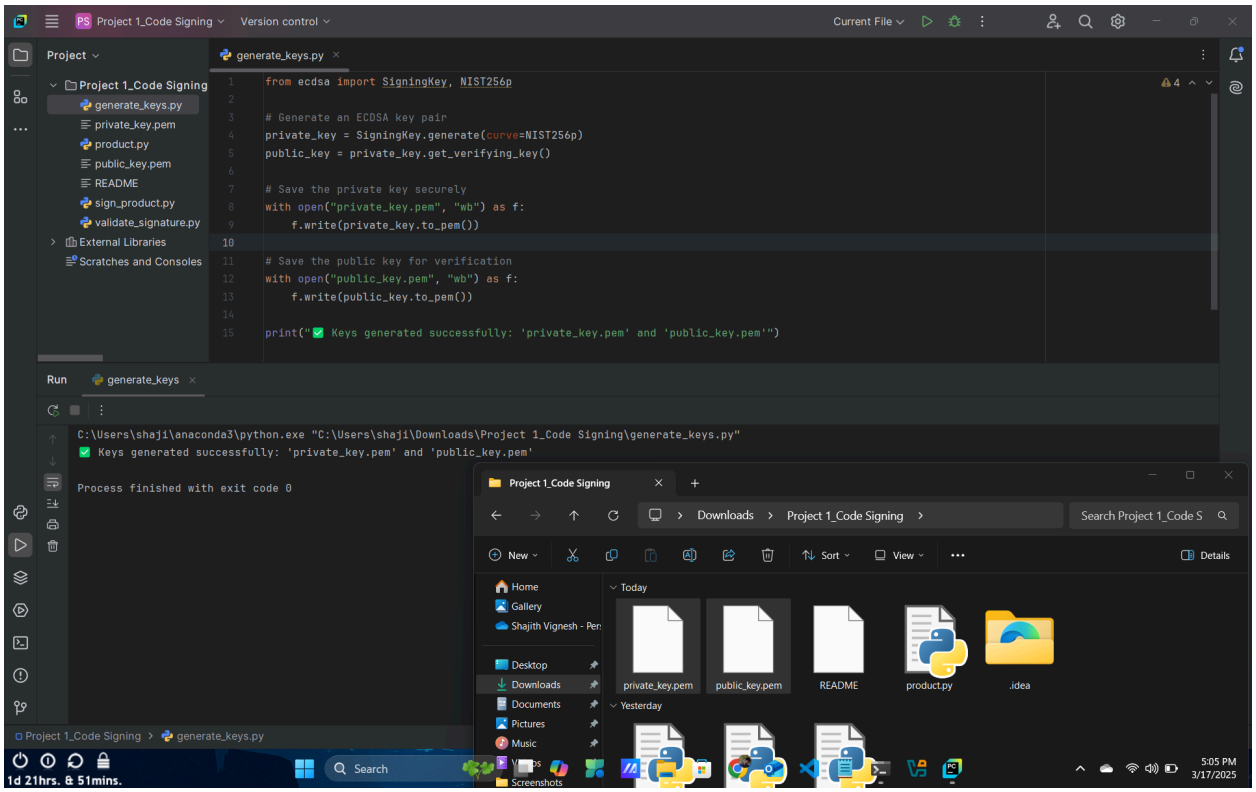
4. "Demystifying Code Signing" Demystifying Code Signing | PACE Anti-Piracy

This article by PACE posted in 2024 highlights what PACE had to say about code signing at the Droidcon London conference. Since this was during a Droidcon, the main focus was on Android applications, they passed on a unique point that every app in the Android ecosystem must be signed to be installed on a phone since it primarily protects the Android ecosystem as a whole rather than individual app developers.

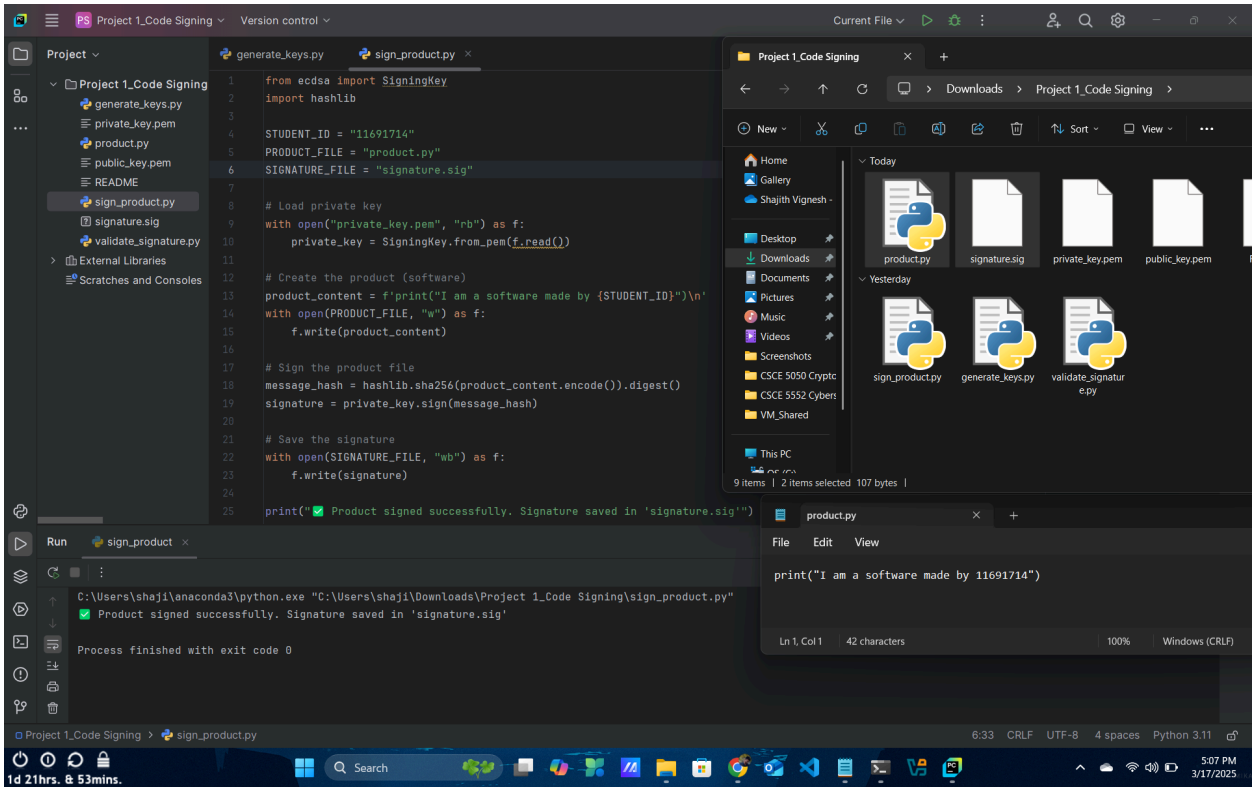
Approach

We will be getting into a deeper understanding of what the 3 main components' roles are. Starting with the key generator.

1. Key generation: This is the first step in the code signing process, where the pair of cryptographic keys are generated, the public key for verification and the private key for signing. This has been done using the ECDSA with the NIST P-256 curve due to its efficiency and security. The authenticity comes in with the private key which is what the developer uses to sign the software which always must be kept safe and secure and is stored in “private_key,pem”. To ensure the software is not tampered with, the public key is used for confirmation and verification, public key can be shared with users and is stored in “public_key.pem”.



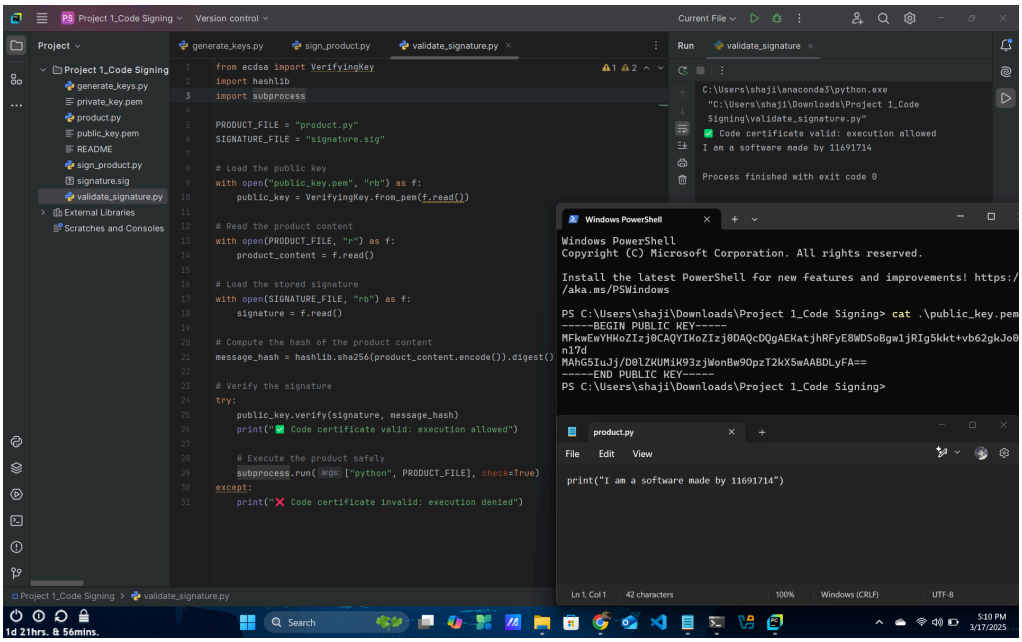
2. Product signing: This is the second step in the code signing process, its goal is to create a digital signature for the software, which is basically the certification. The “product,py” contains the student ID for which the product signer computes a SHA-256 Hash for, which is a unique has that can be detected with the slightest change. It then signs the hash using the private key which creates a digital signature that is stored in “signature.sig”



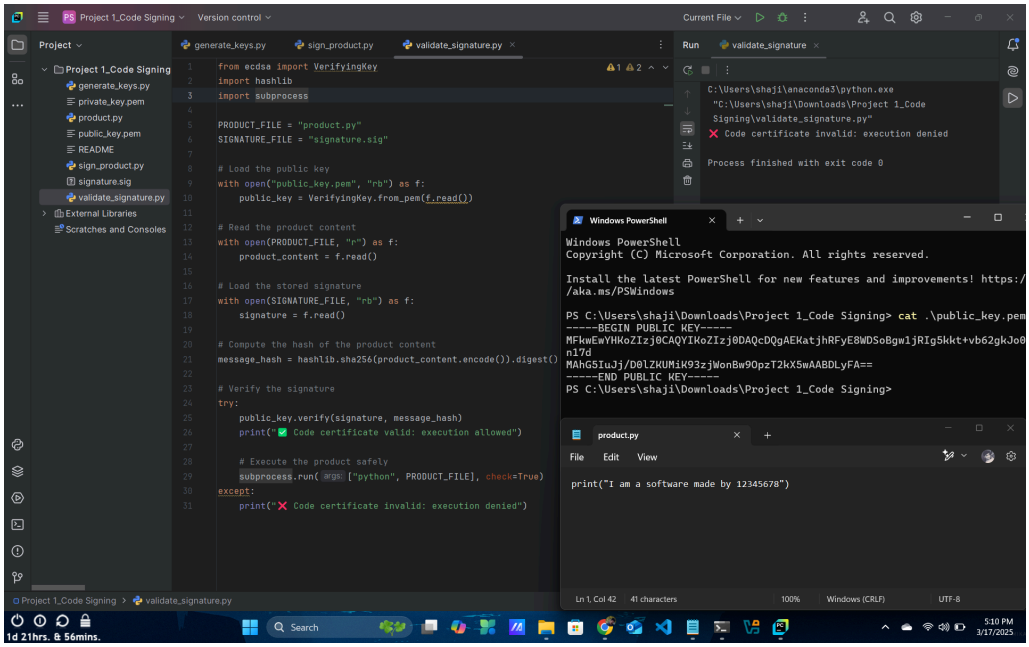
3. Validator: This is the last step in the code signing process, it aims to verify the authenticity and integrity of the software before deciding to continue with the execution or not. What the validator basically does is check if the product signing has been done correctly without any altering. It gets the public key from the “public_key.pem” file and the content in the “product.py” file. It follows the steps of product signing to ensure that the hash has been signed using the specified private key and if they match then the signature is valid and the execution can

go ahead or if it does not match then the signature has been tampered with and the execution is blocked to prevent potential harm.

Valid Execution:



Tampered Execution:



Conclusion

This project successfully demonstrates a secure code signing mechanism using ECDSA. By implementing digital signatures, the integrity and authenticity of software are preserved, preventing unauthorized modifications. The results validate the effectiveness of cryptographic verification in ensuring secure software execution.

Demo Video: Link

References

1. AndrejKovacevic. “All about Code Signing: What It Is, How It Works, and Why You Should Do It - Simple Programmer.” Simple Programmer, 22 Dec. 2021, simpleprogrammer.com/code-signing-primer/.
2. “What Is Code Signing?” Sectigo® Official, Sectigo, 2021, www.sectigo.com/resource-library/what-is-code-signing.
3. “What Are Code Signing Best Practices? | DigiCert FAQ.” Www.digicert.com, www.digicert.com/faq/code-signing-trust/what-are-code-signing-best-practices.
4. "Security Considerations for Code Signing," National Institute of Standards and Technology (NIST), Cybersecurity White Paper, 26 Jan. 2018.
5. Raju, Sreeram. “The Importance of Code Signing Best Practices in the Software Development Lifecycle.” AppViewX, 14 Mar. 2025, www.appviewx.com/blogs/the-importance-of-code-signing-best-practices-in-the-software-development-lifecycle/.
6. eMudhra Limited. “Code Signing Certificate and Best Practices around It.” Emudhra.com, 26 July 2023,

emudhra.com/blog/what-is-a-code-signing-certificate-best-and-effective-practices-to-manage-it.

7. Michie, Neal. “Demystifying Code Signing | PACE Anti-Piracy.” PACE Anti-Piracy, 18 Dec. 2024, paceap.com/demystifying-code-signing/.