A Modular Tool for Exploring the Elliptic Curve Digital Signature Algorithm

José Alberto Guzmán Vega, Rodrigo León Morales, Jorge Gael López Figueras[⋆], Eliseo Sarmiento Rosales, and Francisco Antonio Vidal Ojeda

Escuela Superior de Física y Matemáticas, Instituto Politécnico Nacional, México
`jlopezf2002@alumno.ipn.mx`

Abstract. This study presents the design and implementation of a modular, interactive tool for visualizing and understanding the Elliptic Curve Digital Signature Algorithm (ECDSA). Developed in Python with a Streamlit interface, the tool enables users to explore each phase of the signature process—key generation, message signing, and signature verification—by exposing intermediate values typically hidden in conventional cryptographic libraries. Its pedagogical aim is to bridge the gap between abstract mathematical foundations and practical cryptography, making ECDSA accessible to students and early-stage researchers. The tool includes interactive features such as contextual explanations, dynamic feedback, and simulations of cryptographic pitfalls (e.g., nonce reuse), all designed to enhance learning. Its modular architecture supports classroom use and future extensibility to other cryptographic schemes. The tool has been positively received in undergraduate and postgraduate cryptography courses for improving students' conceptual understanding of ECDSA. Future iterations will expand support for multiple elliptic curves, simulate advanced attack scenarios, and integrate with blockchain platforms to provide a broader cryptographic context.

Keywords: Elliptic curve digital signature algorithm, Digital signatures, Elliptic curve cryptography, Cryptography education

1 Introduction

Digital signatures are a cornerstone of modern cybersecurity, ensuring the integrity and authenticity of online transactions, securing software updates, and enabling identity verification in systems like Bitcoin. In Mexico, the advanced electronic signature (FIEL or *e.firma*) is widely used in legal, tax, and governmental processes. Despite their ubiquity, digital signatures often function as black boxes for both end-users and students in computing or mathematics programs. This disconnect between widespread use and limited understanding introduces pedagogical challenges and risks to transparency and trust [4, 11].

Among the many signature schemes, the Elliptic Curve Digital Signature Algorithm (ECDSA) has emerged as a standard for securing financial systems,

[⋆] Corresponding author

authentication protocols, and blockchain networks [6]. However, students often encounter ECDSA as an opaque, high-level implementation in cryptographic libraries, which prevents them from connecting theoretical concepts—such as group structures, finite fields, and elliptic curves—to their practical applications [1–3].

To address this educational gap, an interactive, browser-based tool was developed in Python with Streamlit to demystify ECDSA by exposing its internal processes. Learners can explore key generation, ephemeral scalar selection, computation of elliptic curve points, and derivation of signature components r and s. This step-by-step visualization enables students to develop both conceptual understanding and practical intuition. Unlike earlier tools such as the Java-based application by Tao et al. [12], the system provides a modular architecture that supports not only elliptic curve operations but also the full digital signature workflow and simulations of cryptographic pitfalls (e.g., nonce reuse or message tampering).

By bridging the gap between abstract mathematics and real-world cryptography, this tool empowers students to engage with ECDSA visually and computationally. Instructors can also benefit from its flexible design, which supports diverse teaching scenarios such as live demonstrations, lab activities, and student-driven explorations. Ultimately, the system provides an accessible environment that connects formal theory with secure systems deployed in practice.

This study is structured as follows: Section 2 reviews related educational tools and visualization frameworks for cryptography. Section 3 describes the methodology and architecture of the tool. Section 4 presents deployment results and a representative case study. Section 5 discusses key findings and future improvements. Section 6 concludes with reflections on its educational impact.

2 Related Work

Several tools have been developed to support cryptography education by making abstract concepts more tangible and accessible. Visualization systems can help students understand mathematical structures such as finite fields and elliptic curves, as well as their role in cryptographic protocols.

Tao et al. [12] introduced a Java-based application for visualizing elliptic curve operations. Their tool provides geometric intuition for scalar multiplication and point addition, allowing learners to explore elliptic curve groups over prime fields. However, it does not simulate the complete workflow of digital signatures nor address security-critical scenarios such as nonce reuse or tampered messages.

Recent studies highlight the importance of interactive learning environments for cryptography. Cattaneo et al. [4] developed visual frameworks to teach public-key cryptography and observed improved conceptual retention among undergraduate students. Similarly, X Simms and H Chi . [10] proposed an educational platform for visualizing digital signature algorithms, though it abstracted away key computational details.

Rayavaram et al. [8] conducted a comparative study of interactive cryptographic applications and concluded that tools offering real-time feedback and transparency significantly enhance learning outcomes.

The approach presented in this study builds on these efforts by offering a modular, browser-based tool that demystifies the Elliptic Curve Digital Signature Algorithm (ECDSA). Unlike previous systems, this tool reveals intermediate values—such as the ephemeral scalar k, curve point $R = kG$, and signature components (r, s)—and enables students to simulate common security pitfalls. This step-by-step interactivity fosters a deeper understanding of both the computational and security aspects of ECDSA, addressing gaps identified in prior educational tools.

3 Methodology

Architecture and Libraries. The tool was developed in Python and structured into three core modules: key generation, message signing, and signature verification. These modules reflect the logical progression of the ECDSA algorithm and aim to expose intermediate values often hidden in standard cryptographic libraries. Cryptographic operations are implemented using the `ecdsa` library for elliptic curve computations and the `cryptography` library for secure hash generation via SHA-256. In contrast to studies focused on improving ECDSA efficiency through algorithmic optimizations [5], our tool prioritizes pedagogical transparency, while maintaining sufficient computational efficiency for educational purposes.

Interface and Accessibility. The interface was built with `Streamlit`, allowing dynamic, browser-based interaction without the need for front-end development or local server configuration. This accessibility made it ideal for use in both in-person and remote learning environments. Its browser-based architecture makes it accessible without requiring specialized hardware or software.

Module 1: Key Generation. Technical Implementation: A random private key $d \in \mathbb{Z}_n$ is generated and the public key is computed as $Q = dG$. Both values are displayed to the user.
Didactic Purpose: Students can regenerate multiple key pairs to observe variability and the deterministic relationship between private and public keys.

Module 2: Message Signing. Technical Implementation: The user provides a plaintext message, which is hashed using SHA-256. A secure ephemeral key k is selected, the point $R = kG$ is computed, and the signature pair (r, s) is generated.
Didactic Purpose: Each value is shown step-by-step with contextual explanations. This enables students to connect algebraic steps to cryptographic outcomes.

Module 3: Signature Verification. Technical Implementation: The system accepts the original message, signature pair, and public key. It recalculates the verification condition and determines validity.
Didactic Purpose: Every intermediate computation is shown with feedback. When a mismatch occurs, the user is informed and shown the computation that caused the failure.

Interactive Learning Features. To enhance its educational value, the tool includes several interactive components designed to promote active learning:

- Contextual tooltips that provide brief definitions for mathematical symbols and variables at each step.
- Real-time value tracing that updates dynamically as users interact with the key, message, or signature inputs.
- Feedback mechanisms that simulate common mistakes. For instance, if the user modifies the message after signing, the tool displays a verification failure along with the expected hash value, reinforcing the concept of message integrity.
- Built-in error triggers that illustrate cryptographic pitfalls. Reusing the same ephemeral key k across signatures prompts a security warning, which explains the vulnerability introduced and its relevance to real-world attacks.

Modifiability and Extensibility. Although only a standard ECDSA over a NIST P-256 curve is implemented, the modular design allows easy substitution of curves (e.g., secp256k1, P-384) or simulated vulnerabilities (e.g., fixed k, invalid point attacks). These features are not yet implemented but the architecture allows their integration.

Classroom Integration. The tool was deployed across three instructional formats:

1. **Live demonstrations** in which the instructor used the tool to walk through examples.
2. **Lab sessions** where students signed and verified messages hands-on.
3. **Student mini-projects** in which learners were challenged to modify aspects of the tool (e.g., hash functions, curve parameters).

Environment and Compatibility. The application was tested on both Linux and Windows environments. Installation requires only a standard Python environment with the required libraries, ensuring that it runs consistently across different operating systems.

4 Results

The tool was successfully deployed in undergraduate and postgraduate cryptography courses, allowing students to explore ECDSA step by step. Its modular structure and interactive feedback enabled learners to visualize key operations and internal variables, enhancing their conceptual understanding of digital signatures.

A representative case study was used to demonstrate the signature process on a sample message with the following parameters:

- **Private key** $d \in \mathbb{Z}_n$:

  `1074949393645155448835840334614153933313161941331778555017415826900044621834 04`

- **Ephemeral key** $k \in \mathbb{Z}_n$:

  `103741840181295446477146623661875051018620753856998458511666888180971368523067`

- **Computed point** $R = kG = (x_1, y_1)$:

  $x_1 =$ `5786544796460403262329725456314759082441565143261091253811106828485351926944 2`
  $y_1 =$ `1088245178973818972302510539740035071441009064204447561894995739521946048820 59`

- **SHA-512 hash of message** h:

  `532859565262330465470231237540213980763521271172101365656088930299332718480 29`

- **Signature components**:

  $r = x_1$
  $s =$ `77956660124118251817030780317899270079882836588953511953207248685357796734992`

During verification, the tool recomputed:

- Scalars $u_1 = s \mod n$, $u_2 = h \mod n$
- Point $(x_1, y_1) = u_1Q + u_2G \mod n$
- $v = x_1 \mod n$, which matched the signature value r

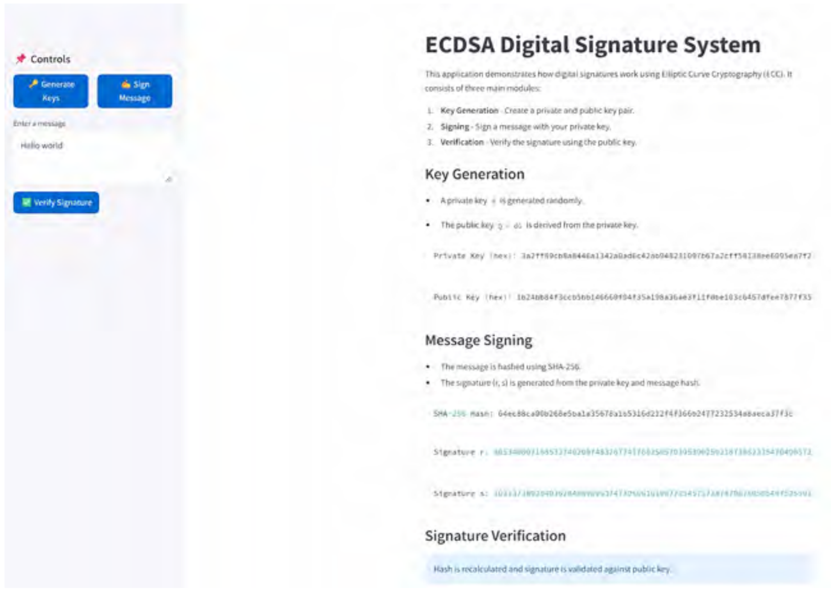
This confirmed the signature's validity. Altering the original message correctly caused verification to fail, reinforcing the importance of integrity in digital signatures.

The tool runs on both Windows and Linux systems with no special configuration, requiring only the `streamlit`, `ecdsa`, and `cryptography` libraries. Its design emphasizes modularity and clarity, enabling learners to both observe and manipulate the cryptographic process in real time.

This study not only operationalizes ECDSA but also bridges the gap between abstract mathematics and practical cybersecurity applications in an accessible, classroom-ready format.

4.1 Interface Snapshot

To complement the technical implementation, Fig. 1 displays a snapshot of the running application. The layout includes modules for key generation, message signing, and signature verification. Each step reveals intermediate values, helping students visualize the cryptographic process in real time.



ECDSA Digital Signature System

This application demonstrates how digital signatures work using Elliptic Curve Cryptography (ECC). It consists of three main modules:

1. **Key Generation** - Create a private and public key pair.
2. **Signing** - Sign a message with your private key.
3. **Verification** - Verify the signature using the public key.

Key Generation

- A private key d is generated randomly.
- The public key $Q = dG$ is derived from the private key.

```
Private Key (hex): 3a2ff89cb8a8446a1342a0ad6c42ab048231097b67a2cff58138ee6095ea7f2
Public Key (hex): 1b24bb84f3ccb5bb146660f04f35a198a36ae3f11f0be103c6457dfee7877f35
```

Message Signing

- The message is hashed using SHA-256.
- The signature (r, s) is generated from the private key and message hash.

```
SHA-256 Hash: 64ec88ca00b268e5ba1a35678a1b5316d212f4f366b2477232534a8aeca37f3c
Signature r: [illegible]
Signature s: [illegible]
```

Signature Verification

Hash is recalculated and signature is validated against public key.

Fig. 1. ECDSA digital signature tool interface: signing and verification

5 Discussion

The interactive tool presented in this study highlights the pedagogical potential of combining modular design with transparent cryptographic computations. By enabling learners to visualize each stage of the ECDSA workflow—key generation, message signing, and verification—the tool addresses a well-documented challenge in cryptography education: bridging the gap between abstract mathematical concepts and their real-world implementations.

Unlike prior visualization tools such as those developed by Tao at al. [12], which primarily focus on illustrating elliptic curve operations, the system presented here provides a comprehensive exploration of the full digital signature process. This includes critical security concepts like ephemeral key management and signature validation, which are often overlooked in high-level educational frameworks. By exposing intermediate values and simulating cryptographic pitfalls, the tool encourages learners to engage actively with both the computational and security aspects of digital signatures.

Students reported increased confidence in understanding elliptic curve cryptography, a technique also recognized as well suited for constrained environments such as IoT and smart devices [7], particularly when transitioning from theoretical representations over $\mathbb{R}$ to finite fields $\mathbb{F}_p$. Instructors noted that the modular interface facilitated interactive lectures and self-guided exploration during laboratory sessions, reinforcing concepts that are traditionally difficult to internalize.

Nevertheless, the current version has limitations. It supports only the NIST P-256 curve and does not yet simulate advanced attack scenarios such as fault injection or side-channel analysis [9]. These features, along with the integration of alternative elliptic curves (e.g., Edwards curves) and blockchain-specific adaptations, are planned for future iterations. Furthermore, accessibility challenges persist in resource-constrained environments; developing a mobile version of the tool could expand its reach to public schools and rural areas.

In comparison to existing educational approaches, this tool prioritizes transparency and hands-on interaction. These characteristics are essential for fostering a deeper understanding of cryptographic algorithms and equipping students with the intuition needed to apply theoretical knowledge in practical cybersecurity contexts.

6 Conclusion and Future Work

This study presents an interactive, didactically motivated tool for teaching the Elliptic Curve Digital Signature Algorithm (ECDSA), one of the most widely used schemes in modern cryptographic systems. The tool's modular structure and transparent computations offer a step-by-step interface that helps students understand the internal mechanics of digital signatures, moving beyond the typical black-box approach found in cryptographic libraries.

From a technical standpoint, one of the main challenges was ensuring compliance with elliptic curve cryptography (ECC) specifications—particularly the correct implementation of key generation and signature verification. Adapting the tool to operate with standard curves like SECP256k1 required detailed study of documentation and precise integration with Python libraries such as `ecdsa` and `cryptography`.

The tool has been successfully tested on both Windows and Linux systems. It runs in a standard Python environment and requires only three libraries: `streamlit`, `ecdsa`, and `cryptography`. Its ease of deployment, combined with browser-based accessibility, makes it suitable for classroom use in a wide range of educational settings.

Pedagogically, this study demonstrates how abstract mathematical concepts—such as elliptic curve arithmetic, modular operations, and hash-based authentication—can become concrete and accessible through interactive visualization. By guiding students through each step of the ECDSA process, the tool helps to close the gap between theoretical mathematics and practical cryptographic applications.

Looking ahead, future iterations of the system may include support for additional elliptic curve formats (e.g., Edwards curves), the simulation of advanced attack scenarios (e.g., fault injection or side-channel analysis), and the integration of learning analytics to assess student progress. Furthermore, we envision developing a mobile version of the tool to facilitate access in contexts, where computer availability is limited, particularly in public schools or rural areas. These enhancements would broaden the tool's reach and impact within diverse educational settings, supporting both instructors and learners in deepening their understanding of modern cryptography.

References

1. A Abidi, B Bouallegue, F Kahri Implementation of Elliptic Curve Digital Signature Algorithm (ECDSA) in *2014 Global Summit on Computer & Information Technology (GSCIT)*, Sousse, Tunisia, 2014, pp. 1–6
2. W Alhamdani Teaching Cryptography Using Design Thinking Approach *Journal of Applied Security Research*, vol 11, pp. 78–89, 2016 doi:10.1080/19361610.2015.1069646
3. J Bos, J A Halderman, N Heninger, J Moore, M Naehrig, E Wustrow Elliptic Curve Cryptography in Practice in *Lecture Notes in Computer Science*, vol. 8437, Springer, 2014
4. G Cattaneo, A De Santis, U Petrillo Visualization of Cryptographic Protocols with GRACE *Journal of Visual Languages & Computing*, vol. 19, pp. 258–290, 2008 doi:10.1016/j.jvlc.2007.05.001
5. Y Genç, E Afacan Design and Implementation of an Efficient ECDSA in *2021 IEEE IEMTRONICS*, 2021, pp. 1–5
6. D Johnson, A Menezes, S Vanstone The Elliptic Curve Digital Signature Algorithm (ECDSA) *International Journal of Information Security*, vol. 1, no. 1, pp. 36–63, 2001 doi:10.1007/s102070100002
7. S Koppula, J Muthukuru Secure Digital Signature Scheme Based on Elliptic Curves for Internet of Things *International Journal of Electrical and Computer Engineering*, vol. 6, no. 3, pp. 1002, 2016 doi:10.11591/ijece.v6i3.9420
8. P Rayavaram, O Ukaegbu, M Abbasalizadeh, K Vellamchetty, S Narain CryptoEL: A Novel Experiential Learning Tool for Enhancing K-12 Cryptography Education in *Proceedings of the 56th ACM Technical Symposium on Computer Science Education*, 2025
9. J-M Schmidt, M Medwed A Fault Attack on ECDSA in *2009 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*, IEEE, 2009, pp. 93–99
10. X Simms, H Chi Enhancing Cryptography Education via Visualization Tools in *Proceedings of the 49th Annual ACM Southeast Conference*, 2011, pp. 344–345
11. D Stinson, A Wager *Teaching Mathematics for Social Justice: Conversations with Educators* National Council of Teachers of Mathematics, 2012
12. J Tao, J Ma, M Keranen, J Mayo, C-K Shene ECvisual: A Visualization Tool for Elliptic Curve Based Ciphers in *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, 2012, pp. 571–576