

# FinTech

Examining the evolving world of finance technology, offering a thorough analysis of trends, innovations, and influences in both digital and traditional finance, from cryptocurrencies and blockchain to conventional systems.

- [The New Age of Value](#)
  - [Bitcoin Blockchain Wars and Its Evolution: Implications for Decentralisation, Economics, and Governance](#)
  - [The Bitcoin Reformation](#)
  - [AI Coins and Decentralized AI Infrastructure](#)
  - [MicroStrategy's \(MSTR\) Bitcoin Investment Strategy: A Comprehensive Overview](#)
  - [Bitcoin's 'Glitch in the Matrix' Has MicroStrategy Stuck in a While Loop](#)
  - [Bitcoin Knots](#)
  - [📊 Bitcoin Price Projections Based on Asset Class Market Caps](#)
- [Monero: Private and Decentralized Cryptocurrency](#)
  - [Monero: Privacy by Design – Origins, Technology, and Ongoing Evolution](#)
  - [Monero Monitoring & Troubleshooting Commands Reference](#)
  - [Monero Wallet Guide](#)
  - [Secure Remote Access to Monero Node Using Tailscale VPN](#)
  - [Monero Node Administration & Maintenance Guide](#)
  - [📝 Making Monero Port 18080 Firewall Rule Persistent on DietPi \(with ts-input Chain\)](#)
  - [📝 Why Earlier Attempts to Persist the iptables Rule for Port 18080 on DietPi Failed](#)
  - [📝 Creating and Managing Additional Accounts for Transaction Segmentation in Monero Wallet](#)

# The New Age of Value

The world of digital assets is rapidly evolving, driven by innovations like cryptocurrencies, NFTs, and decentralized platforms. These technologies are transforming financial systems, industries, and the concept of ownership. Through expert insights and case studies, this guide provides a clear understanding of the digital asset landscape.

# Bitcoin Blockchain Wars and Its Evolution: Implications for Decentralisation, Economics, and Governance

## The Bitcoin Blockchain War (2017): A Turning Point

### The Conflict:

As Bitcoin's popularity surged, its scalability was questioned. The network's 1 MB block size limit, introduced in 2010, restricted transaction throughput to 3-7 transactions per second, leading to delays and high fees during peak demand.

### Two factions emerged:

#### 1. **Block Size Increase Advocates:**

- Believed Bitcoin should scale on-chain by increasing the block size, enabling more transactions per block.
- Argued that larger blocks would reduce transaction fees and support Bitcoin's use as a peer-to-peer electronic cash system, consistent with Satoshi Nakamoto's original vision.
- Viewed concerns about centralization as overstated, asserting that technological advancements in storage and bandwidth would mitigate increased node costs.
- Criticized second-layer solutions like the Lightning Network as overly complex and potentially centralizing due to the reliance on custodial hubs.

#### 2. **SegWit and Lightning Supporters:**

- Proposed keeping the block size small to preserve Bitcoin's decentralization.
- Emphasized that larger blocks would increase the cost of running a full node, reducing the number of participants and risking centralization.
- Advocated for Segregated Witness (SegWit) to optimize block usage and enable second-layer solutions like the Lightning Network for off-chain scaling.
- Argued that a decentralized network was essential to Bitcoin's resistance to censorship and long-term viability.

## The Split:

The disagreement culminated in a hard fork in August 2017, resulting in two chains:

1. **Bitcoin (BTC):**

- Retained the 1 MB block size, adopted SegWit, and shifted toward a "store of value" narrative akin to digital gold.

2. **Bitcoin Cash (BCH):**

- Increased the block size to 8 MB (and later 32 MB), focusing on low-cost, high-speed transactions to maintain Bitcoin's usability for payments.

## Implications:

- By limiting block size, BTC ensured decentralization by keeping the requirements for running a full node accessible. However, this choice constrained Bitcoin's capacity as a global currency.
- BCH sought to enhance usability for everyday payments but sacrificed some decentralization, as larger blocks demand more resources, potentially centralizing the network over time.
- BTC's approach also facilitated second-layer scalability solutions like the Lightning Network, designed to handle microtransactions off-chain while maintaining the integrity of the base layer.

## The Bitcoin Cash Fork (2018): Further Fragmentation

### Dispute Over Direction:

Bitcoin Cash itself split into Bitcoin Cash (BCH) and Bitcoin SV (BSV) in November 2018.

- **Bitcoin SV (Satoshi's Vision):**

- Championed by Craig Wright and Calvin Ayre, this chain implemented massive block size increases (eventually 2 GB) to support high transaction volumes and data-heavy applications, asserting it adhered more closely to Satoshi Nakamoto's original vision.

## Implications:

- **On-Chain Scalability:**

- BSV's ability to handle large-scale transactions and applications demonstrated the potential of a high-capacity blockchain but raised concerns about network centralization, as fewer entities can afford to run nodes.

- **Fragmentation Risks:**

- Each fork diluted the community's resources and focus, potentially slowing adoption and development compared to a unified approach.

# The Evolution of Bitcoin's Role

## Original Vision vs. Reality:

Satoshi Nakamoto's white paper described Bitcoin as a decentralized, peer-to-peer electronic cash system. However, BTC's trajectory has focused on becoming digital gold—a store of value rather than a daily-use currency.

- High fees and limited scalability on the base layer shifted BTC's usability to long-term investment and large transactions, relying on Layer 2 solutions like the Lightning Network for smaller, faster payments.

## Economic Implications of the Shift:

- **Inability to Replace Fiat Currencies:**
  - With its current design, BTC cannot handle the transactional volume required to function as a global currency.
  - This leaves the fiat system intact, maintaining the monopoly of central banks and governments over monetary policy and capital controls.
- **Regulatory Leverage:**
  - By not directly competing with fiat for everyday transactions, Bitcoin avoids triggering aggressive regulation aimed at preserving state control over money. This strategic positioning could be deliberate or a by-product of internal disagreements.

## Centralisation Concerns:

While Bitcoin is decentralized in protocol, its ownership distribution and rising price challenge its founding ideals:

- **Ownership Concentration:**
  - Studies reveal that 2% of wallets hold over 90% of Bitcoin's supply. While some belong to exchanges holding BTC on behalf of users, the concentration among "whales" (large holders) raises concerns about market manipulation and influence.
- **Corporate and Government Accumulation:**
  - Corporations like MicroStrategy and countries like El Salvador hold significant amounts of Bitcoin. If governments or other centralized entities acquire substantial holdings, they could:
    - Influence the market price by coordinating buying or selling strategies.
    - Use Bitcoin holdings as a tool to control or manipulate economies.
    - Undermine Bitcoin's decentralization by concentrating ownership in fewer hands.
- **Impact on Retail Investors:**
  - With BTC prices exceeding \$90,000, owning a full Bitcoin is out of reach for most individuals, exacerbating economic inequalities and reinforcing its image as a "rich man's asset."

- Fractional ownership (satoshis) helps, but the psychological barrier of owning "just a fraction" may deter widespread adoption.

## Long-Term Risks:

- If a small number of entities control the majority of Bitcoin, they could limit its use as an alternative to fiat by:
  - Hoarding supply, reducing circulation.
  - Using their holdings to impose transaction restrictions or fees, undermining Bitcoin's decentralized ethos.

# A Hypothetical Alternative: Bitcoin as Scalable Electronic Cash

Had the community agreed to increase block sizes to enable on-chain scalability, Bitcoin could have developed as a global, decentralized payment system with implications such as:

- **Governments Losing Control Over Money:**
  - Bitcoin could bypass traditional banking systems, enabling global transactions without government oversight.
  - This would challenge the ability of governments to:
    - Collect taxes on income or transactions.
    - Enforce capital controls, such as limiting money movement across borders.
    - Implement monetary policy, as citizens and businesses could opt out of fiat entirely.
- **Financial Inclusion:**
  - With low transaction fees and high scalability, Bitcoin could serve as a practical payment method for the unbanked and underbanked, particularly in developing countries with unstable currencies.
  - This could reduce reliance on remittance services and create economic opportunities.
- **Marketplace Integration:**
  - Scalable on-chain solutions could support decentralized marketplaces, where users buy and sell goods directly using Bitcoin without intermediaries like banks, credit cards, or centralized payment processors.

## Challenges to Scalability:

- Larger block sizes increase centralization risks by making node operation more resource-intensive.
- Governments might aggressively regulate or ban Bitcoin if it competes directly with fiat.

## Concluding Thoughts

The evolution of Bitcoin reflects the tension between decentralization, scalability, and adoption. While BTC's shift toward digital gold ensures its survival as a store of value, it limits its potential as a peer-to-peer electronic cash system. Meanwhile, the rise of institutions and governments in Bitcoin ownership raises concerns about its decentralization and ability to challenge traditional financial systems.

A scalable Bitcoin could have profound implications, including reducing government control over money and enabling financial freedom. However, achieving this vision would require balancing technical feasibility, decentralization, and geopolitical realities—an ongoing challenge for the cryptocurrency community.

# The Bitcoin Reformation

In *The Bitcoin Reformation*, the key idea is that Bitcoin is much more than a financial trend—it's part of a broader revolution similar to the Protestant Reformation. Just like the Reformation shook up the old systems of power in 16th-century Europe, Bitcoin is challenging the modern financial system, particularly the control held by the International Monetary and Financial System (IMFS).

It starts with a historical parallel: during the Reformation, the Catholic Church held a monopoly on religious and spiritual services, which people began to rebel against. In a similar way, Bitcoin is offering a decentralized alternative to today's centralized financial structures.

There are four main reasons why both movements took off:

1. **Monopolistic Service Providers:** The Catholic Church had control over spiritual matters just as the IMFS has control over global finance today. Bitcoin disrupts that, offering an alternative financial system.
2. **Technological Revolution:** The printing press was a game-changer in the 16th century, just like the internet, encryption, and Bitcoin are today. These new technologies make it easier for people to move away from centralized control.
3. **A New Economic Class:** Back then, it was the merchant class that pushed back against old power structures. Now, it's millennials who are sceptical of traditional finance and embracing Bitcoin as an alternative.
4. **Defence and Escape:** Just as Dutch rebels used clever strategies to escape control (like flooding land to fight off invaders), today's "rebels" are using cryptography and decentralized technologies to protect their privacy and financial assets.

Looking ahead, Bitcoin could transform the way we handle money. We might see the rise of full-reserve banking (similar to how banks operated in 17th-century Amsterdam), new forms of peer-to-peer insurance, and the widespread use of Bitcoin as collateral for loans. Derivatives markets around Bitcoin could also grow, just like they did in Amsterdam's financial system during its Golden Age.

In conclusion, the idea here is that Bitcoin, much like the Reformation, represents a massive cultural shift. As more millennials gain economic power and continue to adopt Bitcoin, we could see a real challenge to the centralized financial systems that dominate today. Over time, Bitcoin has the potential to reshape the global economy just as the Reformation transformed Europe centuries ago.

This isn't just about finance—it's about a new way of thinking about money, privacy, and power in the digital age.



# AI Coins and Decentralized AI Infrastructure

## Overview of AI Coins

AI coins are digital assets designed to support and facilitate various functions within the AI ecosystem. They serve as the backbone for enabling decentralized, scalable, and secure AI infrastructure. The key functionalities of AI coins include:

1. **Processing Data:** Managing the computational processes required for AI operations, including data analysis, machine learning training, and inference.
2. **Distributing Power:**
  - AI coins often facilitate the distribution of processing power across networks.
  - They manage servers and computational resources essential for running AI models efficiently.
3. **Managing and Distributing AI Elements:**
  - Coordinating the deployment of specific AI tools, bots, or services.
  - Ensuring equitable and efficient allocation of resources within the network.

## Decentralized Physical Infrastructure Networks (DPINs)

- **Function:** DPINs are specialized decentralized systems designed to handle the physical and computational demands of AI.
- **Key Features:**
  - **Decentralizing AI Power:** These networks redistribute the computational and operational power of AI to ensure a more balanced and inclusive infrastructure.
  - **Anonymity and Privacy:** DPINs are instrumental in making AI operations private and anonymous, a narrative that is expected to gain significant momentum in the near future.

## Significance of Privacy and Anonymity in AI

- Ensuring privacy and user control over data is a transformative aspect of decentralized AI systems.
- Anonymity in AI usage and development can foster greater trust and adoption by protecting users from potential misuse or surveillance.

## Why This Matters

The focus on decentralization and privacy is set to become a major narrative in the evolution of AI. Although currently underrepresented in mainstream discussions, these elements are poised to

shape the future of AI development and adoption. By leveraging decentralized systems like DPINs, AI can:

- Achieve greater scalability.
- Foster innovation through open and equitable resource distribution.
- Address critical concerns around data security and user autonomy.

# MicroStrategy's (MSTR) Bitcoin Investment Strategy: A Comprehensive Overview

## Selling Volatility and Recycling into Bitcoin

Michael Saylor, CEO of MicroStrategy, has outlined the company's innovative approach to capitalizing on market dynamics. Their strategy involves "selling volatility" and reinvesting the proceeds into Bitcoin. The company generates a spread—the difference between the equity premium, convertible bond premium, and Bitcoin premium (measured as the Accretive Earnings Rate [AER] versus the USD). This spread allows them to enhance their Bitcoin holdings per share.

Key steps include:

### 1. Issuing Equity at a Premium:

- MicroStrategy raises capital by issuing shares when their stock price trades at a premium. For example, between November 18-24, the company sold 5.6 million shares at an average price of \$440 per share, raising \$2.46 billion.
- These funds were used to purchase 25,000 Bitcoin at an average price of \$98,000 each.

#### Results:

- Bitcoin holdings increased by ~10% by the end of Q3, reaching 252,000 BTC.
- Shares outstanding rose by only ~2.3%, from ~245 million shares, including in-the-money convertible notes.
- The Bitcoin per 1,000 shares metric rose from 1.03 BTC to 1.1 BTC, creating accretive value for shareholders.

### 2. Monetizing Volatility with Convertible Bonds:

- Convertible bonds provide an alternative way to raise capital without immediate dilution. On November 21, MicroStrategy issued a \$3 billion convertible bond maturing in 2029 at a 0% interest rate. These bonds are convertible into shares at a \$672 strike price, a 55% premium over the stock's then-current price of \$434.
- The proceeds were used to acquire ~30,000 Bitcoin at an average price of \$98,000 each.

#### Risk:

- If Bitcoin's price is below \$98,000 at maturity in 2029, the bonds will not convert, and MicroStrategy must repay the principal, potentially at a loss.

# Accretive Value Creation for Shareholders

MicroStrategy's strategy hinges on ensuring that the percentage increase in Bitcoin holdings exceeds the percentage increase in shares outstanding. By doing so, Bitcoin per share rises, creating accretive value for shareholders. Saylor describes this as effectively "selling \$1 bills for \$3."

## Key Risks

### 1. **Bitcoin Price Volatility:**

- Saylor's projection of Bitcoin appreciating at ~29% annually for the next 21 years underpins the strategy. If Bitcoin fails to meet this expectation, the company faces significant risks, particularly with debt repayment.

### 2. **Stock Price Dependency:**

- The strategy relies on MicroStrategy's stock trading at a premium. If market sentiment shifts, issuing equity becomes less viable, leaving debt as the primary funding option.

### 3. **Leverage and Debt:**

- The company's high leverage magnifies potential losses during prolonged Bitcoin bear markets, potentially impairing operations.

## Why Investors Choose MicroStrategy Over Bitcoin ETFs

MicroStrategy offers leveraged exposure to Bitcoin, unlike ETFs that merely track Bitcoin's price. Investors believe that the company's ability to issue debt and equity to acquire additional Bitcoin could yield higher returns than holding Bitcoin directly or investing in ETFs.

## Conclusion

MicroStrategy's financial engineering—issuing equity at a premium and leveraging convertible bonds—creates a unique avenue for Bitcoin investment. While the strategy provides significant upside for Bitcoin maximalists and bullish investors, it comes with substantial risks. The sustainability of this approach is contingent on Bitcoin's continued appreciation and the market's confidence in MicroStrategy's role as a Bitcoin proxy. Investors must weigh these factors carefully, as the duality of high potential returns and significant risks makes this strategy both innovative and polarizing.

Further reading: [Bitcoin's 'Glitch in the Matrix' Has MicroStrategy Stuck in a While Loop](#)

# Bitcoin's 'Glitch in the Matrix' Has MicroStrategy Stuck in a While Loop



In computer programming, there's a concept known as the "while loop", a piece of code that repeatedly executes a task until a certain condition is met. It seems that MicroStrategy (MSTR) is stuck in this loop.

Many "fundamental" investors are attempting to short the stock, as it trades at multiples far beyond its core net asset or book value. Yet, despite this, the stock shows no signs of relief. Why?

The answer lies in MicroStrategy's leveraged play on Bitcoin. CEO Michael Saylor has discovered a "glitch" that allows him to borrow money at essentially zero cost and pay nothing to lenders, using the proceeds to acquire more Bitcoin. It's a remarkable deal that keeps driving the cycle forward.

Saylor is issuing debt via convertible bonds with a 0% coupon, offering a 55% premium. Even at these terms, demand exceeds supply, creating what seems like an endless arbitrage opportunity. So, why should anyone stop?

The Bitcoin story is well known. As Bitcoin matures into a mainstream asset class and gains institutional adoption from firms like BlackRock (BLK), its value continues to climb. With only 21 million tokens in existence, its price appears to have no ceiling, attracting traders eager to own a piece of it.

Bitcoin has become the favoured asset for those betting against fiat currency debasement. However, it is important to recognise that Bitcoin is a high-risk, high-reward asset, driven by liquidity on steroids. We saw this in August when the Dollar/Yen trade unwound. In times of economic stress or geopolitical conflict, gold—while less "shiny" than Bitcoin—remains the traditional store of value.

Bitcoin's halving cycle, which occurs every four years, also contributes to its price rallies, especially around post-election periods. With President-elect Trump's Bitcoin-friendly policies—suggesting, for example, a strategic Bitcoin reserve—it's easy to see why some projections range from £250,000 to even \$1 million per Bitcoin.

MicroStrategy effectively holds Bitcoin, and as Bitcoin's price rises, so does the company's valuation. But the stock's multiple isn't merely a function of Bitcoin's price; it's amplified by the company's ability to borrow cheaply and use that debt to buy even more Bitcoin, creating a self-reinforcing cycle of upward momentum.

For institutions unable to trade Bitcoin futures—such as the Swiss National Bank—MicroStrategy offers an alternative way to gain exposure to Bitcoin's rise. With endless demand and limited supply, this explains much of the stock's persistence.

Fundamental investors running premium-to-NAV (Net Asset Value) models, citing the 3x+ multiple, are getting burned as they short MSTR. The liquidity is simply insufficient to fight the tide of rising demand.

While the saying "what goes up must come down" holds true, timing is critical. As Bitcoin's price increases, MicroStrategy's stock will likely rise far more than its intrinsic value. To make matters worse for shorts, with Bitcoin now included in the Nasdaq 100 Index (QQQ), the relationship between Bitcoin's performance and broader market indices becomes even stronger. Passive funds tracking these indices would push the stock higher, adding fuel to the fire.

MicroStrategy is trapped in a "while" loop, and only a decline in Bitcoin's price or regulatory intervention could bring it to an end. However, with major institutional investors firmly "hodling" their positions and a Bitcoin-friendly political environment, it's unlikely that regulation will intervene. As long as Bitcoin continues to rise, so too will MicroStrategy's stock.

Further reading: [MicroStrategy's Bitcoin Investment Strategy: A Comprehensive Overview](#)



# Bitcoin Knots

## 1. What is Bitcoin Knots?

- An **alternative implementation** of Bitcoin, derived from **Bitcoin Core** but with enhanced features.
- Provides **more control** over transaction policies, mempool management, and network filtering.
- **Fully compatible** with Bitcoin Core, allowing seamless switching between the two.

## 2. Key Features & Enhancements

- **Advanced Transaction Filtering** – Reject spam, dust, and inefficient transactions.
- **Stricter Mempool Policies** – Reduces blockchain bloat with settings like `Reject` `Parasites` and `Data Carrier Size`.
- **Customizable Pruning** – Handles blockchain storage dynamically based on available space.
- **Faster Feature Adoption** – Incorporates community-driven enhancements before they appear in Bitcoin Core.

## 3. Installation & Setup Notes (Start9 Box)

- Installed via **Start9's Marketplace** (Bitcoin Knots 28.1.0).
- Initial install error: "**Config Generation Error: No Match: blkconstr: Field Is Not Nullable**".
- **Resolution:** A simple restart of the Start9 box allowed Knots to sync and run normally.

## 4. Post-Installation Verification

- **Sparrow Wallet successfully connected** to Bitcoin Knots.
- **Last block synced correctly**, mempool data displayed as expected.
- **Lightning apps (LND, RTL) running without issues**.
- **Small test transaction sent successfully** via Coldcard & Sparrow Wallet.

## 5. Key Takeaways

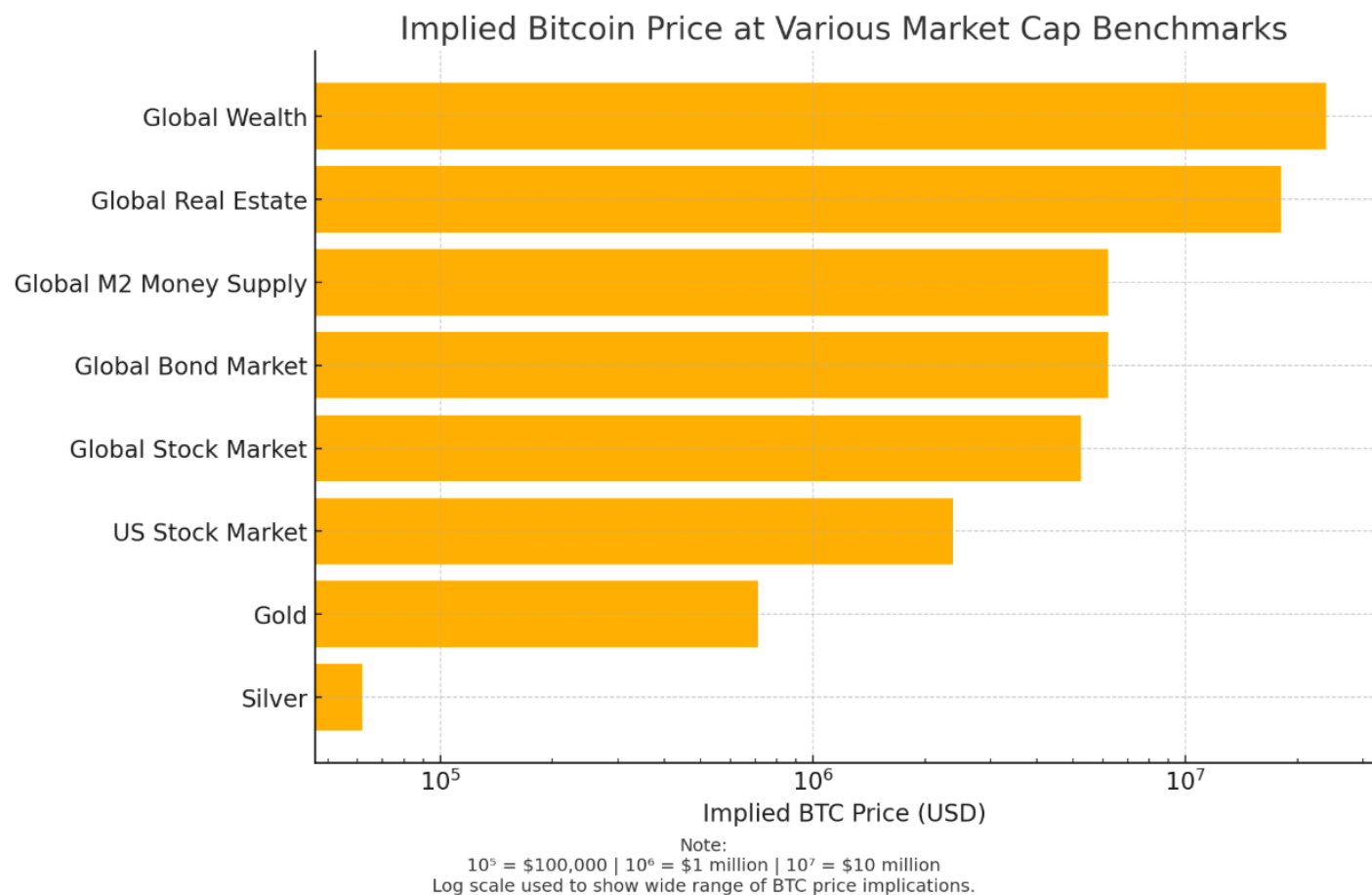
- **Bitcoin Knots runs smoothly** on Start9 after an initial restart.
- **Advanced filtering features help keep the network efficient**.
- **Fully compatible with existing wallets & Lightning services**.



# ? Bitcoin Price Projections Based on Asset Class Market Caps

This chart visualises what the **price of 1 Bitcoin (BTC)** would be if Bitcoin's total market cap grew to match various global asset classes.

Bitcoin's supply is **fixed at 21 million coins**, making it inherently scarce. As demand grows and market adoption increases, Bitcoin's market cap could—hypothetically—compete with other major stores of value.



# ? Asset Classes and Corresponding Implied BTC Prices

Asset Class	Approx. Market Cap (USD)	Implied BTC Price
Silver	\$1.3 trillion	~\$61,900
Gold	\$15 trillion	~\$714,000
US Stock Market	\$50 trillion	~\$2.38 million
Global Stock Market	\$110 trillion	~\$5.23 million
Global Bond Market	\$130 trillion	~\$6.19 million
Global M2 Money Supply	\$130 trillion	~\$6.19 million
Global Real Estate	\$380 trillion	~\$18.1 million
Total Global Wealth	\$500 trillion	~\$23.8 million

■

## ? Calculation Method

Each BTC price is calculated by:

$$\text{Implied BTC Price} = \frac{\text{Asset Class Market Cap}}{21,000,000}$$

For example:

If Bitcoin reaches the size of gold’s market cap:

$$\frac{\$15,000,000,000,000}{21,000,000} \approx \$714,285$$

## ? Reading the Chart Scale

The X-axis uses a **logarithmic scale** to accommodate the wide price range:

- $10^5$  = \$100,000
- $10^6$  = \$1 million
- $10^7$  = \$10 million

This helps show smaller values like silver without visually flattening the higher targets like real estate or global wealth.

# ? Key Insight

Even reaching **10% of gold's market cap** would imply a BTC price over **\$70,000** — already within historical highs. The long-term upside remains significant due to **Bitcoin's absolute scarcity and growing global adoption.**

# Monero: Private and Decentralized Cryptocurrency

This chapter covers the setup, management, and remote access of a self-hosted Monero node and wallet. It includes detailed guides on daemon configuration, wallet CLI usage, Tailscale integration for secure remote connections, troubleshooting, and best practices for maintaining privacy and security within the Monero ecosystem.

# Monero: Privacy by Design – Origins, Technology, and Ongoing Evolution

## Overview

**Monero** is a privacy-focused, decentralized cryptocurrency built to deliver strong financial confidentiality by default. Unlike Bitcoin and other cryptocurrencies that offer optional or partial privacy features, Monero was designed from the ground up to be untraceable and anonymous. It achieves this through the use of advanced cryptographic techniques, including ring signatures, stealth addresses, confidential transactions, and bulletproofs.

At its core, Monero values decentralization and user sovereignty. To maintain a level playing field for miners and avoid hardware-based centralization, the project regularly changes its proof-of-work (PoW) algorithm to resist ASIC dominance.

---

## Privacy Technologies

- **Ring Signatures:** Obscure the origin of a transaction by combining the sender's output with decoy outputs from other users, making it computationally infeasible to determine the actual source.
  - **Stealth Addresses:** Each transaction generates a unique, one-time address for the recipient, preventing transactions from being linked to their wallet address.
  - **Ring Confidential Transactions (RingCT):** Hide the transaction amount in addition to sender and recipient data. Introduced in 2017 and based on a proposal by Bitcoin Core developer Greg Maxwell, RingCT significantly enhances overall privacy.
  - **Bulletproofs:** Implemented in 2018, this zero-knowledge proof technology replaced the older range proofs used in confidential transactions. Bulletproofs reduce transaction size by over 80%, improving scalability and efficiency.
  - **Fungibility:** Since Monero coins have no visible transaction history, they are inherently fungible—each coin is indistinguishable from another, unlike Bitcoin, where coins can be “tainted” based on past activity.
- 

## Origins and History

Monero began as a fork of **Bytecoin**, the first implementation of the **CryptoNote** protocol—a system designed to address privacy issues, mining centralization, and uneven coin distribution in Bitcoin. Bytecoin launched in March 2014 but was marred by controversy over an 80% premine.

In response, a Bitcointalk user known as **thankful\_for\_today** forked the Bytecoin code into a new project called **BitMonero**—a fusion of "Bit" (Bitcoin) and *Monero* (Esperanto for "coin"). The launch of BitMonero was poorly received, prompting seven community members to fork it again under the simplified name **Monero**. This group, led by the pseudonymous developer **Fluffypony** (Riccardo Spagni), became the first Monero Core team.

From its inception in April 2014, Monero has operated without a premine or founder rewards, aligning with its ethos of fairness and decentralization.

Monero's strong privacy features soon made it a favored cryptocurrency among users seeking financial confidentiality. It also became one of the most used currencies on darknet markets—a reality that has contributed to both its popularity and controversy.

---

## Ongoing Development

Monero evolves through **scheduled hard forks**, typically occurring every six months. These upgrades introduce protocol improvements, fix bugs, and often include changes to the PoW algorithm to reinforce decentralization.

Notable milestones include:

- **2017:** Introduction of RingCT, enabling complete transaction privacy.
  - **2018:** Adoption of bulletproofs, reducing transaction size and cost.
  - **2019:** Resignation of lead maintainer Fluffypony, a move intended to further decentralize project governance.
- 

## Regulatory and Social Context

Monero's unwavering focus on anonymity has drawn mixed reactions. Supporters champion it as a vital tool for privacy rights and financial autonomy. Critics, including regulators, express concern about its potential misuse in illicit activity.

Some jurisdictions have considered or enacted restrictions on Monero due to its untraceable nature. Despite this, Monero remains a legitimate cryptocurrency with a dedicated global community and ongoing technical innovation.

---

## Conclusion

Monero stands out in the cryptocurrency space for its strong commitment to privacy, decentralization, and fungibility. While its anonymity features spark ongoing debate, Monero continues to push forward as a leading project for those who believe financial privacy is not just a feature—but a fundamental right.

See also:

- [Monero Wallet Guide](#)
- [Monero Monitoring & Troubleshooting Commands Reference](#)

# Monero Monitoring & Troubleshooting Commands Reference

## 1. Check Monero Daemon Status

```
systemctl status monerod
```

## 2. Start / Stop / Restart Monero Daemon

```
systemctl start monerod  
systemctl stop monerod  
systemctl restart monerod
```

## 3. Check Real-Time Syncing Progress (Logs)

```
journalctl -u monerod.service -n 20 --no-pager
```

Or continuously (live):

```
tail -f /mnt/monero/.bitmonero/monerod.log
```

## 4. Check if Monero is Running

```
ps aux | grep monerod
```

## 5. List All Screen Sessions

```
screen -ls
```



## 6. Reattach to Existing Screen Session

```
screen -r <session_name>  
# e.g., screen -r monero
```

## 7. Detach from Screen (keep running in background)

```
Ctrl + A then D
```

## 8. Check Disk Space

```
df -h
```

Specific mount point:

```
df -h /mnt/monero
```

## 9. Inspect Large Files/Folders

```
du -xh / | grep -P 'G\t' | sort -hr | head -20
```

## 10. Check for Deleted Files Still Using Disk Space

```
lsuf | grep deleted
```

## 11. Check Wallet Sync Status from CLI

Inside `monero-wallet-cli`:

```
status  
net_stats
```

## 12. Manually Launch Wallet CLI

```
/mnt/monero/monero-wallet-cli --wallet-file /mnt/monero/wallets/blueace
```

## 13. Account Tag and Label Commands

Inside wallet CLI:

```
account tag <tag_name> <index>
account tag_description <tag_name> <description>
account label <index> <label text>
```

## 14. Ensure Mount on Boot

In `/etc/fstab`, add:

```
UUID=<your-uuid> /mnt/monero ext4 defaults 0 2
```

Then:

```
mount -a
```

## 15. Quick Ping to Confirm Internet/DNS

```
ping -c 3 deb.debian.org
```

This set of commands should cover most operational and recovery scenarios. Feel free to expand this as your setup evolves (e.g., adding wallet RPC, remote access, VPN, or port monitoring).

See also:

- [Monero Wallet Guide](#)
- [Monero Privacy by Design - Origins, Technology, and Ongoing Evolution](#)

# Monero Wallet Guide

## ? Create a New Wallet

```
/mnt/monero/monero-wallet-cli --generate-new-wallet /mnt/monero/wallets/<wallet_name>
```

- Replace `<wallet_name>` with your desired wallet filename.
- You will be prompted to set a password and make a secure note of your seed phrase.

## ? Open an Existing Wallet

```
/mnt/monero/monero-wallet-cli --wallet-file /mnt/monero/wallets/<wallet_name>
```

- Replace `<wallet_name>` with the name of the wallet you want to open (e.g., `nero`).

## ? Mount Wallet Directory (if needed)

If your wallet directory is not mounted automatically on boot:

```
mount /dev/sda1 /mnt/monero
```

To mount it automatically on boot, add the following line to `/etc/fstab`:

```
/dev/sda1 /mnt/monero ext4 defaults,noatime 0 2
```

## ?? Address Labeling

**Create a new subaddress with a label:**

```
address new <label>
```

- Example: `address new exchange_withdrawal`

**View all addresses and their labels:**

```
address all
```

### Change an existing label:

```
address label <index> <new label>
```

- Index is the number listed with `address all` (e.g., `1`).

## ? Tag Management

### Assign a tag to a specific account index:

```
account tag <tag_name> <account_index>
```

- Example: `account tag personal 0`

### Set a description for a tag:

```
account tag_description <tag_name> <description>
```

- Example: `account tag_description personal "casual spending"`

## ? Copy Wallet Files to Your Local Machine (via SCP)

From your **local machine**, run:

```
scp root@<remote_ip>:/mnt/monero/wallets/<wallet_name>* ~/Downloads/
```

```
scp 'root@192.168.1.155:/mnt/monero/wallets/nero*' ~/Downloads/ # on Mac
```

- Replace `<remote_ip>` with your Pi's IP address (e.g., `192.168.1.155`).
- Replace `<wallet_name>` with your actual wallet name (e.g., `blueace`).

## ? How to Use Wildcards with `scp` on macOS (zsh)

When using the `scp` command in the **macOS Terminal**, you might run into this error:

```
zsh: no matches found: root@192.168.1.155:/mnt/monero/wallets/nero*
```

This happens because **zsh**, the default shell on macOS, tries to expand the `*` **wildcard** **locally on your Mac** before sending the command to the remote machine. Since there's no local file named `nero*`, it throws an error.

## ? Solution

To prevent zsh from expanding the wildcard, **quote the remote path** in your `scp` command:

```
scp 'root@192.168.1.155:/mnt/monero/wallets/nero*' ~/Downloads/ # copy Monero wallet to Mac
```

“`''` Always use **single quotes** `' '` to preserve the literal meaning of wildcards for the remote shell.

Let me know if you'd like this saved in a reusable note or command reference

## ? Why Avoid Using the Primary Address

- **Primary address reuse reduces privacy** by making it easier to associate transactions.
- **Best practice:** Always use a new subaddress per transaction or recipient.
- Monero's built-in subaddress system enhances unlinkability and protects financial metadata.

## ? Sweep Balance to Another Wallet

To transfer all funds to another address:

```
sweep_all <destination_address>
```

- Used to migrate to a new wallet like `nero` from `blueace`.

See also:

- [Monero Privacy by Design - Origins, Technology, and Ongoing Evolution](#)
- [Monero Monitoring & Troubleshooting Commands Reference](#)

# Secure Remote Access to Monero Node Using Tailscale VPN

**Q: What VPN solution are we using to securely access the Monero node remotely?**

A: We use **Tailscale**, a user-friendly mesh VPN based on WireGuard, to create a secure, private network between your devices.

**Q: How do I install Tailscale on my DietPi (Monero) server?**

A: Run this command on the DietPi terminal to install Tailscale:

```
curl -fsSL https://tailscale.com/install.sh | sh
```

**Q: How do I start and enable Tailscale service on DietPi?**

A: After installation, start the service and enable it to run at boot:

```
sudo systemctl enable --now tailscaled
```

**Q: How to authenticate and connect the DietPi server to your Tailscale network?**

A: Run this to authenticate:

```
sudo tailscale up
```

**Q: How do I check that my DietPi server is connected to the Tailscale network?**

A: Use the status command:

```
tailscale status
```

You should see your DietPi listed with its Tailscale IP address.

## Q: How to install Tailscale on your local Mac (or other client machine)?

A: Download the Tailscale app from [tailscale](https://tailscale.com) and install it. Then sign in with the same Tailscale account you used for the server.

## Q: How do I confirm the local machine is connected and can reach the DietPi server?

A: On your Mac terminal, ping the DietPi Tailscale IP:

```
ping <dietpi-tailscale-ip>
```

Replace `<dietpi-tailscale-ip>` with the actual Tailscale IP you see in `tailscale status`.

## Q: How do I connect my Monero wallet CLI on my Mac to the remote monerod daemon over Tailscale?

A: Use the remote daemon IP and port in your wallet CLI command:

```
./monero-wallet-cli --wallet-file /path/to/yourwallet --daemon-address <dietpi-tailscale-ip>:18081 --trusted-daemon
```

## Q: What if I encounter connection errors or timeouts?

A: Verify:

- The monerod daemon is running and listening on `0.0.0.0:18081` on DietPi.
  - Firewall rules on DietPi allow incoming connections on port 18081.
  - Tailscale network shows both devices connected.
  - You can ping and `nc` (netcat) test the port from your local machine.
- 

## Q: How to stop or disable Tailscale on DietPi if needed?

A:

```
sudo systemctl stop tailscaled  
sudo systemctl disable tailscaled
```



# Monero Node Administration & Maintenance Guide

## 1. Checking for Duplicate Files and File Usage

- **Find duplicate files by name** (search for specific file names):  
`find / -type f \( -name "output0.csv" -o -name "output1.csv" \) 2>/dev/null`
- **Check file disk usage** (with depth level, e.g., 1 for directory contents):  
`du -h --max-depth=1 /path/to/dir`

## 2. Open Ports and What Services They Listen To

- **Check for open ports and the services they listen to** (view listening ports and active connections):  
`netstat -tuln`
- **Find which services are using specific ports (e.g., port 18080):**  
`lsof -i :18080`
- **Check if specific ports are open (e.g., 18080 or 22):**  
`netstat -tuln | grep 18080`
- **List open ports with `ss` (useful alternative to `netstat`):**  
`ss -tuln`

## 3. Checking iptables and UFW Rules

### iptables Commands:

- **View iptables rules:**  
`iptables -L`
- **View specific chain rules (e.g., INPUT chain):**  
`iptables -L INPUT -n --line-numbers`

- **Show a specific port's status in iptables** (e.g., port 22 or 18080):  
`iptables -L INPUT -n --line-numbers | grep -E '22|18080'`
- **View specific rules in a named chain (e.g., `ts-input`)**:  
`iptables -L ts-input -n --line-numbers`
- **Delete an iptables rule** (e.g., delete rule number 3 in the INPUT chain):  
`iptables -D INPUT 3`
- **List all iptables rules including the ones set by UFW**:  
`iptables -L`
- **ufw (Uncomplicated Firewall)**:  
`ufw status`
- **Enable UFW**:  
`ufw enable`
- **Allow incoming traffic on port 18080** (for Monero node):  
`ufw allow 18080/tcp`
- **Check UFW status and rules**:  
`ufw status verbose`
- **Disable UFW** (if necessary):  
`ufw disable`

## 4. Network and Firewall Troubleshooting

- **Check IP routing or address information**:  
`ip a`
- **Check active connections and the state of the ports**:  
`ss-tuln`

## 5. Commands for Managing Services and Systemd

- **Check the status of a service (e.g., monerod)**:  
`systemctl status monerod`

- **Stop a service (e.g., monerod):**  
`systemctl stop monerod`
- **Start a service (e.g., monerod):**  
`systemctl start monerod`
- **Enable a service to start on boot (e.g., monerod):**  
`systemctl enable monerod`
- **Disable a service from starting on boot:**  
`systemctl disable monerod`

## 6. Mail Setup (msmtp, Mailutils)

- **Send a test email using msmtp:**  
`echo "Test email" | msmtp --debug user@example.com`
- **Check mail logs for issues** (e.g., for msmtp sending failures):  
`cat /var/log/mail.log`
- **Configure msmtp for sending emails (via SMTP):** Edit the `~/.msmtprc` file to include the email credentials and SMTP server details.

## 7. Directory and File Clean-Up

- **Remove directories and files** (e.g., remove `.bitmonero`):  
`rm -rf /path/to/directory`
- **Delete unnecessary files (e.g., log files):**  
`rm /path/to/file.log`

## 8. System Information and Disk Usage

- **Show system disk usage:**  
`df -h`
- **View detailed file sizes in directories:**  
`du -sh /path/to/directory`
- **Show the current system resource usage:**  
`top`

## 9. Miscellaneous Useful Commands

- **Check the current kernel version:**  
`uname -r`
- **Search for a specific string in files:**  
`grep "search_string" /path/to/file`
- **View the contents of a file:**  
`cat /path/to/file`

## 10. File Transfer Commands

- **Transfer files via SCP** (e.g., from your Pi to local):  
`scp root@your_pi_ip:/path/to/file /local/destination`
- **Transfer files via SCP with wildcards:**  
`scp root@your_pi_ip:/path/to/files/* /local/destination/`
- **Transfer files via SCP from Pi to Mac** (Mac uses a similar approach to Linux for SCP):  
`scp 'root@your_pi_ip:/path/to/file' ~/local/destination`  
Example:  
`scp 'root@192.168.1.155:/mnt/monero/wallets/nero*' ~/Downloads/`

# ? Making Monero Port 18080 Firewall Rule Persistent on DietPi (with ts-input Chain)

## ? Problem Summary

- Your DietPi uses a custom iptables chain called `ts-input` (e.g., created by Tailscale).
  - The Monero daemon needs TCP port 18080 open inbound for P2P connections.
  - Direct rule persistence (iptables-persistent, cron @reboot) **fails** because the `ts-input` chain is created **late** in boot or dynamically.
  - Applying `iptables -I ts-input 1 -p tcp --dport 18080 -j ACCEPT` too early errors out:  
`iptables: No chain/target/match by that name.`
- 

## ?? Workaround Overview

- Use a **custom script** that:
    - **Waits** for `ts-input` chain to exist (polls for up to 60 seconds)
    - **Removes duplicate** existing rules for port 18080
    - **Inserts the rule if missing**
  - Run this script via a **systemd service** triggered after the network and Tailscale daemon start.
- 

## ? Step 1 — Create the script

```
/root/scripts/fix-monero-fw.sh
```

```
#!/bin/bash
```

```
# Wait up to 60 seconds for ts-input chain to appear
for i in {1..60}; do
    if iptables -L ts-input &>/dev/null; then
```

```

    echo "$(date) - Chain ts-input exists, proceeding"
    break
else
    echo "$(date) - Chain ts-input does not exist yet, waiting..."
    sleep 1
fi
done

# Exit if chain never appears
if ! iptables -L ts-input &>/dev/null; then
    echo "$(date) - ERROR: Timeout waiting for ts-input chain. Exiting."
    exit 1
fi

# Remove duplicate rules for tcp port 18080 in ts-input, keep only first
RULE_COUNT=$(iptables -L ts-input -n --line-numbers | grep -c 'tcp dpt:18080')

while [ "$RULE_COUNT" -gt 1 ]; do
    echo "$(date) - Removing duplicate rule at position 2"
    iptables -D ts-input 2
    RULE_COUNT=$(iptables -L ts-input -n --line-numbers | grep -c 'tcp dpt:18080')
done

# Insert rule if missing
if ! iptables -C ts-input -p tcp --dport 18080 -j ACCEPT &>/dev/null; then
    echo "$(date) - Rule not found, inserting rule"
    iptables -I ts-input 1 -p tcp --dport 18080 -j ACCEPT
else
    echo "$(date) - Rule already exists, no action needed"
fi

exit 0

```

**Make executable:**

```
chmod +x /rootscripts/fix-monero-fw.sh
```

---

## ?? Step 2 — Create systemd service

```
/etc/systemd/system/fix-monero-  
fw.service
```

[Unit]

Description=Add iptables rule for Monero port 18080 after tailscaled starts

After=network-online.target tailscaled.service

Wants=network-online.target tailscaled.service

[Service]

Type=oneshot

ExecStart=/root/scripts/fix-monero-fw.sh

RemainAfterExit=yes

StandardOutput=append:/var/log/fix-monero-fw.log

StandardError=append:/var/log/fix-monero-fw.log

[Install]

WantedBy=multi-user.target

## ? Step 3 — Enable and start the service

systemctl daemon-reload

systemctl enable fix-monero-fw.service

systemctl start fix-monero-fw.service

---

## ? Step 4 — Verify after reboot

Check the rule applied:

```
iptables -L ts-input -n --line-numbers | grep 18080
```

Check logs:

```
tail /var/log/fix-monero-fw.log
```

---

## ? Optional — Log rotation

**Create** `/etc/logrotate.d/fix-monero-fw`

```
/var/log/fix-monero-fw.log {  
    daily  
    rotate 7  
    missingok  
    notifempty  
    compress  
    delaycompress  
    copytruncate  
}
```

```
}
```

**Test rotation:**

```
logrotate --force /etc/logrotate.d/fix-monero-fw
```

# ? Result

- Firewall rule for port 18080 in `ts-input` chain is applied reliably **after reboot**.
- Duplicate rules are automatically cleaned up.
- Logs are captured and rotated for easy troubleshooting.

# ? Additional Useful Commands

Command	Description
<code>lsof -i :18080</code>	Lists processes using TCP port 18080
<code>journalctl -u fix-monero-fw.service -b -n 50</code>	Shows last 50 logs from the fix-monero-fw.service for current boot
<code>cat /var/log/fix-monero-fw.log</code>	Displays the content of the fix-monero-fw.log file
<code>journalctl -u monerod.service -f</code>	Follows live logs of the Monero daemon service
<code>`ps aux`</code>	<code>grep monerod`</code>
<code>systemctl daemon-reexec</code>	Reloads the systemd manager configuration
<code>systemctl daemon-reload</code>	Reloads systemd units and configurations
<code>systemctl enable iptables-custom-rule.service</code>	Enables the iptables custom rule service
<code>iptables -L ts-input -n --line-numbers</code>	Lists all rules in <code>ts-input</code> chain with line numbers
<code>`iptables -L ts-input -n --line-numbers`</code>	<code>grep 18080`</code>
<code>sudo crontab -l</code>	Lists root user's cron jobs





# ? Why Earlier Attempts to Persist the iptables Rule for Port 18080 on DietPi Failed

## ? Current Setup Summary

- **System:** DietPi (Debian-based, minimal)
  - **Firewall:** `iptables` with a `ts-input` chain (Tailscale or custom)
  - **Monero Daemon:** `monerod` running with `--p2p-bind-ip=0.0.0.0`
  - **Problem:** No incoming P2P connections
- 

## ?? Core Issue: The `ts-input` Chain Isn't Available Early Enough

- The custom iptables chain `ts-input` (used by Tailscale or similar) is created **late in the boot process** or dynamically **after** many system services start.
- Any firewall rule referencing `ts-input` applied **before this chain exists will fail** silently or produce errors like:

```
“ iptables: No chain/target/match by that name.
```

- This timing issue is the root cause why most standard methods to persist rules don't work on this setup.
- 

## ? Review of Earlier Approaches and Why They Failed

### 1. ? Using iptables-persistent

- Saves rules to `/etc/iptables/rules.v4` and restores them on boot automatically.
- **Fails because the `ts-input` chain doesn't exist yet** when the restore process runs.
- Result: The rule referencing `ts-input` is ignored or dropped and does not persist after reboot.

## 2. ?? Basic systemd Service Applying the Rule on Boot

- A one-shot systemd service running after `network-online.target` tried to insert the rule.
- Since `ts-input` chain creation happens **after** or asynchronously to the network target, the service runs **too early**.
- Rule insertion fails with “No chain/target/match by that name,” and no retry is attempted.
- No rule is applied.

## 3. ? Cron Job with @reboot Directive

- Cron jobs run early after reboot, often before `ts-input` chain exists.
- A `sleep` delay helps but is unreliable due to race conditions in chain creation timing.
- Lack of checks and retries means the rule often doesn't get applied.
- Logs for troubleshooting may be missing or incomplete.

---

## ? Why Our Final Workaround Works

- The custom script **polls repeatedly (up to 60 seconds)**, waiting for the `ts-input` chain to appear before applying any rule.
- It **cleans up duplicate rules** to keep firewall rules tidy on repeated runs.
- The systemd service **depends on both network and Tailscale daemons** to start, improving timing.
- Logs output are captured for troubleshooting and verification.
- Together, these ensure the rule is **only applied once the chain exists**, solving the timing/race condition problem.

---

## ? Summary

Approach	Outcome	Reason for Failure
iptables-persistent	Rule not applied after reboot	<code>ts-input</code> chain missing when rules restored

Approach	Outcome	Reason for Failure
Basic systemd service	Rule insertion fails early	Runs before <code>ts-input</code> chain is created
Cron @reboot + sleep	Unreliable, race conditions persist	Chain not guaranteed to exist after sleep delay
Final script + systemd	Reliable rule application	Waits and retries until chain exists before insert

# ? Creating and Managing Additional Accounts for Transaction Segmentation in Monero Wallet

## ? Create a New Account

```
account new <label>
```

- Example: `account new donations`

## ?? Create and Label a Subaddress

```
address new <label>
```

- Example: `address new Dev_Team`

## ? Assign a Tag to an Account

```
account tag <tag_name> <account_index>
```

- Example: `account tag donate 1`

## ? Set a Tag Description

```
account tag_description <tag_name> <description>
```

- Example: `account tag_description donate "Donations to causes and projects"`

## ? Send Funds from Account 0 to Account 1

First, switch to account 0 (if not already selected):

```
account switch 0
```

Then send funds to the Dev Team subaddress in account 1:

```
transfer <destination_address> <amount>
```

- Example:

```
transfer 88xkUv...xyz 0.04
```

---

## ? Notes on Monero Account and Address Behaviour

- Monero **treats each account holistically**. Even if funds are sent from a subaddress, the account's **primary address** may become marked as "used" after a transaction.
- This does not compromise privacy — it simply reflects internal wallet indexing.
- Using **subaddresses** for sending/receiving helps keep your financial metadata compartmentalised.
- **New accounts** can be used to segregate funds for donations, savings, or specific activities.

---

## ?? Address Labelling

**Create a new subaddress with a label:**

```
address new <label>
```

- Example: `address new exchange_withdrawal`

**View all addresses and their labels:**

```
address all
```

**Change an existing label:**

```
address label <index> <new label>
```

- Index is the number listed with `address all` (e.g., `1`).

---

# ? Tag Management

## Assign a tag to a specific account index:

```
account tag <tag_name> <account_index>
```

- Example: `account tag personal 0`

## Set a description for a tag:

```
account tag_description <tag_name> <description>
```

- Example: `account tag_description personal "casual spending"`
-