RSK: Bitcoin Merge Mining is Here to Stay

Published on: 16 July, 2019

Read Time - 14 mins

By Sergio Demian Lerner, RSK Chief Scientist

Introduction to Bitcoin Mining

Bitcoin Mining is the process used by Bitcoin to achieve Sybil-attack resistance and forms the basis for Nakamoto Consensus. The process involves solving computationally difficult puzzles while selecting a valid set of transactions to be applied to the ledger. Nowadays most Bitcoin mining is performed by mining pools. Mining pools exist to reduce payout variance for miners, but also to reduce their maintenance costs and bandwidth requirements. Mining pools follow the client-server architecture where the miners or “workers” (the clients) connect to the poolserver (the server). The poolserver runs one of the mining pool server software. Some common softwares are Ckpool, Btcpool, and Eloipool.

The poolserver generally communicates with a bitcoind instance through an RPC channel over unencrypted TCP/IP. The channel is used for obtaining information about the current best branch, and the parent block hash for the block to be mined. Also the poolserver obtains a valid set of pre-selected transactions. This is performed using the bitcoind RPC command getblocktemplate. The poolserver will choose a subset of those transactions to include them in the new block. Additionally, the poolserver generally polls bitcoind every few seconds to detect changes in the parent block. To reduce the delay in detecting new blocks, some poolserver implementations connect to bitcoind over the p2p protocol or use the bitcoind blocknotify command line option to be notified as soon as possible when a block arrives. The poolserver can make use of one or more standard p2p connections to the Bitcoin network in order to submit the newly created blocks, and it can communicate with fast-block transfer backbones such as FALCON or FIBRE.

Scroll down

Merge-Mining

Merge mining is a technique for using the same mining hashrate that secures a primary blockchain to secure a secondary blockchain. Namecoin was the first cryptocurrency to merge-mine with Bitcoin. Merge mining is performed by embedding the block id from the secondary blockchain (a cryptographic hash of the new block recently constructed) somewhere in the block of the primary blockchain that is being mined. This secondary hash, prefixed by some short descriptive text or magic bytes is called the merge-mining “tag”. The prefix enables the secondary blockchain to locate the tag. However, there must be no ambiguity on where the tag is located: a block from the primary-blockchain must get associated with none or at most one block of a secondary blockchain. While the use of cryptographic hashing for the linkage prevents cheating, the linkage security requirements are much lower than what’s required in traditional cryptography. Informally, the only security requirement for merge-mining is that it must be more difficult to create a primary-blockchain block that can be associated with two blocks from the same secondary blockchain than to mine two different primary-blockchain blocks, one for each association, at the difficulty of the secondary blockchain. For RSK, the primary blockchain is, of course, Bitcoin, and the secondary is RSK. The RSK block difficulty currently compares to 70-bit security, while Bitcoin difficulty compares to 74 bits.

Proof of Work Proxy

In merge-mining, a Bitcoin header serves only as a proof-of-work proxy. The RSK blockchain must interpret the PoW of a Bitcoin block header and search in the Bitcoin block for the tag that uniquely establishes the relation with an RSK block header. Therefore it transitively translates the Bitcoin block PoW to an RSK block PoW. As mentioned before, the difficulty for the RSK blockchain is lower than the difficulty for the Bitcoin blockchain, so many Bitcoin block headers that do not solve the Bitcoin PoW puzzle will be valid solutions to the RSK PoW puzzle. Let's dig a bit deeper into this distinction. Each blockchain computes an expected difficulty for every block. This difficulty is defined by all prior blocks, in order to keep the average time between blocks approximately constant. Internally, the difficulty is translated into a "target", which is inversely proportional to the difficulty. The target is a 256-bit unsigned integer.

Show 10 entries

Search:

Blockchain	Target
Bitcoin	000000000000000000165exxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
RSK	00000000000000000db5a4xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Namecoin	00000000000000000019xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Showing 1 to 3 of 3 entries

Previous Next

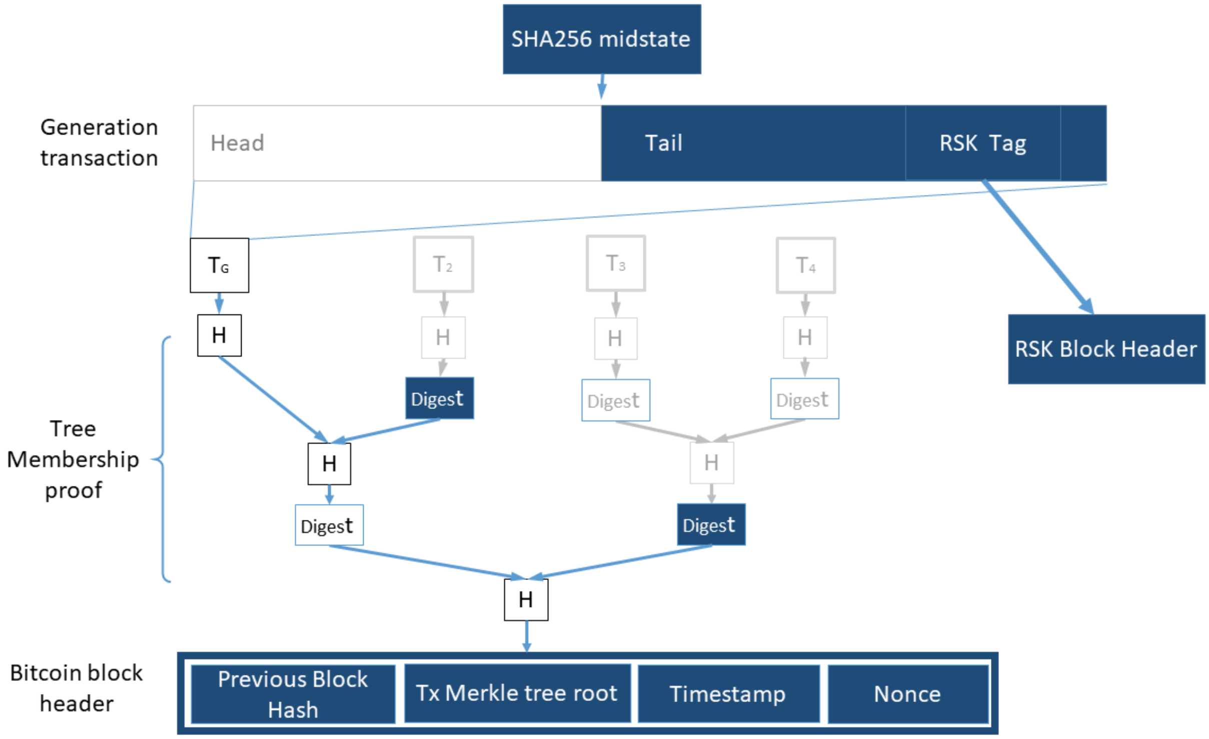
Approximate targets for merge-mined blockchain, on the same day. RSK has a higher target because blocks are 20 times more frequent

Due to the unpredictability of cryptographic hash digest, a block header id, which is a cryptographic hash digest of the block header contents, is assumed to represent a uniformly random variable. While this may not be theoretically true, it works in practice as there are no known practical methods to reverse the hash functions used in Bitcoin (double SHA256) or in RSK (Keccak). This hash digest, when interpreted as an unsigned number, must be lower than the target, for the block to represent a solution to the PoW puzzle. Therefore the lower the target is, the more difficult is to solve the PoW puzzle.

Scroll down

SPV Proofs

The secondary blockchain does not require the full Bitcoin block to validate the PoW of the Bitcoin header and associate it with the RSK header. This association can be proven by an SPV proof (which mainly comprises a Merkle tree membership proof). The following diagram depicts the relation between the different parts of the proof. The blue boxes represent the information that is contained in the SPV proof, and needs to be transmitted along the RSK block.



A Bitcoin SPV Proof Making Possible to Compress Merge-mined PoW proofs

Targets Hierarchy

Scroll down

The RSK target will be generally higher than the Bitcoin target, because RSK blocks are more frequent than Bitcoin blocks, so the RSK puzzle is less difficult to solve. Therefore a Bitcoin block header that solves the RSK PoW puzzle may not be accepted by the Bitcoin network. Note that both puzzles involve working with Bitcoin headers, and actually solving them involves the same try-and-error process. So how can the pool detect and inform about the more frequently solved RSK puzzles while miners are looking to solve the more difficulty Bitcoin puzzle? The answer is that the miners are already looking to solve blocks at much lower difficulty than what Bitcoin requires. These intermediate blocks are called “shares”, and they are required by the poolsever for accounting. In fact, it is the poolserver software the one that will command miners to try to solve the much simpler puzzle (specifying a much higher target). You may think that shares are some kind of intermediate solutions that help to solve the real Bitcoin puzzle but they are not. Shares can’t be extended to achieve a real block solution. However, the more powerful a miner is, the more shares she will create for the pool between real solutions. Shares, therefore, provide higher granularity for accounting miners’ contributions. The shares are transmitted to the poolserver regularly so the server can fairly split future earnings between all involved clients, weighting their hashing contributions. But shares are also transmitted because one of them can be (by chance) a solution to the current Bitcoin PoW puzzle. So miners don’t need to receive the real Bitcoin PoW puzzle difficulty (or target) from the poolserver, and they do not normally know if they have solved a Bitcoin block until this is communicated back by the pool. The poolserver checks each received share, reconstructs the block header, and if the header double SHA256 hash digest is numerically lower than the target associated with the current Bitcoin difficulty, it forwards the block to the bitcoind daemon, which spreads it over the network. Since each secondary blockchain may have a different difficulty, a merge-mined capable poolserver has to do this check for every secondary blockchain it handles. If the Bitcoin header represents a valid solution for the RSK blockchain PoW puzzle, it sends the bitcoin block header to rskj, which will append the associated RSK block and forward it as valid to the RSK network.

Note that for RSK the poolserver has the option to transmit to rskj only an SPV proof of the tag embedded in the Bitcoin block (more on this later), to reduce bandwidth requirements.

The following table shows approximate difficulties (in terms of average number of nonces iterated to find a puzzle solution, as of June 2019) of Bitcoin, RSK, and shares:

Show 10 entries

Search:

Solution destination	Block interval	Average number of nonce iterations to find a solution	Assumptions
Bitcoin	10 minutes	2^{74}	100% Bitcoin hashrate
RSK	30 seconds	2^{69}	50% merge-mining
Mining pool share	3.3 seconds per client	2^{52}	20% hashing power, 4000 clients Ckpool software

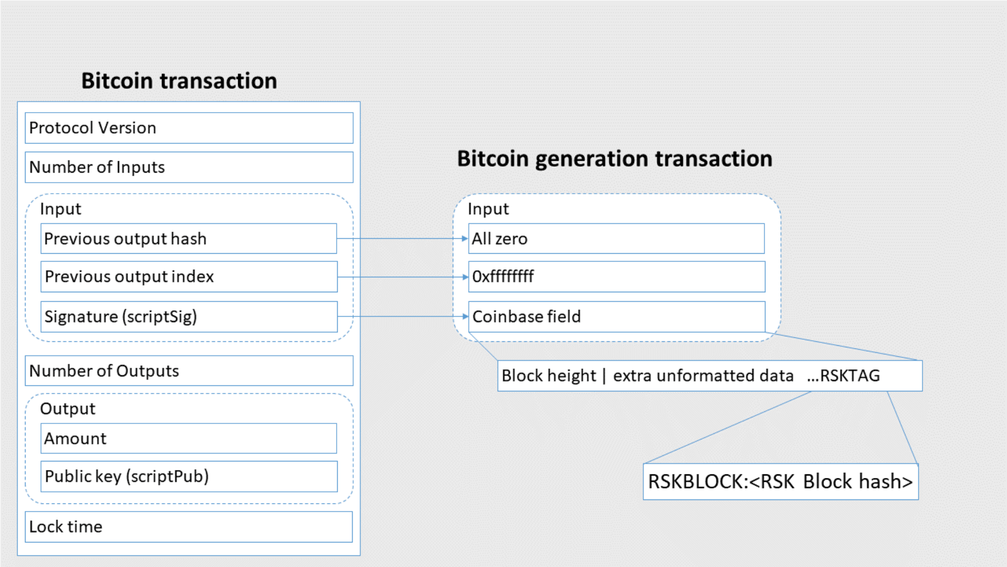
Showing 1 to 3 of 3 entries

Previous Next

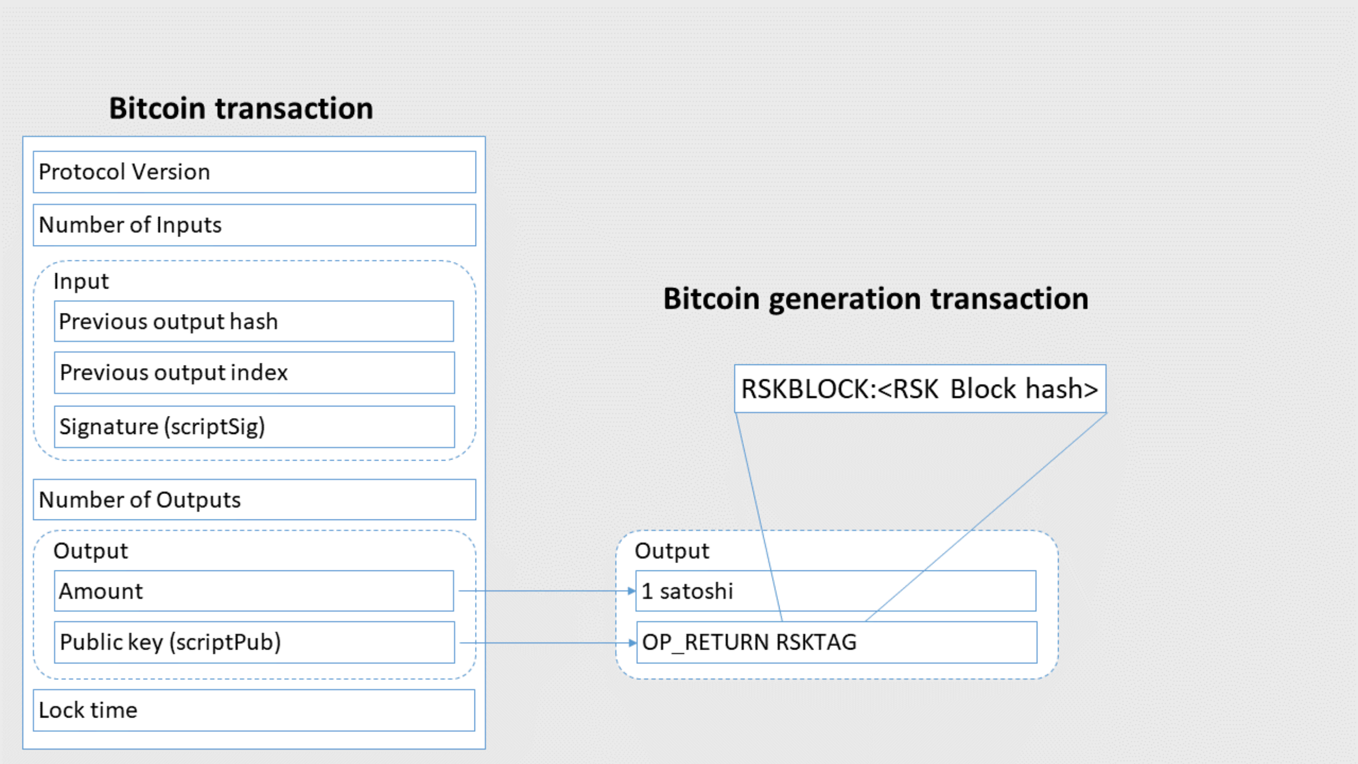
RSK tag Embedding

The RSK tag consists of the ASCII identifier "RSKBLOCK:" concatenated with a chunk of binary data that includes the hash digest of the RSK block header being mined. As previously stated, the tag must be identified without ambiguity (there should not be a way to create a Bitcoin block that can be associated with two different RSK blocks). While Namecoin defined a scheme for packing tags for different merge-mined blockchains, the proposed scheme was never standardized. Therefore, in order to enhance future compatibility and to allow more versatility for minerpool softwares, the RSK tag can be located anywhere in the generation transaction (the first transaction of a block, also sometimes called coinbase). The RSK tag can be located in the coinbase field of the generation transaction, or in any of the outputs of the generation transaction (generally as an OP_RETURN payload). The following diagrams show the two possibilities.

Scroll down



The RSK Tag Stored in the Coinbase Field



The RSK Tag Stored in One Output of the Generation Transaction

The Current RSK Tag Format

Scroll down

The current format of the RSK tag is: *RSKBLOCK:RskBlockHeaderHash*

“RSKBLOCK:” is the ASCII string consisting of the bytes: 52 53 4b 42 4c 4f 43 4b 3a.

RskBlockHeaderHash is the Keccak hash digest of the RSK Block header in binary format, without the merge-mining fields, which are filled after the PoW is solved.

When the RSK tag is included in an output script, it should be included after the OP_RETURN OP_PUSHDATA1 opcodes, to prevent spamming the Bitcoin UTXO, but this is not mandatory by consensus.

The following additional restrictions apply:

- The number of bytes immediately after RskBlockHeaderHash, up to the end of the coinbase transaction, must be lower than or equal to 128 bytes.
- The trailing raw bytes must not contain the binary string “RSKBLOCK:” (52 53 4b 42 4c 4f 43 4b 3a)
- If the RSK tag is located in a non-last output script, there exists a negligible probability of the RSK tag to appear by chance in the bytes of a following output. The poolserver software must not rule out the possibility of a rogue Bitcoin address included in the generation transaction having the tag embedded, and being used as an attack to break the validity of the merge-mined header. This could be a problem for decentralized pools (such as p2pool) that distributes revenues in the generation transaction itself. Therefore we recommend to using the last output script for the RSK tag.
- If the RSK tag is located in the coinbase field, there exists a negligible probability for the “RSKBLOCK:” to appear by chance in the ExtraNonce2 data field whose content is provided by the miner who solves the block,

as part of the Stratum protocol. Also, the miner could include the tag in the ExtraNonce2 maliciously. This is not a problem as long as the poolserver adds the RSKBLOCK: tag after the ExtraNonce2 chunk.

A standard P2SH output consumes 34 bytes so the trail length limitation (the maximum number of bytes past the tag) generally means that the tag must be located in the coinbase field or within the last 4 outputs of the coinbase transaction.

The trailing bytes restriction allows RSK full node to create a compressed SPV proof that consists of:

- The Bitcoin header (80 bytes)
- A Merkle Branch to the Coinbase transaction (approximately 320 bytes)
- A mid-state of SHA-256 consuming the head of the coinbase transaction (32 bytes)
- A 64 byte aligned chunk containing a trail of the coinbase transaction, including the RSK tag (max. 169 bytes). The use of a trail allows the protocol to use a proof that the trail belongs to the coinbase transaction as a free-start hash starting with the given mid-state.
- Currently the maximum size of an SPV merge-mining proof is 780 bytes.

The poolsever software can send to the rskj daemon the full block or this SPV proof. If rskj receives a block, it will parse it and extract the necessary fields to build the SPV proof.

The RskBlockHeaderHash is created by the standard RSK node (rskj daemon). Rskj exposes a mining RPC-JSON interface containing the command “getwork”. A poolserver plugin polls the rskj daemon and maintains the latest RskBlockHeaderHash value to provide to the poolserver. If a share hash is low enough so the RSK PoW puzzle is solved, the poolserver informs the plugin, which in turns informs to the rskj daemon.

Scroll down

Merge-Mining improvements in the Upcoming Network Upgrade (a.k.a Armadillo)

The upcoming release 1.0.0 features a network upgrade that improves the merge-mining tag format. The new format is specified by RSKIP110. The change adds additional information to RSK merge-mining tags so that users or automated systems can make informed decisions about the network health. Using tag monitoring tools, the RSK network nodes can also respond autonomously to abnormal situations in a way that protects nodes from double-spends, and also in the future nodes could broadcast succinct cryptographic proofs of anomalous state to other nodes of the network.

The 32-byte block header hash is replaced by a 32-byte byte array with the following format:

- 20-byte prefix of the hash-for-merge-mining (PREFIX)
- 7-byte Commit-to-Parents-Vector (CPV)
- 1-byte number of uncles in the last 32 blocks (NU), limited by 255.
- 4-byte Block Number (BN)

The four fields must be checked in consensus. In a nutshell, the CPV, NU and BN fields enable any node monitoring the Bitcoin blockchain to create a graph of the parallel competing chains of RSK blocks being built, even if not broadcasted to the RSK network. The hash of the RSK header is truncated to 20 bytes so that the tag still occupies 32 bytes, therefore maintaining compatibility with poolserver softwares.

RSK Merge-Mining Security

The theory of Nakamoto consensus using proof-of-work is based on thermodynamic and game theoretic security, not cryptographic security. RSK merge-mining is safe from any irrational attacker that can compute 2^80 hash operations in less than 30 seconds. A rational attacker would prefer to behave honestly and merge-mine an RSK block requiring only about 2^69 operations (the current difficulty of and RSK block) and being fully subsidized by Bitcoin, rather that to carry on other attacks. Setting aside the remote possibility of the composability of this attack with other merge-mined systems, an irrational attacker that attempts to perform 2^80 hashing operations would need to invest 2000 times more hardware than the rational miner, and his electricity consumption would not be subsidized. The investment would amount about five trillion dollars (5e12), assuming he uses hardware similar to state-of-the-art Bitcoin ASICs. But things are even worse for the irrational attacker, as he can only produce one or more blocks that share the proof of work for the same RSK block height, and we'll see later that this does not generally provide any monetary gain to the attacker (1). So that's five trillion dollars wasted. Therefore the 80-bit-security is well balanced across all components, and there is no component with substantially weaker security that can be a target of attack. To justify this, we'll shortly and informally review possible targets of attack.

First, RSK compresses the generation transaction with a non-standard cryptographic trick. Instead of providing the full generation transaction, it transmits only the tail. To still be able to produce the right cryptographic hash of this message, it starts hashing from a midstate of the Merkle–Damgård construction, instead of the initial state. This 64-byte midstate is transmitted along the tail. Securely using this trick requires assuming a stronger property from SHA256, known as "freestart collision" resistance, and we need this to be at least as secure as brute-forcing 80 bits, which is our target security threshold. SHA-1 (an already broken hash function) has a best known free-start attack that requires brute-forcing about 80 bits. No freestart collision has been found in SHA256, and the best results (2) correspond to fiding semi-free-start collisions in a reduced-round version of SHA256 (38 of the 64 rounds, at a cost of 2^65 operations), so we can conclude that RSK merge-mining is safe. It must be noted that a supposed free-start attack on SHA256 requiring 2^80 operations would be devastating for SHA256 and would render it useless as a secure standard hash function, and Bitcoin would be in trouble anyway. But if such attack is ever discovered, the RSK platform could be easily protected by a network upgrade not to use this cryptographic trick for SPV compression, at the expense of a small increase in block size. However, as we stated before, even if the attack would cost less than merge-mining in terms of bit-security, it would not be cost-efficient.

Scroll down

Second, with respect to the truncation of the hash digest in the upcoming 1.0.0 release, the new scheme provides the equivalent of 80-bit security for tag collisions. However, the attack is theoretically, economically and computationally irrational. From the computational complexity point of view, an 80-bit collision attack would require the use of an unrealistic amount of memory (3). Also, CPU cost of the collision attack is more than 2000 times higher than the cost of solving the RSK PoW puzzle (69 vs 80 bits).

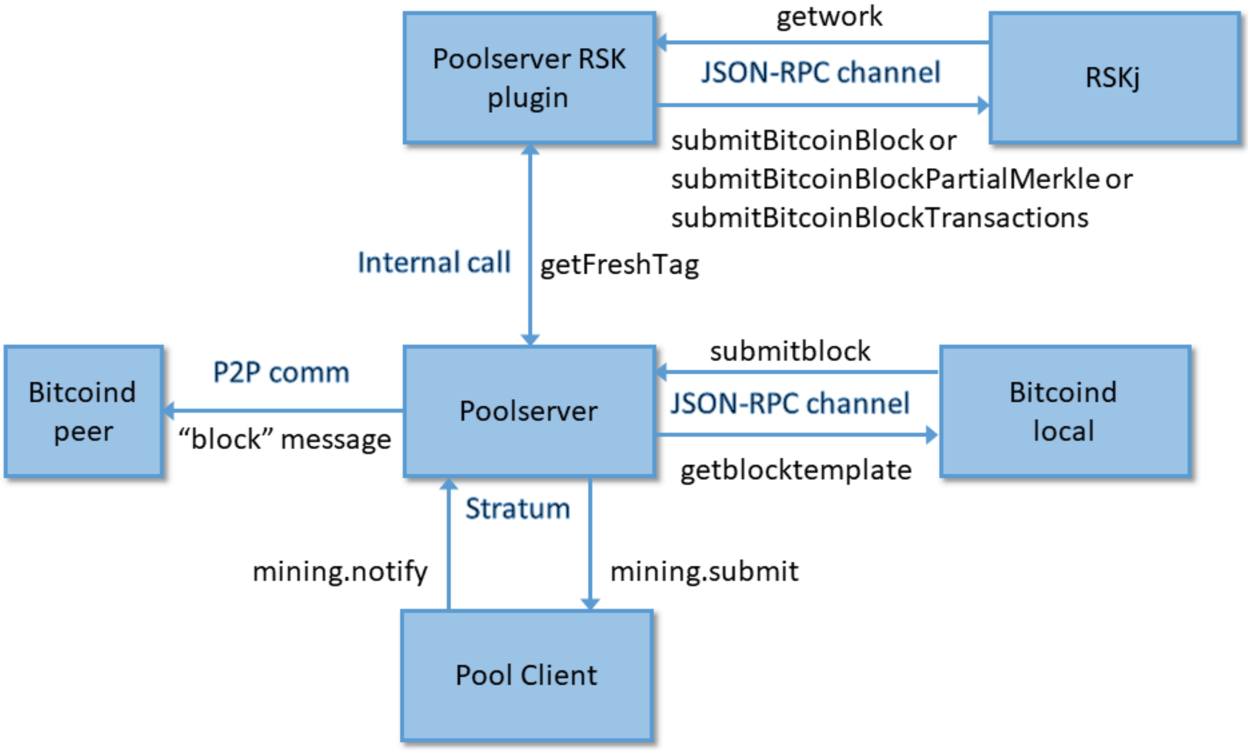
From the theoretical point of view and due to the DECOR+ consensus used in RSK, because the collision can only occur at the same RSK block height, the colliding blocks would share the block reward, so there is no benefit to find new siblings of past blocks if the past block reward is approximately equal to the reward of a new block. The attacker would be mainly competing with himself. For the same reason, there is no benefit to find collisions for new blocks, with the added difficulty that it must be performed within 30 seconds on average, instead of 5 minutes available to reference uncles.

From the economic point of view, an honest merge-miner earns Bitcoin transaction fees, so merge-mining is subsidized by Bitcoin, while the attacker has to pay the full cost of the collision attack. This makes almost any attack on the PoW linkage not cost-efficient. This is valid for both, hash digest collisions on the RSK block hash and the free-start collisions on the coinbase transaction hash.

Therefore we think the tag is secure for the next 20 years, even considering a breakthrough in computing efficiency. However, if computing trends radically change, a future network upgrade could easily expand back the size of the hash to the full 32 bytes.

RSK Merge Mining Plug-in Development

The following figure shows a standard poolserver architecture, including an RSK merge-mining plugin:



Poolserver with the RSK merge-mining plugin connected with the different networks

The poolserver RSK plugin communicates with the rskj daemon using two JSON-RPC methods: **mnr_getwork** and **mnr_submitBitcoinBlock**. Other submission methods enable sending less information to the rskj node, to reduce bandwidth consumption. The JSON-RPC connection port of the rskj daemon can be configured, and the default value is 4242. A description of the methods and the data exchange format can be found here: https://github.com/rsksmart/rskj/wiki/JSON-RPC-API. Those methods are implemented along with many others that belong to the Web3 interface, which is the standard interface to connect to the rskj node.

The RSK team has developed several fully working plugins for several pool implementations such as CoiniumServ, Ckpool, Btcpool, and Eloipool. Other pools have implemented their own plugins. We recommend Ckpool over other mining pool softwares as it is optimized and we extensively verified that merge-mining does not affect mining performance of Bitcoin mining.

Soon we'll open our guide for merge-mining plugin development with additional tips to help pools build efficient and reliable merge-mining modules.

It's important to note that because RSK uses the DECOR+ protocol, the poolserver software can keep processing all RSK PoW solutions found by the pool clients, even if the RSK parent block has changed. All competing blocks (called uncles) are rewarded.

Summary

Between 40% and 51% of Bitcoin miners are currently merge-mining RSK, which makes RSK the most secure smart-contract platform in the planet, in "thermodynamic" security (4). Merge mining RSK is easy, but care must be taken not to alter the normal functioning of the poolserver software, even under the most harsh conditions, such as direct attacks to the RSK network. RSK Labs developed several plugins that were extensively tested against a number of failure conditions, to verify the continuity of the mining operation. Also, efficiency measures were performed on mining pool softwares to attest the mining software efficiency is unaltered. We encourage pooled and solo miners to merge-mine RSK and take part of the Bitcoin smart-contract revolution. If you need help setting up your merge-mining operation, find us on RSK gitter community forum.

(1) Another interesting attack would be to use a tag collisions to create blocks for several competing chains in parallel, all with the same accumulated difficulty, and feed each other miner with a different “best chain” in order to prevent them to work in the same best chain. It requires having direct peer-to-peer connections with the other miners. It could work if the attacker has less hashing power than the remaining miners together but vast more hashing power than the remaining miners separated. Still this attack would be much more costly than just performing a 51% attack to ignore other miner’s blocks.

(2) https://eprint.iacr.org/2015/350.pdf

(3) The RAM memory required by a simple collision attack is 2^80 hash digests (32 million exabytes). Using the Oorschot/Wiener algorithm, and assuming a single processor can perform 2^40 sequential hash operations in less than 30 seconds, the RAM memory requirement could be reduced to 32 terabytes.

(4) Once a transaction is confirmed in a block it can’t be reversed without someone expending a minimum amount of energy to rewrite the chain. This energy is known as the “thermodynamic” security of a PoW-based blockchain.

Scroll down

Developers

Developer Portal
GitHub
Bounty Program
RSKIPs

Rootstock is the most secure smart contract network in the world and enables decentralized applications secured by the Bitcoin Network to empower people and improve the quality of life of millions.

Contact us.

Resources

Community
Ambassador
Help Desk
Block Explorer
Stats

White Paper

Original
Updated

Scroll down

Terms & Conditions

Privacy Policy

Rootstock (RSK) Public Key (69F6 F997 2497 8762 D541 8AE4 58D2 260D 5998 6758)