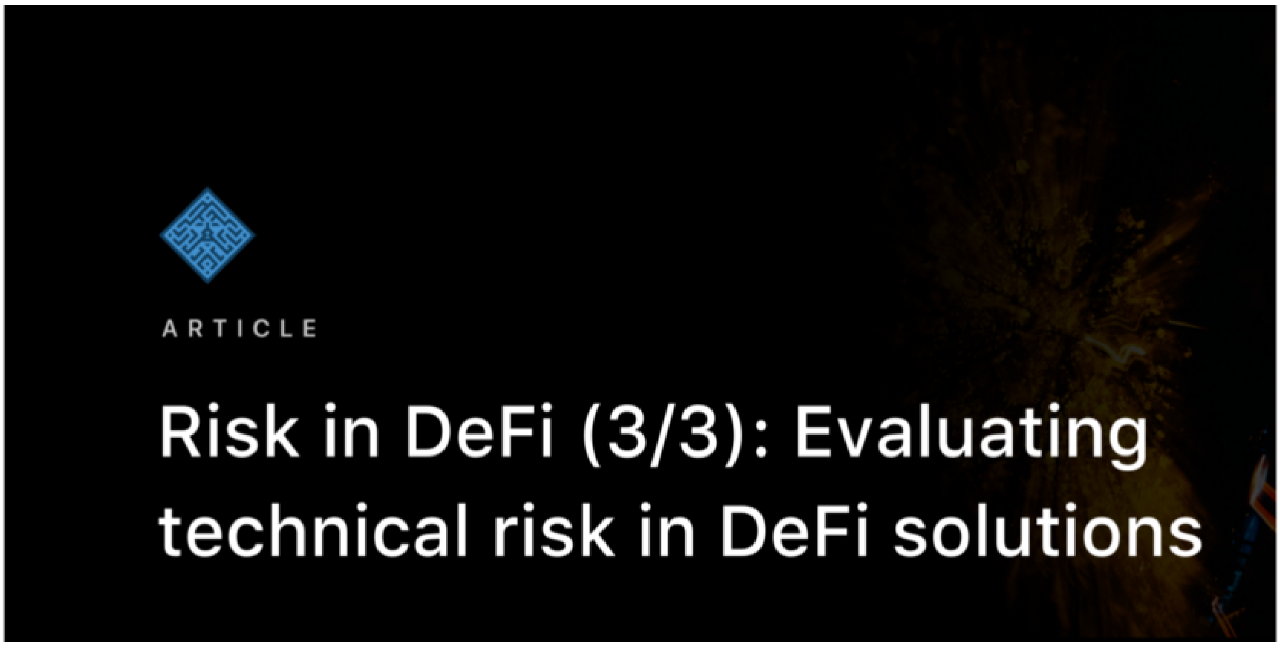
# Risk in DeFi (3/3): E DeFi solutions

Plutus DeFi
Jan 14 · 6 min read


ARTICLE
Risk in DeFi (3/3): Evaluating technical risk in DeFi solutions

Welcome to the third installment in our series on “Risk in DeFi!” This article will explore the technical risks associated with using decentralized financial (DeFi) solutions.

In Part 1 of our series, we looked at procedural risk and highlighted some of the ways hackers might manipulate users into compromising their security and best practices for avoiding such issues.

Part 2 of our series explored financial risk and how to assess the potential risks and rewards associated with investment opportunities in DeFi and how to compare them with the opportunities available in the world of traditional finance.

**Technical risk** refers to problems in the hardware, software, and protocols that inform a product or service and pose an actual threat.

Read on to learn more about the major technical risks that can compromise new technologies and the common risks as contracts.

## Risk in DeFi

Before we talk about technical risk, he to consider when dealing with DeFi sc

In addition to technical risks, the other forms of risk to consider are financial and procedural risk.

**Financial risk** involves comparing the potential risk and reward associated with various investment opportunities with the goal of constructing a successful portfolio in accordance with a person or organization’s tolerance for risk.

**Procedural risk** examines the ways in which users might be manipulated into using the product in unintended ways that could compromise their security.

## Technical Risk

When examining DeFi solutions for technical vulnerabilities we have to consider the “normal” software, the smart contracts deployed to the blockchain or distributed ledger, and the hardware used to interface with all of that software.

It’s important to distinguish risks from vulnerabilities. Anything that poses an actual threat to an individual or organization can be considered a risk, while vulnerabilities refer to weaknesses that can be exploited.

In other words, vulnerabilities may expose worthless commands or data that don’t pose a problem but risks refer to situations in which there’s definitely a problem.

## Categorizing Risks

Whether we’re thinking about risks in “normal” software, smart contracts, or hardware, the major risks to watch out for include:

- **Memory safety** considers problems that can interrupt access to memory, including access errors, uninitialized variables, and memory leaks. Common examples include buffer overflows, dangling pointers, and stack exhaustion.

- **Race conditions** occur when the outcome of an event depends on the sequence or timing of other events. The potent consider when working with smar

- **Improper API use** can occur due easily as carelessness.

- **Improper use case and exceptio** and insufficient testing to ensure adequate consideration for

- **Input/Output handling** must be set up correctly to ensure that the exchange of data between devices and files can't be manipulated or exploited in nefarious ways.

## Software Risks

Here's a list of the risks to consider when evaluating whether or not a piece of software may be considered trustworthy:

- **Dedicated Denial-of-Service (DOS or DDOS)** is a common technique for disrupting a website or service. A DDOS attack involves flooding something like a website, application, or server with useless requests that prevent real users from being able to submit their own.

The transaction fee associated with many blockchains and distributed ledgers helps prevent DDOS attacks by making it expensive to submit a lot of transactions.

That said, a DDOS attack could hypothetically be executed against a blockchain or a website or application interfacing with a blockchain and disrupt access to a decentralized application within a given time period.

- **Injection** occurs when software that allows users to manage data using the command line or makes use of SQL databases contains vulnerabilities that allow an attacker to access commands and change data in unintended ways.

- **Overflow** occurs because processes like data buffering and integer calculation are executed in specific ways that may cause unintended results if code doesn't account for them.

For example, buffer overflow happens when a piece of software tries to add more data to the buffer than the storage capacity of the buffer allows for, a problem that can corrupt data, execute malicious code, and disable the software.

- **Uncontrolled Format Strings** can become a problem because the forms that allow users to submit data need to be se format strings to execute function

## Smart Contract Risks

One of the things that distinguish a sn “regular” software runs on a compute individual computers.

Smart contracts, on the other hand, get deployed to a blockchain or some other form of distributed ledger as a transaction and provide rules for how the “regular” software and the distributed ledger interact.

As a relatively new technology, best practices for smart contract security remain under development but here are some of the major vulnerabilities to look out for:

- **Forcibly Sending Ether to a Contract** can cause unintended consequences in smart contracts, including self-destruction of the contract.
- **Front-Running** refers to the potential for attackers to watch the mempool for transactions that have not yet been included in a block and use that information to manipulate the outcome of an event.
- For example, an attacker might see a transaction executed by a decentralized exchange and quickly submit a transaction or series of transactions to execute a trade of their own with a higher gas fee to “front-run” the first transaction and make sure the second transaction gets recorded before the first.
- **Insufficient gas griefing** happens when attackers initiate transactions with enough gas to complete the transaction but not enough to complete a subcall included as part of the transaction.
- **Integer Overflow and Underflow** can occur when a set of code fails to consider that the *uint* variable has a maximum value of $2^{256}$, after which it resets to zero.
- **Reentrancy** happens when an external contract hijacks the intended flow of events to make unintended changes to data. The infamous DAO hack in 2016 involving the theft of $50 million in Ether happened as a result of an attacker or group of attackers exploiting a reentrancy vulnerability.

- **Timestamp Dependence** refers to the fact that miners can manipulate the timestamp of a block. Functions th to manipulation if not coded prop

## Hardware Risks

Unless a decentralized application (dA or "crypto debit card," it's easy to forg the hardware used to interact with dApps.

Kraken Security Labs recently released a report on a vulnerability discovered in Keepkey's hardware wallet solution enabling someone to expose the private key with roughly $75 and 15 minutes of physical access to the device.

Here are some of the major vulnerabilities to consider when evaluating the risk associated with the hardware components involved in a product:

- **Incompatibility** occurs when hardware devices aren't set up to work together or don't have the proper drivers installed to enable them to work together.
- **Power problems** can occur when electrical outlets provide an inconsistent supply of voltage. Whether it's a spike in voltage, low voltages, or shifts in frequency, fluctuations in the power supply can have harmful consequences.
- **Sensitivity** to humidity, dust, degradation, and unprotected storage can cause a product to behave abnormally or stop functioning altogether.
- **Voltage glitching (also known as fault injection)** involves manipulating hardware like clocks and temperature sensors in ways that trick the software into acting in unintended ways. The vulnerability in Keepkey hardware wallets exposed by Kraken involves voltage glitching.

## Ready to DeFi?

Throughout this series we've gained an understanding of the procedural, financial, and technical risks associated with DeFi solutions.

Now that we know the major vulnerabilities to look out for, we're better equipped to evaluate the opportunities available!

If you still have questions, feel free to ask our community on Telegram and let us know what interests you about DeFi!

*If you value privacy in your DeFi solutio*
*our anonymous decentralized lending aggregator.*

*Please sign up on our website at https://plutusdefi.com/.* 😀

Decentralized Decentralization Smart Contract Security Smart Contracts
Ethereum Blockchain