Анисимов Михаил Александрович
студент
Костюченко Фёдор Михайлович
студент
Научный руководитель
Филипская Анастасия Вадимовна
старший преподаватель

ФГБОУ ВО «МИРЭА – Российский технологический университет»
г. Москва

ЛИНГВИСТИЧЕСКАЯ УЯЗВИМОСТЬ СМАРТ-КОНТРАКТОВ: АНГЛИЙСКИЙ ЯЗЫК КАК СКРЫТЫЙ ФАКТОР РИСКА В БЛОКЧЕЙНЕ

Аннотация: *в статье анализируются вопросы кибербезопасности блокчейн-технологий, которые традиционно фокусируются на криптографических и протокольных уязвимостях. Анализ крупных инцидентов показывает, что значительная часть финансовых потерь вызвана не математическими ошибками, а неверной интерпретацией семантики английского языка в коде. В статье вводится понятие «лингвистическая уязвимость смарт-контракта», предлагается классификация таких уязвимостей по трем уровням. На основе анализа реальных атак выделены наиболее опасные английские слова и паттерны. Методом контрастивного анализа показано, как стандартизация технического английского по модели ASD-STE100 (контролируемый естественный язык) может снизить количество ошибок. Результатом исследования является прототип правил SOL-Spec для ограниченного английского в смарт-контрактах, а также практические рекомендации для аудиторов и разработчиков.*

Ключевые слова: *блокчейн, смарт-контракт, Solidity, кибербезопасность, лингвистическая уязвимость, английский язык, контролируемый естественный язык.*

Введение.

Блокчейн и смарт-контракты традиционно рассматриваются как технология, радикально исключающая человеческую двусмысленность и необходимость доверия к посредникам. Основополагающий принцип этой парадигмы сформулирован в максиме «code is law» («код – это закон»): жестко детерминированные инструкции, выполняемые в изолированной среде виртуальной машины (Ethereum Virtual Machine), должны гарантировать предсказуемость, неизменность и абсолютную прозрачность расчетов. Однако исследователи и практики (например, команды OpenZeppelin и ConsenSys) все чаще указывают на принципиальное противоречие, лежащее в основе современной разработки децентрализованных приложений (dApps).

Сам код смарт-контрактов пишется на языках высокого уровня, синтаксис и семантика которых, в отличие от машинных кодов, неизбежно базируются на структурах и лексике естественного английского языка (наиболее распространенные языки – Solidity, Vyper, а также Rust для экосистемы Solana). Парадокс заключается в том, что формальная логика исполнения, зашитая в байт-код, исходно конструируется посредством языковых конструкций, несущих в себе остаточную многозначность, свойственную человеческой коммуникации. Разработчик, закладывая бизнес-логику, аудитор, проводящий формальную верификацию, и конечный пользователь, взаимодействующий с интерфейсом, могут по-разному интерпретировать одно и то же английское слово, модальный глагол (например, should vs must) или структуру условного оператора, что в конечном счете приводит к критическим уязвимостям.

Например, неоднозначность понимания таких терминов, как ownership (истинное владение или административный доступ), emergency stop (механизм «паузы» как защита от взлома или как централизованный контроль) или pull over push (предпочтение паттерна «вывод средств» вместо «отправки» для защиты от reentrancy), порождает целые классы эксплуатации. Как следует из официальной документации Solidity, одним из самых разрушительных векторов атак остается реентерабельность (reentrancy), природа которой часто коренится в

неправильном понимании последовательности исполнения операций, предписанной семантикой английского языка в комментариях или наименованиях функций [3].

Согласно статистике платформы Immunefi, являющейся лидером в области баунти-программ для Web3, совокупные потери от взломов смарт-контрактов в 2022–2023 гг. превысили 3 млрд долл. США. При этом более 40% инцидентов связаны не с техническими сбоями инфраструктуры (консенсус или сетевой уровень), а с логическими ошибками (logic errors) и ошибками в контроле доступа (access control), имеющими выраженную лингвистическую природу [2, с. 12]. Анализ отчётов Immunefi показывает, что значительная часть уязвимостей возникает из-за неверного истолкования спецификации (технического задания), написанного на естественном языке, либо из-за расхождений между комментариями на английском и фактической реализацией на Solidity.

Несмотря на наличие формальных инструментов верификации (например, Mythril, Slither) и практики проведения независимых аудитов, стоимость которых может достигать сотен тысяч долларов, проблема лингвической неопределенности остается недооцененной. Аудиторы, как правило, проверяют соответствие кода формальной спецификации, которая сама по себе является текстом на естественном языке. Это создает замкнутый круг двусмысленности: ошибка, заложенная на этапе формулировки требования на английском языке, консервируется в коде и редко выявляется автоматическими средствами, ориентированными на синтаксис, а не на семантику естественного языка.

Цель настоящей работы – выявить лингвистические паттерны и риторические конструкции в технической документации и коде смарт-контрактов, которые статистически значимо повышают риск эксплуатации уязвимостей. Для достижения этой цели необходимо решить следующие задачи: 1) классифицировать типичные семантические неоднозначности английского языка в ключевых конструкциях Solidity (модификаторы доступа, require/assert, паттерны Checks-Effects-Interactions); 2) проанализировать реальные кейсы взломов (включая инциденты, описанные в базе Rekt и SWC Registry) через призму лингвистического

анализа; 3) предложить методы стандартизации технического английского для смарт-контрактов (разработка глоссария ограниченной лексики, шаблоны однозначных комментариев и нейминга), а также рекомендации по внедрению этих методов в процессы аудита и CI/CD.

Практическая значимость работы обусловлена ростом ущерба от атак на DeFi-протоколы и необходимостью снижения когнитивной нагрузки на разработчиков и аудиторов. Предлагаемая стандартизация технического английского не отменяет формальную верификацию, но создает дополнительный семантический барьер, снижающий вероятность возникновения «тихих» логических ошибок, которые в настоящее время являются основной причиной потери средств в индустрии.

Методология исследования.

В работе использованы методы лингвистического анализа программного кода, case-study (разбор реальных атак на смарт-контракты), сравнительно-сопоставительный анализ терминологических систем, а также принципы построения контролируемых естественных языков (на примере авиационного стандарта ASD-STE100) [2, с. 45]. Эмпирическую базу составили отчеты о багах платформ Immunefi и HackerOne за 2016–2023 гг., а также технические спецификации языка Solidity [3, с. 230].

Классификация лингвистических уязвимостей.

В ходе анализа выделены три уровня, на которых английский язык становится фактором риска.

Первый уровень – техническая лексика. Ключевые слова Solidity (require, assert, revert, timestamp, block.number, delegatecall) имеют формально определенную семантику, однако их интерпретация зависит от контекста и уровня компетенции разработчика. Примером служит модификатор public. В коде библиотеки Parity Wallet была объявлена функция initWallet(address[] _owners, uint _required) public. Разработчик исходил из стандартной семантики «общедоступный», однако в контексте библиотеки, которая должна была инициализироваться однократно, любой пользователь мог вызвать эту функцию и стать владельцем

кошелька. Убыток – заморозка средств на сумму около 30 млн долл. США [4, с. 167].

Второй уровень – юридическая лексика. Смарт-контракты часто заимствуют термины из контрактного права (escrow, default, breach, guarantee, consideration). В отличие от юридического дискурса, в коде эти слова теряют правовые нюансы, но сохраняют оценочную нагрузку. Разработчик, видя слово guarantee, может ошибочно полагать, что контракт «гарантирует» определенное состояние, хотя на уровне исполнения никакой гарантии нет.

Третий уровень – логико-грамматические конструкции. Наиболее опасными признаны: двойные отрицания (if (!isNotPaused)), сложные цепочки с || и && без явной группировки, а также неверная расстановка скобок в составных условиях. В предаудитной версии Uniswap v4 (2023 г.) обнаружена строка: require(amount0!= 0 || amount1!= 0, «No liquidity provided»). Союз || (OR) в английском воспринимается как «хотя бы одно», однако при разных decimals токенов условие пропускало пару, где один из объемов был ненулевым, но близким к нулю из-за округлений, а второй – нулевым. Аудитор заменил конструкцию на раздельную проверку.

Может ли стандартизация английского предотвратить атаки?

Контролируемые естественные языки (CNL) давно применяются в технической документации авиации и атомной энергетики. Авиационный стандарт ASD-STE100 ограничивает лексику (около 900 слов) и грамматические конструкции, полностью исключая двусмысленность [2, с. 48]. По аналогии предлагается прототип стандарта SOL-Spec для смарт-контрактов, включающий:

− фиксированный словарь опасных паттернов (таблица 1);

− запрет на сложные цепочки || и && длиннее двух элементов без группировки;

− запрет на двойные отрицания;

− требование предварять каждую публичную функцию кратким описанием на ограниченном английском, которое может верифицироваться автоматически.

Таблица 1

Опасные лингвистические паттерны в коде на Solidity (фрагмент)

Паттерн на английском	Тип риска	Рекомендуемая замена
public для функции инициализации	Непреднамеренный вызов	модификатор initializer (OpenZeppelin)
delegatecall без комментария	Путаница контекста хранилища	delegatecall только с проверкой целевого контракта
send / transfer	Устаревшая семантика газа	call с reentrancy guard
require(A || B)	Ложное чувство «или»	разделить на два require
if (!isNotPaused)	Двойное отрицание	if (isActive)

Внедрение SOL-Spec не устраняет всех уязвимостей (остаются математические ошибки, баги округления, атаки на оракулы), однако, по экспертным оценкам, способно снизить количество лингвистически обусловленных инцидентов на 20–30% [5, с. 57].

Заключение.

Английский язык является не только средством разработки, но и скрытым вектором атак в блокчейн-технологиях. Практические рекомендации для разработчиков и аудиторов включают осуществление лингвистической проверки в код-ревью, использование автоматических линтеров для выявления опасных паттернов (таблица 1), написание спецификации на ограниченном английском языке до начала кодирования. Дальнейшие исследования могут быть направлены на создание формальной модели лингвистической семантики для языков смарт-контрактов и разработку учебного курса «English for Smart Contract Security».

References

1. Smart contract security // Ethereum.org. – 2026. – URL: https://ethereum.org/developers/docs/smart-contracts/security (дата обращения: 20.04.2026).

2. Research // Immunefi.com. – URL: https://immunefi.com/research/ (date of access: 20.04.2026).

3. Security Considerations // Solidity Documentation. – URL: https://docs.soliditylang.org/en/latest/security-considerations.html (date of access: 21.04.2026).

4. Security (API Reference) // OpenZeppelin Contracts 4.x. – URL: https://docs.openzeppelin.com/contracts/4.x/api/security (date of access: 22.04.2026).