Teaching Asymmetric Cryptography: A Three-Level, History-Grounded Laboratory Model

Piotr Witkowski, Bartosz Sawicki
Warsaw University of Technology,
Warsaw, Poland
Email: {piotr.witkowski, bartosz.sawicki}@pw.edu.pl

Aleksandra Kawala-Sterniuk
Wrocław University of Science and Technology,
Wroclaw, Poland
Email: aleksandra.kawala-sterniuk@pwr.edu.pl

Urszula Solarz
Email: urszulasolarz@gmail.com

***Abstract*—Courses on asymmetric cryptography often suffer from a gap between mathematical theory and practical configuration. Mathematical treatments focus on number theory without practical context, while operational tutorials teach command sequences without explaining underlying principles. This paper proposes a curriculum framework that bridges this gap by organizing material into three abstraction levels: (1) deployed products and configuration, (2) algorithm implementation, and (3) mathematical theory. Each level pairs system mechanisms with historical attacks that exploited them, providing clear rationale for specification constraints and security standards. We instantiate this model using RSA in a laboratory sequence implemented at two technical universities, mapping each abstraction level to student assignments, historical vulnerabilities, and core engineering principles. Annual course evaluations demonstrate high student interest and engagement. As a methodological contribution, this paper analyzes the curriculum design and its trade-offs rather than reporting formal control-group empirical results.**

***Index Terms*—Cryptography education, curriculum design, RSA, padding oracle, SSL/TLS, OpenSSL.**

I. Introduction

Public-key cryptography poses distinct challenges for engineering education. While correctness relies on number-theoretic proofs, real-world vulnerabilities rarely stem from mathematical flaws in core algorithms. Consequently, the theoretical foundations and practical failure modes exist at different abstraction levels. Taught in isolation, a purely theoretical approach produces students who can prove correctness (e.g., $M^{ed} \equiv M \pmod{n}$) but deploy insecure textbook RSA, whereas a purely practical approach yields students who execute `openssl req` commands without understanding certificate semantics or trust models.

Resolving this disconnect requires more than adding isolated topics like padding schemes to a theoretical course; it requires a coherent structural ordering. Specifications such as PKCS #1 v2.2 [1] or TLS 1.3 [2] often appear to students as arbitrary lists of constraints. However, each constraint addresses a specific historical vulnerability: OAEP was introduced because PKCS #1 v1.5 was vulnerable to chosen-ciphertext attacks [3], TLS 1.3 removed static RSA key transport due to recurring padding oracle variants [4], [5], and RSA blinding was mandated to mitigate remote timing side-channel attacks [6]. Taught alongside the vulnerabilities that prompted them, cryptographic mitigations become well-motivated solutions rather than arbitrary rules to memorize.

This paper formalizes this approach into a structured curriculum model based on three abstraction levels. Each level presents system mechanisms alongside the historical attacks that compromised them at that specific abstraction layer. Sections II–III detail the conceptual framework, Sections IV–VI describe its implementation using RSA at Warsaw University of Technology, and Section VII analyzes the trade-offs and potential limitations of the model. The contribution is methodological: while annual course evaluations show positive student feedback, this paper presents curriculum design analysis rather than formal control-group empirical evaluation.

II. Related Work

Prior literature on cryptography pedagogy generally falls into two categories. The first comprises *tool-centered* approaches: CrypTool 2 provides interactive visualizations and cryptanalysis modules enabling students to inspect algorithm mechanics without deep mathematical prerequisites [7], while SEED offers a repository of hands-on security laboratory exercises for direct adoption [8]. While these approaches demonstrate that practical manipulation aids comprehension, they do not specify instructional sequencing across abstraction levels. The second category comprises *stage-centered* frameworks: Feng et al. propose a progression covering theory, algorithm, practice, and application [9], addressing topic sequence through a bottom-up structure based on activity types.

Our curriculum model differs from existing frameworks in two key respects. First, the framework organizes content around *system abstraction levels* (product, implementation, and theory). As a result, core artifacts (such as RSA keys) are examined repeatedly across different system layers rather than partitioned by activity type. Second, historical vulnerabilities serve as explicit structural components: each abstraction level pairs a target mechanism with its corresponding historical exploit.