MASTER OF ENGINEERING IN TELECOMMUNICATIONS

A Comprehensive Evaluation of Ethereum, Solana, and Avalanche in Addressing the Blockchain Trilemma

Author: Álvaro Bayona Bultó

Supervisor: Javier Matanza Domingo

Madrid

2023

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

A Comprehensive Evaluation of Ethereum, Solana, and Avalanche in Addressing the Blockchain Trilemma

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2022/23 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: Alvaro Bayona Bulto Fecha: 28/08/2023

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: Javier Matanza Domingo Fecha: 28/08/2023

AUTORIZACIÓN PARA LA DIGITALIZACIÓN, DEPÓSITO Y DIVULGACIÓN EN RED DE PROYECTOS FIN DE GRADO, FIN DE MÁSTER, TESINAS O MEMORIAS DE BACHILLERATO

1º. Declaración de la autoría y acreditación de la misma.

El autor D. Alvaro Bayona Bulto DECLARA ser el titular de los derechos de propiedad intelectual de la obra: A Comprehensive Evaluation of Ethereum, Solana, and Avalanche in Addressing the Blockchain Trilemma, que ésta es una obra original, y que ostenta la condición de autor en el sentido que otorga la Ley de Propiedad Intelectual.

2º. Objeto y fines de la cesión.

Con el fin de dar la máxima difusión a la obra citada a través del Repositorio institucional de la Universidad, el autor **CEDE** a la Universidad Pontificia Comillas, de forma gratuita y no exclusiva, por el máximo plazo legal y con ámbito universal, los derechos de digitalización, de archivo, de reproducción, de distribución y de comunicación pública, incluido el derecho de puesta a disposición electrónica, tal y como se describen en la Ley de Propiedad Intelectual. El derecho de transformación se cede a los únicos efectos de lo dispuesto en la letra a) del apartado siguiente.

3º. Condiciones de la cesión y acceso

Sin perjuicio de la titularidad de la obra, que sigue correspondiendo a su autor, la cesión de derechos contemplada en esta licencia habilita para:

a) Transformarla con el fin de adaptarla a cualquier tecnología que permita incorporarla a internet y hacerla accesible; incorporar metadatos para realizar el registro de la obra e incorporar "marcas de agua" o cualquier otro sistema de seguridad o de protección.
b) Reproducirla en un soporte digital para su incorporación a una base de datos electrónica, incluyendo el derecho de reproducir y almacenar la obra en servidores, a los efectos de garantizar su seguridad, conservación y preservar el formato.
c) Comunicarla, por defecto, a través de un archivo institucional abierto, accesible de modo libre y gratuito a través de internet.
d) Cualquier otra forma de acceso (restringido, embargado, cerrado) deberá solicitarse expresamente y obedecer a causas justificadas.
e) Asignar por defecto a estos trabajos una licencia Creative Commons.
f) Asignar por defecto a estos trabajos un HANDLE (URL *persistente)*.

4º. Derechos del autor.

El autor, en tanto que titular de una obra tiene derecho a:

a) Que la Universidad identifique claramente su nombre como autor de la misma
b) Comunicar y dar publicidad a la obra en la versión que ceda y en otras posteriores a través de cualquier medio.
c) Solicitar la retirada de la obra del repositorio por causa justificada.
d) Recibir notificación fehaciente de cualquier reclamación que puedan formular terceras personas en relación con la obra y, en particular, de reclamaciones relativas a los derechos de propiedad intelectual sobre ella.

5º. Deberes del autor.

El autor se compromete a:

a) Garantizar que el compromiso que adquiere mediante el presente escrito no infringe ningún derecho de terceros, ya sean de propiedad industrial, intelectual o cualquier otro.
b) Garantizar que el contenido de las obras no atenta contra los derechos al honor, a la intimidad y a la imagen de terceros.
c) Asumir toda reclamación o responsabilidad, incluyendo las indemnizaciones por daños, que pudieran ejercitarse contra la Universidad por terceros que vieran infringidos sus derechos e

intereses a causa de la cesión.

d) Asumir la responsabilidad en el caso de que las instituciones fueran condenadas por infracción de derechos derivada de las obras objeto de la cesión.

6º. Fines y funcionamiento del Repositorio Institucional.

La obra se pondrá a disposición de los usuarios para que hagan de ella un uso justo y respetuoso con los derechos del autor, según lo permitido por la legislación aplicable, y con fines de estudio, investigación, o cualquier otro fin lícito. Con dicha finalidad, la Universidad asume los siguientes deberes y se reserva las siguientes facultades:

- La Universidad informará a los usuarios del archivo sobre los usos permitidos, y no garantiza ni asume responsabilidad alguna por otras formas en que los usuarios hagan un uso posterior de las obras no conforme con la legislación vigente. El uso posterior, más allá de la copia privada, requerirá que se cite la fuente y se reconozca la autoría, que no se obtenga beneficio comercial, y que no se realicen obras derivadas.
- La Universidad no revisará el contenido de las obras, que en todo caso permanecerá bajo la responsabilidad exclusive del autor y no estará obligada a ejercitar acciones legales en nombre del autor en el supuesto de infracciones a derechos de propiedad intelectual derivados del depósito y archivo de las obras. El autor renuncia a cualquier reclamación frente a la Universidad por las formas no ajustadas a la legislación vigente en que los usuarios hagan uso de las obras.
- La Universidad adoptará las medidas necesarias para la preservación de la obra en un futuro.
- La Universidad se reserva la facultad de retirar la obra, previa notificación al autor, en supuestos suficientemente justificados, o en caso de reclamaciones de terceros.

Madrid, a 28 de agosto de 2023

ACEPTA

Fdo Alvaro Bayona Bulto

MASTER OF ENGINEERING IN TELECOMMUNICATIONS

A Comprehensive Evaluation of Ethereum, Solana, and Avalanche in Addressing the Blockchain Trilemma

Author: Álvaro Bayona Bultó

Supervisor: Javier Matanza Domingo

Madrid

2023

EVALUACIÓN DE ETHEREUM, SOLANA Y AVALANCHE EN RELACIÓN CON EL "BLOCKCHAIN TRILEMMA"

Autor: Bayona Bulto, Alvaro.
Director: Matanza Domingo, Javier.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

Este estudio implica un análisis exhaustivo de tres tecnologías líderes en blockchain: Ethereum, Solana y Avalanche. El enfoque se centra en cómo cada una aborda el "Blockchain Trilemma" al equilibrar la descentralización, la seguridad y la velocidad. Además, el proyecto incluye la creación de una base de datos y un panel de control para monitorear métricas clave en tiempo real.

Palabras clave: Blockchain, Ethereum, Solana, Avalanche, Blockchain Trilemma.

1. Introducción

En este proyecto, nos embarcamos en una exploración exhaustiva del papel estratégico de GSR como un creador de mercado y colaborador en el ecosistema dentro del dinámico mundo de las criptomonedas. Con una gran mayoría de sus actividades centradas en la creación de mercado, que constituye un 80% de su alcance operativo, dirigimos nuestra atención hacia desentrañar las complejidades de esta función crítica. Los creadores de mercado desempeñan un papel fundamental al infundir liquidez en el mercado a través de un proceso continuo de cotización de precios de compra y venta, asegurando así transacciones sin problemas y evitando cuellos de botella de liquidez.

El objetivo principal de nuestro estudio es proporcionar a GSR una recomendación imparcial y bien fundamentada para la selección del ecosistema de blockchain más prometedor. En el corazón de esta recomendación yace una evaluación integral del "Blockchain Trilemma", un desafío fundamental que involucra la interacción entre tres atributos esenciales: descentralización, seguridad y escalabilidad. En particular, investigamos cómo tres plataformas de blockchain prominentes: Ethereum, Solana y Avalanche, navegan por este intrincado trilema. Nuestra evaluación abarca análisis tanto cualitativos como cuantitativos, siendo estos últimos facilitados por la creación de un panel de control dinámico en tiempo real.

2. Evaluación de la Posición del Trilema de Cada Tecnología

Ethereum

En términos de **seguridad**, Ethereum enfrenta posibles vulnerabilidades debido a su tiempo de bloque reducido, lo que puede aumentar el riesgo de ataques a largo plazo y exponer vulnerabilidades de contratos inteligentes. El cambio a Prueba de Participación (PoS, por sus siglas en inglés) introduce el desafío de "nada en riesgo", donde los validadores podrían validar bloques conflictivos para maximizar recompensas, comprometiendo potencialmente la seguridad de la red. La innovadora característica de

contratos inteligentes de Ethereum, si bien es revolucionaria, introduce preocupaciones únicas de seguridad, ya que cualquier fallo en el código puede ser explotado. Además, la red enfrenta problemas como el "front-running" y el reordenamiento de transacciones, lo que requiere una gestión vigilante.

La **descentralización** sigue siendo un principio fundamental para Ethereum, facilitada por la adopción de PoS para reducir barreras de entrada y fomentar una participación más amplia. Sin embargo, al igual que con cualquier sistema PoS, surgen riesgos de centralización debido al efecto de "los ricos se vuelven más ricos", donde los validadores más ricos tienen una mayor probabilidad de ser seleccionados, lo que podría concentrar el poder en unos pocos participantes. El modelo de gobernanza de Ethereum, si bien promueve la toma de decisiones descentralizada, también plantea desafíos para llegar a un consenso sobre las actualizaciones del protocolo.

La **velocidad** se aborda con el equilibrio entre la reducción de los tiempos de bloque y el mantenimiento de la descentralización. Si bien un mayor rendimiento de transacción es ventajoso, puede llevar a problemas de latencia de red y sincronización que necesitan una gestión cuidadosa para garantizar una participación equitativa y evitar la centralización debido a ventajas en la minería.

Solana

En términos de **seguridad**, Solana introduce soluciones innovadoras como la Prueba de Historia (PoH, por sus siglas en inglés) y tiempos de confirmación simplificados, mejorando la postura de seguridad de la plataforma. Esto permite una confirmación de transacciones mejorada y agrega una capa adicional de resistencia a manipulaciones.

La **descentralización** se beneficia del uso de PoS en Solana, lo que reduce las barreras de entrada y fomenta una participación más amplia. Sin embargo, al igual que en Ethereum, el desafío de la centralización persiste debido a la concentración del poder de apuesta entre unos pocos participantes.

La **velocidad** es una característica destacada de Solana, impulsada por su mecanismo de consenso combinando PoS con PoH. La capacidad de la red para procesar múltiples transacciones simultáneamente mejora el rendimiento de transacción y contribuye a su escalabilidad.

Avalanche

La **seguridad** en Avalanche se refuerza mediante la implementación de una estructura de Grafo Acíclico Dirigido, que mitiga los riesgos asociados con los ataques de doble gasto. El algoritmo de consenso único de la plataforma fortalece aún más su seguridad al dificultar los ataques a largo plazo.

La **descentralización** se aborda a través de subredes y la Red Principal. Si bien las subredes ofrecen personalización, también deben protegerse contra el riesgo de centralización no deseada en estos segmentos especializados.

La **velocidad** se revoluciona mediante la estructura de Grafo Acíclico Dirigido y el mecanismo de consenso de Avalanche. Esto permite la confirmación asíncrona y paralela

de transacciones, mejorando significativamente el potencial de escalabilidad de la plataforma.

3. Herramienta Práctica

Dentro del paisaje en constante evolución de la tecnología blockchain, el proyecto se convierte en un activo invaluable para GSR al ofrecer una herramienta práctica para monitorear métricas a lo largo del tiempo. Esta herramienta ayuda a rastrear el progreso de diversas blockchains con respecto al "Blockchain Trilemma". El proyecto involucra dos archivos Python, creando una solución de seguimiento de datos para GSR.

El primer archivo Python se enfoca en la extracción y compilación de datos. Emplea técnicas de web scraping, utilizando el paquete Selenium para extraer datos de diversas plataformas en línea. Estos datos, después de ser limpiados y refinados, se almacenan en un archivo de Excel. El libro de Excel contiene hojas separadas para Ethereum, Solana y Avalanche, que albergan datos de series temporales para variables como 'Número de Validadores' y 'TPS' (transacciones por segundo). Otra hoja llamada 'Aggregated' compila todas las métricas para comparaciones holísticas.

El segundo archivo Python crea un panel de control dinámico para la visualización interactiva de datos. Emplea la versatilidad y las bibliotecas de Python para generar cuatro gráficos principales: un Gráfico de Radar que muestra el rendimiento de las blockchains en diferentes métricas, una Tabla Agregada que proporciona valores métricos no normalizados, un gráfico de series temporales para 'Número de Validadores', y otro para 'TPS'. El panel de control interactivo permite a los usuarios activar casillas de verificación para ver tecnologías específicas, mejorando la experiencia analítica personalizada.

4. Conclusiones

En resumen, Solana está bien posicionada para resolver potencialmente el "Blockchain Trilemma" a través de sus notables atributos de Escalabilidad. Por otro lado, Avalanche ha evitado hábilmente las limitaciones del Trilema al garantizar la fusión fluida de Escalabilidad, Seguridad y Descentralización dentro de su Red Principal. Además, la ingeniosidad de las subredes posiciona a Avalanche como una solución optimizada y versátil de blockchain, aumentando su potencial de liderazgo en la industria. Así, considerando el panorama integral, Avalanche emerge como un competidor formidable, con mayores perspectivas de ascender como la preeminente Tecnología Blockchain.

A Comprehensive Evaluation of Ethereum, Solana, and Avalanche in Addressing the Blockchain Trilemma

Author: Bayona Bulto, Alvaro.
Supervisor: Matanza Domingo, Javier.
Collaborating Entity: ICAI – Universidad Pontificia Comillas

ABSTRACT

This study involves a comprehensive analysis of three leading blockchain technologies: Ethereum, Solana, and Avalanche. The focus is on how each addresses the "Blockchain Trilemma" by balancing decentralization, security, and speed. Additionally, the project includes creating a database and dashboard to monitor key metrics in real-time.

Keywords: Blockchain, Ethereum, Solana, Avalanche, Blockchain Trilemma

1. Introduction

In this project, we embark on a comprehensive exploration of GSR's strategic role as a prominent market maker and ecosystem collaborator within the dynamic realm of cryptocurrencies. With a substantial majority of their activities centered around market making, constituting an impressive 80% of their operational scope, we direct our focus towards unraveling the intricacies of this critical function. Market makers play a pivotal role by infusing liquidity into the market through a continuous process of quoting both buying and selling prices, thereby ensuring seamless transactions and averting liquidity bottlenecks.

The primary objective of our study is to furnish GSR with an impartial and well-informed recommendation for the selection of the most promising blockchain ecosystem. At the heart of this recommendation lies a comprehensive evaluation of the "Blockchain Trilemma," a fundamental challenge involving the interplay between three essential attributes: decentralization, security, and scalability. In particular, we investigate how three prominent blockchain platforms – Ethereum, Solana, and Avalanche – navigate this intricate trilemma. Our assessment encompasses both qualitative and quantitative analyses, the latter facilitated by the creation of a dynamic real-time dashboard.

2. Assessing the Trilemma Position of Each Technology

Ethereum

In terms of **security**, Ethereum grapples with potential vulnerabilities due to its reduced block time, which can increase the risk of long-range attacks and expose smart contract vulnerabilities. The shift to Proof of Stake (PoS) introduces the "nothing-at-stake" challenge, where validators might validate conflicting blocks to maximize rewards, potentially compromising the network's security. Ethereum's groundbreaking smart contract feature, while revolutionary, introduces unique security concerns as any flaws

in the code can be exploited. Additionally, the network faces issues like front-running and transaction reordering, necessitating vigilant management.

Decentralization remains a core principle for Ethereum, facilitated by the adoption of PoS to lower entry barriers and encourage wider participation. However, as with any PoS system, centralization risks emerge due to the "rich get richer" effect, where wealthier validators have a higher chance of being selected, potentially concentrating power within a few participants. Ethereum's governance model, while promoting decentralized decision-making, also poses challenges in reaching consensus on protocol upgrades.

Scalability is approached with the balance between reduced block times and maintaining decentralization. While faster transaction throughput is advantageous, it can lead to network latency and synchronization issues that need careful management to ensure equitable participation and prevent centralization due to mining advantages.

Solana

In terms of **security**, Solana introduces innovative solutions like Proof of History (PoH) and streamlined confirmation times, enhancing the platform's security posture. This enables improved transaction confirmation and adds an extra layer of tamper resistance.

Decentralization benefits from Solana's use of PoS, reducing barriers to entry and encouraging broader participation. However, as with Ethereum, the challenge of centralization remains due to the concentration of staking power among a few participants.

Scalability is a standout feature of Solana, driven by its PoH and PoS consensus mechanism. The network's ability to process multiple transactions simultaneously enhances transaction throughput and contributes to its scalability.

Avalanche

Security in Avalanche is fortified through the implementation of a Directed Acyclic Graph (DAG) structure, which mitigates risks associated with double-spending attacks. The platform's unique consensus algorithm further bolsters its security by rendering long-range attacks more difficult.

Decentralization is approached through subnets and the Primary Network. While subnets offer customization, they must also guard against the risk of unintended centralization within these specialized segments.

Scalability is revolutionized by Avalanche's DAG structure and consensus mechanism. This enables the asynchronous and parallel confirmation of transactions, significantly enhancing the platform's scalability potential.

3. Practical Takeaways

Within the ever-evolving landscape of blockchain, the project becomes an invaluable asset for GSR by offering a practical tool to monitor metrics over time. This tool aids in tracking the progress of various blockchains in addressing the Blockchain Trilemma's challenges. The project involves two Python files, creating a data-tracking solution for GSR.

The first Python file focuses on data extraction and compilation. It employs web scraping techniques, using the Selenium package to extract data from various online platforms. This data, after being cleaned and refined, is stored in an Excel file. The Excel workbook contains separate sheets for Ethereum, Solana, and Avalanche, housing time series data for variables like 'Number of Validators' and 'TPS'. Another sheet named 'Aggregated' compiles all metrics for holistic comparisons.

The second Python file creates a dynamic dashboard for interactive data visualization. It employs Python's versatility and libraries to generate four main graphs: a Radar Plot showcasing blockchain performance across metrics, an Aggregated Table providing unnormalized metric values, a Time Series graph for 'Number of Validators', and another for 'TPS'. The interactive dashboard empowers users to toggle checkboxes to view specific technologies, enhancing the tailored analytical experience.

4. Conclusions

In summation, the analysis yields intriguing insights: Solana is well poised to potentially resolve the Blockchain Trilemma through its notable Scalability attributes. On the other hand, Avalanche has deftly evaded the constraints of the Trilemma by ensuring the seamless fusion of Scalability, Security, and Decentralization within its Primary Network. Additionally, the ingenuity of subnets positions Avalanche as an optimized and versatile blockchain solution, augmenting its potential for industry leadership. Thus, considering the comprehensive panorama, Avalanche emerges as a formidable contender, boasting higher prospects to ascend as the preeminent Blockchain Technology.

Memory Index

Figures Index

Chapter 1: Introduction

1.1. GSR and motivation of the project

GSR is a crypto market maker and ecosystem partner founded in 2013. Although they have several other services such as an OTC trading business, an active venture investment arm, and several asset management funds, market making is responsible for 80% of their business and, therefore, will be the project's main focus.

A market maker provides liquidity [1] to a market by continuously quoting prices at which it will buy and sell (Bloomenthal, 2021). Without them, one owner of security may want to sell but find out there are no buyers, or one buyer may want to buy and find out there are no sellers. Therefore, they act as continuous buyers and sellers to always have transactions in the market.

Specifically, GSR provides market-making services to cryptocurrency projects and cryptocurrency exchanges and, in exchange for choosing GSR as the market-maker, GSR provides these clients with KPIs, performance measuring through its unique software, and daily market reports. Moreover, for unlisted projects, GSR can also provide tokenomics advice and introductions to funds and investors, among many other services.

However, being a market maker has the risk of buying a security and seeing its price decline, obtaining a loss for holding the security. This loss is not very drastic as they are continuously quoting prices, which means that they adapt quickly, reducing the loss. Market makers use spreads between their offers to compensate for this

[1] Liquidity is the degree to which an asset can be quickly bought or sold without notably affecting the stability of its Price.

possible loss. For instance, they may provide a bid price (buying offer) of 100$ and an asking price (selling offer) of 100.05$, creating a spread and a profit of 0.05$. Due to the high-volume trading, this can create huge benefits for the market maker.

All in all, market makers earn money through the spread and by the increase in the value of the securities. On the one hand, the spread can be increased if there is little competition among other market makers for that same token. That is, if GSR is the only market maker for a specific token, they control the offers and can charge a higher difference between them. On the other hand, the value of the securities depends on the success of the specific project or exchange in which GSR is the market maker.

Considering all this information, it is clear that for GSR it is essential to choose the most successful blockchain ecosystem in which to focus their sales and investment resources. By doing this, they will earn significant financial benefits, through stronger returns, minimum losses, and charging bigger spreads, as well as reputational benefits which will help position GSR as top of mind for clients and job seekers.

For all these reasons GSR has proposed a project to obtain an unbiased opinion through the assessment of the different blockchain ecosystems from a technological point of view.

1.2. The Blockchain Trilemma

The Blockchain Trilemma, termed by Vitalik Buterin (founder of Ethereum), states that there are three crucial properties that a blockchain must aim to achieve: decentralization, security, and scalability. However, one blockchain cannot maximize the three of them without having to do trade-offs in at least one of the properties. This leads to the creation of many different blockchains that address the trilemma in their unique way. An explanation of the properties will make this trilemma clearer.

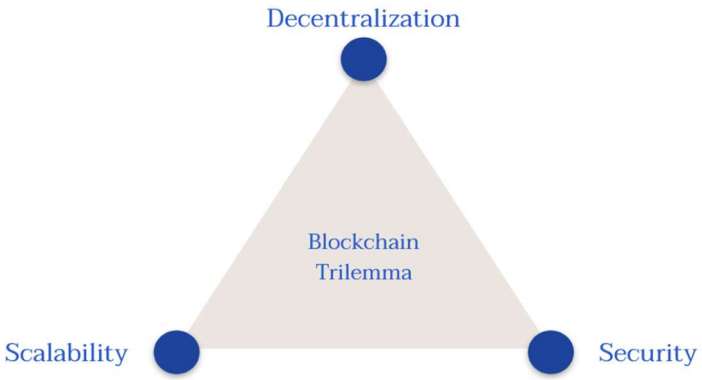

Figure 1. The Blockchain Trilemma

First, decentralization, states that one blockchain must not rely on a central point of control but, instead, distribute the control equally to all participants. This is the main reason why blockchains are so popular as it is the main difference from traditional systems such as banks. In today's world, people trust that banks are going to operate integrally but, if they decide not to, they have the power to, for instance, freeze all the accounts. Decentralization is primarily proportional to the number of participants in the network, but other characteristics, such as their geographical distribution, are also important.

Regarding the second property, security is the ability of a blockchain to maintain veridic and irrevocable transactions. The most common attack to corrupt the network is the 51% attack, where one entity can control 51% of the network and, therefore, when all the nodes vote, this entity has the majority and can introduce false transactions. The way to prevent this is by forcing nodes to spend resources to participate in the network and, therefore, if someone wants to achieve 51% of the network, it will have to spend a ridiculous number of resources. Although this is not the only attack a blockchain can suffer, the rest of the attacks are related to the fact that it is open-source, meaning that anyone can read the code and try to hack it. Therefore, these attacks are difficult to prevent and measure when assessing the security of a blockchain.

Last, scalability is the number of transactions that a blockchain can handle. This property is essential for the widespread adoption of the blockchain as, if it is low, users

will have to wait a considerable amount of time for their transaction to be accepted and, to incentivize miners to select their transaction before the others, the user will have to pay a high fee to the miner. An example of why this is a problem will be, for instance, if you want to run an application like Spotify in a decentralized manner. When you decide to listen to a song it will take some time before your petition is processed and you can listen to it. In addition, you will have to pay a high quantity for listening to it, discouraging users from using the application.

The trilemma appears when addressing how each property interplays with the others. The clearest example is between decentralization and scalability. Having a large number of participants in the network leads to higher decentralization, but it also leads to less scalability as every participant has to agree on the validity of a transaction, and, therefore, having more participants will take more time for the transaction to be final. Thus, the relationship between decentralization and security can be written as follows.

$$scalability \propto \frac{1}{decentralization}$$

Moreover, regarding the relationship between scalability and security, it can be demonstrated that it follows the same path as its predecessor. Improving scalability means reducing the block interval but, to do so, the number of resources that one participant must spend decreases and, therefore, security decreases.

$$scalability \propto \frac{1}{security}$$

Finally, the relationship between security and decentralization is directly proportionate as a higher number of nodes will increase decentralization and it will also mean that one entity has to control much more resources to obtain 51% of the network. Thus, the relationship can be written as follows.

$$security \propto decentralization$$

These relations lead to blockchain networks finding it easier to maximize decentralization and security at the expense of scalability, as is the case of Ethereum. However, as has already been discussed, the importance of scalability is notorious and several blockchains have been created to address this trilemma in different ways.

1.3. Objectives

GSR relies heavily on market making and the success of this business is determined by whether its resources are allocated in the correct blockchain ecosystem or not. Therefore, the main objective of this project is to provide GSR with an outside and unbiased recommendation on which blockchain should they focus their efforts on.

The blockchain trilemma, and how each blockchain addresses it, will serve as the method to estimate the future success of a blockchain and the basis of the final recommendation. The reason behind this is that the three properties of the trilemma are the key to the success and widespread adoption of a blockchain, meaning that whoever has the most favorable trade-offs will in turn be the most successful blockchain and where GSR should focus its efforts.

To achieve this objective, the following milestones are established.

- Understanding the general aspects of how every blockchain works from a theoretical point of view by analyzing Bitcoin, the first blockchain.
- Study in a theoretical way the blockchains that, as of today, are most likely to succeed based on GSR: Ethereum, Solana, and Avalanche. Moreover, analyze how they have addressed The Blockchain Trilemma in a qualitative way based on its technical characteristics.
- Analyze quantitatively The Blockchain Trilemma for the three blockchains that GSR wants to assess. To do so, the metrics that will

indicate where each blockchain stands in the trilemma must be determined.

- Development of a real-time dashboard with all the quantitative measures for the blockchains studied to provide GSR with a visual and updated snapshot of the blockchain ecosystem.

The reason behind giving so much emphasis to Ethereum, Solana, and Avalanche is that, after talking to GSR and based on their expertise in the field, they suggested that those were the blockchains that were gaining more traction among the community, and, therefore, want to have more insights about. Moreover, the three of them address the trilemma in very different ways, and is interesting to see how each technical decision affects their success in the trilemma, serving as the pinpoint for future blockchain developments.

All these objectives will serve the end purpose of providing a final recommendation to GSR with the addition of a tool to help them make their assessment at any future point in time in case the blockchain ecosystem changes. Moreover, this dashboard can help to identify trends that can add valuable information on where to allocate their resources as it can suggest where the interest of the market is going.

1.4. Work Methodology

The project will be divided into 4 main parts.

- Technical understanding of Bitcoin to get basic knowledge of how a blockchain works.
- Technical understanding of Ethereum, Solana, and Avalanche. This will provide the knowledge to understand why each blockchain stands where it stands in The Blockchain Trilemma, as well as what

are the constraints to maximize all the properties and if they are revocable in the future.

- Identify the metrics needed to describe each property in the trilemma and measure them in the case of Ethereum, Solana, and Avalanche.
- Creation of a dashboard with all the information of the metrics in real-time restricted to Ethereum, Solana, and Avalanche.

The work methodology regarding the theoretical study in the first and second parts of the project, as well as the identification of the metrics in the third part, will be based on research papers, especially the white papers of each blockchain, as well as information that can be found on the official websites.

Moreover, the methodology employed in the fourth phase of the project revolves around the utilization of web scraping techniques to acquire real-time information for each metric of interest, subsequently integrating this data into a dynamic dashboard. This process is meticulously crafted in Python, leveraging two pivotal libraries that cater to the specific requirements of this task: Streamlit and Selenium. Streamlit, an open-source Python library, serves as the backbone for crafting interactive and visually appealing dashboards with minimal effort. On the other hand, Selenium, a powerful web scraping and automation framework, empowers the extraction of real-time data by simulating user interactions with web pages. By harmoniously merging the capabilities of Streamlit and Selenium, the project ensures a seamless pipeline for extracting, processing, and displaying real-time metrics within an engaging and user-friendly dashboard interface.

Chapter 2: Bitcoin

As previously mentioned, Bitcoin serves as the foundation for understanding the functioning of a blockchain. By comprehending its process, we can gain insight into how the alterations introduced by other blockchain technologies affect the trilemma's key components: decentralization, security, and scalability. Consequently, this chapter aims to elucidate the entire process, starting from a user's creation of a transaction to the ultimate confirmation of a transaction's finality.

Before delving into the intricacies of the Bitcoin process, it is essential to shed light on the concept of hashes, which are mathematical functions widely employed in blockchains for various purposes. In simple terms, a hash is a cryptographic algorithm that converts any given input into a fixed-length string of characters, accomplished through encryption and within a remarkably short span of time (Frankenfield, What Is a hash? Hash Functions and Cryptocurrency Mining, 2023). This encryption process ensures that even with knowledge of the output hash, determining the original input is practically impossible, as it would require an immense amount of time and an arduous trial-and-error approach. Consequently, hashes function as one-way functions, possessing the characteristic of being relatively easy to calculate but extremely challenging to reverse-engineer.

The remarkable property of hashes lies in their ability to consistently produce the same output hash for a given input. However, even the slightest alteration in the input results in an entirely different hash being generated. This property of hashes becomes immensely valuable in two fundamental ways. Firstly, hashes serve as a means to verify the integrity and unchanged nature of the input data if the output hash is known. By comparing the output hash of a given input with the previously obtained hash, one can ensure that the data has remained unaltered.

Secondly, hashes play a pivotal role in reducing the size of data within blockchains. As hashes are of fixed length, they enable the representation of large amounts of data in a concise and manageable manner. This characteristic proves highly advantageous, particularly in distributed systems like blockchains, where the efficient storage and transmission of data are crucial. By employing hashes, the blockchain can validate the integrity of information while significantly minimizing the amount of data that needs to be stored or transmitted.

2.1. **Bitcoin Process** (Antonopoulos, 2014)

2.1.1. Generate an address

To actively engage in a Bitcoin process, the client must possess a valid pair of public and private keys, serving as the cryptographic means for secure access to their bitcoins. The private key is essentially a random number ranging from 1 to 2^{256}, and various methods exist for generating this randomness. Conversely, the public key is derived from the private key using elliptic curve cryptography (which is not pertinent to the thesis objective) (Tech Target, 2021). Presented below is an illustration of these keys.

Private key
1E99423A4ED27608A15A2616A2B0E9E52CED330AC530EDCC32C8FFC6A526AEDD

Public key
F028892BAD7ED57D2FB57BF33081D5CFCF6F9ED3D3D7F159C2E2FFF579DC341A

Figure 2. Private and Public Key Example

The utilization of keys revolves around a fundamental concept: while anyone can access the client's public key, deciphering the client's private key remains an

insurmountable task. This principle facilitates two primary applications: digital signatures and encryption.

- **Digital signatures** leverage the fact that the client's public key is widely known, while their private key remains undisclosed. By encrypting data with the client's private key, anyone possessing the client's public key can decrypt the message. If the message is decrypted successfully using the client's public key, it confirms that the client's private key was used for encryption. Given that only the client possesses knowledge of their private key, this cryptographic technique verifies that the client transmitted the information, thereby ensuring its authenticity.

- **Encryption**, built upon the same underlying principles, applies when another individual desires to transmit a message to the client. In this scenario, the sender employs the client's public key to encrypt the message, ensuring that only the client's private key can decrypt it. Consequently, the confidentiality of the message content is guaranteed, as only the client can access its decrypted form.

In a decentralized structure where every participant receives all transactions, the importance of utilizing keys for system security becomes evident. However, Bitcoin's core principle revolves around transparency, wherein everyone can view the content of all transactions. As a result, the emphasis lies primarily on the utilization of digital signatures rather than encryption.

Consequently, the pair of keys serve distinct purposes as follows:

- **Public key:** The public key, accessible to everyone without revealing the private key, functions as the designated address to which individuals can send BTC. It functions akin to an email address, providing a means for others to send BTC securely to the intended recipient. Although the address is not precisely the public key, it is derived directly from it.

- **Private key:** The private key is utilized to sign transactions, ensuring that only the legitimate owner of the BTC can access and manage their digital assets. By employing the private key for transaction signing, it guarantees the authenticity and integrity of the sender, preventing unauthorized access to the BTC holdings.

2.1.2. Create a transaction

Upon acquiring the pair of keys, the client possesses the essential information required for sending and receiving transactions. However, before engaging in such transactions, the client must initiate the creation process. To illustrate the necessary details involved in a transaction, let us consider an example.

Alice wants to send 0.6 BTC to Bob. To proceed with the transfer, Alice must first ensure that she possesses the required amount of BTC. To confirm her available balance, Alice needs to examine all the transactions recorded in the blockchain where she is listed as the recipient of BTC.

This particular step introduces a challenge in the process. Since the system operates in a decentralized manner, numerous nodes, if not all of them, must validate the accuracy of a transaction. One of the criteria for validation involves ensuring that the sender possesses sufficient funds. Interestingly, each validating node needs to perform the same balance-checking process that Alice conducted earlier. This verification task can be exceptionally demanding and time-consuming.

To alleviate this workload, Alice employs a selective approach. Instead of presenting the entire transaction history where she received BTC, she specifically chooses the relevant transactions that collectively add up to 0.6 BTC, demonstrating her possession of the necessary funds. This strategy proves beneficial because the validating node only concerns itself with verifying if Alice has precisely 0.6 BTC available for the specific transaction she wishes to execute. The total amount of BTC in her possession, beyond the required funds, becomes irrelevant in this context.

Returning to the example, Alice proceeds to examine the complete list of transactions where she is listed as the recipient of BTC. Her objective is to select the necessary transactions that, when combined, add up to 0.6 BTC—the amount she intends to transfer. After careful examination, Alice discovers two relevant transactions: one where she received 0.2 BTC and another where she received 0.5 BTC, totaling 0.7 BTC. However, there are two considerations to take into account.

Firstly, Alice recognizes that 0.7 BTC is 0.1 BTC more than what she actually intends to send. Secondly, she is aware that a transaction fee must be paid to process the transaction. While a detailed explanation of transaction fees will be provided in subsequent steps, it is sufficient for now to understand that a fee is necessary.

For the sake of illustration, let's assume Alice wants to pay a fee of 0.01 BTC. To address these two issues, the transaction will be structured as follows:

- Alice will utilize the 0.7 BTC from the two aforementioned transactions.
- Bob will receive 0.6 BTC from Alice as the intended transfer.
- The miner facilitating the transaction will receive the fee of 0.01 BTC from Alice.
- Alice will receive back the remaining 0.09 BTC, as change, completing the transaction.

By structuring the transaction in this manner, Alice ensures that the correct amount is transferred to Bob, the transaction fee is paid, and any excess amount is returned to her as change.

Therefore, to take all that has been mentioned into account, the structure of a transaction is as follows.

- **Inputs:** A collection of pointers representing the transactions selected by Alice from which she intends to source the funds. Each pointer comprises the transaction hash to which it is referring and an index indicating the specific

output within that transaction. It's important to note that a single transaction can have multiple outputs, as previously demonstrated.

Inputs
51285729d21f871d7f91f7432cd088a819c8d049d7a3985e002a6f458216ac8c : 0
Fc1acd96728ca18828f31b6f814854cf7d7191fe51e8798b24875ce08eb25e3f : 0

Figure 3. Input Structure

- **Outputs or UTXO (Unspent Transaction Outputs):** A list that specifies all the addresses to which the BTC should be sent and the corresponding amounts to be transferred. Each output represents an unspent transaction output, meaning that the specified amount is available to be used as input in future transactions.

Outputs
Bob's Address : 0.6 BTC
Alice's Address : 0.09 BTC

Figure 4. Output Structure

As shown in Figure 4, the fees do not have a specific field in the outputs, but rather are calculated by the miner through the following equation:

$$Transaction\ Fee = \sum inputs - \sum outputs$$

However, there is one more important element that needs to be included in the transactions to ensure their integrity and security. As mentioned earlier, the inclusion of keys in transactions serves the purpose of validating the identity of the sender and ensuring that the funds are being withdrawn from the correct address. To achieve this, two additional components are incorporated into the transaction structure:

- **Locking Script:** This script is added to the outputs and specifies the specific conditions that must be fulfilled to spend the output at a later time. The locking script essentially acts as a set of requirements that need to be met for the funds to be accessed.

- **Unlocking Script:** On the other hand, the unlocking script is included in the inputs and serves to fulfill the conditions established by the corresponding locking script of the output being utilized. The unlocking script is designed to satisfy the predetermined requirements specified in the locking script.

To illustrate this process, let's consider the example we mentioned earlier involving Alice and Bob. Suppose Alice wants to send BTC to Bob. In this case, Alice would need to include a locking script in the transaction output, which defines the conditions that Bob must meet to spend the received BTC in the future.

Consequently, when Bob wishes to utilize the BTC he received from Alice, he must provide an unlocking script that fulfills the conditions outlined in the locking script associated with the output he intends to use. The unlocking script typically consists of Bob's signature and his public key.

For successful transaction validation, the locking script is responsible for verifying the correctness of Bob's signature and ensuring that his address matches the output he is referring to. If the locking script yields a satisfactory result, it signifies that Bob is indeed the rightful owner of the BTC and grants him the ability to spend those funds securely.

2.1.3. Transmit and validate a transaction

Once all the necessary information regarding a transaction has been finalized, it is ready to be sent and validated by each node in the network. The method of sending transactions is known as flooding, where each node transmits the transaction to its immediate neighbors, who then relay it to their respective neighbors. Before sending the transaction to its neighbors, each node performs a series of validations to ensure its integrity. This flooding approach offers security benefits when compared to a broadcast method, as it mitigates the risk of spamming and denial-of-service attacks. In this method, a malicious transaction would only propagate to a limited number of nodes, preventing its widespread dissemination.

When a node receives a transaction, it undergoes a thorough validation process, checking for the following criteria:

- **Syntax and Data Structure:** The transaction's syntax and data structure, including values and conditions such as size, input formats, and limit values, must adhere to the correct specifications.

- **Script Validation:** The unlocking script is examined to ensure that it only pushes numbers onto the stack, while the locking script must conform to isStandard forms. Additionally, the unlocking scripts for each input must be validated against the corresponding output locking scripts.

- **Output Existence:** For each input, the transaction is rejected if the referenced output does exist in any other transaction within the transaction pool.

- **Output Availability:** The node searches both the main branch and the transaction pool to locate the referenced output transaction for each input. If the output transaction is missing, the transaction is considered an orphan. Furthermore, if the referenced output has already been referenced, the transaction is rejected.

- **Input and Output Balance:** The transaction is rejected if the sum of input values is less than the sum of output values, as this would indicate an imbalance or insufficient funds.

- **Transaction Fee:** If the transaction fee is deemed too low to be included in an empty block, the transaction is rejected.

After performing these validation checks, the node categorizes the transaction into one of three possible states:

- **Acceptance:** If the transaction successfully passes all the validation steps, the node adds it to the transaction pool and transmits it to its neighboring nodes.

- **Rejection:** If the transaction fails to meet any of the validation criteria, the node rejects it and refrains from transmitting it to further nodes.

- **Orphan Transaction:** Transactions in which a referenced output (from a previous transaction) has not yet arrived at the node are considered orphan transactions. These transactions cannot be fully validated and added to the pool until their referenced parent transaction is received. In such cases, the orphan transaction is added to the orphan transaction pool. Once the referenced parent transaction arrives and successfully validates, the orphan transaction can proceed with its validation. If the number of transactions in the orphan transaction pool exceeds a predefined limit, the node removes some transactions from the pool to manage its size.

2.1.4. Mine a block

Simple Case: One node in charge of mining

If we were to envision a scenario where the responsibility of mining blocks falls upon a single node, the process would become significantly simplified and straightforward. Let's delve into the details of this streamlined procedure:

- To commence, the node would undertake the critical task of **selecting which transactions** from the transaction pool it wishes to include in the block. A sophisticated algorithm governs this selection process, weighing various factors such as the duration spent by transactions in the pool and the fees they offer. This algorithmic approach favors transactions that yield higher rewards for the miner while ensuring that no transaction languishes indefinitely in the transaction pool.

- Once the node has finalized its selection of transactions, it proceeds to add an essential transaction known as the **Generation Transaction**. This particular transaction plays a pivotal role in transferring the fees and rewards associated with mining a block (as with each newly mined block, fresh BTC is generated) directly to the miner.

- Lastly, the miner **appends the block header** to the completed block and disseminates it to the remaining nodes within the network. The block header in this case would be relatively succinct, necessitating the inclusion of only two fundamental components, alongside relevant metadata: the hash of the previous block header, which facilitates the creation of the blockchain's chain of blocks, and the Merkle Root.

The addition of the Merkle Root to the block header holds significant importance due to Bitcoin's utilization of Merkle Trees. This innovative methodology empowers nodes to search through the blockchain rapidly, considerably expediting the process of validating referenced transactions (Frankenfield, Merkle Tree in Blockchain: What it is and How it Works, 2021). To illustrate this concept further, let us consider Figure 5 as an exemplary depiction of a Merkle Tree. The tree's construction commences by independently applying the hash function to each individual transaction. Subsequently, pairs of transaction hashes are amalgamated, and the resulting string is then hashed. This process continues iteratively until the Merkle Root finally emerges.

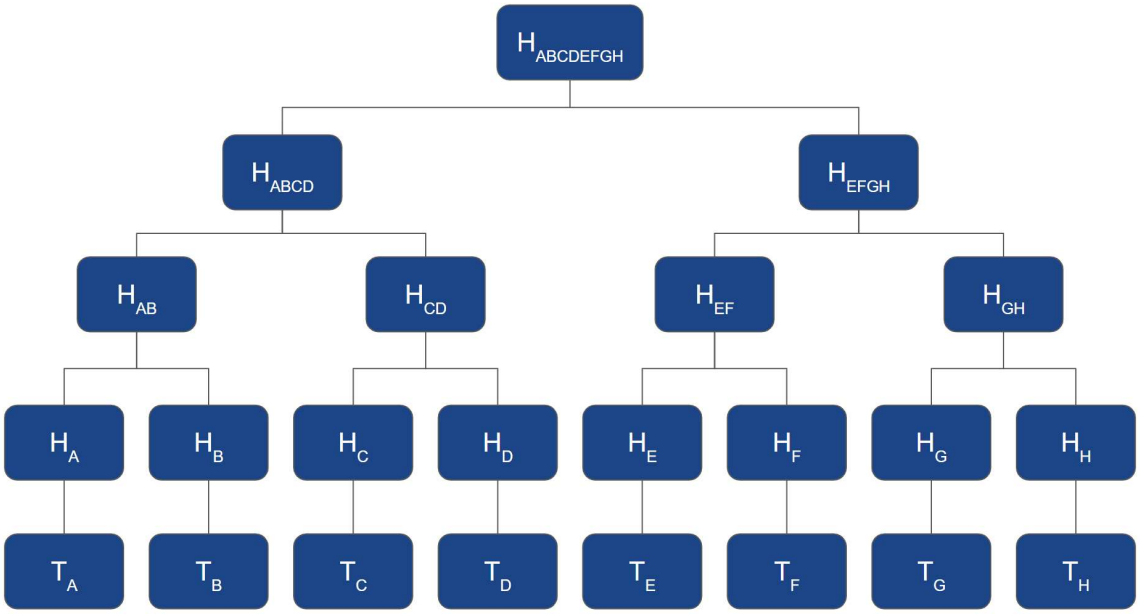

Figure 5. Merkle Tree

The principal reason behind the value of Merkle Trees in searching transactions within the blockchain stems from their ability to ascertain whether a particular transaction (e.g., TA) exists within a block. By solely computing the hash of the transaction (H_A) and knowing H_B, H_{CD}, H_{EFGH}, and $H_{ABCDEFGH}$, one can determine the Merkle Root and compare it with the block's Merkle Root found in the header. If both Merkle Roots align, it confirms the presence of the transaction within that specific block, obviating the need to compute all the individual hashes and requiring only the computation of four hashes instead.

However, it is essential to acknowledge that Bitcoin's fundamental concept revolves around complete decentralization, wherein no central node exclusively mines blocks. Instead, the process is open to anyone within the network. This decentralized nature poses challenges concerning the selection of the miner for each turn (explained in the proof of work section) and the establishment of a consensus regarding the uniformity of the blockchain across all participants (explained in Section 2.1.5).

Proof of Work

The method known as Proof of Work emerges as the solution to the quandary surrounding the selection of a mining node. To determine the chosen node, a competitive environment is established wherein each participating node engages in a puzzle-solving endeavor. The node that successfully solves the puzzle first becomes the designated miner responsible for mining the subsequent block.

This intriguing puzzle entails the constant change of a random number referred to as the nonce. The resulting block header is then subjected to a hash function. To emerge victorious, the resulting hash must possess, at the beginning, a certain number of zeros equal to or greater than the difficulty level stipulated by the network. For instance, Figure 6 serves as an illustrative representation of this process, assuming a difficulty level of 3.

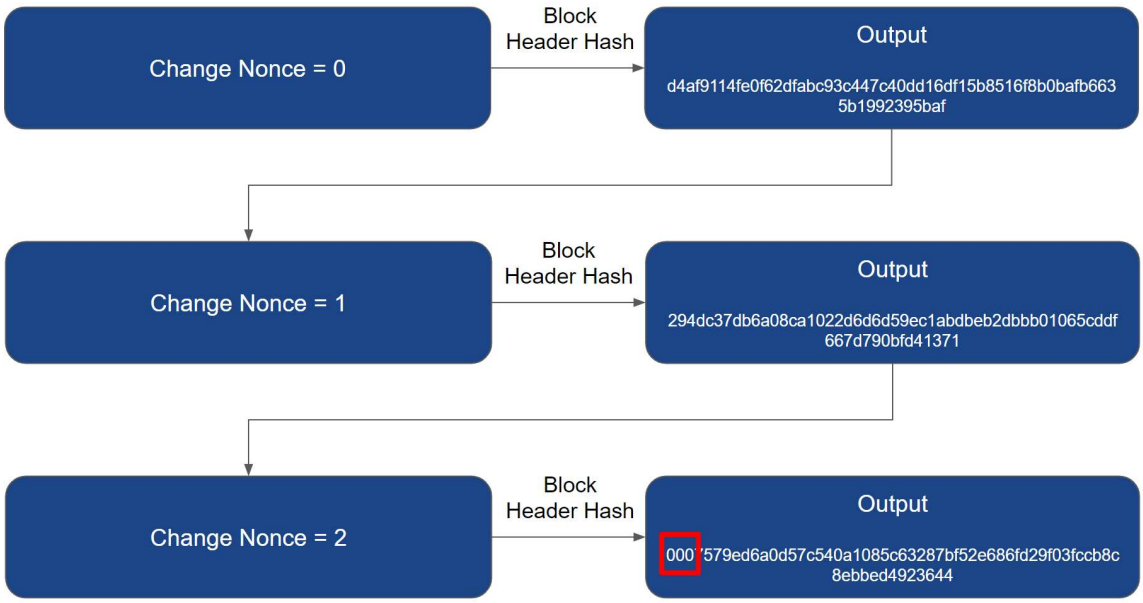

Figure 6. Proof of Work

Let us closely examine the example provided. The node in question initiates its puzzle-solving journey with a nonce value of 0. However, as the resulting block header hash fails to commence with three zeros, the node proceeds to attempt the puzzle with a nonce value of 1. Unfortunately, the outcome remains the same. Undeterred, the

node perseveres and tries once more, this time employing a nonce value of 2. Remarkably, this iteration yields a hash that commences with the desired three zeros, thus achieving the desired outcome and claiming victory.

It is crucial to acknowledge the immense computational demand imposed by this method, as the nature of hash functions leaves no alternative but to resort to a trial-and-error approach. The absence of a predetermined formula or shortcut necessitates arduous computational efforts to find the appropriate solution. The utilization of this method becomes imperative when considering alternatives like complete random selection, which would enable individuals to create an unlimited number of nodes at minimal cost, consequently elevating their chances of being chosen as the miner. Such a scenario would render the network vulnerable to malicious actors who could exploit the system by amassing an excessive number of nodes, enabling them to manipulate the blockchain in their favor. Therefore, the adoption of Proof of Work not only introduces an element of randomness through the reliance on random nonce selection but also serves as a safeguard against such detrimental attacks by imposing a cost to being selected as a miner.

Implementing this method changes the simple case scenario shown above by adding two more fields to the block header: the nonce and the difficulty target.

2.1.5. Transmit and validate a block – Consensus Algorithm

Once a miner successfully solves the puzzle, achieving the proof of work, a moment of triumph ensues, as the miner proceeds to broadcast the newly minted block to the other nodes within the network. This broadcast serves as a declaration of victory, indicating that the miner has emerged as the chosen one. Upon receiving a new block, each node undertakes the crucial task of validating its contents. If the validation process proves successful, the node ceases its efforts to solve the block it was previously working on. It promptly removes the transactions included in the received block from its transaction pool and proceeds to share the block with its neighboring

nodes, employing a mechanism akin to the transmission of regular transactions. Once this process is complete, the node promptly redirects its attention toward the pursuit of solving the next block, continuing its mining endeavors.

The validation process comprises several vital aspects that warrant careful consideration. Let us explore these facets in detail:

- **Syntax and Data Structure:** verifies the block's adherence to the correct specifications regarding its data structure, size, and time limitations between blocks.

- **Proof of Work:** involves verifying that the block header hash falls below the target difficulty level. This requirement ensures that the miner has invested substantial computational effort in solving the puzzle, fortifying the security of the blockchain.

- **Transactions:** Each transaction included within the block must undergo thorough verification, following the principles elucidated in Section 2.1.3. This verification process serves as a safeguard against fraudulent or invalid transactions, further bolstering the integrity of the blockchain.

However, it is important to acknowledge the possibility of encountering a scenario known as a fork, which arises when two miners successfully solve the puzzle nearly simultaneously. In such instances, a fork occurs when a node has already received and validated a block, subsequently adding it to the main chain, only to receive another correctly solved block with the same parent as the one just received. The parent-child relationship between blocks is indicated by the previous block header field within the block's header. This situation is visually depicted in Figure 7, providing a clear illustration of the concept.

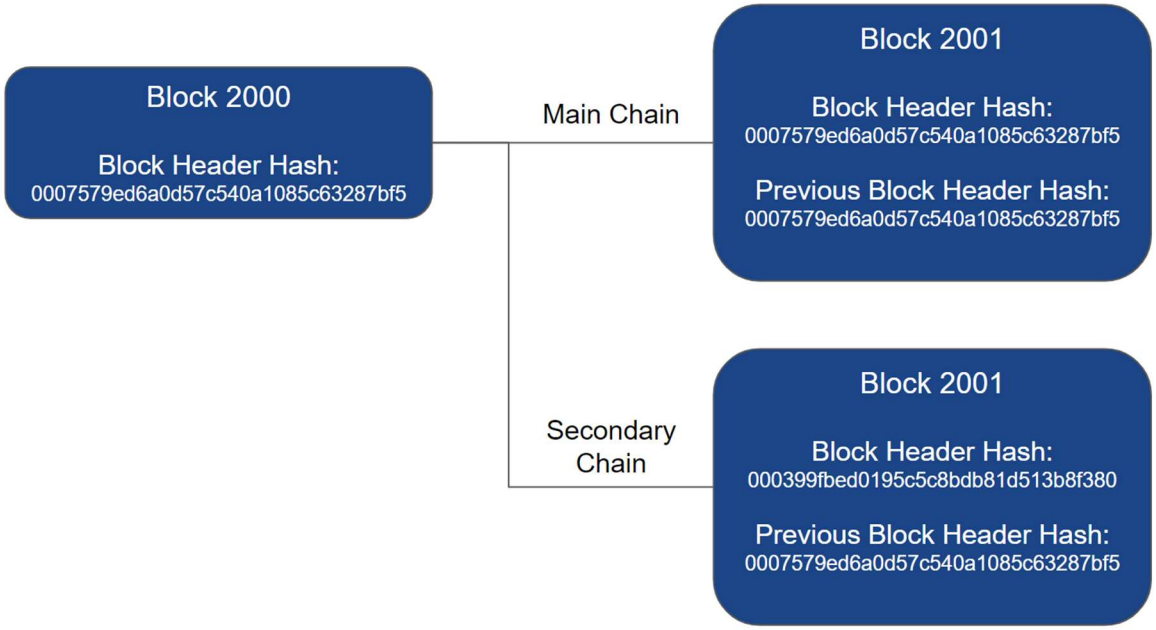

Figure 7. Fork

Once confronted with a fork, the node in question initiates the creation of a secondary chain by appending the newly received block to it. At the same time, the node continues mining the next block with the previous block (from the main chain) as its parent. From this juncture, two potential outcomes may arise:

- The node receives yet another block, with its parent being the one from the **main chain:** In this case, the main chain becomes longer than the secondary chain. Consequently, the secondary chain is discarded, and the transactions contained within it are reintroduced to the transaction pool, ready to be included in future blocks.

- The node receives another block, with its parent being the one from the **secondary chain:** This scenario leads to the secondary chain becoming longer than the main chain. As a result, the secondary chain supersedes the main chain, assuming the mantle of the new main chain. Consequently, the previous main chain is discarded, and its associated transactions are added back to the transaction pool.

To maintain the stability and predictability of the network, Bitcoin has established a target time of approximately 10 minutes for solving each block. This duration is achieved by periodically adjusting the difficulty target (i.e., the number of leading zeros the block header hash must possess) every 2016 blocks. This adjustment mechanism serves to regulate the overall mining speed, ensuring that the probability of encountering a fork remains exceedingly low, practically diminishing to zero when considering the occurrence of two simultaneous forks.

Chapter 3: Ethereum

Upon grasping the fundamentals of Bitcoin's functioning and gaining insights into the basics of blockchains, we can delve into elucidating the distinctive technical features of Ethereum, Solana, and Avalanche in comparison to Bitcoin. This exploration will enable us to analyze how each divergence influences the Blockchain Trilemma.

Consequently, in this chapter, our primary focus will be directed toward the exploration of the first technology, Ethereum, which will be presented in two comprehensive sections. The initial segment will delve into an in-depth explanation of the various technical differences that set Ethereum apart from Bitcoin. This will include a thorough examination of their contrasting architectures, consensus mechanisms, and smart contract capabilities (Kasireddy, 2017).

Moving on to the second section, we will conduct a meticulous analysis of the far-reaching implications that each of these disparities has on the delicate balance of the Blockchain Trilemma. As we evaluate the trade-offs between decentralization, scalability, and security, we will gain valuable insights into how Ethereum's unique technological design shapes its position in the ever-evolving landscape of blockchain solutions.

3.1. Technical Differences With Bitcoin

3.1.1. **Smart Contracts** (Frankenfield, What Are Smart Contracts on the Blockchain and How They Work, 2023)

In the early days of Bitcoin, its primary and somewhat restrictive application was limited to the straightforward transaction of coins between accounts. Such a limitation posed challenges to the widespread adoption of blockchain technology. To overcome this constraint and unlock the true potential of decentralized systems, innovative minds sought to create more versatile blockchains capable of accommodating a variety of applications demanded by the users.

To achieve this, these alternative blockchains adopted an approach that involved hardcoding each desired application into their systems. Essentially, every time a specific application was needed, the developers of that blockchain would have to manually craft the necessary code modifications to accommodate the new functionality. For example, if a voting mechanism was demanded, the blockchain's underlying code would be tailored to accommodate the voting application. However, this approach proved to be quite limited and inefficient in the long run.

In contrast, Ethereum emerged as a groundbreaking solution that revolutionized the landscape of blockchain technology. It introduced a paradigm shift by treating the blockchain as an operating system where anyone could create and deploy applications without the need to modify the blockchain's core code or its inherent characteristics.

This groundbreaking capability was made possible through the invention of smart contracts. Smart contracts are self-executing pieces of code that operate on the Ethereum blockchain. They function as automated agreements that execute predetermined actions when certain conditions are met. Figure 8 showcases a smart contract intended to serve the same function as Bitcoin - exchanging tokens. As depicted, the smart contract is essentially a piece of code containing functions that can

be called and, consequently, executed autonomously on the Ethereum network. This ability to execute predefined functions upon certain conditions is what sets smart contracts apart from conventional contracts and marks a major breakthrough in blockchain technology.

```
contract TokenExchange
{
  mapping (address => uint) balances;

  function BuyTokens() payable
  {
    balances[msg.sender] += msg.value;
  }

  function SellTokens(uint amount)
  {
    if (balances[msg.sender] >= amount)
    {
      if (msg.sender.call.value(amount)() == false) // send money to caller
        throw;
      balances[msg.sender] -= amount;
    }
  }
}
```

Figure 8. Smart Contracts

However, the true revolution lies beyond this single example. Ethereum's groundbreaking aspect is that it goes beyond mere token exchange and empowers developers to unleash their creativity and build custom programs tailored to their unique needs. Unlike traditional blockchains, Ethereum's openness allows developers to conceive and implement a diverse array of applications and decentralized services without necessitating any fundamental alterations to the underlying blockchain protocol.

The implications of this paradigm shift are far-reaching. From decentralized finance (DeFi) platforms enabling borderless lending and borrowing, to non-fungible token (NFT) marketplaces revolutionizing digital ownership and art, the possibilities are virtually limitless. Ethereum's flexible and programmable smart contracts have fueled an explosion of innovation, giving rise to an ever-expanding ecosystem of decentralized applications that shape industries across the globe.

Implementation of Smart Contracts

The implementation of smart contracts brought about significant changes to the traditional process of creating and sending transactions within the Ethereum blockchain.

The first modification was the introduction of two types of accounts:

- **Externally Owned Account (EOA):** Much like Bitcoin accounts, an Externally Owned Account is defined by a pair of cryptographic keys - a public key for receiving funds and a private key for controlling the account. The owner of the private key holds full authority over this account, enabling secure control over their assets.

- **Contract Account:** Contract Accounts, on the other hand, act as repositories for the smart contract code. However, it is important to note that creating and maintaining Contract Accounts comes at a cost, as they consume valuable network storage resources.

The distinction between these two types of accounts is pivotal in understanding the mechanics of transactions in the Ethereum ecosystem. When an Externally Owned Account initiates a transaction, it can send two types of messages (or transactions), contingent upon the type of account receiving the message:

- **Externally Owned Account to another Externally Owned Account:** In this scenario, the transaction entails a simple transfer of Ether (ETH) from one Externally Owned Account to another. This type of transaction is analogous to conventional cryptocurrency transfers.

- **Externally Owned Account to a Contract Account:** In this case, the message sent from the Externally Owned Account to the Contract Account serves to activate the smart contract. Furthermore, this message can contain the necessary

inputs and parameters, enabling the smart contract to perform its intended function as predetermined by its code.

In contrast, the messages that Contract Accounts generate are a result of the execution of its code and, therefore, they can't initiate any transaction unless it has received one from either an Externally Owned Account or another Contract Account. The transactions they generate are referred to as "Internal Transactions".

The second modification is that, to fully harness the potential of smart contracts, developers require the necessary tools to craft and deploy their custom code. For this purpose, Ethereum provides the powerful programming language Solidity. With Solidity, developers can create sophisticated smart contract code, enabling them to implement a diverse array of applications and services on the Ethereum blockchain.

3.1.2. **Proof of Stake** (Ethereum, 2023)

As of 2022, Ethereum underwent a significant transformation by transitioning from the energy-intensive Proof of Work (PoW) consensus mechanism to the more eco-friendly Proof of Stake (PoS) protocol. This shift marked a major milestone in Ethereum's evolution, addressing the sustainability concerns associated with PoW.

To better comprehend the transition, let's first revisit the mechanics of Proof of Work. In the PoW system, miners engage in a competitive race to solve complex cryptographic puzzles by continuously hashing the block header. This process demands an enormous amount of computational power and, consequently, results in significant energy consumption. While PoW has proven to be highly secure, the environmental impact of its energy-intensive mining activities has been a growing concern for the blockchain community.

To tackle these sustainability challenges, Ethereum embraced Proof of Stake as an alternative consensus mechanism. Unlike PoW, where miners invest considerable computational resources to solve puzzles, PoS introduces a novel approach by

requiring participants, known as validators, to stake a certain amount of Ethereum (ETH) to become eligible for block creation.

Here's how Proof of Stake operates:

- **Staking ETH:** In the PoS system, each validator is required to stake a desired amount of ETH into the network. This amount represents their "stake" and serves as a measure of their commitment to the network's security and integrity.

- **Selection Process:** The probability of a validator being chosen to propose and validate a new block is directly influenced by the amount of ETH they have staked relative to the total amount staked by all participants in the network. In other words, the more ETH a validator stakes, the higher their chances of being selected as the block proposer.

- **Validation:** Validation in the Proof of Stake (PoS) consensus mechanism follows a similar process to that of Proof of Work (PoW). Each node in the network verifies the correctness of a proposed block. However, in the event that a Validator submits a block that is incorrect or invalid, their staked ETH is put at risk. In other words, if the proposed block is deemed invalid, the Validator may lose a portion or all of their staked ETH as a penalty for their erroneous action. This shift in the validation process introduces a powerful incentive for Validators to act honestly and validate transactions correctly, as they have a financial stake at risk.

To illustrate this process, let's consider the scenario where Alice stakes 2 ETH, and the total amount of ETH staked by all validators in the network amounts to 10 ETH. In this case, Alice's probability of being selected as the block proposer is calculated as 2 ETH / 10 ETH = 0.2 or 20%.

3.1.3. Importance of Fees (Gas)

In the world of Bitcoin, transaction fees were implemented by sending a commission to the miner with every transaction. However, the diverse and expansive capabilities of Ethereum, which extend far beyond simple money transfers, rendered this commission-based system inadequate. To address this challenge, Ethereum introduced a more sophisticated fee system, where fees are incurred each time a code is executed as a result of an incoming transaction.

The Ethereum fee system revolves around two critical variables: gas price and gas limit. The gas price indicates how much ETH a user is willing to pay for executing a certain amount of code (e.g., 1 line of code equals 1 ETH). On the other hand, the gas limit represents the maximum amount of ETH the user is willing to expend in the process.

Here's how the fee process unfolds:

- **Determining the Maximum Fee:** To initiate a transaction, the sender establishes the gas price and gas limit, which, when multiplied together, calculates the total maximum amount of ETH they are willing to pay for the transaction's execution.

- **Transaction Execution and Gas Consumption:** The transaction is then sent to a Contract Account, and its code is executed. As the code executes, it consumes a specific amount of ETH, which is sent to the miner executing the code. In some cases, the Contract Account may send an Internal Transaction to another Contract Account, triggering the need for additional code execution and subsequent ETH consumption by the miner.

- **Transaction Finalization:** Once the code execution is complete, one of two scenarios may unfold: either the sender runs out of gas, rendering the transaction invalid, and all the staked ETH is lost, or the transaction is successfully finalized, and any excess ETH staked by the sender is returned.

The gas price plays a vital role in incentivizing miners to prioritize certain transactions. Higher gas prices mean greater rewards for miners, making those transactions more appealing to them. As a result, miners have the freedom to select which transactions they want to validate or ignore. To assist senders in determining an appropriate gas price, miners can communicate the minimum gas price they are willing to accept for transaction execution.

This fee system in Ethereum holds numerous advantages:

- **Network Sustainability:** Ethereum's design ensures that every operation executed on the network is simultaneously processed by every full node. To maintain network efficiency and prevent overtaxing, imposing fees encourages users to utilize Ethereum smart contracts for simpler tasks, such as running basic business logic or verifying cryptographic objects, rather than resource-intensive processes like file storage, email, or machine learning.

- **Halting Problem Mitigation:** Ethereum's Turing-complete language allows for loops, making it susceptible to the halting problem. Without fees, a malicious actor could exploit this vulnerability by executing an infinite loop within a transaction, disrupting the network without any consequences. Fees, through the concept of Gas Limit, protect the network from such deliberate attacks.

- **Encouraging Efficiency:** Ethereum's block size becomes variable, as there is no inherent limit to the code that a Contract Account can hold. However, as code size increases, so do the fees required for its execution. Consequently, fees serve as a motivating factor for developers to create efficient and optimized smart contracts, promoting better resource management and network health.

3.1.4. Block Time

One of the most significant concerns that plagued Bitcoin was its slow scalability due to the 10-minute block time, resulting in a limited number of transactions that could be processed. Nevertheless, this characteristic also had its advantages, as it greatly reduced the possibility of forks and consecutive forks, contributing to the network's stability.

Forks occur when multiple miners successfully validate different blocks at the same height and compete to become the valid chain. This can cause temporary divergence in the blockchain, leading to uncertainties about the canonical chain. The risk of forks not only raises security concerns but also has implications for centralization.

Ethereum decided to reduce its block time to approximately 15 seconds, which significantly impacted the potential for forks to occur. This shift, while providing faster transaction finalization, introduced new challenges in terms of security and centralization risks, as discussed in the following section.

Finally, with an increased likelihood of forks in Ethereum, the frequency of orphan blocks (blocks that are not part of the main chain) also rises. These orphan blocks occur when multiple valid blocks are created at the same height, but only one can be included in the blockchain. The rest become orphans, causing a temporary divergence in the network.

Instead of discarding these orphan blocks, Ethereum implements a mechanism to handle them. Orphan blocks are preserved as valid blocks and can be included in the blockchain by a miner with a lower fee. This approach ensures that the efforts put into creating these blocks are not wasted and provides an opportunity for miners with lower fees to participate in the network's block creation process.

3.2. Implications to the Blockchain Trilemma

3.2.1. Security

After conducting a thorough technological analysis of Ethereum, several security concerns have emerged, warranting a closer examination of the network's resilience and vulnerabilities. In this section, we will delve into the key security aspects of Ethereum, exploring issues such as the impact of its reduced block time, the nothing-at-stake problem in proof of stake, and the vulnerabilities associated with smart contracts.

- **Long-Range Attacks and Forks:** As mentioned earlier, Ethereum's reduced block time, while enhancing transaction throughput, increases the likelihood of forks in the blockchain. This creates a higher probability of long-range attacks, where an attacker attempts to rewrite the entire blockchain's history from a specific block in the past. Since there is more time for new forks to emerge, attackers can potentially construct longer chains, undermining the blockchain's integrity.

- **Nothing-at-Stake Problem:** The nothing-at-stake problem refers to the possibility that validators in a proof-of-stake (PoS) system may attempt to validate multiple conflicting blocks in an attempt to maximize their chances of being rewarded. Unlike in proof-of-work (PoW) where miners have significant costs associated with mining a block, PoS validators do not face any real-world costs. Thus, they might be incentivized to validate multiple forks simultaneously, increasing the chances of blockchain divergence.

- **Immaturity of Proof of Stake:** While PoS holds great promise as a more energy-efficient consensus mechanism compared to PoW, it is relatively less battle-tested in real-world scenarios. As Ethereum transitions to PoS with Ethereum 2.0, it exposes itself to potential vulnerabilities and challenges that may not have

been fully explored during its testing phase. This can lead to unforeseen security risks during the transition period.

- **Smart Contract Vulnerabilities:** Ethereum's groundbreaking capability of enabling smart contracts also introduces a unique security challenge. Smart contracts are self-executing pieces of code that operate on the Ethereum blockchain. Any vulnerability or flaw in the code can be exploited by malicious actors, leading to potentially catastrophic consequences. High-profile incidents, such as the DAO hack in 2016 (Cryptopedia, 2022), have demonstrated the importance of rigorous code auditing and security best practices for smart contract development.

- **Front-Running and Transaction Reordering:** Front-running occurs when a malicious actor anticipates a pending transaction and quickly executes a transaction with higher gas fees to profit from the original transaction's actions. Additionally, transaction reordering can lead to a change in the order of transactions in a block, which can impact the outcome of certain applications, like decentralized exchanges (DEXs). Both of these issues require careful consideration and mitigation strategies to ensure fairness and prevent exploitation.

- **Governance and Protocol Upgrades:** Decentralized governance plays a significant role in Ethereum's security. It involves making important decisions about the protocol and upgrades through community consensus. While this promotes decentralization, it can also lead to contentious debates and potential hard forks if disagreements arise.

Despite these challenges, Ethereum has demonstrated resilience and adaptability throughout its development. The community continuously addresses security concerns, enhances the protocol, and promotes responsible development practices. Regular audits, bug bounties, and responsible disclosure policies contribute to making Ethereum more robust and secure over time.

3.2.2. Decentralization

Decentralization lies at the heart of Ethereum's vision, aiming to empower a trustless and inclusive ecosystem. However, the decentralization of Ethereum is a nuanced subject, influenced by various factors, including its consensus mechanism, block time, and barriers to entry. In this analysis, we will explore the decentralization aspects of Ethereum, considering the impact of its proof-of-stake (PoS) consensus, its lower barriers to entry, and the effect of reduced block time on network participation.

- **Proof of Stake and Centralization Issues:** Ethereum's transition to a PoS consensus mechanism is intended to reduce energy consumption and enhance scalability. However, PoS has raised concerns about centralization due to the "rich get richer" effect. In PoS, validators are chosen based on the amount of cryptocurrency they stake as collateral. The more cryptocurrency a validator holds, the higher the chance of being selected. This creates an advantage for wealthier participants, leading to a concentration of power among a few prominent stakeholders. As a result, the decentralization ideals of Ethereum face challenges in achieving a more distributed validator network.

- **Lower Barriers to Entry and Increased Decentralization:** One advantage of PoS over traditional PoW is its lower barriers to entry. PoW mining demands expensive hardware and electricity costs, limiting participation to those with significant financial resources. In contrast, PoS requires validators to hold and stake a certain amount of cryptocurrency, making it more accessible to a broader range of participants. This reduced barrier fosters greater decentralization, as more individuals can actively engage in the network's validation process, furthering the ethos of inclusivity.

- **Impact of Reduced Block Time:** Ethereum's reduced block time, while enhancing transaction throughput, raises concerns about centralization. A shorter block time means more frequent block creation, intensifying competition among miners to validate and propagate their blocks quickly.

Larger mining operations with better resources and network connectivity may have an advantage in this competitive landscape, potentially leading to the centralization of mining power. The challenge lies in striking a balance between transaction speed and maintaining a decentralized network.

- **Governance and Protocol Upgrades:** Decentralized governance is vital to the long-term decentralization of Ethereum. The decision-making process for protocol upgrades and network improvements requires community consensus. While decentralized governance promotes inclusivity, it also presents challenges in reaching agreements that may lead to hard forks.

3.2.3. Scalability

Scalability is a critical consideration for any blockchain network seeking widespread adoption and utility. As Ethereum continues to solidify its position as a leading decentralized platform, addressing scalability challenges becomes paramount. Among the key factors impacting Ethereum's scalability, the reduction of block time to 15 seconds per block stands out as a significant technological change.

- **Reduced Block Time:** One of the notable efforts to enhance Ethereum's scalability was the reduction of block time from approximately 10 minutes in Bitcoin to around 15 seconds per block in Ethereum. This reduction significantly improves the transaction throughput of the network, allowing more transactions to be processed within a shorter time frame. Faster block times lead to quicker transaction finalization, improving the user experience and facilitating real-time applications on the blockchain.

- **Block Size and Gas Limit:** With the decrease in block time, there is a need to strike a balance between the block size and the gas limit, which determines the maximum computational capacity of each block. Larger block sizes can accommodate more transactions, improving scalability, but they also increase

the time and resources required for block validation and propagation. Setting an appropriate gas limit is essential to prevent bloated blocks and potential centralization of mining power.

- **Network Latency and Synchronization:** Reduced block time can lead to higher network latency, especially in situations of increased network congestion. As blocks propagate through the network, nodes need to synchronize quickly to maintain a consistent blockchain state. Delays in synchronization may cause temporary forks or conflicts in the network, impacting scalability and user experience.

Chapter 4: Solana

Moving on to our exploration of the second technology under scrutiny, Solana, we will adopt a parallel structure to our examination of Ethereum. Our approach will involve a thorough examination of Solana's technological underpinnings and operational mechanisms (Solana, 2023). Subsequently, in the following section, we will consolidate our insights as we explore the ramifications of Solana's implementation within the context of the Blockchain Trilemma.

4.1. Technical Differences With Ethereum

4.1.1. Proof of History

In traditional blockchain systems, such as Bitcoin and Ethereum, transactions are grouped into blocks for synchronization. This means that a transaction must wait until a predefined period known as the "block time" has elapsed before it can be processed. In the Proof of Work (PoW) consensus protocol, longer block times (around 10 minutes) are set to reduce the likelihood of multiple validators creating valid blocks simultaneously. This precaution is necessary due to the competitive nature of PoW mining and the energy-intensive computations involved.

Proof of Stake (PoS) consensus, on the other hand, doesn't have the same rigid block time requirements as PoW. PoS relies on validators who hold a stake in the network, and thus the need for lengthy block times is mitigated. This theoretically allows for faster transaction confirmations.

However, PoS introduces a unique challenge. Without accurate timestamps, validators struggle to establish the correct order in which blocks are generated. This temporal uncertainty can disrupt the smooth progression of the blockchain, as the correct sequence of events is essential for its integrity.

This challenge is where Proof of History (PoH) steps in. Solana's innovative approach combines PoS with PoH to address this problem. PoH introduces reliable timestamps, providing a solution to the issue of ordering blocks. By doing so, it enhances the overall performance and efficiency of the blockchain network as the time required for validation reduces considerably.

Timestamp Creation

The creation of accurate timestamps within distributed networks is a complex challenge. Conventional clocks aren't suitable due to the inherent variations in timekeeping among different computers. Solana, however, takes a distinctive approach by prioritizing the establishment of an event sequence rather than achieving clock synchronization across all nodes.

Solana's innovative solution involves utilizing hashes as a measure of time, mirroring how regular clocks use seconds. This concept is visually illustrated in Figure 9, providing a tangible representation of this novel approach.

Index	Input 1	Input 2	Output
1	"Hello World"		Hash1
2	Hash1		Hash2
3	Hash2	Transaction A	Hash3
4	Hash3		Hash4
5	Hash4	Transaction B	Hash5
6	Hash5		Hash6

Figure 9. Proof of History Timestamp Generation

Central to this concept is the creation of a sequential chain of hashes. This begins with the initial hash, derived from a random string (e.g., "Hello World"). The resulting output becomes the input for the subsequent hash computation. Consequently, the second hash is determined by hashing the previous Hash1, and this pattern continues, forging an unbroken chain of interconnected hashes.

The adoption of hashes as temporal units carries significant importance. These cryptographic hashes establish a logical flow of events by ensuring that one event can be proven to have occurred before another. This sequential relationship is akin to the way traditional clocks convey time progression in seconds.

However, the significance of employing hashes doesn't end here. Their cryptographic nature imbues the chain with properties that enhance security and verifiability. Each new hash not only incorporates the data of the previous hash but also inherently ties the new block to the entire history that came before it. This property forms a critical component of Solana's Proof of History mechanism, solidifying the integrity and immutability of the chronological sequence.

The last phase in establishing accurate timestamps involves seamlessly connecting these timestamps with their respective events. This crucial step is illustrated in Figure 9, specifically highlighted in the row labeled with index 3. This visualization demonstrates how events and timestamps are linked.

When a new transaction arrives, it becomes part of the data inputs used in generating timestamps. This incoming transaction is combined with the hash output created in the previous steps. This hash acts like a unique data identifier, encapsulating both the earlier data and the current transaction.

The resulting hash, termed Hash3, acts as the link between the specific transaction (let's say, Transaction A) and the corresponding timestamp (for instance, timestamp 3). This ensures that Transaction A is securely tied to its correct timestamp, preserving an accurate record in the blockchain's history.

Leader Selection and Rotation

As previously highlighted, it's crucial to clarify that Proof of History doesn't function as a distinct consensus mechanism but rather serves to reinforce the Proof of Stake framework. In parallel with the Proof of Stake approach, the mechanics of Solana permit only one validator to produce ledger entries. (It's worth noting that Solana differs from conventional block-based systems, as it operates at the transaction/entry level.)

The singular leadership in producing ledger entries yields a significant benefit: it ensures that all validators can recreate identical replicas of the ledger. However, this approach also carries a drawback: the potential for a malicious leader to wield censorship power over votes and transactions. Detecting this censorship becomes complex since it's indistinguishable from network packet losses. Consequently, the straightforward solution of indefinitely assigning a single node as the leader is infeasible.

Solana addresses this challenge by adopting a strategy that mitigates the influence of a malicious leader. This strategy revolves around the concept of leadership rotation, where the role of the leader shifts periodically. This rotation mechanism effectively curtails the impact that a malicious leader could exert over the network.

The operational framework for this rotation entails the selection of a leader for a predefined timeframe known as an epoch. Remarkably, the duration of an epoch is quantified using the unit of hashes as a temporal measure. Once designated, the leader commences the process of receiving transactions and building the ledger, a procedure analogous to the depiction in Figure 9. As the epoch concludes, often after a specific number of hashes (for instance, 100), the mantle of leadership transitions to the next designated leader.

This selection of leaders is orchestrated in advance, with the leader for the next epoch being identified while the current epoch is underway. This foresighted scheduling ensures a seamless transition between leadership roles. It's essential to

underscore that this selection process adheres to the same Proof of Stake principles employed in Ethereum, adhering to established and tested mechanisms.

Validation through Parallel Computing

As previously mentioned, in the context of earlier blockchain systems, transactions necessitated being grouped within blocks for transmission. This methodology, however, imposed a constraint wherein a transaction could only be processed once a specific time interval, known as the "block time," had elapsed.

Solana, in contrast, operates on a different premise. The unequivocal order of events, fortified by the Proof of History mechanism, renders the notion of waiting for transactions to aggregate within blocks obsolete. This distinctive attribute obviates the need for a waiting period before transmitting and validating transactions. Instead, transactions can be promptly transmitted once they are appended to the ledger.

Consequently, when a designated leader receives a transaction and incorporates it into the ledger, the leader promptly disseminates the most current version of the ledger to the network's validators. This ledger transmission enables the validators to engage in the validation process, scrutinizing the ledger's updates and verifying the accuracy of the included transactions.

This streamlined approach yields a tangible advantage: By the time validators engage in voting and adhere to the consensus algorithm, the validation process has already been completed. Consequently, the validators' efforts are streamlined, and no additional time is expended. This efficiency is a direct result of Solana's innovative approach, allowing validators to seamlessly transition from transaction validation to the consensus algorithm, maximizing the utilization of available resources.

Furthermore, aside from the mentioned benefits, validators have the option to leverage parallel computing techniques in order to expedite the validation process for transactions. This is achieved by breaking down the validation tasks into smaller

segments, also known as chunks. These chunks are essentially distinct portions of the ledger. By utilizing parallel computing, individual CPUs within a validator's setup can be allocated to validate specific chunks simultaneously. This division of labor capitalizes on the fact that, for validation purposes, validators only require access to the previous output. As a result, the combination of segmenting the ledger and employing parallel computing leads to a reduction in the time needed for the entire validation process.

4.1.2. Confirmation Times

In comparison to Ethereum's block time, which stands at approximately 15 seconds, the actual confirmation time for transactions often extends far beyond this due to considerable congestion. This congestion tends to drive up gas fees and introduces substantial variability in confirmation times.

However, Solana takes a notably different approach by combining Proof of History (PoH) with Proof of Stake (PoS) to shape its blockchain dynamics. In Solana's context, instead of the conventional notion of "blocks," the interval between votes serves as an analogous concept. Remarkably, this interval is remarkably brief, clocking in at around 800 milliseconds.

This strategic incorporation of Proof of History and Proof of Stake has significant implications for the network's efficiency. Solana's adoption of such a short interval between votes mitigates congestion to a great extent. The outcome is a network marked by consistently low and stable confirmation times, accompanied by fees that remain predictable.

4.1.3. Turbine Block Propagation

The Turbine block propagation method is an innovative approach to efficiently distribute new blocks across its blockchain network. It's designed to minimize duplicate messages, optimize bandwidth usage, and enhance the speed and reliability of block propagation. The process is as follows.

- **Neighborhood Formation:** Solana's network is divided into neighborhoods, each containing a group of validators. Validators with higher stakes are placed in higher layers of neighborhoods, creating a hierarchy based on their influence in the network.

- **Focal Nodes:** Within each neighborhood, a validator is designated as a "focal node." The focal node acts as a central hub for receiving and distributing chunks of transactions.

- **Chunk Creation:** When a new block is to be propagated, it's divided into smaller segments called "chunks." Each chunk contains a subset of transactions. These chunks can be processed and validated independently. Data is divided into chunks for a specific reason: the utilization of Erasure Codes. These codes enable validators to reconstruct the complete original dataset using just half of the available chunks. This incorporation enhances the method's resilience against malicious actors.

- **Local Validation:** The focal node of a neighborhood receives a chunk first. It validates the transactions within the chunk to ensure their authenticity and correctness.

- **Selective Forwarding:** Instead of broadcasting the entire chunk to all validators, the focal node selectively forwards the chunk to neighboring validators within the same neighborhood. This minimizes the duplication of messages and optimizes bandwidth usage.

- **Aggregated Messages:** Validators within a neighborhood aggregate the validated chunks into larger messages. These aggregated messages contain multiple chunks and are more efficient to transmit compared to individual chunks.

- **Communication Through Spokes:** Selected validators, known as "spokes," are responsible for maintaining communication between different neighborhoods. Spokes help relay aggregated messages from one neighborhood to another, ensuring cross-neighborhood connectivity.

4.2. Implications to the Blockchain Trilemma

4.2.1. Security

Although focusing on scalability, the technology features implemented in Solana do have an impact on security, trying to reduce several issues in Ethereum.

Proof of History (PoH) forms a core element of Solana's security framework, providing an immutable timestamp that guarantees the chronological order of transactions. This timestamp-based ledger eradicates concerns about fork-related issues, enhancing the security and resilience of the blockchain against tampering.

Confirmation times, streamlined by the turbine block propagation method, play a crucial role in reducing network congestion (Solana Floor Content, 2023). In addition, the integration of parallel processing further accelerates confirmation times, which reduces the susceptibility to double-spending attacks. Moreover, Turbine's approach using Erasure Codes further bolsters the network's defense against potential malicious actors as it ensures data reconstruction even in scenarios of missing or corrupted data, strengthening the security posture.

However, the Solana blockchain exhibits three significant vulnerabilities within its security framework. These vulnerabilities warrant careful consideration due to their potential implications for the network's robustness and overall reliability.

The first vulnerability pertains to the integrity of the Proof of History mechanism, a key element of Solana's architecture. This mechanism heavily relies on the accuracy of the initial timestamp. In the unfortunate event that this initial timestamp is tampered with, the entire integrity of the Proof of History could be compromised. This potential vulnerability emphasizes the criticality of ensuring the tamper-proof nature of this timestamp and highlights the need for rigorous security measures to safeguard against malicious tampering attempts.

A second notable vulnerability stems from the relative newness of Solana in the blockchain landscape. While the platform boasts innovative features and high scalability potential, its limited track record in real-world scenarios raises concerns about battle-tested security. Given the rapid pace of evolution in emerging threats and attack vectors, the absence of substantial real-world testing could potentially leave the network susceptible to vulnerabilities that have yet to be identified or addressed. This underscores the importance of continuous security auditing, stress testing, and community collaboration to identify and rectify vulnerabilities proactively.

Furthermore, a third vulnerability involves the network's decentralization and scalability dynamics. Inherent challenges within the current technological landscape make it complex to become a validator on the Solana network. This limited accessibility to validation roles could result in performance disparities that deviate from the theoretical ideals of decentralization and scalability. These disparities not only impact the network's overall security posture but also render it susceptible to Denial of Service (DoS) attacks, particularly in light of the network's ability to maintain impressively low transaction fees.

4.2.2. Decentralization

Regarding decentralization, Solana features don't produce much impact compared to Ethereum. As it also relies on Proof of Stake (PoS) the decentralization characteristics introduce both familiar benefits and risks.

As with Ethereum, Solana's PoS obtains a reduction in the barriers to entry. PoS allows participants to engage in consensus by staking tokens, requiring less energy-intensive computations than traditional Proof of Work. This lowers the entry threshold, fostering broader participation and contributing to a more decentralized validator landscape.

However, PoS also brings to the forefront the "rich get richer" phenomenon. Validators with larger stakes have proportionately greater influence in block validation and consensus decisions. This dynamic can lead to the concentration of power and resources in the hands of a few, potentially undermining the foundational principle of decentralization.

However, there is a negative impact of Solana's technology on decentralization due to its confirmation time optimization. While rapid confirmation times enhance user experience and reduce congestion, they could inadvertently amplify the risk of centralization. Validators with superior computational capabilities might have a higher likelihood of being selected as leaders more frequently due to their faster responses. This could tilt the playing field in favor of those with better resources, potentially diminishing the diverse and decentralized nature of the validator set.

Moreover, a key factor impacting decentralization is the hardware requirement for becoming a validator on Solana. The need for a robust setup — including a minimum of a 12-core/24-thread CPU, 128 GB RAM, and 500 GB disk space — creates a barrier to entry, limiting potential validators. This reduction in participants hampers decentralization and, consequently, has a direct and adverse effect on scalability.

By excluding many due to the demanding hardware prerequisites, Solana inadvertently shrinks its validator pool. This lack of diversity in validators contradicts

the essence of decentralization, which relies on broad participation. The network's scalability ambitions are undermined as the limited validator count inhibits efficient transaction processing for a growing user base.

4.2.3. Scalability

In contrast with security and decentralization, Solana's technology features have a profound impact on its scalability. By leveraging innovative solutions and optimizing data processing, Solana has crafted a framework that enhances its scalability in significant ways.

The usage of Proof of History (PoH) significantly impacts Solana's scalability by contributing to the platform's ability to handle a high volume of transactions efficiently. By providing an immutable and verifiable timestamp for events on the blockchain, PoH establishes a reliable chronological order. This chronological ordering aids in the parallel processing of transactions, enabling multiple transactions to be executed concurrently rather than sequentially. As a result, PoH's accurate timestamps, combined with parallel processing, alleviate bottlenecks that often limit the scalability of other blockchain networks.

In addition, the Turbine block propagation serves as the cornerstone of the scalability framework by aggregating validated chunks of data and minimizing redundant messages allowing Solana to optimize the data transmission process. This not only conserves bandwidth but also streamlines the propagation of transactions and blocks across the network, ensuring that scalability remains unhindered even as transaction volumes increase.

All this allows for rapid confirmation times ensuring a quick transaction finality which reduces the backlog of unprocessed transactions and prevents network congestion.

However, as previously discussed, a significant challenge facing the Solana network revolves around the scarcity of validators due to the demanding requirements for assuming this crucial role. This challenge presents a complex situation with two primary consequences that could impact the network's transaction handling ability and make it vulnerable to potential Denial of Service (DoS) attacks. This concern becomes even more relevant considering the network's practice of offering consistent and low transaction fees.

The connection between the limited number of validators and the network's transaction capacity is a crucial aspect to consider. With the pool of validators constrained due to the high hardware demands, the network's ability to process and confirm transactions could be hampered. This limitation could hinder the network's goal of efficiently handling a large volume of transactions, which is a fundamental requirement for a scalable and practical blockchain platform.

Chapter 5: Avalanche

The next technological advancement meriting examination is Avalanche. Just as in the cases of Ethereum and Solana, our exploration framework shall entail a meticulous dissection of the key distinctions inherent to Avalanche when juxtaposed with the aforementioned technologies. This will be succeeded by an exhaustive evaluation of how these divergences wield an influence on each facet encapsulated within the paradigm of the Blockchain Trilemma.

Ethereum and Solana have garnered significant attention due to their unique features and functionalities. Avalanche, in a similar vein, has carved its niche within the blockchain landscape with its distinctive consensus mechanism and scalable architecture. A rigorous comparative analysis of these platforms serves as the bedrock of our examination.

5.1. Technical Differences with Ethereum and Solana

5.1.1. DAG Structure

In the preceding blockchain technologies we have examined, information was encoded within a block structure. This structural arrangement was imperative in arranging data systematically, ensuring a universally acknowledged sequence of blocks, and facilitating the uniform distribution of the blockchain ledger among all participants, thereby upholding its integrity. This simplifies the validation process for each new block and, therefore, consensus can be reached more efficiently.

Moreover, this block structure introduced a pivotal advantage in the form of a linear, chronological sequence, aptly named the blockchain. This linear concatenation serves as a potent deterrent against long-range attacks, a security concern wherein a malicious actor attempts to rewrite past transactions in an effort to alter the history of the blockchain. The inherent immutability of the chain makes such endeavors laborious and resource-intensive, thus bolstering the overall security and reliability of the blockchain network.

However, employing a block structure presents several inherent drawbacks. The primary inconvenience stems from its propensity to result in diminished throughput. In cases where the volume of transactions surpasses the capacity of a single block, certain transactions inevitably confront a delay until the subsequent block is formed. This latency in transaction confirmation can undermine the efficiency and responsiveness of the blockchain network.

A secondary concern arises in the form of potential forks. The necessity to ensure the chronological sequence of blocks necessitates a minimal time interval between block creations to account for network latency. This time constraint, while ensuring order, introduces the possibility of forks, wherein competing versions of the blockchain emerge due to slight variations in block arrival times. This divergence undermines the network's cohesion and introduces complexities for participants.

These challenges were notably conspicuous within the Bitcoin and Ethereum ecosystems. A paradigm shift transpired with Solana, which adeptly mitigated these predicaments through the incorporation of the Proof of History (PoH) mechanism alongside the Proof of Stake (PoS) consensus protocol. This strategic fusion enabled Solana to operate at the transaction level, obviating the constraints of the block-based approach. In a contrasting vein, Avalanche undertakes a complete reimagining of the quandary. It forgoes the conventional Blockchain data structure and embraces a Directed Acyclic Graph (DAG) architecture.

A Directed Acyclic Graph (DAG) is a specific type of data structure that represents a set of elements (usually called nodes) connected by directed edges, where

the connections between nodes have a defined direction and do not form any cycles. The term "acyclic" signifies that there are no loops or closed paths in the structure. Each edge points from one node to another, indicating a specific relationship between them.

In the context of blockchain technology, a DAG is an alternative to the linear block structure commonly associated with traditional blockchains like Bitcoin. In a DAG-based system, transactions or data units are represented as nodes in the graph, and the edges represent relationships or dependencies between these transactions. Unlike the linear, sequential blocks in traditional blockchains, where each block references the previous block, transactions in a DAG can reference multiple previous transactions, creating a more complex network of connections. Figure 10 represents this structure.

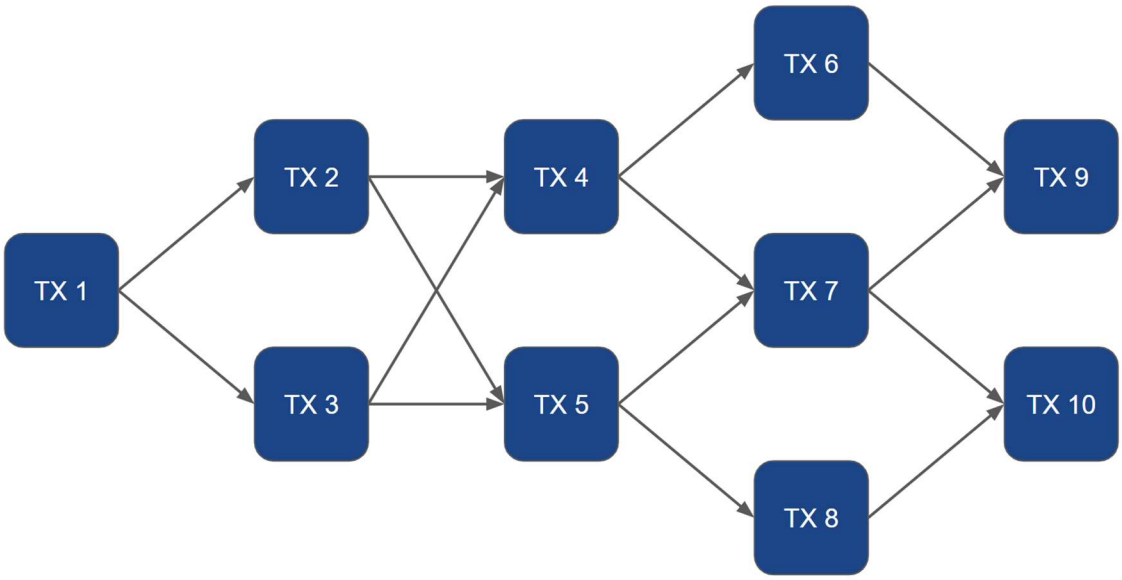

Figure 10. DAG Structure

Figure 10 provides a visual depiction that underscores the primary differentiators between the DAG architecture and the structure characteristic of the previously examined blockchains. Notably, two pivotal distinctions are readily discerned. Firstly, the DAG framework operates at the transaction level, a departure from the block-centric approach observed in conventional blockchains. Secondly, this configuration engenders the existence of multiple forks, as evidenced in the instances

of TX2 and TX3. In such cases, the precise chronological order of these transactions becomes indeterminable.

The adoption of this structure holds the potential to yield notable advantages compared to the classical Blockchain. The avoidance of temporal delays associated with block arrivals and the elimination of constraints tied to block sizes emerge as immediate benefits. Furthermore, the pivotal capability to parallelize the validation process (for instance TX2 and TX3 could be validated in parallel) manifests as a catalyst for heightened throughput and expedited transaction confirmation times.

Certainly, the infeasibility of achieving these objectives served as the impetus for various other blockchain platforms to embrace the conventional Blockchain structure. This transition became a necessity due to the challenges posed by operating at the transaction level, as it heightened the propensity for forks, a complexity that was arduous to effectively manage.

However, a paradigm shift is evident through Avalanche's innovative approach to its consensus algorithm. This pioneering alteration has paved the way for the actualization of the DAG structure, diverging from the well-trodden path of classical Blockchain architecture. This strategic evolution has empowered Avalanche to seamlessly adopt and capitalize on the inherent potential of the DAG framework. The culmination of these advancements is a remarkable surge in throughput, a testament to the efficacy of the DAG structure in addressing the longstanding issues that constrained previous attempts.

5.1.2. Avalanche Consensus Algorithm

As previously noted, the adoption of the DAG structure owes its realization to the pioneering Avalanche consensus algorithm. Nevertheless, achieving this milestone was far from effortless and required a series of progressive updates, with each iteration building upon the advancements of its forerunners. The initial iteration, termed

"Slush," laid the groundwork, followed by subsequent versions—namely "Snowflake," "Snowball," and culminated in the pinnacle iteration, "Avalanche" (Buttolph, 2022).

Slush

The initial action taken by each node involves casting a vote to determine the validity of the received transaction. After this step, each node initiates the process of querying a randomly selected group of nodes to ascertain their voting stance. This procedural sequence is elucidated in Figure 11.

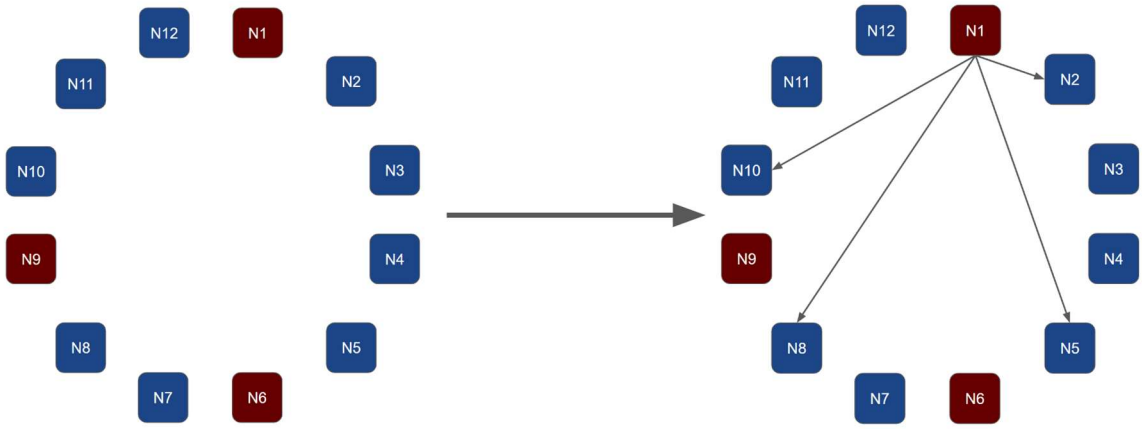

Figure 11. Slush Algorithm

In the depicted instance within Figure 11, Node N1 undertakes inquiries directed towards Nodes N10, N8, N5, and N2. Subsequently, Node N1 accumulates responses indicating that the majority among these nodes have voted in favor of validity. Consequently, Node N1 adjusts its initial vote from "invalid" to "valid," in alignment with the prevailing majority.

This iterative process persists across all nodes, advancing through diverse rounds of queries, until a unanimous consensus is achieved across the entire network. This consensus necessitates a comprehensive agreement, rather than a mere majority consensus (e.g., a scenario where 90% of the network votes "valid" is insufficient; 100% alignment is obligatory). However, a potential vulnerability emerges when a malicious

node deliberately skews its voting response to align with the minority instead of the majority. For instance, if Node N1 was malevolent, it could persistently uphold its original "invalid" vote, thwarting convergence to a definitive consensus. This disruptive behavior triggers an indefinite loop as the network remains devoid of a harmonized consensus resolution.

Snowflake

To address the issue of continuous looping within the Slush algorithm, Avalanche introduced an upgraded solution named Snowflake. This enhanced approach incorporates a "counter" mechanism to effectively manage the looping concern.

Here's how Snowflake works straightforwardly: In the Snowflake algorithm, each participating node is assigned a counter. This counter keeps track of how consistently a node sticks to its initial voting choice across several rounds. If a node maintains its initial vote through consecutive rounds, the counter increases by one. Conversely, if a node changes its vote, the counter resets to its starting value.

When a node's counter reaches a specified threshold, the node enters a "locked" state. In simpler terms, this means the node's voting decision becomes fixed and remains unchanged in subsequent rounds. This locking mechanism is a crucial way to prevent never-ending loops. If a node doesn't achieve the locked state even after multiple rounds, it might raise concerns about that node's intentions.

However, while Snowflake effectively addresses looping concerns, there's a trade-off to consider in terms of efficiency. The counter's periodic resetting before reaching the locked state can slow down the process. This elongates the time needed to reach a consensus among the participating nodes.

Snowball

In response to the challenge of diminished efficiency stemming from recurrent counter resets, Avalanche devised the Snowball algorithm. This innovative iteration of the consensus mechanism addresses the efficiency concern by altering the counter behavior. Unlike its predecessor, Snowflake, where the counter could reset, the Snowball algorithm maintains an unbroken record of query outcomes.

In essence, Snowball functions as follows: When a node participates in the consensus process, it initiates a series of rounds of queries. In each round, the node gathers information from its peers regarding their voting choices. The novelty lies in the fact that the counter's integrity is maintained across all these rounds, eliminating any possibility of resetting.

For instance, let's consider a scenario where a node commences the consensus process with an initial round of queries. If, after this first round, the accumulated vote tally reflects a majority in favor of "valid," the "valid" counter will increase accordingly. As these rounds of queries progress and culminate, the counter will embody a comprehensive summary of the overall outcomes. For example, the counter could display a total of 7 rounds endorsing "valid" and 2 rounds supporting "invalid." Based on the outcome, where a greater number of rounds resulted in "valid" votes, the node will subsequently cast its vote as "valid."

This mechanism introduces a level of aggregation and deliberation that leverages the collective wisdom of the network to reach a consensus. By maintaining a holistic view of all the queries and their results, the Snowball algorithm empowers nodes to base their ultimate votes on a comprehensive perspective, derived from multiple rounds of input.

Avalanche

Avalanche distinguishes itself from its predecessors through the culmination of its consensus mechanism, which enables the implementation of the previously discussed DAG (Directed Acyclic Graph) structure.

Central to Avalanche's distinctiveness is its ability to generalize the Snowball algorithm, thereby addressing the intricate challenge associated with implementing the DAG structure. Inherent to DAGs are the continuous occurrences of forks, complicating the consensus process and presenting the potential for disagreements regarding the validity of transactions. This phenomenon paves the way for the double spend problem, a significant concern in the blockchain domain.

The double spend problem, in essence, refers to a scenario where a malicious actor attempts to spend the same cryptocurrency funds more than once. This can occur due to the lack of a central authority in decentralized systems to ensure that transactions aren't duplicated. This issue jeopardizes the integrity and security of a blockchain network, undermining the trust that it seeks to establish among participants.

To surmount this challenge, Avalanche creatively applies the foundational principles of the Snowball algorithm. However, there's a crucial departure: instead of nodes voting solely on a single transaction, they extend their scrutiny to encompass a range of transactions, specifically including several levels of predecessors.

For instance, envision the illustrative Figure 10. In Avalanche's context, the process of validating a transaction like TX8 extends beyond merely assessing its individual validity. Instead, the validating node evaluates not only TX8 but also its preceding transactions. In a hypothetical scenario where three levels of predecessors are considered, the validation process would encompass TX5, TX2, TX3, TX1, and TX8.

This multifaceted approach shifts the consensus landscape from one of binary voting outcomes—valid or invalid—to one of nuanced confidence levels. Rather than definitively labeling a transaction as correct or incorrect, nodes express their

confidence levels based on the collective assessment of multiple transactions. For instance, a node might assert that out of five transactions analyzed, one is erroneous, resulting in a confidence level of 80% that the chain analyzed is correct.

After this complex voting process, nodes proceed to engage in a sequence of querying, analogous to the Snowball algorithm. Over time, consensus coalesces around transactions that demonstrate correctness. Importantly, while the order of these transactions might remain ambiguous, the potential security vulnerabilities associated with forks are mitigated.

In sum, Avalanche's ultimate innovation lies in its ability to navigate the intricate landscape of the DAG structure. By synthesizing the principles of Snowball, extending transaction assessment to predecessors, and embracing confidence-based voting, Avalanche achieves a robust consensus mechanism that not only accommodates the challenges of forks but also fortifies the security of the blockchain ecosystem.

5.1.3. Subnets

Ethereum gained immense popularity for its unparalleled flexibility in developing specialized applications atop its blockchain, negating the need to create separate blockchains for each application. However, this shared blockchain ecosystem posed a challenge: while each application was distinctive, they were obliged to adhere to common features such as the consensus algorithm and protocols.

Avalanche, in its pursuit of even greater flexibility, introduces a groundbreaking concept called "subnets." Subnets constitute discrete, isolated networks operating within the broader Avalanche framework, meticulously tailored to cater to specific usage scenarios, security prerequisites, or organizational imperatives.

Fundamentally, subnets function as tailored environments housing transactions, validators, and participants uniquely configured for specific purposes. This partitioning of the overarching network empowers Avalanche to accommodate a diverse array of applications and requirements without compromising network cohesion or operational efficiency.

Subnets grant users the authority to configure their blockchain networks in alignment with their precise intentions. Consider, for example, an entity aiming for heightened privacy; such an entity could establish a subnet fortified with stringent privacy measures. Conversely, a decentralized finance (DeFi) initiative might fashion a subnet optimized for high-speed transactions and seamless interaction with assorted financial instruments.

The segmentation of the network into subnets presents Avalanche with an extraordinary level of adaptability. Each subnet operates with a degree of autonomy, enabling it to define its distinct rules, parameters, and consensus mechanisms. This decentralized approach facilitates the diversification of applications, accommodating a wide spectrum of requirements within a unified network architecture.

A critical aspect of the subnetting concept is its contribution to Avalanche's horizontal scalability prowess. This dynamic capacity to scale horizontally entails the establishment of new subnets as transaction demand escalates or novel specialized use cases arise. The introduction of fresh subnets serves to avert congestion while preserving the efficacy of network operations.

5.1.4. Primary Network

The intricate and versatile infrastructure of subnets within the Avalanche ecosystem is meticulously organized under the guidance of the Primary Network (Sekniqi, Laine, Buttolph, & Gün Sirer, 2020). This foundational layer constitutes the cornerstone upon which the entire Avalanche architecture is built. Comprising three essential blockchains—the X-Chain, the C-Chain, and the P-Chain—the Primary

Network orchestrates the multifaceted functionalities that underpin Avalanche's diverse applications and subnets.

- **X-Chain (Exchange Chain):** The X-Chain assumes a pivotal role in the Avalanche landscape by overseeing the management of token transactions and the creation of tokens themselves. It acts as the conduit for seamless transfers of tokens, providing the backbone for economic activities within the ecosystem. Its responsibilities encompass facilitating rapid peer-to-peer transactions and token issuance, contributing to the fluidity and efficiency of Avalanche's economic transactions.

- **C-Chain (Contract Chain):** Central to Avalanche's versatility is the C-Chain, entrusted with managing smart contracts and overseeing the dynamic landscape of applications constructed within the ecosystem. It serves as the hub for the execution of code-based applications, enabling developers to craft decentralized solutions with ease. Of significance is its interoperability with Ethereum's Virtual Machine, positioning Avalanche as an attractive destination for Ethereum developers seeking to harness the benefits of the Avalanche platform without the complexities of migration.

- **P-Chain (Platform Chain):** Steering the orchestration of subnets and the intricate dance of validators is the P-Chain. Tasked with maintaining the integrity of the entire ecosystem, the P-Chain assumes the vital role of coordinating validators within subnets and orchestrating the consensus mechanism that guarantees network security. This control mechanism acts as the linchpin ensuring the collaborative harmony of Avalanche's distributed infrastructure.

5.2. Implications to the Blockchain Trilemma

Similar to the examination conducted on the other technologies explored, the analysis culminates with an assessment of the implications that this particular technology holds for the Blockchain Trilemma.

5.2.1. Security

The implementation of a Directed Acyclic Graph (DAG) structure heralds a fundamental transformation in transaction confirmation and security paradigms. This structural innovation serves as a robust defense mechanism against the risk of "double spending" attacks, a threat ubiquitous in conventional blockchains. Unlike the linear progression of transactions within traditional blocks, the DAG structure ushers in a realm of parallel and asynchronous transaction confirmations. This pivotal feature dramatically truncates the temporal window within which malicious actors could exploit transactions for double spending, thus augmenting security against this specific form of attack.

Moreover, Avalanche's distinctive consensus algorithm further fortifies its security architecture by thwarting the ominous specter of the "long-range attack." This vector of attack involves manipulating transactions from bygone epochs in the blockchain's history. The probabilistic nature of Avalanche consensus, buttressed by its repeated voting cycles, erects an imposing barrier to long-range attacks. The swift and iterative convergence of consensus renders the endeavor of revising historical transactions resource-intensive and implausible, thus engendering an enhanced security layer against this particular breed of threat.

Finally, the integrative utilization of Subnets and the Primary Network affords the Avalanche ecosystem an intricate tapestry of security enhancements. Tailored to specific use cases, Subnets serve as bulwarks against the relentless tide of "transaction

spam attacks." By configuring subnets to optimally accommodate distinct functionalities, Avalanche curtails the effectiveness of attacks that seek to inundate the network with low-value transactions. Moreover, the vigilant oversight of the Primary Network acts as a guardian against the amplification of spam attacks across subnets, orchestrating the identification and mitigation of such threats.

However, the adoption of novel technologies and approaches, like the DAG structure and the Avalanche consensus algorithm, introduces a concomitant risk profile. The introduction of novel elements can potentially unveil uncharted attack vectors or interaction intricacies, demanding a learning curve for developers and security experts to discern and rectify potential vulnerabilities.

Furthermore, akin to other blockchain frameworks, Avalanche is not immune to vulnerabilities stemming from smart contracts and the potential for centralization. However, unlike conventional 51% control mechanisms, Avalanche's consensus requires an 80% network control threshold, rendering such control less attainable.

5.2.2. Decentralization

Regarding decentralization, only the integration of subnets and the primary network brings forth a nuanced impact on decentralization. On one hand, subnets offer the advantage of customization for specific use cases, fostering diverse and specialized applications within the ecosystem. The primary network's oversight ensures that while customization is encouraged, certain network standards are maintained, providing a balanced environment for tailored operations.

However, a potential downside is the risk of centralization within subnets. If a subset of validators within a subnet gains disproportionate control, the decentralized nature of the network within that subnet could be compromised. This highlights the need for ongoing vigilance to prevent unintended centralization within specialized segments of the ecosystem.

5.2.3. Scalability

The introduction of the DAG structure in the Avalanche Blockchain has a profound impact on scalability. The asynchronous and parallel transaction confirmation enabled by the DAG structure holds the potential to dramatically enhance scalability. Unlike traditional linear blockchains, where sequential transaction confirmation can lead to congestion, the DAG structure allows for multiple transactions to be confirmed simultaneously. This not only expedites the validation process but also reduces the risk of bottlenecks during peak usage.

Moreover, the consensus algorithm employed by Avalanche significantly impacts the scalability of the technology by allowing transactions to be confirmed rapidly. This rapid confirmation process enhances the overall throughput of the network, catering to high transaction volumes. However, the algorithm's assumption of honest participation might introduce limitations to scalability. In scenarios where a substantial portion of participants collude or behave maliciously, the decentralized validation process could be hindered as more rounds would be needed to achieve consensus. This might potentially slow down transaction throughput and impact overall scalability.

The incorporation of subnets and the primary network further influences scalability within the Avalanche ecosystem. Subnets, tailored to specific use cases, offer the potential to improve scalability by enabling specialized transaction environments. This ensures that transactions with distinct functionalities can be processed independently, preventing congestion on the primary network. The primary network's oversight also maintains consistency across subnets.

Chapter 6: Quantitative Analysis

After thoroughly examining the technical distinctions among Ethereum, Solana, and Avalanche, it becomes imperative to delve beyond theoretical disparities and grasp their tangible repercussions in practical contexts. To achieve this, the execution of a quantitative analysis becomes paramount. This analytical approach not only grants us profound insights into the practical implications but also furnishes us with a systematic framework for juxtaposing these technologies.

Hence, the focal point of this chapter shifts towards an initial process of handpicking and elucidating the metrics that will serve as our compass in this comparative exploration. Additionally, the acquisition of metric-specific data for each technology takes precedence. This data-driven endeavor is fundamental in enabling us to conduct an insightful comparative analysis that encapsulates diverse facets of these technologies.

4.1. Metrics

As mentioned earlier, this section involves the selection and explanation of various metrics used to assess the real-world effectiveness of each technology. Additionally, it encompasses outlining a method for comparing the studied technologies.

Moreover, we will categorize the choice of metrics according to the Blockchain Trilemma. This categorization will grant us insight into how each blockchain addresses the Trilemma and evaluates the effectiveness of their respective approaches.

4.1.1. Security

Nakamoto Coefficient

The Nakamoto Coefficient draws inspiration from the GINI Index, a tool utilized to assess wealth distribution within a nation. In its blockchain context, the Nakamoto Coefficient is constructed as the minimum count of entities necessary to collectively command authority over the network (Bybit, 2022). This metric serves as a direct gauge to comprehend the feasibility of executing a 51% attack on the associated blockchain.

The metric is obtained through the following formula.

$$K = \min\left\{n \in N: \sum_{i=1}^{n} x_i > p \sum_{i=1}^{N} x_i\right\}$$

On the left-hand side of the equation, we observe the accumulation of the control percentages attributed to 'n' validators. Conversely, the right-hand side encapsulates the summation of control percentages attributed to the entire 'N' validators present in the network. This value is then multiplied by the predetermined requisite percentage 'p' needed to acquire control over the network—where 'p' corresponds to the necessary proportion, such as 0.51 if a 51% threshold is essential for network control.

Downtime

As the digital realm becomes increasingly intertwined with our daily lives, the uninterrupted and secure operation of blockchain networks becomes essential. Downtime, often associated with technical glitches, network failures, or malicious attacks, can significantly impact the reliability and integrity of a blockchain. This has prompted the exploration of downtime as a security metric, shedding light on its potential significance in the realm of blockchain analysis.

Downtime in the context of blockchain refers to periods during which the network is unavailable, incapable of processing transactions, or executing smart contracts. While downtime might appear to be a conventional operational concern, it possesses far-reaching implications in terms of security. The downtime metric underscores the vulnerability of a blockchain to various threats, including distributed denial-of-service (DDoS) attacks, consensus mechanism vulnerabilities, software bugs, and even internal misconfigurations.

The concept of employing downtime as a security metric aligns with the principle of availability, one of the key pillars of information security. Availability ensures that a blockchain network remains operational, accessible, and resilient against disruptions. When downtime occurs, it not only hampers the network's performance but also exposes it to potential exploitation by malicious actors seeking to take advantage of lapses in security protocols.

Measuring downtime involves tracking the duration and frequency of periods when the blockchain is inactive or inaccessible. This metric can be quantified in terms of hours, minutes, or even seconds, depending on the granularity required for analysis. A blockchain with prolonged or recurrent downtime might indicate underlying issues that compromise its security posture. Consequently, this metric serves as an indirect indicator of the blockchain's resilience against both accidental disruptions and intentional attacks.

4.1.2. Decentralization

Token Distribution Entropy

Token distribution entropy captures the diversity and evenness of the allocation of tokens across network participants. In essence, it quantifies the randomness and unpredictability of the distribution pattern. By scrutinizing the entropy of token distribution, blockchain stakeholders can gain a deeper understanding of the equitable distribution of influence and control within the network.

To apply token distribution entropy as a decentralization metric, one must analyze the spread of tokens among users, accounts, or addresses within the blockchain ecosystem. A higher entropy value signifies a more evenly dispersed token ownership landscape, suggesting that control and economic power are not concentrated in the hands of a select few entities. This resonates with the core principle of decentralization – a network where no single entity commands undue influence.

The formula to calculate the token distribution entropy is the following.

$$S = -\sum_{i=1}^{N} p_i \times log_2(p_i) \quad where\ p_i = \frac{x_i}{\sum_{i=1}^{N} x_i}$$

Based on the aforementioned formula, an S value of 0 signifies minimal decentralization (or heightened centralization), whereas an S value of log(N) signifies maximal decentralization (or minimal centralization).

Number of Validators

The number of validators refers to the count of independent entities responsible for validating transactions and maintaining the consensus mechanism within a blockchain network. This metric captures the diversity of participants contributing to the network's operation, reflecting the distribution of control and influence among stakeholders. Analyzing the number of validators offers valuable insights into the robustness of the network's decentralization architecture.

Applying the number of validators as a decentralization metric entails assessing the participation and representation of various entities within the network. A higher number of validators generally indicates a more decentralized ecosystem, as control is distributed among a larger pool of participants. This diversity of validators minimizes the risk of concentration of influence, promoting a healthier and more resilient network.

4.1.3. Scalability

Transactions Per Second

TPS quantifies the number of transactions a blockchain network can process within a single second. This metric is a direct reflection of the network's capacity to handle a larger volume of transactions in a timely manner. Evaluating TPS as a scalability metric provides invaluable information about the network's readiness to support real-world applications that demand rapid transaction confirmation.

Applying TPS as a scalability metric entails assessing the blockchain's responsiveness and performance under varying transaction loads. A higher TPS value suggests that the network possesses the infrastructure and protocols to swiftly process a substantial number of transactions without compromising its efficiency. This is particularly crucial for blockchain platforms seeking to accommodate applications ranging from financial transactions to supply chain management.

Time to Finality

Time to Finality refers to the duration it takes for a transaction to be fully confirmed and considered irreversible on the blockchain. This metric offers a clear measurement of how quickly a transaction achieves consensus and becomes an immutable part of the blockchain's history. Evaluating Time to Finality as a scalability metric is vital for assessing the network's responsiveness in processing transactions and maintaining an efficient user experience.

Applying Time to Finality as a scalability metric involves assessing the time it takes for a transaction to go through the consensus process and receive sufficient confirmations to be considered finalized. A shorter Time to Finality implies that transactions can be confirmed more rapidly, enabling the blockchain to handle a higher throughput of transactions. This becomes particularly significant for blockchain

applications that require real-time confirmation, such as payment systems or supply chain tracking.

4.2. Metrics Values per Blockchain

Now that we have a clear understanding of the variables involved, we can move forward with a detailed analysis of each technology. It's important to note that a solid grasp of these variables is essential for drawing accurate insights. With a good understanding of these variables in mind, we are ready to thoroughly assess quantitatively each of the technologies in question.

As of the current date, August 23rd, 2023, the specific values for each variable are provided in Figure 12, obtained through the corresponding official webpages and other resources (Ethereum, s.f.) (Solana Beach, s.f.) (Avalanche, s.f.) (Circle, 2023). This visual representation serves as a useful reference point, ensuring our analysis is grounded in the most recent and pertinent data. By using these current variable values as a foundation, we can confidently begin our assessment, which will be both well-informed and rigorous. Within this context, we will now proceed to carefully examine the distinct merits and attributes associated with each technology on our agenda.

Metric	Ethereum	Solana	Avalanche
Nakamoto Coefficient	379886	30	0
Token Distribution Entropy	19.5	8.883481435	0
Number of Validators	746265	1964	1354
TPS	29.33	4501	179
Time to Finality [s]	180	0.4	2

Figure 12. Metrics Values per Blockchain

Upon a thorough examination of the provided dataset, a conspicuous observation emerges: Avalanche notably lacks both the Nakamoto Coefficient and Token Distribution Entropy. This intriguing absence can be attributed to the distinctive nature of Avalanche's consensus algorithm, wherein its voting mechanism remains detached from token dependencies. Consequently, the utilization of these particular metrics within the framework of Avalanche's analysis becomes impractical.

Furthermore, it is noteworthy that the metric of Downtime finds no representation within Figure 12. This omission is grounded in the understanding that Downtime is a multifaceted concept, necessitating an intricate analysis beyond a mere numerical depiction. Downtime, as a measure, extends its roots into a plethora of causative factors, encompassing a spectrum that goes beyond potential attacks. Therefore, its absence from the visual representation is a result of its inherent complexity, demanding a dedicated analysis in order to capture its diverse manifestations comprehensively.

Nakamoto Coefficient

Starting with the Nakamoto Coefficient, it is evident that Ethereum requires a notably higher number of validators to reach the critical 51% control threshold. This observation is in line with the analysis that highlighted the resource-intensive nature of being a Solana validator. The demanding resource requirements of the Solana ecosystem inherently contribute to the practicalities of achieving validation within its network.

Moreover, an important aspect to consider is the temporal dimension that sets Ethereum apart from Solana. Ethereum's longer presence in the blockchain arena has allowed it to cultivate a more extensive and diversified community. The extended timeframe has provided Ethereum with the opportunity to nurture a broad user base, a factor that naturally contributes to a more distributed decision-making process, ultimately reflecting in its higher Nakamoto Coefficient.

Furthermore, Ethereum's historical resilience against challenges, forks, and adversities has solidified its position as a reliable and secure platform. This stability fosters ongoing participation and engagement from a diverse range of stakeholders, which, in turn, contributes to a robust Nakamoto Coefficient.

Adding another layer to the analysis, Ethereum's global recognition and early adoption have played a pivotal role in shaping its high Nakamoto Coefficient. Being a trailblazer in the decentralized technology landscape has afforded Ethereum an extensive legacy. Its influence on the blockchain domain has led to widespread acknowledgment, drawing enthusiasts and experts alike to contribute to its ecosystem.

Lastly, it's worth noting that Solana's threshold for network control stands at 33%, in contrast to the conventional 51%, which inherently contributes to a comparatively lower Nakamoto Coefficient.

Token Distribution Entropy

Revisiting the analytical landscape, an interesting facet emerges as we delve into the metrics at hand. It's crucial to acknowledge that Avalanche's non-participation in this particular metric adds a layer of distinctiveness to the analysis. In alignment with the established trend highlighted by the Nakamoto Coefficient, Ethereum once again takes the lead, surpassing Solana. Notably, Ethereum's distribution profile stands out prominently, nearing the threshold of maximum distribution.

Conversely, Solana positions itself as a contender with a notable distribution value, albeit distinct from Ethereum. This value is calculated to be close to the peak of decentralization, registering at 10.93. This numeric representation, derived from the logarithm base 2 of the validator count, provides an insightful assessment of decentralization levels.

On the Solana front, the proximity of its distribution value to the theoretical maximum suggests an interesting narrative. While the raw count of validators might

not be extensive, the network's architecture and consensus mechanisms promote a relatively higher level of decentralization per validator. This architectural choice encourages a distributed decision-making process, contributing to the observed distribution value.

Number of Validators

When we transition our focus to Solana and Ethereum, a striking parallel unfolds, echoing the analytical patterns witnessed in the Nakamoto Coefficient assessment. Nevertheless, the advent of this analysis ushers in the prominent inclusion of Avalanche, rendering the discourse multifaceted. Avalanche, in its distinctiveness, steps forward as a participant, albeit with a notable distinction: it holds the record for the lowest number of validators in comparison to Solana and, notably, Ethereum. This disparity becomes a noteworthy point of discussion, which prompts a deeper exploration into the intricacies that underlie this unique facet.

Avalanche's validator count, situated at the nadir of this comparison, raises pertinent questions about the reasons behind this occurrence. To unravel this narrative, it's crucial to delve into the core attributes that define Avalanche's validator ecosystem. One pivotal aspect lies in the exclusivity of Avalanche's validator Primary Network from which the data is extracted. With a limited number of positions available, the selection process is designed to ensure optimal network performance and security. While this selectivity underscores the network's robustness, it could potentially contribute to the observed disparity in validator count.

A noteworthy nuance emerges in the form of the variability in validator numbers across different subnets within Avalanche. This inherent fluctuation is attributed to the unique characteristics and demands of each subnet. As a result, a comprehensive analysis of the validator count necessitates a holistic consideration of the broader context, encompassing the distinct subnets' contributions to the overarching Avalanche ecosystem.

TPS

This juncture marks a pivotal shift in our analysis, especially when contrasting Solana and Ethereum. Here, the discrepancy between the two platforms becomes as pronounced as the variance in their respective validator counts. However, it is worth noting a particularly intriguing metric: that which pertains to Avalanche. Despite its consensus mechanism addressing scalability concerns, Avalanche remains in the shadow of Solana, revealing an area where it has considerable room for improvement.

Time to Finality

Parallel to the TPS analysis, the outcomes presented here closely mirror those observed in the TPS assessment. These results accentuate the evident scalability limitations in Ethereum, stemming from its block-based consensus mechanism, which stands in contrast to the transaction-based approach adopted by Solana and Avalanche. Moreover, within the context of Avalanche, the incorporation of multiple rounds of voting inadvertently results in the squandering of valuable time, contributing to its suboptimal performance.

Downtime

While lacking a concrete quantification for this metric, its significance in portraying network security necessitates an examination of each blockchain in this context. As of the present day, Ethereum has remained uninterrupted, showcasing its robust security posture. In contrast, Solana has encountered instances of downtime on 10 separate occasions, a noteworthy concern given its relatively brief existence, signaling vulnerabilities in this domain (Mitchelhill, 2023). On the other hand, Avalanche stands out for never having experienced a network downtime, underscoring its strength, particularly when considering its comparatively younger lifespan.

4.3. Conclusions Regarding the Blockchain Trilemma

Drawing from an in-depth analysis of the technological facets, along with the subsequent practical implications elucidated by the aforementioned metrics, we can arrive at insightful conclusions concerning how each blockchain addresses the complexities of the Blockchain Trilemma.

- Ethereum: While it distinctly outshines Solana and Avalanche in terms of Security and Decentralization, its shortcomings in Scalability and procedural adaptability impede the seamless integration of modern Decentralized Applications. The challenges it faces in achieving high transaction throughput while maintaining network efficiency can potentially hinder its efficacy in accommodating the demands of contemporary blockchain applications.

- Solana: Standing as a clear frontrunner in Scalability, Solana's commendable performance is juxtaposed with marked concerns related to Decentralization and Security. The susceptibility to Downtime, as evidenced by the metric discussed, casts a shadow over its security profile. Without addressing these vulnerabilities, only through rigorous testing and the passage of time can Solana's standing remain unimpaired. In the realm of Decentralization, despite the apparent low validator count, a promising horizon emerges. The dynamic nature of blockchain ecosystems can lead to an increase in validators as participation becomes more accessible, bolstering Solana's distribution, as indicated by Token Distribution Entropy.

- Avalanche: Distinguished by its departure from the traditional Blockchain Trilemma paradigm, Avalanche showcases an innovative path that prioritizes flexibility. Operating outside the bounds of conventional categories, Avalanche strikes a balance between Scalability, Security, and Decentralization, aligning its Primary Network with optimal performance in these domains. This network serves as the bedrock, demonstrating robust Scalability, Security, and

Decentralization. What further sets Avalanche apart is its visionary approach to subnets. By offering the capacity for users to establish their own blockchains within these subnets, Avalanche manifests unparalleled flexibility. This adaptability empowers the platform to allocate resources in alignment with specific blockchain requirements, marking it as a pioneer in customization and optimization.

In summation, the analysis yields intriguing insights: Solana is well poised to potentially resolve the Blockchain Trilemma through its notable Scalability attributes. On the other hand, Avalanche has deftly evaded the constraints of the Trilemma by ensuring the seamless fusion of Scalability, Security, and Decentralization within its Primary Network. Additionally, the ingenuity of subnets positions Avalanche as an optimized and versatile blockchain solution, augmenting its potential for industry leadership. Thus, considering the comprehensive panorama, Avalanche emerges as a formidable contender, boasting higher prospects to ascend as the preeminent Blockchain Technology.

Chapter 7: Practical Takeaway

Beyond the theoretical explorations meticulously presented in the preceding chapters, this project extends its impact to the practical realm, rendering it an invaluable asset for GSR. Within the dynamic landscape of the blockchain ecosystem, characterized by continuous evolution, this endeavor offers a tangible takeaway that holds profound significance. Specifically, the project introduces a tool designed to monitor and track the metrics delineated in Chapter 6 over time, thus furnishing a continuous feedback mechanism regarding the progression of each blockchain in tackling the intricate challenges posed by the Blockchain Trilemma.

To achieve this ambitious goal, a two-fold approach has been undertaken, involving the development of two distinct Python files. These files encapsulate the essence of a comprehensive data-tracking solution, affording GSR the means to gather insights and assess the ever-evolving state of various blockchains.

The first Python file is tailored to the extraction and compilation of data from diverse resources. This intricate process is meticulously orchestrated to ensure the accurate collection of relevant metrics. The collated data is then judiciously saved, laying the foundation for a meticulously curated database that evolves over time. This database becomes a repository of valuable insights, affording GSR the ability to gauge the trajectory of each blockchain's performance in addressing the Blockchain Trilemma.

Complementing the data extraction and compilation endeavor, the second Python file takes the form of a dashboard. This sophisticated dashboard is ingeniously designed to provide an easily accessible and analyzable interface for the amassed data. The dashboard serves as a visual conduit, rendering complex information comprehensible at a glance. Its user-friendly design empowers GSR to delve into the

nuances of the tracked metrics, fostering informed decision-making and insights generation.

In the subsequent chapter, our exploration will delve deep into the intricacies of both these Python files, unraveling the technical underpinnings that bring this visionary project to life. Through this pragmatic lens, the project evolves from theoretical discourse to practical utility, forging a potent synergy between theoretical insight and real-world application. Both codes will be displayed in the Annex.

7.1. get_data.py

As previously highlighted, the first Python file operates in a two-fold manner: it procures data from a myriad of diverse resources and subsequently organizes and stores this information within an Excel file. Consequently, this process involves a series of sequential steps, each of paramount importance.

Web Scraping

To initiate the data procurement endeavor, the pivotal technique of web scraping takes center stage. This procedure hinges on the utilization of specialized tools designed to extract data from various online platforms. In the context of blockchain data acquisition, each blockchain possesses its official platforms replete with an array of statistics. While real-time data collection is indeed attainable, obtaining historical data is often a formidable challenge. It is here that the significance of crafting a comprehensive dataset, exclusive to GSR, becomes evident.

To fulfill this ambitious data compilation, a potent ally in the form of the Selenium package is employed. This Python package constitutes a powerful automation tool, granting the capability to programmatically interact with web pages.

In essence, Selenium empowers the script to navigate, access, and retrieve information from websites as if an actual user were performing these actions.

The Selenium package's salient feature lies in its capacity to simulate human-like interaction with web pages. This includes actions such as clicking buttons, entering text, navigating through menus, and more. By accurately mimicking these user interactions, Selenium effectively bypasses potential obstacles posed by website structures designed to prevent automated data collection.

The emulation of human behavior through the utilization of the Selenium package bears particular significance, especially in scenarios where the objective is to extract intricate data nuances that replicate the token distribution across various validators. A common scenario entails the token distribution data being presented in tabular form, often with a limited number of entries viewable initially. To access additional entries, a requisite action, such as clicking a button, is needed to unveil further information.

To shed light on the technical intricacies, the process unfolds as follows:

- **Initializing the Selenium Driver:** This pivotal initial step involves launching the Selenium driver, which acts as a virtual browser to navigate and interact with web content. This initiation creates a seamless connection between the script and the target webpage, paving the way for subsequent interactions.

- **Webpage Exploration and Element Retrieval:** Within the virtual browsing environment, the script systematically scans through the webpage's structure. Guided by the objective of retrieving specific elements, it seeks out the relevant tokens of data, often employing techniques such as locating the element using its Full X-Path. The emulation of human-like browsing is pivotal here, enabling the script to seamlessly interact with the webpage elements.

- **Execution of Interaction:** Once the target element is identified, the Selenium script effectively simulates the human action necessary to retrieve the desired

data. This could involve clicking buttons, selecting dropdowns, or performing any action that triggers the display of additional data entries.

- **Closing the Driver:** Concluding the interaction, the Selenium driver is gracefully closed, effectively terminating the virtual browsing session. This phase is integral in maintaining system resources and ensuring a controlled conclusion to the data extraction process.

Data Storage

With the successful extraction of data accomplished, the subsequent phase seamlessly transitions into data storage, underpinned by a preliminary but crucial preparatory step: data cleaning. This preparatory process becomes pivotal as the data harvested through Web Scraping techniques typically manifests in text form, often accompanied by contextual elements that include more than just the numerical values sought after. A common occurrence involves instances such as '28,389 ETH', where the relevant numerical value is coupled with extraneous textual descriptors.

In light of this, the process of data storage necessitates the integration of effective data-cleaning methodologies. The ultimate objective is to refine the extracted data, stripping away non-desirable elements and rendering it amenable for seamless conversion into the intended data types.

An example would be instances such as '28,389 ETH', where data cleaning routines target the removal of extraneous characters, including commas and currency symbols (such as 'ETH'). This action streamlines the data, leaving behind only the numeric component which can then be converted into the corresponding data type, being in this case, an integer.

Transitioning into the storage phase, the initial pivotal consideration revolves around the determination of the optimal storage destination. At this juncture, a

strategic choice must be made regarding the repository that will house the meticulously acquired and refined data.

The underpinning rationale for this data storage decision is tied to the intended execution frequency of the code. It is meticulously designed to be executed daily. Operating within such a narrow temporal interval, changes within the metrics may not exhibit significant variations, rendering a shorter interval redundant. Consequently, this daily execution rhythm culminates in the accumulation of a mere 365 new data rows each year. Given this relatively modest dataset size, leveraging tools such as Excel is a viable and judicious approach.

In essence, the decision hinges on practicality and efficiency. The Excel platform, known for its user-friendly interface and adeptness at managing smaller datasets, aligns harmoniously with the limited data volume anticipated within the context of this project. This strategic alignment streamlines both data storage and subsequent analysis, rendering the decision to harness Excel as the storage medium not only practical but also astute in optimizing resource allocation.

As a result, the Excel workbook has been thoughtfully compartmentalized into four distinct Sheets, each serving a specific purpose:

- **Sheets 'Ethereum', 'Solana', and 'Avalanche':** These sheets are individually designated to house the time series data for each respective technology. Each sheet meticulously captures variables that exhibit temporal variations. Specifically, these variables include the dynamic 'Number of Validators' and 'TPS'. Each sheet boasts a streamlined structure comprising three columns, where the first two columns correspond to the mentioned variables, and the third column is dedicated to recording the precise date of data extraction. This organizational architecture is visually exemplified in Figure 13. Notably, in the case of Solana, the metrics are expanded to incorporate two additional variables: the Nakamoto Coefficient and the Token Distribution Entropy, both of which exhibit temporal variations.

Date	Number of Validators	TPS	Nakamoto Coefficient	Entropy
22/08/2023	1962	4865	30	8.883473863
23/08/2023	1964	4501	30	8.883481435
24/08/2023	1964	5064	30	8.883481376

Figure 13. Solana Data Stored

- **Sheet 'Aggregated':** This sheet, illustrated in Figure 14, emerges as a pivotal component, designed to encapsulate a comprehensive aggregation of all studied metrics across the various blockchains. The strategic rationale underlying this sheet lies in the convenience of facilitating holistic comparisons. Here, an all-encompassing table is created, incorporating all metrics studied, providing a readily accessible means of comparing these metrics across the diverse blockchain platforms. The structure of this sheet is a masterstroke, with the majority of metrics remaining static, barring those subject to temporal variation. For the dynamically evolving metrics, such as 'Number of Validators' and 'TPS', the Excel automatically populates the most recent available data, ensuring the table remains consistently updated with the latest insights. In essence, this table harmonizes the dual characteristics of being both static and dynamic, enhancing its utility as a reference tool.

Metric	Ethereum	Solana	Avalanche
Nakamoto	379886	30	0
Entropy	19.5	8.883481376	0
Number of Validators	748559	1964	1354
TPS	29.33	5064	179
Time to Finality	180	0.4	2

Figure 14. Aggregated Data Stored

7.2. dashboard.py

Given the perpetual influx of updated data, the imperative of a dynamic dashboard emerges, one that possesses the agility to extract this evolving data and seamlessly update its graphical representations. Beyond mere data presentation, the dashboard assumes the pivotal role of enabling effective comparisons between different metrics and blockchain technologies. To empower users with the ability to control the displayed information, granting them the discretion to toggle between displayed and hidden metrics, an interactive element is a prerequisite.

In light of these multifaceted requirements, Python rises once again as the platform of choice. Python's inherent versatility, coupled with its rich ecosystem of libraries and tools, uniquely positions it to fulfill the dynamic and interactive needs of the envisioned dashboard. Python's capability to harness real-time data, process it, and then translate it into dynamic visualizations aligns harmoniously with the project's overarching goals.

Drawing from the available dataset, a constellation of four pivotal graphs harmoniously coalesce to offer a comprehensive panorama of how diverse blockchains fare when confronted with the intricacies of the Blockchain Trilemma.

- **Radar Plot:** Illustrated in Figure 15, the inaugural graph materializes as a Radar Plot, adeptly capturing the performance of each blockchain across the array of metrics under consideration. This visual representation leverages the aggregated data stored in the 'Aggregated' table of the Excel file. The Radar Plot ingeniously condenses complex data into an intuitive visual format, enhancing comprehension and comparison.

  Notably, this Radar Plot employs scaling mechanisms to standardize the data's range to [0, 1]. Within this normalized range, the technology attaining the least favorable metric outcome receives a value of 0, while the technology excelling in the same metric attains a value of 1. Technologies achieving intermediate results are assigned values that interpolate between the extremities, predicated

upon their proximity to the maximal and minimal values. This scaling harmonizes diverse metrics and technologies, facilitating an equitable and cohesive evaluation.

To empower user engagement and tailor insights, the dashboard facilitates interaction. Users are empowered to toggle checkboxes, selecting specific technologies for display. This intuitive feature amplifies the user's agency, allowing them to customize their view of the Radar Plot based on their analytical inclinations. Within Figure 15, Ethereum and Solana are both elected for display, underscoring the versatility of the interactive dashboard in catering to user preferences.

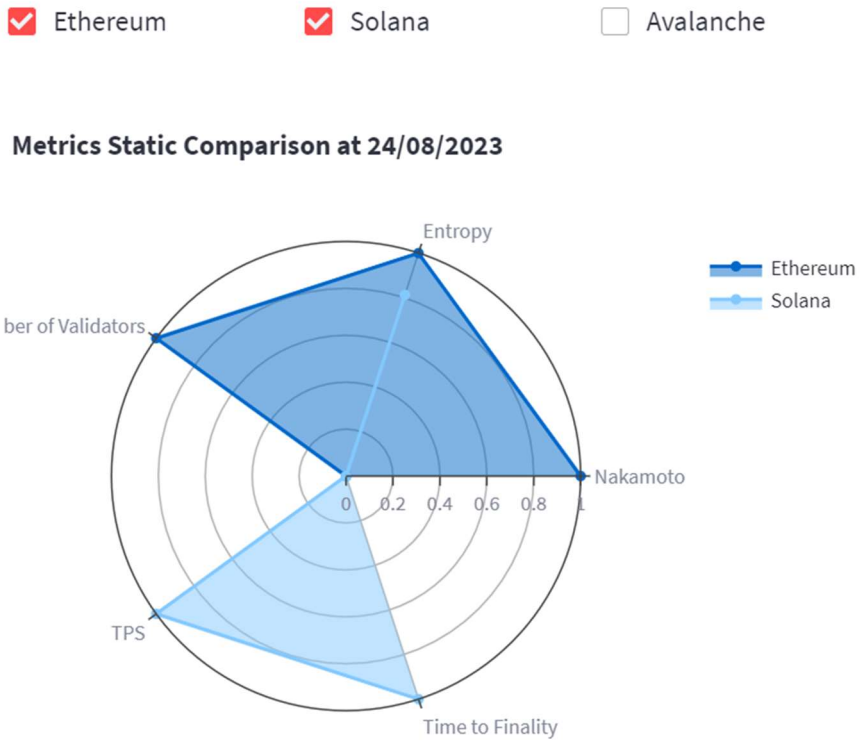

Figure 15. Radar Plot

- **Aggregated Table:** Evident in Figure 16, the Aggregated Table assumes a prominent position within the visual repertoire. Conceived as an unabridged replication of the 'Aggregated Table', its purpose harmonizes with that of the Radar Plot, but with a distinct intent. This table mirrors the 'Aggregated Table'

in its entirety, meticulously capturing the unadulterated and up-to-date metric values without resorting to scaling.

The rationale for incorporating the Aggregated Table lies in its capacity to complement the insights gleaned from the Radar Plot. Unlike the scaled values of the Radar Plot, the Aggregated Table showcases the precise, real-world metric values, preserving their veracity without any normalization. This facet augments the dashboard's capacity to cater to a spectrum of analytical preferences, ensuring that users can choose to engage with unscaled, absolute data when required.

	Metric	Ethereum	Solana	Avalanche
0	Nakamoto	379886.000000	30.000000	0.000000
1	Entropy	19.500000	8.883481	0.000000
2	Number of Validators	748559.000000	1964.000000	1354.000000
3	TPS	29.330000	5064.000000	179.000000
4	Time to Finality	180.000000	0.400000	2.000000

Figure 16. Aggregated Table

- **Number of Validators Time Series:** Presented in Figure 17, the ensuing visualization is dedicated to spotlighting the temporal flux in the number of validators across distinct blockchains. This chart is thoughtfully designed to encapsulate the dynamic changes that unfurl over time, specifically procuring data from the 'Ethereum', 'Solana', and 'Avalanche' sheets within the Excel file. Parallel to the Radar Plot, this visualization invites user interaction through checkboxes. Users are empowered to select and display the blockchain of their choosing, fostering a tailor-made analytical journey. By toggling checkboxes, users can summon a focused representation of the number of validators for their preferred blockchain(s).

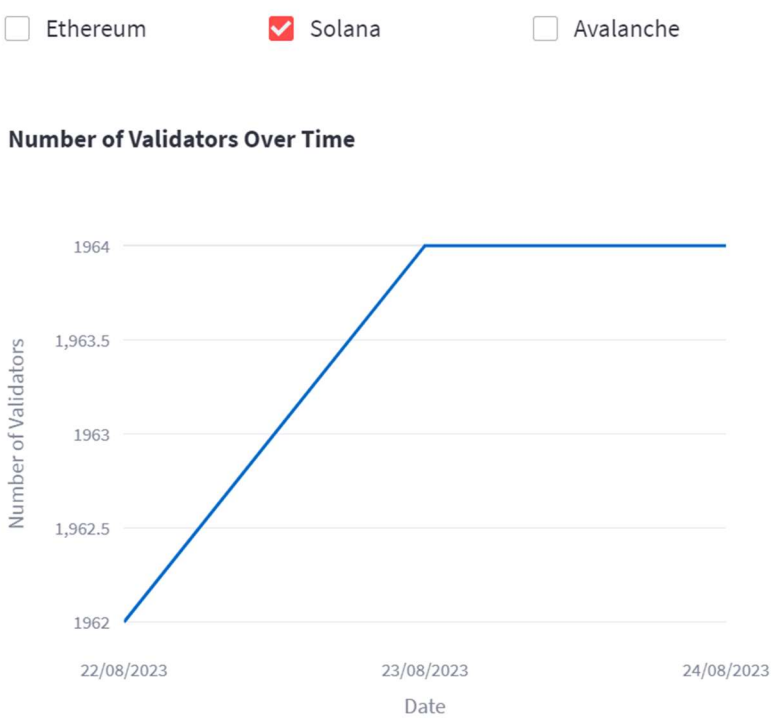

Figure 17. Number of Validators Time Series

- **TPS Time Series:** Unveiled within Figure 18, the culminating visualization mirrors the structure of the Number of Validators Time Series. However, in this iteration, the spotlight pivots toward tracking the Time-Per-Second (TPS) metrics. This dynamic chart illuminates the temporal shifts in TPS across blockchain platforms. Similar to its counterpart, this visualization orchestrates an intricate dance with user engagement, allowing for the selection of preferred blockchains through intuitive checkboxes.

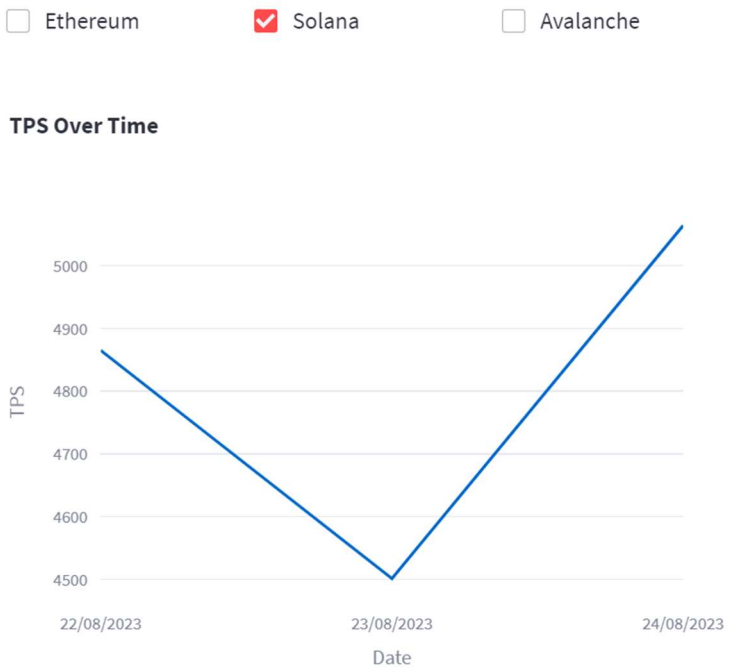

Figure 18. TPS Time Series

References

Antonopoulos, A. M. (2014). *Mastering Bitcoin.* O'Reilly Media, Inc.

Avalanche. (n.d.). *Avalanche Dashboard*. Retrieved from Avalanche: https://stats.avax.network/dashboard/overview/

Bloomenthal, A. (2021, August 31). Retrieved from Investopedia: https://www.investopedia.com/terms/m/marketmaker.asp

Buttolph, S. (2022). Snowman Consensus. *Avalanche Summit 2022.* Monastery.

Bybit. (2022, July 18). *Nakamoto Coefficient: An Accurate Indicator for Blockchain Decentralization?* Retrieved from Bybit: https://learn.bybit.com/blockchain/nakamoto-coefficient-decentralization/

Circle. (2023). *Developers Circle: Blockchain Confirmations*. Retrieved from Circle: https://developers.circle.com/developer/docs/confirmations

Cryptopedia. (2022, March 17). *What was the DAO?* Retrieved from Cryptopedia: https://www.gemini.com/cryptopedia/the-dao-hack-makerdao

Ethereum. (2023, July 26). *Proof-Of-Stake (PoS)*. Retrieved from Ethereum: https://ethereum.org/en/developers/docs/consensus-mechanisms/pos/#:~:text=Ethereum%20switched%20on%20its%20proof,proof%2Dof%2Dwork%20architecture

Ethereum. (n.d.). *BeaconScan*. Retrieved from BeaconScan: https://beaconscan.com/

Frankenfield, J. (2021, July 26). *Merkle Tree in Blockchain: What it is and How it Works*. Retrieved from Investopedia: https://www.investopedia.com/terms/m/merkle-tree.asp

Frankenfield, J. (2023, May 31). *What Are Smart Contracts on the Blockchain and How They Work*. Retrieved from Investopedia: https://www.investopedia.com/terms/s/smart-contracts.asp

Frankenfield, J. (2023, May 28). *What Is a hash? Hash Functions and Cryptocurrency Mining*. Retrieved from Investopedia: https://www.investopedia.com/terms/h/hash.asp

Kasireddy, P. (2017, September 13). *How does Ethereum work, anyway?* Retrieved from Preethi Kasireddy: https://www.preethikasireddy.com/post/how-does-ethereum-work-anyway

Mitchelhill, T. (2023, February 28). *The Solana Network Has Gone Down 10 Times. Will It Recover?* Retrieved from The Chainsaw: https://thechainsaw.com/defi/altcoins/solana-down-network-freeze-ten-times-recover/#:~:text=How%20many%20times%20has%20Solana,a%20total%20of%20ten%20times

Sekniqi, K., Laine, D., Buttolph, S., & Gün Sirer, E. (2020). *Avalanche Platform.*

Solana. (2023). *Solana Documentation*. Retrieved from Solana: https://docs.solana.com/es/

Solana Beach. (n.d.). *Solana Beach*. Retrieved from Solana Beach: https://solanabeach.io

Solana Floor Content. (2023, April 19). *Turbine: Solana's Revolutionary Block Propagation Protocol*. Retrieved from Solana Floor Content: https://ghost.step.finance/turbine-solanas-revolutionary-block-propagation-protocol/

Tech Target. (2021, June). *Data Security and Privacy: Public Key*. Retrieved from Tech Target: https://www.techtarget.com/searchsecurity/definition/public-key#:~:text=A%20public%20key%20can%20be,is%20sent%20over%20the%20internet

Annex I: Code

get_data.py

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import numpy as np
import pandas as pd
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.common.keys import Keys
import math
import time
import re
from datetime import datetime


# Get Date
today = datetime.now()
formatted_date = today.strftime("%d/%m/%Y")

################################## ETHEREUM

##### Number of Validators
# Initialize the Selenium WebDriver
driver = webdriver.Chrome()

# URL of the website to scrape
url = "https://beaconscan.com/validators#active"
driver.get(url)
driver.implicitly_wait(30)

xpath = "/html/body/main/section[1]/div/ul/li[2]/a"
element = driver.find_element(By.XPATH, xpath)
match = re.search(r'\((\d+)\)', element.text)
number_validators = int(match.group(1))
print(number_validators)

# Close the Driver
driver.quit()


##### Transactions Per Second
# Initialize the Selenium WebDriver
driver = webdriver.Chrome()

url =
"https://ethtps.info/#:~:text=Ethereum%20is%20doing%2029.33%20TPS&text=Drag%20the%20slider%20above%20to,compare%20the%20historical%20TPS%20distribution."
driver.get(url)
```

```
driver.implicitly_wait(10)

xpath = "/html/body/div/div[1]/center[2]/h4"
element = driver.find_element(By.XPATH, xpath)
match = re.search(r'([\d.]+)', element.text)
tps = float(match.group(1))
print(tps)

# Close the Driver
driver.quit()


# Read existing data from Excel file (specific sheet)
excel_file = "Data.xlsx"
sheet_number = 0
df = pd.read_excel(excel_file, sheet_name=sheet_number)

# Create a new row as a dictionary
new_row = {
    'Date': formatted_date,
    'Number of Validators': int(number_validators),
    'TPS': float(tps)
}

# Add the new row to the DataFrame
df = df.append(new_row, ignore_index=True)

# Write the updated data to the same sheet in Excel file
with pd.ExcelWriter(excel_file, engine='openpyxl', mode='a',
if_sheet_exists='replace') as writer:
    df.to_excel(writer, sheet_name='Ethereum', index=False, header=True)

################################### SOLANA

##### Transactions Per Second

# Initialize the Selenium WebDriver
driver = webdriver.Chrome()

# URL of the website to scrape
url = "https://solanabeach.io"
driver.get(url)

# Obtain Element
xpath =
"/html/body/div[1]/div[3]/div/div/div/div[6]/div/div[1]/div/div[1]/div/div/p/span"
timeout = 10
element = WebDriverWait(driver, timeout).until(
    EC.visibility_of_element_located((By.XPATH, xpath))
)
tps = element.text
tps = tps.replace(',', '')
print(tps)

# Close the Driver
driver.quit()


##### Number of Validators
```

```
# Initialize the Selenium WebDriver
driver = webdriver.Chrome()

# URL of the website to scrape
url = "https://solanabeach.io/validators"
driver.get(url)
driver.implicitly_wait(10)

# Obtain Element
selector_path = "#app > div.sc-dfVpRl.dEiGAe > div > div > div > div.sc-
kPVwWT.kRENfF.card.undefined > div > div > div:nth-child(1) > div > div > div > div
> p > span"
element = driver.find_element(By.CSS_SELECTOR, selector_path)
number_validators = element.text
print(number_validators)


##### Nakamoto Coefficient and Entropy

# Scroll down all the page
actions = ActionChains(driver)
scroll_increment = 3000  # Adjust as needed
scroll_count = 0

while scroll_count < 350:  # Perform scrolling a few times for demonstration
    actions.send_keys(Keys.PAGE_DOWN).perform()
    scroll_count += 1

# Obtain Element
stake = np.array([])
number_validators = number_validators.replace(",", "")
for i in range(int(number_validators)+1):
    if i != 30:
        xpath =
"/html/body/div[1]/div[3]/div/div/div/div[4]/div/table/tbody/tr[{}]/td[2]/span[1]".
format(i+1)
        element = driver.find_element(By.XPATH, xpath)
        stake = np.append(stake, element.text)


stake = np.array([int(s.replace(',', '')) for s in stake])
total_amount = np.sum(stake)

# Entropy
entropy = 0
for i in range(len(stake)):
    individual_stake = stake[i]/total_amount
    entropy = entropy + individual_stake*math.log2(individual_stake)
entropy = -entropy
print(entropy)

# Nakamoto Coefficient
cumulative_stake = 0
i = 0
while i<len(stake):
    individual_stake = stake[i]/total_amount
    cumulative_stake = cumulative_stake + individual_stake
    i = i+1
```

```
    if cumulative_stake>1/3:
        break

nakamoto = i
print(nakamoto)

# Close the Driver
driver.quit()


# Read existing data from Excel file (specific sheet)
excel_file = "Data.xlsx"
sheet_number = 1
df = pd.read_excel(excel_file, sheet_name=sheet_number)

# Create a new row as a dictionary
new_row = {
    'Date': formatted_date,
    'Number of Validators': int(number_validators),
    'TPS': float(tps),
    'Nakamoto Coefficient': int(nakamoto),
    'Entropy': float(entropy)
}

# Add the new row to the DataFrame
df = df.append(new_row, ignore_index=True)

# Write the updated data to the same sheet in Excel file
with pd.ExcelWriter(excel_file, engine='openpyxl', mode='a',
if_sheet_exists='replace') as writer:
    df.to_excel(writer, sheet_name='Solana', index=False, header=True)




################################## AVALANCHE

##### Number of Validators

# Initialize the Selenium WebDriver
driver = webdriver.Chrome()

# URL of the website to scrape
url = "https://stats.avax.network/dashboard/network-status/"
driver.get(url)
driver.implicitly_wait(30)

# Avalanche page is an iFrame
iframe_xpath = "/html/body/div/main/div/iframe"
iframe_element = driver.find_element(By.XPATH, iframe_xpath)

# Switch to the iframe context
driver.switch_to.frame(iframe_element)

# Search inside the iFrame
number_validators =
driver.find_element(By.XPATH,"/html/body/div[1]/div/div/main/div/div/div/div[2]/div
/div/div/div/div[2]/span/h1").text
number_validators = number_validators.replace(',', '')
```

```
print(number_validators)


##### TPS

# URL of the website to scrape
url = "https://stats.avax.network/dashboard/overview/"
driver.get(url)
driver.implicitly_wait(10)

# Avalanche page is an iFrame
iframe_xpath = "/html/body/div/main/div/iframe"
iframe_element = driver.find_element(By.XPATH, iframe_xpath)

# Switch to the iframe context
driver.switch_to.frame(iframe_element)

# Search inside the iFrame
tps =
driver.find_element(By.XPATH,"/html/body/div[1]/div/div/main/div/div/div/div[2]/div
/div/div/div/div[2]/span/h1").text
print(tps)

# Close the Driver
driver.quit()


# Read existing data from Excel file (specific sheet)
excel_file = "Data.xlsx"
sheet_number = 2
df = pd.read_excel(excel_file, sheet_name=sheet_number)

# Create a new row as a dictionary
new_row = {
    'Date': formatted_date,
    'Number of Validators': int(number_validators),
    'TPS': float(tps),
}

# Add the new row to the DataFrame
df = df.append(new_row, ignore_index=True)

# Write the updated data to the same sheet in Excel file
with pd.ExcelWriter(excel_file, engine='openpyxl', mode='a',
if_sheet_exists='replace') as writer:
    df.to_excel(writer, sheet_name='Avalanche', index=False, header=True)
```

dashboard.py

```
# streamlit run dashboard.py

import streamlit as st
import pandas as pd
import plotly.express as px
import openpyxl # To read excel
import xlwings as xw # To retrieve calculated values
from sklearn.preprocessing import MinMaxScaler
import plotly.graph_objects as go
import math


################ Loading Data

# Load the Excel
excel_file = "Data.xlsx"
sheet_name = "Aggregated"
wb = xw.Book(excel_file) # Read the excel getting only the calculated values
ws = wb.sheets[sheet_name] # Filter to get only sheet 3

ws0 = wb.sheets["Ethereum"]
ws1 = wb.sheets["Solana"]
ws2 = wb.sheets["Avalanche"]

# Get the used range of the sheet
used_range = ws.used_range
used_range0 = ws0.used_range
used_range1 = ws1.used_range
used_range2 = ws2.used_range

# Convert the used range to a DataFrame
df = pd.DataFrame(used_range.value[1:], columns=used_range.value[0])
df = df.dropna(how="all")
df = df.iloc[:, :-1]

df0 = pd.DataFrame(used_range0.value[1:], columns=used_range0.value[0])
df0 = df0.dropna(how="all")

df1 = pd.DataFrame(used_range1.value[1:], columns=used_range1.value[0])
df1 = df1.dropna(how="all")

df2 = pd.DataFrame(used_range2.value[1:], columns=used_range2.value[0])
df2 = df2.dropna(how="all")

# Close the workbook
wb.close()

################ Creating Dashboard
st.set_page_config(layout="wide")

st.markdown('<h1 style="text-align: center;">Comparative Analysis of Blockchain
Metrics</h1><br><br><br>', unsafe_allow_html=True)

## Left section (1/2 width)
```

```
col1, col2, col3 = st.columns([5, 1, 5])


# Column 1
with col1:

    # Scale the dataset
    scaler = MinMaxScaler()
    df_scaled = df.copy()
    first = df_scaled.pop('Metric')
    for i in range(len(df_scaled)):
        if i == 1:
            number_validators = [df0['Number of Validators'].iloc[-1], df1['Number
of Validators'].iloc[-1], df2['Number of Validators'].iloc[-1]]
            scaled_values = [df_scaled.iloc[i,0]/math.log2(number_validators[0]),
df_scaled.iloc[i,1]/math.log2(number_validators[1]),
df_scaled.iloc[i,2]/math.log2(number_validators[2])]
            scaled_series = pd.Series(scaled_values, index=['Ethereum', 'Solana',
'Avalanche'])
        else:
            reshaped_series = df_scaled.iloc[i].values.reshape(-1, 1)
            scaled_values = scaler.fit_transform(reshaped_series)
            scaled_series = pd.Series(scaled_values.flatten(), index=['Ethereum',
'Solana', 'Avalanche'])
        if i == len(df_scaled)-1:
            scaled_values = 1 - scaled_values
            scaled_series = pd.Series(scaled_values.flatten(), index=['Ethereum',
'Solana', 'Avalanche'])
        df_scaled.loc[i] = scaled_series

    df_scaled.insert(0, 'Metric', first)

    # Create the radar plot
    # Create checkboxes for selecting columns
    checks = st.columns(3)
    with checks[0]:
        show_ethereum = st.checkbox('Ethereum', value=True)
    with checks[1]:
        show_solana = st.checkbox('Solana')
    with checks[2]:
        show_avalanche = st.checkbox('Avalanche')

    # Create a custom radar plot using Plotly
    fig = go.Figure()
    fig.update_layout(
        polar=dict(
            radialaxis=dict(
                visible=True,
                range=[0, 1]  # Adjust the range to match your data
            )
        ),
        showlegend=True,
        title='Metrics Static Comparison at {}'.format(df0['Date'].iloc[-1])
    )

    if show_ethereum:
        fig.add_trace(go.Scatterpolar(
            r=df_scaled['Ethereum'],
            theta=df_scaled['Metric'],
```

```
            fill='toself',
            name='Ethereum'
        ))

    if show_solana:
        fig.add_trace(go.Scatterpolar(
            r=df_scaled['Solana'],
            theta=df_scaled['Metric'],
            fill='toself',
            name='Solana'
        ))

    if show_avalanche:
        fig.add_trace(go.Scatterpolar(
            r=df_scaled['Avalanche'],
            theta=df_scaled['Metric'],
            fill='toself',
            name='Avalanche'
        ))


    col1.plotly_chart(fig, use_container_width=True)

    col1.table(df.style.hide_index())


# Column 3
with col3:

    # Create checkboxes for selecting datasets
    checks = st.columns(3)
    with checks[0]:
        show_ethereum = st.checkbox('Ethereum ', value=True)
    with checks[1]:
        show_solana = st.checkbox('Solana ')
    with checks[2]:
        show_avalanche = st.checkbox('Avalanche ')

    # Create a line chart using Plotly with independent traces for each selected
dataset
    data = []

    if show_ethereum:
        trace_ethereum = go.Scatter(x=df0['Date'], y=df0['Number of Validators'],
mode='lines', name='Ethereum')
        data.append(trace_ethereum)

    if show_solana:
        trace_solana = go.Scatter(x=df1['Date'], y=df1['Number of Validators'],
mode='lines', name='Solana')
        data.append(trace_solana)

    if show_avalanche:
        trace_avalanche = go.Scatter(x=df2['Date'], y=df2['Number of Validators'],
mode='lines', name='Avalanche')
        data.append(trace_avalanche)

    layout = go.Layout(title='Number of Validators Over Time', xaxis_title='Date',
yaxis_title='Number of Validators')
```

```
    fig = go.Figure(data=data, layout=layout)

    col3.plotly_chart(fig, use_container_width=True)

    # Create checkboxes for selecting datasets
    checks = st.columns(3)
    with checks[0]:
        show_ethereum = st.checkbox('Ethereum  ', value=True)
    with checks[1]:
        show_solana = st.checkbox('Solana  ')
    with checks[2]:
        show_avalanche = st.checkbox('Avalanche  ')

    # Create a line chart using Plotly with independent traces for each selected
dataset
    data = []

    if show_ethereum:
        trace_ethereum = go.Scatter(x=df0['Date'], y=df0['TPS'], mode='lines',
name='Ethereum')
        data.append(trace_ethereum)

    if show_solana:
        trace_solana = go.Scatter(x=df1['Date'], y=df1['TPS'], mode='lines',
name='Solana')
        data.append(trace_solana)

    if show_avalanche:
        trace_avalanche = go.Scatter(x=df2['Date'], y=df2['TPS'], mode='lines',
name='Avalanche')
        data.append(trace_avalanche)

    layout = go.Layout(title='TPS Over Time', xaxis_title='Date',
yaxis_title='TPS')
    fig = go.Figure(data=data, layout=layout)

    col3.plotly_chart(fig, use_container_width=True)
```

Annex II: Sustainable Development Goals

In the fast-evolving landscape of blockchain technology, the project assumes a pivotal role by conducting an in-depth evaluation of three prominent blockchain platforms—Ethereum, Solana, and Avalanche. This comprehensive analysis is not only a strategic endeavor but also aligns profoundly with the United Nations' Sustainable Development Goals (SDGs). Specifically, the project resonates with SDG 8: Decent Work and Economic Growth, and SDG 9: Industry, Innovation, and Infrastructure.

Project Alignment with SDG 8: Decent Work and Economic Growth

The project's central focus on evaluating three prominent blockchain technologies, namely Ethereum, Solana, and Avalanche, demonstrates a profound alignment with Sustainable Development Goal 8: Decent Work and Economic Growth. Through an extensive and meticulous analysis, the project endeavors to provide essential insights to a key partner, a crypto currency market maker. By equipping the market maker with these valuable insights, the project empowers them to make strategic resource allocation decisions and thereby engender well-informed choices within the market.

In effect, the project directly contributes to the optimization of market-making strategies, potentially yielding an enhanced level of efficiency in market operations. This heightened operational efficiency, spurred by the analytical insights delivered by the project, is a catalyst for economic growth within the spheres of blockchain

technology and financial technology at large. The underlying objective of stimulating economic activities and consequently generating novel job prospects is a testament to the project's synergy with the principles outlined in SDG 8.

Project Alignment with SDG 9: Industry, Innovation, and Infrastructure

The project's profound engagement with the intricate world of blockchain technologies inherently aligns with the core tenets of Sustainable Development Goal 9: Industry, Innovation, and Infrastructure. By meticulously scrutinizing the nuances of Ethereum, Solana, and Avalanche, the project propels innovation within the realm of blockchain technology through a comprehensive and balanced analysis.

The evaluation of these blockchains, both from theoretical and practical standpoints, provides a foundation for meaningful innovation by identifying their respective strengths and limitations. As the project endeavors to construct a real-time database and a visual dashboard that succinctly captures these findings, it cultivates a resource that stands to benefit stakeholders seeking a deeper understanding of these intricate systems. The project's commitment to enhancing comprehension of blockchain technologies is inherently aligned with SDG 9's pursuit of fostering resilient and robust industry practices.