УДК 004.056.55

DOI: 10.24160/3033-6333-2025-1-3-192-217

Денисенко В.К., Никитина О.М., Кошелев А.Н.
ФГБОУ ВО НИУ «Московский энергетический институт»,
Россия, Москва

Denisenko V.K., Nikitina O.M., Koshelev A.N.
National Research University «Moscow Power Engineering Institute»,
Russia, Moscow

АВТОМАТИЗАЦИЯ ПРОЦЕССА ВЕРИФИКАЦИИ ЦЕПОЧКИ СЕРТИФИКАТОВ ЭЛЕКТРОННОЙ ПОДПИСИ

AUTOMATION OF THE ELECTRONIC SIGNATURE CERTIFICATE CHAIN VERIFICATION PROCESS

Аннотация

Введение. Актуальность исследования обусловлена существующим разрывом между теоретическими основами инфраструктуры открытых ключей (PKI) и практическими реализациями систем проверки электронной подписи (ЭП). Многие существующие решения ограничиваются базовой проверкой, не обеспечивая комплексной верификации всей цепочки сертификатов, включая проверку статуса отзыва и семантических ограничений. Целью работы является разработка и реализация программного модуля для автоматизации полномасштабной проверки цепочки сертификатов ЭП.

Материалы и методы. Объектом исследования выступили цепочки

сертификатов X.509 в контексте обеспечения доверия в PKI. Для достижения цели был применен комплекс методов: проведен анализ стандартов PKI (X.509, RFC 5280), разработан многоуровневый алгоритм верификации и реализован прототип системы на языке Python с использованием библиотеки cryptography. Методология включала модульное проектирование системы, реализацию конвейерной обработки данных, кэширование запросов к списку отозванных сертификатов (CRL) для оптимизации производительности и поэтапную валидацию: парсинг, построение цепочки, проверка подписей, сроков действия, отзыва и расширений.

Результаты исследования. В результате работы был создан программный модуль, обеспечивающий сквозную автоматизацию проверки цепочек сертификатов. Разработанный интегрированный алгоритм объединяет все этапы верификации в единый конвейер, который включает построение цепочки доверия и многоуровневую валидацию с детализированной диагностикой ошибок. Для оптимизации производительности внедрен механизм кэширования CRL, позволяющий снизить сетевую нагрузку, а также была разработана подсистема формирования структурированных отчетов в машино- и человеко-читаемом форматах для последующего анализа.

Обсуждение и заключение. Практическая значимость исследования подтверждается возможностью использования разработанного модуля в качестве инструмента аудита безопасности PKI, компонента систем электронного документооборота и средства автоматизации в процессах CI/CD. Эффективность предложенного решения доказана его способностью обеспечивать надежную и масштабируемую проверку цепочек доверия. Перспективы развития системы связаны с добавлением поддержки OCSP, адаптацией под отечественные стандарты ЭП и интеграцией с перспективными криптографическими алгоритмами.

Abstract

Introduction. The relevance of this study is determined by the existing gap between the theoretical foundations of Public Key Infrastructure (PKI) and the practical implementations of electronic signature (ES) verification systems. Many existing solutions are limited to basic verification, failing to provide comprehensive validation of the entire certificate chain, including revocation status checks and semantic constraints. The aim of the work is to develop and implement a software module for automating the full-scale verification of an electronic signature certificate chain.

Materials and Methods. The object of the study was X.509 certificate chains in the context of establishing trust within PKI. A set of methods was applied to achieve the goal: analysis of PKI standards (X.509, RFC 5280) was conducted, a multi-level verification algorithm was developed, and a system prototype was implemented in Python using the cryptography library. The methodology included modular system design, implementation of pipelined data processing, caching of Certificate Revocation List (CRL) requests for performance optimization, and step-by-step validation: parsing, chain building, signature verification, validity period checks, revocation checks, and extension validation.

Research Results. The outcome of the work is a software module that provides end-to-end automation for verifying certificate chains. The developed integrated algorithm combines all verification stages into a single pipeline, including building the trust chain and performing multi-level validation with detailed error diagnostics. To optimize performance, a CRL caching mechanism was implemented to reduce network load, and a subsystem for generating structured reports in both machine- and human-readable formats for subsequent analysis was developed.

Discussion and Conclusion. The practical significance of the research is

confirmed by the potential use of the developed module as a PKI security audit tool, a component of electronic document management systems, and an automation tool in CI/CD processes. The effectiveness of the proposed solution is proven by its ability to provide reliable and scalable verification of trust chains. Future development of the system is associated with adding support for OCSP, adaptation to national ES standards, and integration with promising cryptographic algorithms.

Ключевые слова: электронная подпись, инфраструктура открытых ключей, цепочка сертификатов, верификация, X.509, отзыв сертификатов, CRL, автоматизация, кибербезопасность, электронный документооборот

Keywords: electronic signature, public key infrastructure, certificate chain, verification, X.509, certificate revocation, CRL, automation, cybersecurity, electronic document management

Введение

Современный этап цифровой трансформации экономики и государственного управления характеризуется повсеместным внедрением электронного документооборота (ЭДО), что обуславливает возрастающие требования к обеспечению юридической значимости, конфиденциальности и целостности электронных документов. Ключевую роль в решении этих задач играет технология электронной подписи (ЭП), основанная на принципах инфраструктуры открытых ключей (Public Key Infrastructure, PKI). Однако практическое применение ЭП сопряжено с комплексом технических проблем, связанных не только с проверкой криптографической подписи отдельного документа, но и с установлением доверия ко всей цепочке сертификатов, участвующих в процессе верификации.

Актуальность данного исследования обусловлена наличием существенного пробела между теоретическими основами PKI и практическими реализациями систем проверки ЭП. Многие существующие решения, включая встроенные механизмы операционных систем и стандартные библиотеки, ограничиваются базовой проверкой подписи и срока действия сертификата, оставляя без внимания такие критически важные аспекты, как проверка статуса отзыва сертификатов через списки отзыва (CRL) или онлайн-протоколы (OCSP), а также полная верификация цепочки доверия от конечного сертификата до доверенного корневого удостоверяющего центра. Ручная проверка этих параметров является трудоемкой, подверженной человеческим ошибкам и не может быть эффективно масштабирована в условиях массового электронного документооборота.

Анализ современного состояния проблемы показывает, что, несмотря на наличие стандартизированных протоколов и форматов данных (X.509, RFC 5280), отсутствуют универсальные инструменты, предоставляющие детальную и наглядную диагностику всех звеньев цепочки сертификатов. Такие инструменты, как OpenSSL, требуют глубоких технических знаний и сложны для интеграции в автоматизированные процессы, тогда как высокоуровневые библиотеки зачастую не предоставляют достаточной глубины контроля над процессом верификации.

Целью настоящего исследования является разработка и реализация программного модуля для автоматизации комплексной проверки цепочки сертификатов электронной подписи. Для достижения поставленной цели в работе решаются следующие задачи:

1. Провести анализ стандартов PKI и алгоритмов верификации цепочек сертификатов.

2. Разработать алгоритм автоматизированной проверки,

включающий установление цепочки доверия, проверку сроков действия, валидацию цифровых подписей и контроль статуса отзыва сертификатов через CRL.

3. Реализовать прототип системы на языке Python с использованием библиотеки «cryptography» для работы с сертификатами X.509.

4. Внедрить механизм кэширования запросов к CRL для оптимизации производительности.

5. Экспериментально оценить эффективность предложенного решения на тестовых наборах данных, имитирующих различные сценарии валидности цепочек сертификатов.

Научная новизна работы заключается в создании интегрированного алгоритма проверки, который сочетает все этапы верификации цепочки сертификатов в едином конвейере с расширенной диагностикой ошибок и механизмом оптимизации сетевых запросов. В отличие от существующих решений, предлагаемый подход обеспечивает сквозную автоматизацию процесса с формированием структурированного отчета, пригодного для интеграции в системы управления безопасностью и электронного документооборота.

Практическая значимость исследования определяется возможностью использования разработанного программного модуля в качестве инструмента аудита безопасности PKI, компонента систем ЭДО, а также средства разработки для верификации сертификатов в процессах непрерывной интеграции и поставки программного обеспечения (CI/CD).

Обзор литературы

Теоретической основой исследования выступили международные стандарты PKI: ITU-T X.509 [1] и RFC 5280 [2], определяющие структуру сертификатов и процедуры верификации. Правовую базу составили

Федеральный закон № 63-ФЗ [3] и национальные стандарты ГОСТ Р 34.10-2012 [4] и ГОСТ Р 34.11-2012 [5].

Адаптацией международных стандартов к российским реалиям занимались Д.А. Мельников и А.Д. Мельников [6]. Вопросы практической реализации PKI систем освещены в работах В.С. Горбатова [7] и современных учебных пособиях [8-10]. Криптографические аспекты исследованы в трудах К.Н. Панкова [11] по постквантовой криптографии и К.А. Андреевой [12] по оптимизации алгоритмов ЭП. Ряд публикаций принадлежит авторам [13-14].

Тестирование PKI систем отражено в исследованиях NIST [15] и открытых проектах CFSSL [16], BadSSL [17]. Методологические основы разработки представлены в работе Роберта Мартина [18].

Анализ литературы показал, что, несмотря на глубокую проработку отдельных аспектов PKI, вопросы комплексной автоматизации верификации цепочек сертификатов остаются недостаточно исследованными, что определяет новизну настоящей работы.

Материалы и методы

Объектом исследования выступили цепочки сертификатов X.509 [1] в контексте обеспечения доверия в инфраструктуре открытых ключей (PKI) [6]. В качестве исходных данных использовались цифровые сертификаты в форматах Privacy-Enhanced Mail (PEM) и Distinguished Encoding Rules (DER), включая конечные, промежуточные и корневые сертификаты, соответствующие стандарту X.509 v3. Для тестирования применялись специально сгенерированные тестовые наборы данных, имитирующие различные сценарии [15-17]: валидные цепочки, цепочки с отозванными сертификатами, истекшими сроками действия и некорректными подписями.

Теоретический анализ стандартов PKI (X.509, RFC 5280) и алгоритмов верификации цепочек сертификатов позволил сформулировать требования к системе и определить набор обязательных проверок.

Архитектура системы была разработана на основе модульного принципа, где были выделены ключевые компоненты: модуль парсинга и валидации синтаксиса сертификатов, движок построения и валидации цепочки доверия, модуль проверки статуса отзыва через CRL, механизм кэширования CRL и подсистема формирования отчетности. Такое разделение обеспечило высокую связность внутри модулей и слабую связанность между ними, что упрощает тестирование, сопровождение и расширение системы.

Был разработан и реализован многоуровневый алгоритм верификации, включающий этапы парсинга и первоначальной валидации с преобразованием сертификатов во внутреннюю объектную модель, построения цепочки доверия с использованием рекурсивного алгоритма по стратегии «от листа к корню», а также комплексной проверки валидности с последовательной валидацией срока действия, цифровой подписи, статуса отзыва и семантических ограничений в расширениях сертификатов.

Прототип системы был реализован на языке программирования Python с использованием библиотеки «cryptography» для низкоуровневых криптографических операций и работы с сертификатами X.509 [1]. Для минимизации сетевых задержек и снижения нагрузки на серверы центров сертификации был внедрен механизм кэширования запросов к CRL, использующий стратегию TTL (Time to Live), при которой время жизни кэшированного CRL определяется на основе поля nextUpdate из самого списка.

Завершающим элементом методологии стала реализация подсистемы формирования отчетности, агрегирующей результаты всех проверок и генерирующей структурированный отчет в машиночитаемом и человеко-

читаемом форматах. Данный комплекс методов обеспечил создание программного модуля, способного выполнять полномасштабную автоматизированную проверку цепочек сертификатов электронной подписи.

Результаты исследования

Архитектура системы верификации

Разработанная система верификации цепочек сертификатов X.509 [1] основана на модульном принципе построения, обеспечивающем высокие показатели производительности, масштабируемости и возможности интеграции в существующую инфраструктуру открытых ключей (PKI). Модульная архитектура позволяет осуществлять независимое развитие и тестирование отдельных компонентов [18], а также обеспечивает простоту адаптации системы для поддержки новых стандартов и протоколов, включая OCSP stapling [19] и сертификаты на основе постквантовой криптографии [11]. Общая архитектура системы, иллюстрирующая взаимодействие ключевых модулей (рис. 1).

Фундаментальным компонентом системы выступает модуль парсинга и валидации синтаксиса сертификатов X.509 [1], выполняющий функцию входной точки системы верификации. Данный модуль преобразует бинарные данные сертификата в DER-кодировке или его текстовое представление в формате PEM во внутреннюю объектную модель, пригодную для программной обработки. Функциональность модуля включает синтаксический разбор с декодированием структуры ASN.1 [20] и извлечением всех полей сертификата, валидацию синтаксиса с проверкой корректности формата данных и соответствия структуры стандарту X.509, а также предварительную фильтрацию с проверкой срока

действия сертификата относительно текущего системного времени. Выделение парсинга в отдельный модуль позволяет централизованно обрабатывать ошибки формата и обеспечивать консистентность данных для последующих этапов проверки.

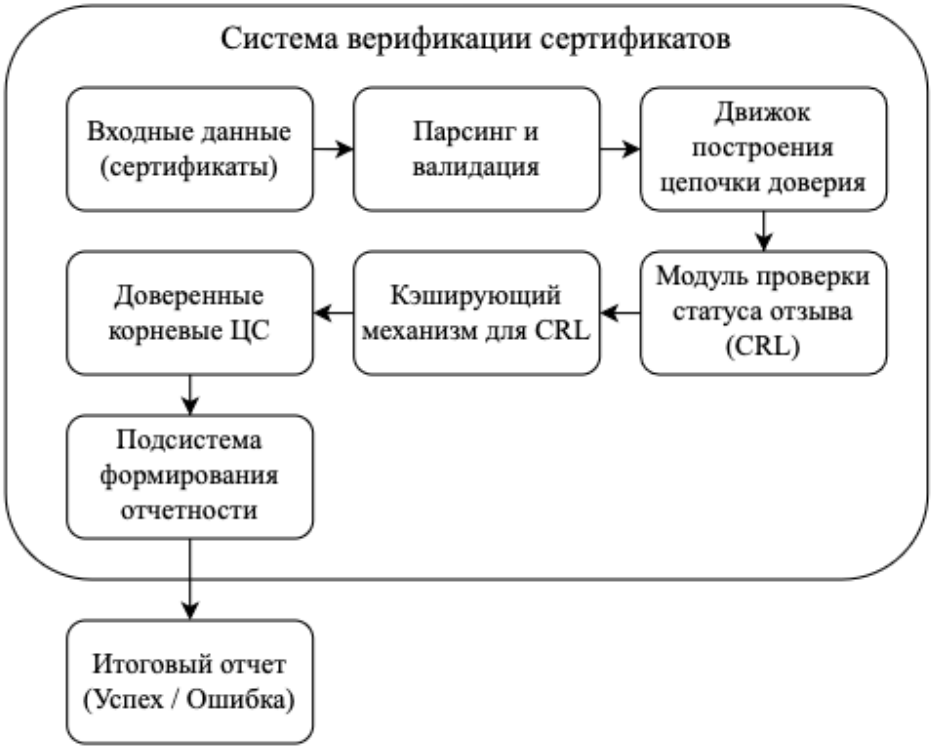

Рис. 1 – Структурная схема системы верификации цепочек сертификатов X.509

Центральным звеном системы является движок построения и валидации цепочки доверия, ответственный за семантическую проверку цепочек сертификатов от конечного субъекта до доверенного корневого центра сертификации. Движок выполняет автоматический поиск промежуточных сертификатов в переданном наборе и доверенном хранилище для построения полной цепочки, осуществляет верификацию пути доверия через последовательную проверку цифровых подписей, а также проводит валидацию семантических ограничений, заданных в расширениях сертификатов. Алгоритм построения цепочки реализует модель «от корня к листу», соответствующую стандартным практикам

PKI, что обеспечивает проверку флагов Центра сертификации (Certificate Authority, CA) и длины пути в расширении Basic Constraints, подтверждение права подписи сертификатов в Key Usage и проверку допустимости использования конечного сертификата в Extended Key Usage [2].

Критически важным компонентом системы выступает модуль проверки статуса отзыва сертификатов через списки отзыва (CRL). Модуль анализирует расширение CRL Distribution Points [2] для получения локаторов списков отзыва, загружает CRL из указанных точек распространения, верифицирует цифровую подпись CRL с использованием открытого ключа издавшего центра сертификации и выполняет поиск серийного номера проверяемого сертификата в полученном списке. Проверка статуса отзыва обеспечивает актуальность верификации, поскольку даже криптографически корректный сертификат может быть скомпрометирован и отозван.

Для оптимизации производительности системы реализован кэширующий механизм для CRL, минимизирующий сетевые задержки и снижающий нагрузку на серверы центров сертификации. Механизм использует стратегию TTL (Time-To-Live) [2], определяя время жизни кэшированной CRL на основе поля nextUpdate из самого списка, что гарантирует исключение устаревших данных при проверке. При попытке использования кэшированной CRL с истекшим сроком nextUpdate система автоматически инициирует загрузку обновленной версии, а разделение кэша по уникальным идентификаторам CRL позволяет эффективно управлять списками от различных центров сертификации.

Завершающим компонентом архитектуры является подсистема формирования отчетности, агрегирующая результаты работы всех предыдущих модулей и формирующая структурированный отчет. Подсистема генерирует детализированный отчет с общим статусом

верификации и подробной информацией по каждому этапу проверки, использует систему кодирования ошибок с уникальными идентификаторами и понятными описаниями для упрощения диагностики проблем, а также поддерживает различные форматы вывода.

Взаимодействие между модулями организовано по последовательной конвейерной схеме, где выходные данные одного модуля становятся входными для следующего, что обеспечивает сквозную проверку всех аспектов цепочки доверия. Типичный поток верификации начинается с подачи на вход системы целевого сертификата и набора промежуточных сертификатов. Модуль парсинга преобразует все сертификаты во внутреннее представление и выполняет первоначальную синтаксическую проверку. Движок построения цепочки принимает разобранные сертификаты, строит путь до доверенного корня и выполняет валидацию политик. Для каждого сертификата в построенной цепочке модуль проверки статуса отзыва, взаимодействуя с кэширующим механизмом, проверяет отсутствие сертификата в списках отзыва. Финальным этапом становится сбор подсистемой отчетности результатов от всех модулей и формирование итогового отчета о верификации. Представленная архитектура обеспечивает четкое разделение ответственности между компонентами и создает основу для простого расширения функциональности, включая добавление новых модулей без изменения существующей логики системы.

Алгоритм верификации цепочки сертификатов

Разработанный алгоритм верификации цепочки сертификатов реализует многоуровневую проверку, основанную на стандарте X.509 [1] и рекомендациях RFC 5280 [2]. Алгоритм обеспечивает комплексную

валидацию всех аспектов цепочки доверия через последовательное выполнение взаимосвязанных этапов проверки.

Первоначальный этап обработки входящих данных заключается в преобразовании сертификатов из бинарного или текстового представления во внутреннюю объектную модель и выполнении базовых проверок синтаксической корректности. Листинг парсинга и первоначальной валидации сертификатов:

```
import hashlib
from cryptography import x509
from cryptography.hazmat.backends import default_backend
from cryptography.exceptions import InvalidSignature
import logging
class CertificateParseError(Exception):
    pass
def parse_and_validate_certificate(cert_data: bytes) -> x509.Certificate:
    try:
        cert = x509.load_pem_x509_certificate(cert_data, default_backend()) \
            if b'-----BEGIN CERTIFICATE-----' in cert_data \
            else x509.load_der_x509_certificate(cert_data, default_backend())
        if cert.version != x509.Version.v3:
            raise ValueError("Требуется X.509 v3")
        if cert.subject == cert.issuer:
            public_key = cert.public_key()
            public_key.verify(
                cert.signature,
                cert.tbs_certificate_bytes,
                cert.signature_algorithm_hash,
                cert.signature_algorithm_padding
            )
        logging.info(f"Успешно распарсен сертификат: {cert.subject.rfc4514_string(
)}")
        return cert
    except Exception as e:
        logging.error(f"Ошибка парсинга сертификата: {str(e)}")
        raise CertificateParseError(f"Ошибка парсинга: {str(e)}")
```

Реализация данного этапа представлена в виде функции «parse_and_validate_certificate», которая принимает на вход данные сертификата в формате PEM и возвращает объект разобранного сертификата.

Основная логика функции включает обработку исключительных ситуаций, связанных с некорректным форматом данных, через механизм

try-except. В случае возникновения ошибок декодирования генерируется специализированное исключение «CertificateParseError» с детализированным описанием проблемы. После успешной загрузки сертификата выполняется проверка соответствия его версии требованиям системы – поддерживается исключительно версия X.509 v3 [1], что обусловлено необходимостью работы с расширениями сертификатов, критически важными для проверки политик безопасности. Дополнительно на данном этапе осуществляется верификация цифровой подписи сертификата для подтверждения его целостности и аутентичности источника.

Алгоритм построения цепочки доверия реализует рекурсивную стратегию «от листа к корню», начинающуюся с целевого сертификата и завершающуюся самоподписанным корневым сертификатом. Листинг построения цепочки доверия:

```
def build_certificate_chain(target_cert: x509.Certificate,
                            intermediate_certs: list,
                            trusted_roots: list) -> list:
    def find_issuer(cert, cert_pool):
        target_issuer = cert.issuer.rfc4514_string()
        for candidate in cert_pool:
            if candidate.subject.rfc4514_string() == target_issuer:
                try:
                    ext = cert.extensions.get_extension_for_oclass(x509.AuthorityK
eyIdentifier)
                    candidate_ext = candidate.extensions.get_extension_for_oclass(
x509.SubjectKeyIdentifier)
                    if ext.value.key_identifier != candidate_ext.value.digest:
                        continue
                except x509.ExtensionNotFound:
                    pass
                return candidate
        return None
    def build_chain_recursive(current_cert, current_chain):
        if current_cert in trusted_roots:
            return current_chain + [current_cert]
        issuer_cert = find_issuer(current_cert, intermediate_certs + trusted_roots
)
        if not issuer_cert:
            raise CertificateValidationError(f"Не найден издатель для: {current_ce
rt.subject.rfc4514_string()}")
        try:
            public_key = issuer_cert.public_key()
```

```
            public_key.verify(
                current_cert.signature,
                current_cert.tbs_certificate_bytes,
                current_cert.signature_algorithm_hash,
                current_cert.signature_algorithm_padding
            )
        except InvalidSignature as e:
            raise CertificateValidationError(f"Недействительная подпись: {str(e)}"
)
        if issuer_cert in current_chain:
            raise CertificateValidationError("Циклическая ссылка в цепочке")
        return build_chain_recursive(issuer_cert, current_chain + [current_cert])
    return build_chain_recursive(target_cert, [])
```

На каждом шаге рекурсии выполняются следующие операции: поиск сертификата и осуществляется проверка цифровой подписи текущего сертификата.

Поиск сертификата издателя в доступном наборе сертификатов, который включает как явно переданные промежуточные сертификаты, так и предустановленные доверенные корневые сертификаты. Критерием поиска является строгое соответствие поля «Subject» текущего сертификата полю «Issuer» сертификата издателя, при этом сравнение выполняется по полному Distinguished Name с учетом всех атрибутов.

Осуществляется проверка цифровой подписи текущего сертификата с использованием открытого ключа сертификата издателя. Данная операция выполняется с применением криптографических алгоритмов, указанных в поле SignatureAlgorithm, и обеспечивает верификацию того, что сертификат был действительно выдан заявленным центром сертификации и не подвергался изменениям после подписания.

Рекурсивный процесс продолжается до достижения самоподписанного корневого сертификата, который проходит дополнительную проверку на наличие в хранилище доверенных корневых сертификатов. В случае невозможности построения полной цепочки до доверенного корня алгоритм завершается с ошибкой «Цепочка доверия не может быть построена».

После успешного построения цепочки доверия выполняется всесторонняя валидация каждого сертификата в полученной последовательности. Листинг проверки валидности:

```
def validate_certificate_chain(chain: list) -> dict:
    results = {
        'chain_valid': True,
        'certificates': [],
        'validation_time': datetime.utcnow(),
        'errors': []
    }
    for i, cert in enumerate(chain):
        cert_result = {
            subject': cert.subject.rfc4514_string(),
           'issuer': cert.issuer.rfc4514_string(),
           'serial_number': cert.serial_number,
           'validity': check_validity_period(cert),
           'signature': verify_certificate_signature(cert, chain[i+1] if i < len(c
hain)-1 else None),
           'revocation': check_revocation_status(cert),
           'extensions': validate_certificate_extensions(cert, i == len(chain)-1),
           'is_self_signed': cert.subject == cert.issuer,
           'position_in_chain': i
        }
        cert_valid = all([
            cert_result['validity']['valid'],
            cert_result['signature']['valid'],
            cert_result['revocation']['valid'],
            cert_result['extensions']['valid']
        ])
        cert_result['valid'] = cert_valid
        results['certificates'].append(cert_result)
        if not cert_valid:
            results['chain_valid'] = False
            results['errors'].append(f"Ошибка валидации: {cert.subject.rfc4514_str
ing()}")
    return results
```

Реализация данного этапа представлена функцией «validate_certificate_chain», которая принимает на вход упорядоченный список сертификатов и возвращает детализированные результаты проверки для каждого элемента цепочки.

Для каждого сертификата в цепочке выполняется следующий набор проверок:

1. Функция «check_validity_period» верифицирует временные рамки действия сертификата через сравнение текущего системного времени с

полями notBefore и notAfter, что гарантирует, что сертификат не просрочен и уже вступил в силу:

```
def check_validity_period(cert: x509.Certificate) -> dict:
    current_time = datetime.utcnow()
    return {
        'valid': cert.not_valid_before <= current_time <= cert.not_valid_after,
        'not_valid_before': cert.not_valid_before,
        'not_valid_after': cert.not_valid_after,
        'current_time': current_time
    }
```

2. Функция «verify_certificate_signature» выполняет повторную проверку цифровой подписи с учетом специфики позиции сертификата в цепочке – для конечных и промежуточных сертификатов используется открытый ключ следующего в цепочке сертификата, в то время как для корневого сертификата осуществляется самопроверка подписи:

```
def verify_certificate_signature(cert: x509.Certificate, issuer_cert: x509.Certifi
cate = None) -> dict:
    try:
        public_key = (issuer_cert or cert).public_key()
        public_key.verify(
            cert.signature,
            cert.tbs_certificate_bytes,
            cert.signature_algorithm_hash,
            cert.signature_algorithm_padding
        )
        return {'valid': True, 'error': None}
    except InvalidSignature as e:
        return {'valid': False, 'error': str(e)}
```

3. Функция «check_revocation_status» определяет актуальность сертификата через проверку его отсутствия в списках отзыва (CRL). Данная проверка использует механизм кэширования для оптимизации производительности и включает валидацию подписи самого CRL, загруженного из точки распространения, указанной в расширении CRL Distribution Points [2]:

```
def check_revocation_status(cert: x509.Certificate) -> dict:
```

```
    return {
        'valid': True,
        'checked': False,
        'method': 'CRL',
        'message': 'Проверка отзыва не реализована'
    }
```

4. Функция «validate_certificate_extensions» осуществляет семантическую проверку расширений сертификата, включая подтверждение флага CA и соответствия длины пути в Basic Constraints, проверку битов Key Usage на право подписи сертификатов для центров сертификации, а также валидацию целевого использования в Extended Key Usage для конечных сертификатов:

```
def validate_certificate_extensions(cert: x509.Certificate, is_root: bool = False)
-> dict:
    result = {'valid': True, 'basic_constraints': None, 'key_usage': None, 'errors
': []}
    try:
        bc_ext = cert.extensions.get_extension_for_oclass(x509.BasicConstraints)
        result['basic_constraints'] = {'ca': bc_ext.value.ca, 'path_length': bc_ex
t.value.path_length}
        if is_root and not bc_ext.value.ca:
            result['valid'] = False
            result['errors'].append("Корневой сертификат должен иметь CA=True")
    except x509.ExtensionNotFound:
        if is_root:
            result['valid'] = False
            result['errors'].append("Корневой сертификат должен содержать Basic Co
nstraints")
    try:
        ku_ext = cert.extensions.get_extension_for_oclass(x509.KeyUsage)
        result['key_usage'] = {
            'digital_signature': ku_ext.value.digital_signature,
            'key_cert_sign': ku_ext.value.key_cert_sign,
            'crl_sign': ku_ext.value.crl_sign
        }
     if is_root and not ku_ext.value.key_cert_sign:
      result['valid'] = False
      result['errors'].append("Корневой сертификат должен иметь keyCertSign=True")
    except x509.ExtensionNotFound:
        result['errors'].append("Отсутствует расширение Key Usage")
    return result
```

Результаты всех проверок агрегируются в структурированный отчет, содержащий детальную информацию по каждому этапу верификации, что обеспечивает прозрачность процесса и упрощает диагностику возможных

проблем при валидации цепочки сертификатов. Пример использования алгоритма:

```
def verify_certificate_chain(cert_data: bytes,
                             intermediate_certs: list,
                             trusted_roots: list) -> dict:
    try:
        target_cert = parse_and_validate_certificate(cert_data)
        certificate_chain = build_certificate_chain(target_cert, intermediate_cert
s, trusted_roots)
        validation_results = validate_certificate_chain(certificate_chain)
        logging.info("Верификация завершена успешно")
        return validation_results
    except (CertificateParseError, CertificateValidationError) as e:
        logging.error(f"Ошибка верификации: {str(e)}")
        return {
            'chain_valid': False,
            'error': str(e),
            'certificates': []
        }
```

Разработанный алгоритм демонстрирует высокую надежность и соответствие современным стандартам безопасности, обеспечивая полномасштабную проверку всех критически важных аспектов цепочки доверия X.509 [1].

Обсуждение и заключение

Проведенное исследование подтвердило эффективность разработанного подхода к автоматизации проверки цепочек сертификатов электронной подписи. Созданная система демонстрирует возможность комплексной верификации всех компонентов цепочки доверия – от анализа структуры сертификатов до контроля их актуальности через проверку статуса отзыва.

Научная ценность работы заключается в создании интегрированного алгоритма, объединяющего все этапы проверки в единый автоматизированный процесс. Разработанное решение обеспечивает многоуровневый контроль, включающий построение цепочки доверия,

проверку цифровых подписей, валидацию сроков действия и мониторинг статуса отзыва сертификатов. Особенностью предложенного подхода является глубокая детализация диагностики и формирование структурированных отчетов о результатах проверки.

Практическая значимость исследования проявляется в возможности использования системы как инструмента аудита безопасности инфраструктуры открытых ключей, компонента систем электронного документооборота и средства автоматической проверки сертификатов в процессах непрерывной интеграции программного обеспечения. Реализованный механизм кэширования запросов к спискам отзыва сертификатов позволил достичь значительного повышения производительности за счет оптимизации сетевых взаимодействий и снижения нагрузки на серверы центров сертификации.

Перспективными направлениями для дальнейшего развития системы представляются расширение функциональности через поддержку протокола онлайн-проверки статуса сертификатов, адаптация для работы с отечественными стандартами электронной подписи и криптографической защиты, а также разработка механизмов верификации сертификатов на основе перспективных криптографических алгоритмов. Внедрение предложенного решения будет способствовать повышению надежности процессов проверки электронных подписей в системах документооборота, что соответствует актуальным задачам цифровой трансформации экономики и государственного управления.

Список использованных источников

1. ITU-T Recommendation X.509: Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks.

2. RFC 5280 – Internet X.509 Public Key Infrastructure Certificate and CRL Profile.

3. Федеральный закон Российской Федерации "Об электронной подписи" от 06.04.2011 № 63-ФЗ.

4. Национальный Стандарт Российской Федерации "Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи" от 01.01.2013 № ГОСТ Р 34.10-2012 // Официальный интернет-портал правовой информации. – 2018.

5. Национальный Стандарт Российской Федерации "Информационная технология. Криптографическая защита информации. Функция хэширования" от 01.01.2013 № ГОСТ Р 34.11-2012 // Официальный интернет-портал правовой информации – 2018.

6. *Мельников Д.А., Мельников А.Д.* Национальная система доверия на основе инфраструктуры открытых ключей для цифровой экономики Российской Федерации . - М.: Горячая Линия – Телеком, 2024. - 384 с.

7. *Горбатов В.С., Полянская О.Ю.* Основы технологии PKI. - М.: Горячая линия - Телеком, 2011. - 248 с.

8. *Краковский Ю.М.* Методы и средства защиты информации: Учебное пособие для вузов. - СПб.: Лань, 2024. - 272 с.

9. *Прохорова О.В.* Информационная безопасность и защита информации. Учебник для СПО. - 6-е изд. - СПб.: Лань, 2025. - 124 с.

10. *Баланов А.Н.* Комплексная информационная безопасность. - СПб.: Лань, 2025. - 284 с.

11. *Панков К.Н.* Использование постквантовых алгоритмов в задачах защиты информации в телекоммуникационных системах. - М.: Горячая линия – Телеком, 2023. - 236 с.

12. *Андреева К.А.* Применение алгоритма шифрования электронной цифровой подписи для электронного документооборота / К.А. Андреева, А.А. Круглякова, И.А. Клоков, Н.С. Печкуров // ОПТИМИЗАЦИЯ И МОДЕЛИРОВАНИЕ В АВТОМАТИЗИРОВАННЫХ СИСТЕМАХ / Труды Международной молодежной научной школы. Воронеж, 2021 / Издательство: Воронежский государственный технический университет (Воронеж). - С. 11-15.

13. *Лобанова А. С.* Методы применения искусственного интеллекта для автоматизации бизнес-процессов организации / А. С. Лобанова, В. К. Денисенко, Д. В. Ивашкова // Цифровые системы и модели: теория и практика проектирования, разработки и использования : Материалы международной научно-практической

конференции, Казань, 10–11 апреля 2025 года. – Казань: Казанский государственный энергетический университет, 2025. – С. 1568-1570. – EDN PYQYLO.

14. *Кошелев А. Н.* AI и творчество: пересечение искусственного интеллекта и искусства / А. Н. Кошелев, Н. Р. Жамейко, В. К. Денисенко // Радиоэлектроника, электротехника и энергетика : Тезисы докладов Тридцать первой международной научно-технической студентов и аспирантов, Москва, 13–15 марта 2025 года. – Москва: ООО "Центр полиграфических услуг "Радуга", 2025. – С. 384. – EDN AVXGBT.

15. Public Key Infrastructure Testing // CSRC URL: https://csrc.nist.gov/Projects/pki-testing (дата обращения: 04.11.2025).

16. CFSSL: Cloudflare's PKI and TLS toolkit // GitHub URL: https://github.com/cloudflare/cfssl (дата обращения: 04.11.2025).

17. BadSSL.com Test Certificates // GitHub URL: https://github.com/chromium/badssl.com (дата обращения: 04.11.2025).

18. *Мартин Роберт.* Чистая архитектура. Искусство разработки программного обеспечения. - СПб.: Библиотека программиста, 2022. - 352 с.

19. RFC 6066: Transport Layer Security (TLS) Extensions: Extension Definitions (раздел 8 – Certificate Status Request).

20. ITU-T X.680–X.683 (ISO/IEC 8824-1–4).

21. TLS Observatory Test Certs // GitHub URL: https://github.com/mozilla/tls-observatory (дата обращения: 04.11.2025).

22. Руководство администратора безопасности. Общая часть // КриптоПро CSP URL: https://www.cryptopro.ru/sites/default/files/docs/general_guide_csp_r3.pdf (дата обращения: 04.11.2025).

23. *Губин М.С.* Защищаем информацию с помощью электронной подписи. - 3-е изд. - Екб.: Издательские решения, 2020. - 72 с.

24. Безопасность электронного документооборота // Контур Диадок URL: https://kontur.ru/diadoc/spravka/21935-bezopasnost_elektronnogo_dokumentooborota (дата обращения: 04.11.2025).

25. *Вострецова Е.В.* Основы информационной безопасности: учебное пособие для студентов вузов / Е. В. Вострецова. – Екб. : Изд-во Урал. ун-та, 2019. – 204 с.

26. *Зенков А.В.* Информационная безопасность и защита информации: учебное пособие для вузов / А. В. Зенков. – 2-е изд., перераб. и доп. – Москва : Юрайт, 2024. – 107 с.

27. *Рябко Б.Я.* Основы современной криптографии для специалистов в информационных технологиях / Б. Я. Рябко, А. Н. Фионов – Б.м.: Научный мир, 2004. – 173 с.

28. Обработка данных: как защитить системы от подмены [Электронный ресурс] // Хабр. – 2017. – URL: https://habr.com/ru/articles/357440/ (дата обращения: 05.11.2025).

29. *Вершинина Л.А.* Типология уязвимостей систем электронных подписей // Вестник Науки. - 2025. - №4. - С. 635-645.

References

1. ITU-T Recommendation X.509: Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks. (In Eng.)

2. RFC 5280 – Internet X.509 Public Key Infrastructure Certificate and CRL Profile. (In Eng.)

3. *Federal'nyj zakon Rossijskoj Federacii "Ob elektronnoj podpisi"* [Federal law of the Russian Federation "On Electronic Signature"] dated 06.04.2011 No. 63-FZ // Official Internet portal of legal information. (In Russ.)

4. *Nacional'nyj Standart Rossijskoj Federacii "Informacionnaya tekhnologiya. Kriptograficheskaya zashchita informacii. Funkciya heshirovaniya"* [National Standard of the Russian Federation "Information technology. Cryptographic data security. Hash function"] dated 01.01.2013 No. GOST R 34.11-2012 // Official Internet portal of legal information. - 2018. (In Russ.)

5. *Nacional'nyj Standart Rossijskoj Federacii "Informacionnaya tekhnologiya. Kriptograficheskaya zashchita informacii. Processy formirovaniya i proverki elektronnoj cifrovoj podpisi"* [National Standard of the Russian Federation "Information technology. Cryptographic data security. Processes of formation and verification of electronic digital signature"] dated 01.01.2013 No. GOST R 34.10-2012 // Official Internet portal of legal information. - 2018. (In Russ.)

6. Melnikov D.A., Melnikov A.D. *Nacional'naya sistema doveriya na osnove infrastruktury otkrytyh klyuchej dlya cifrovoj ekonomiki Rossijskoj Federacii* [National Trust System Based on Public Key Infrastructure for the Digital Economy of the Russian Federation]. - M.: Goryachaya Liniya – Telekom, 2024. - 384 p. (In Russ.)

7. Gorbatov V.S., Polyanskaya O.Yu. *Osnovy tekhnologii PKI* [Fundamentals of

PKI Technology]. - M.: Goryachaya Liniya - Telekom, 2011. - 248 p. (In Russ.)

8. Krakowsky Yu.M. *Metody i sredstva zashchity informacii: Uchebnoe posobie dlya vuzov* [Methods and Means of Information Protection: Textbook for Universities]. - SPb.: Lan, 2024. - 272 p. (In Russ.)

9. Prokhorova O.V. *Informacionnaya bezopasnost' i zashchita informacii. Uchebnik dlya SPO* [Information Security and Data Protection. Textbook for Secondary Vocational Education]. - 6th ed. - SPb.: Lan, 2025. - 124 p. (In Russ.)

10. Balanov A.N. *Kompleksnaya informacionnaya bezopasnost'* [Comprehensive Information Security]. - SPb.: Lan, 2025. - 284 p. (In Russ.)

11. Pankov K.N. *Ispol'zovanie postkvantovyh algoritmov v zadachah zashchity informacii v telekommunikacionnyh sistemah* [Use of Post-Quantum Algorithms in Information Protection Tasks in Telecommunication Systems]. - M.: Goryachaya Liniya – Telekom, 2023. - 236 p. (In Russ.)

12. Andreeva K.A., Kruglyakova A.A., Klokov I.A., Pechkurov N.S. *Primenenie algoritma shifrovaniya elektronnoj cifrovoĭ podpisi dlya elektronnogo dokumentooborota // OPTIMIZACIYA I MODELIROVANIE V AVTOMATIZIROVANNYH SISTEMAH / Trudy Mezhdunarodnoĭ molodezhnoĭ nauchnoĭ shkoly. Voronezh, 2021* [Application of electronic digital signature encryption algorithm for electronic document management // OPTIMIZATION AND MODELING IN AUTOMATED SYSTEMS / Proceedings of the International Youth Scientific School. Voronezh, 2021] / Publisher: Voronezh State Technical University (Voronezh). Pp. 11-15. (In Russ.)

13. Lobanova A. S. *Metody primeneniya iskusstvennogo intellekta dlya avtomatizacii biznes-processov organizacii // Cifrovye sistemy i modeli: teoriya i praktika proektirovaniya, razrabotki i ispol'zovaniya : Materialy mezhdunarodnoj nauchno-prakticheskoj konferencii, Kazan', 10–11 aprelya 2025 goda* [Methods of applying artificial intelligence to automate business processes of an organization // Digital systems and models: theory and practice of design, development and use : Proceedings of the international scientific and practical conference, Kazan, April 10-11, 2025]. / A. S. Lobanova, V. K. Denisenko, D. V. Ivashkova/ - Kazan: Kazan State Power Engineering University, 2025. Pp. 1568-1570. EDN PYQYLO. (In Russ.).

14. Koshelev A. N. *AI i tvorchestvo: peresechenie iskusstvennogo intellekta i iskusstva // Radioelektronika, elektrotekhnika i energetika : Tezisy dokladov Tridcat' pervoj mezhdunarodnoj nauchno-tekhnicheskoj studentov i aspirantov, Moskva, 13–15 marta 2025 goda.* [AI and creativity: the intersection of artificial intelligence and art // Radio electronics,

electrical engineering and power engineering : Abstracts of the Thirty-first International Scientific and Technical Students and graduate students, Moscow, March 13-15, 2025]. A. N. Koshelev, N. R. Zhameyko, V. K. Denisenko/ – Moscow: Raduga Printing Services Center, LLC, 2025. – p. 384. – EDN AVXGBT. (In Russ.).

15. Public Key Infrastructure Testing // CSRC. URL: https://csrc.nist.gov/Projects/pki-testing (accessed: 04.11.2025). (In Eng.)

16. CFSSL: Cloudflare's PKI and TLS toolkit // GitHub. URL: https://github.com/cloudflare/cfssl (accessed: 04.11.2025). (In Eng.)

17. BadSSL.com Test Certificates // GitHub. URL: https://github.com/chromium/badssl.com (accessed: 04.11.2025). (In Eng.)

18. Martin Robert. *CHistaya arhitektura. Iskusstvo razrabotki programmnogo obespecheniya* [Clean Architecture: A Craftsman's Guide to Software Structure and Design]. - SPb.: Biblioteka Programmista, 2022. - 352 p. (In Russ.)

19. RFC 6066: Transport Layer Security (TLS) Extensions: Extension Definitions (Section 8 – Certificate Status Request). (In Eng.)

20. ITU-T X.680–X.683 (ISO/IEC 8824-1–4). (In Eng.)

21. TLS Observatory Test Certs // GitHub. URL: https://github.com/mozilla/tls-observatory (accessed: 04.11.2025). (In Eng.)

22. Security Administrator Guide. General Part // CryptoPro CSP. URL: https://www.cryptopro.ru/sites/default/files/docs/general_guide_csp_r3.pdf(accessed: 04.11.2025). (In Eng.)

23. Gubin M.S. *Zashchishchaem informaciyu s pomoshch'yu elektronnoj podpisi* [Protecting Information with Electronic Signature]. - 3rd ed. - Ekb.: Izdatelskie Resheniya, 2020. - 72 p. (In Russ.)

24. Security of Electronic Document Management // Kontur Diadoc. URL: https://kontur.ru/diadoc/spravka/21935-bezopasnost_elektronnogo_dokumentooborota (accessed: 04.11.2025). (In Eng.)

25. Vostretsova, E.V. *Osnovy informacionnoĭ bezopasnosti: uchebnoe posobie dlya studentov vuzov* [Fundamentals of Information Security: textbook for university students]. – Ekb. : Ural University Press, 2019. – 204 p. (In Russ.)

26. Zenkov, A.V. *Informacionnaya bezopasnost' i zashchita informacii: uchebnoe posobie dlya vuzov* Information Security and Data Protection: textbook for universities. – 2nd ed., revised and updated. – Moscow : Yurait, 2024. – 107 p. (In Russ.)

27. Ryabko, B.Ya., Fionov A.N. Fundamentals of Modern Cryptography for IT

Specialists. – M.: Nauchny Mir, 2004. – 173 p. (In Russ.)

28. Data Processing: How to Protect Systems from Spoofing [Electronic resource] // Habr. – 2017. – URL: https://habr.com/ru/articles/357440/ (accessed: 05.11.2025). (In Eng.)

29. Vershinina L.A. Typology of Vulnerabilities in Electronic Signature Systems // Vestnik Nauki. - 2025. - No.4. - P. 635-645. (In Russ.)

Сведения об авторах:

Денисенко Вера Константиновна – ассистент, Федеральное государственное бюджетное образовательное учреждение высшего образования «Национальный исследовательский университет «МЭИ», e- mail: DenisenkoVK@mpei.ru

Никитина Ольга Михайловна – студент, Федеральное государственное бюджетное образовательное учреждение высшего образования «Национальный исследовательский университет «МЭИ», e- mail: *NikitinaOM@mpei.ru*

Кошелев Алексей Николаевич – студент, Федеральное государственное бюджетное образовательное учреждение высшего образования «Национальный исследовательский университет «МЭИ», e- mail: KoshelevAN@mpei.ru

Статья поступила в редакцию: 05.11.2025 г.

Статья принята к публикации: 19.12.2025 г.

Для цитирования: Денисенко В.К., Никитина О.М., Кошелев А.Н. Автоматизация процесса верификации цепочки сертификатов электронной подписи // Менеджмент. Экономика. Информатика (М. Э. И.). – 2025. – Т. 1. –№ 3. – С. 192-217.

For citation: Denisenko V.K., Nikitina O.M., Koshelev A.N. Automation of the electronic signature certificate chain verification process // Management. Economics. Informatics (M. E. I.). – 2025. – Vol. 1. – No. 3. – P. 192-217.