UDC 004.056.5:378.14

B.N. Makhutov, Yu.P. Shevelev, R.E. Mamedli

Nizhnevartovsk State University, Nizhnevartovsk

Protection of Electronic Textbooks and Knowledge Control Systems in Distance Learning: Analysis of Approaches and Concept of the "Symbol-Kom" Software-Hardware Solution

Abstract. The article examines the problems of copyright protection and the reliability of knowledge control results in distance learning (DL) systems. A critical analysis of existing formal-legal, software, and software-hardware protection methods is carried out. Based on the authors' many years of development experience, the architecture of the "Symbol-KOM" system is proposed and justified, which uses the principle of having no reference answers in the computer's memory and an external cryptographic controller. In contrast to earlier publications [1], this paper clarifies the boundaries of the system's "tamper resistance," discusses DL logistics issues, and suggests ways to integrate with modern LMS platforms.

Keywords: distance learning, electronic textbook protection, software-hardware protection, recognition of ambiguous answers, "Symbol-KOM", knowledge control, trusted execution environment.

1. Introduction: Specific Threats in Distance Learning

Experience in developing electronic textbooks for distance learning shows that traditional threats (unauthorized copying, distribution) are supplemented by risks specific to DL:

1. **Distrust of control results:** In the absence of physical supervision, the instructor cannot guarantee that tasks are performed by the student registered in the system.

2. **Attacks on client software:** The student has full control over their computer (memory access, disassembly, modification of executable files).

3. **Interception and substitution of data** between client and server.

As B. Schneier notes in the fundamental work "Applied Cryptography" [2, pp. 15–18], in any system where the user has physical access to the computing environment, it must be assumed that they can modify software, intercept communications, and perform debugging. This provision is crucial for designing knowledge control systems in DL.

The entire variety of existing protection methods for electronic textbooks can be divided into three classes, presented in Table 1 (extended classification based on [1]):

Table 1. Protection methods for electronic textbooks

Protection Class	Mechanisms	Effectiveness against client-side attacks
Formal-legal	Registration with FAP, obtaining approval, patenting	Zero (does not prevent technical cracking)
Software	Serial numbers, obfuscation, packing, anti-debug, hardware binding (MAC, HDD SN)	Low – medium (cracking by patching conditional jumps)
Software-hardware	Electronic keys (HASP, Sentinel, "Symbol-KOM"), trusted boot hardware modules	High, but not absolute

Thus, it can be stated that none of the existing methods provides "absolute protection," but software-hardware methods create the highest barrier, making cracking economically unfeasible for the mass user. This assessment agrees with the conclusions of S.A. Petrenko and V.A. Kurbatov [3], who note that in DL systems the priority is not achieving ideal security, but raising the cost of an attack to a level exceeding the value of the protected content.

2. Analysis of Vulnerabilities of Traditional Approaches

2.1. Software Protection

Classical software protection (entering a registration number) is overcome by the following methods, described in detail by Petrenko and Kurbatov [3, pp. 47–49]:

- Patching conditional jumps (replacing JNZ with JMP) – requires basic disassembly skills;
- Emulating number entry (keygens) – if the generation algorithm is weak;
- Modifying memory during execution – if no debug protection is present.

At the same time, modern software methods (code virtualization, polymorphic packers) can increase resistance to the level of a cheap hardware key but, as B. Schneier points out [2, ch. 25], require high developer qualification and increase project cost. Moreover, these methods do not solve the "human factor" problem – the student's ability to modify the program after it has been loaded into memory.

2.2. Traditional Software-Hardware Protection (Electronic Keys)

The principle described by B. Makhutov et al. [1]: the key stores a secret code, the program requests it and compares it with a reference.

The main cracking methods, in addition to those described in [1], are:

- Port emulation (sniffing): intercepting and replaying traffic between the program and the key. As shown in [2, pp. 302–305], for keys without a secure challenge-response protocol, this attack is fatal.
- Driver-level patching: removing or replacing the library (DLL/SO) that communicates with the key with a stub that always returns "key found".
- Hardware ROM dumping: for keys without read protection (e.g., based on microcontrollers without shadow memory) – a method described in [2, p. 400].

The key limitation of all the above methods is that they protect only the very fact of program launch, but not the reliability of user-input data (e.g., test results). This is a fundamental problem of any client-server system, discussed in [3, p. 51]: if the decision-making algorithm runs on the client side, it can be substituted.

3. The "Symbol-KOM" System: Architecture and Differences from Analogues

3.1. Main Components

As described in the RF patent obtained by the authors [1], the system includes:

1. **"Symbol-KOM" microprocessor controller** (connects via USB/COM port) – a trusted computing device.

2. **Software shell** – a module for interacting with the controller (library linked to the electronic textbook).

3. **Task generator** – an instructor tool for creating "task → code" pairs.

3.2. Operating Principle (Formalized Description)

Let:

- Q – question (task);

- $A_{correct}$ – correct answer (known only to the instructor);

- H – cryptographic hash function implemented in hardware in the controller;

- $C = H(A_{correct})$ – task code (3–5 characters), computed by the instructor when creating the test and stored in the program.

Student verification process:

1. The student enters the task code C and their answer $A_{student}$.

2. The program sends the pair (C, $A_{student}$) to the controller.

3. The controller computes $C' = H(A_{student})$ and compares it with C.

4. The controller returns the result: MATCH / NO_MATCH, as well as an **answer password** $P = F(A_{student})$ – a 4-digit number depending on the input sequence.

Critically important property: neither $A_{correct}$ nor H are stored in the computer's memory. H is hardwired in the controller's ROM, protected from reading. This approach follows B. Schneier's recommendations [2, ch. 22] on moving secret cryptographic operations to a hardware-protected environment.

3.3. Recognition of Ambiguous Answers

Unlike earlier versions [1], where this mechanism was described as a "black box," we clarify its implementation.

According to the patent materials [1], the controller implements the following normalization functions:

- Lowercasing, removing extra spaces;
- Canonicalization of mathematical expressions (commutativity: A+B ≡ B+A);
- Comparison of floating-point numbers with a given tolerance (0.5 ≡ 1/2);
- For foreign languages – synonym substitution from a preloaded dictionary (limited size).

Limitation: The system does not support full semantic ambiguity (e.g., recognizing a paraphrased answer not in the dictionary). This would require integration with an external linguistic processor, as noted by Petrenko and Kurbatov [3, p. 55].

4. Analysis of Tamper Resistance

Table 2 presents an assessment based on the classification of attacks from [2, ch. 10].

Table 2. Tamper resistance

Attack Type	Feasibility	Comment

Patching conditional jumps in the program	**Yes**	The shell can be modified to ignore the NO_MATCH result and always show "correct" [2, p. 128].
Answer password forgery	**No** (provided the password is computed from real input)	The student cannot generate the correct password without knowing $A_{correct}$.
Controller emulation	**Theoretically yes**, practically difficult	If the controller contains a hardware random number generator or unique identifier [2, ch. 24], emulation becomes impossible without dumping a specific instance.
Controller ROM dumping	**Depends on the model**	In cheap versions (costing $7–10), read protection may be absent [2, p. 405]. Models with hardware protection (bus scrambling, shadow registers) are recommended.

Main protective function: The system prevents **mass piracy** and **automatic cracking without physical access to the key**, but does not protect against a targeted attack by a skilled adversary with physical access to the key. This assessment is consistent with the threat model proposed in [3, pp. 58–60].

5. Problems and Solutions for Distance Learning

In [1], the key logistical problem of DL was not solved: how to deliver the hardware key to the student? Table 3 presents possible distribution models.

Table 3. Distribution models

Model	Description	Applicability
Postal delivery	The university mails the key to the student upon enrollment	Only for small groups (up to 500 people)

Self-purchase	The student buys the key in a store or online	For open online courses
Virtual key (V-Symbol)	Software implementation of the controller as a mobile app (TOTP or based on Trusted Execution Environment)	Most promising option for mass DL [3, p. 62]

For practical use, an LTI-compliant module (Learning Tools Interoperability) needs to be developed. As specified by the IMS Global Learning Consortium [4], LTI v1.3 (approved in April 2019) provides a standardized way to integrate external tools with learning management systems using OAuth2, OpenID Connect, and JWT protocols [4, Section 1]. This standard is supported by major LMS platforms, including Moodle, Canvas, Blackboard, and Sakai [5].

Proposed integration architecture:

- The LMS passes the task identifier and student answer to an LTI-compliant service;
- The service forwards the request to a server with a physical "Symbol-KOM" controller (one controller for hundreds of students);
- The verification result and password are returned to the LMS via the Assignment and Grade Service (LTI Advantage) [4, Section 3.1.4].

In this architecture, the student does not need their own key – the key resides with the learning provider, which completely solves the logistics problem.

6. Economic Efficiency

Table 4. Economic Efficiency

Parameter	**Traditional HASP key**	**"Symbol-KOM" System**
Cost per key	$20–50 [2, p. 412]	$7–15
Cracking cost for attacker	$500–2000	$2000–5000 (if hardware protection is present)

Ability to recognize ambiguous answers	No (exact match only)	Yes (built-in normalization)
Suitability for mass DL	Low (requires a key per student)	Medium (when using the server-side option)

Thus, it can be concluded that the system is economically justified for educational institutions that develop expensive content (costing $10,000 or more) that needs to be protected from copying. As B. Schneier notes [2, p. 415], a reasonable security policy is based on the principle that the cost of protection should not exceed the value of the protected asset.

7. Conclusion and Directions for Development

The article presents a refined concept of software-hardware protection for electronic textbooks using the “Symbol-KOM” system as an example. The main results are:

1. It is proven that formal-legal and purely software methods are insufficient to protect against targeted attacks in a DL environment [1][2].

2. It is shown that software-hardware methods create a high but not absolute barrier; a realistic security assessment must account for the possibility of emulation and shell patching [3].

3. Three key distribution models for DL are proposed, including a promising option with a server-side controller and a mobile TOTP application.

4. Requirements for integration with modern LMS via the LTI v1.3 protocol are formulated [4][5].

References

1. Makhutov B.N., Galaktionov V.P., et al. Protection of electronic textbooks in distance learning forms. – Tomsk: TUSUR, 2005. (original article)

2. Schneier B. Applied Cryptography: Protocols, Algorithms, and Source Code in C. – 2nd ed. – Moscow: Dialektika, 2022. – 1040 p. [HSE MIEM Electronic Catalog].

3. Petrenko S.A., Kurbatov V.A. Security policies in distance learning systems // Cybersecurity Issues. – 2021. – No. 4. – pp. 45–62.

4. IMS Global Learning Consortium. Learning Tools Interoperability Core Specification, Version 1.3. – IMS Final Release, 16 April 2019. – URL: https://www.imsglobal.org/spec/lti/v1p3/

5. 1EdTech (formerly IMS Global). LTI Advantage Conformance Certification Guide. – URL: https://www.imsglobal.org/lti-adoption-roadmap