Scaling Definitions

1) Consensus (Old)

Define

Z_M = 300000 bytes ; Minimum penalty free zone.

M_B = Block weight in bytes.

M_L = The median over the last 100000 blocks of max((min (M_B , $1.7M_L$), Z_M , $M_L/1.7$) ; recursive calculation for M_L with M_L starting at M_L of previous 100001 block (currently = Z_M); Long term median

M_L ; Dynamic penalty free zone.

M_S = the median over the last 100 blocks of max(M_B , M_L) ; Effective short term median.

$M_N = \min(M_S , 50M_L)$; Median for Penalty calculation.

R_{Base} = Block Reward.

$0 < M_B \leq 2M_N$; Requirement for a valid block.

$B = M_B /M_N - 1$ where $-1 < B \leq 1$

$P_B = R_{Base}B^2$ for $B > 0$; Monero applies a penalty P_B, to increase the block weight by B; $P_B = 0$ for $B \leq 0$

Proposed Scaling Definitions (November 2025 update)

1) Consensus (New)

Define

Z_M = 1000000 bytes ; Minimum penalty free zone.

M_B = Block weight in bytes. For transaction weights see section 5.

M_L = The median over the last 100000 blocks of max((min (M_B , $2M_L$), Z_M , $M_L/2$) ; recursive calculation for M_L with M_L starting at M_L of previous 100001 block (currently = Z_M); Long term median

M_L ; Dynamic penalty free zone.

M_S = the median over the last 100 blocks of max(M_B , M_L) ; Effective short term median.

$M_N = M_S$; Median for Penalty calculation.

R_{Base} = Block Reward.

$0 < M_B \leq \min(2M_N , 16M_L)$; Requirement for valid block.

$B = M_B /M_N - 1$ where $-1 < B \leq 1$

$P_B = R_{Base}B^2$ for $B > 0$; Monero applies a penalty P_B, to increase the block weight by B; $P_B = 0$ for $B \leq 0$

Changes

1) The requirement for a valid block is now $0 < M_B \leq \min(2M_N , 16M_L)$. This caps the maximum M_B to $16M_L$ as opposed to $32M_L$. It also means that the maximum allowed M_B is reduced for $M_S > 8M_L$. For $M_S = 16 M_L$, M_B is capped by M_S
2) Maximum growth of M_S is reduced from $50M_L$ to $16M_L$ and maximum growth of M_B is reduced from $100M_L$ to $16 M_L$.
3) Rate of growth and decline of M_L is increased from 1.7x to 2x
4) Z_M is increased from 300000 bytes to 1000000 bytes

Note 1: (Optional) Transitional Hard Fork Keeping RingCT

The above can be implemented as a transitional stage 1 hard fork with the current RingCT using Z_M = 250000 bytes. On the stage 2 hard fork for FCMP++ the only consensus change that is then needed is to multiply the Z_M constant by 4. This allows for a 4000000 byte maximum on ramp to 50000 blocks to address the current code constants vulnerability.

2a) Minimum Fee For Node Relay (Old)

We add a, penalty attracting, transaction T with a size of T_T to a block of weight M_B

Define

T_R = 3000 bytes ; Reference Transaction weight. Note: T_R must be greater than T_2. T_2 equals the weight in bytes of a 2 input and 2 output transaction.

$B_T = T_T / M_N$

$P_{BT} = R_{Base}(B+B_T)^2 = R_{base}(B^2 + 2BB_T + B_T^2)$; The new penalty, where $B + B_T > 0$

$P_T = P_{BT} – P_B = R_{Base}(2BB_T + B_T^2)$; Increase in penalty from adding transaction T

$F_T = R_{Base}(2BB_T + B_T^2)$; The additional fee required to overcome the increase in penalty P_T

For the case B = 0 this reduces to $F_T = R_{Base}B_T^2$

$M_F = M_L$; Median for minimum fee calculation

To calculate the minimum fee we consider a transaction of weight T_R at the start of the penalty, B = 0 with $M_N = M_F$ 95% of the fee required to pay the penalty incurred is the minimum fee.

$B_{RL} = T_R / M_F$;

$F_R = R_{Base}B_{RL}^2$; Fee required to pay the penalty incurred

$f_R = R_{Base}B_{RL}/M_F$; Fee required to pay the penalty incurred per byte for a given M_F

$f_I = 0.95f_R$; Minimum fee per byte

2a) Minimum Fee For Node Relay (New)

We add a, penalty attracting, transaction T with a size of T_T to a block of weight M_B

Define

T_R = 10000 bytes ; Reference Transaction weight. Note: T_R must be greater than T_2. T_2 equals the weight in bytes of a 2 input and 2 output transaction.

$B_T = T_T / M_N$

$P_{BT} = R_{Base}(B+B_T)^2 = R_{base}(B^2 + 2BB_T + B_T^2)$; The new penalty, where $B + B_T > 0$

$P_T = P_{BT} – P_B = R_{Base}(2BB_T + B_T^2)$; Increase in penalty from adding transaction T

$F_T = R_{Base}(2BB_T + B_T^2)$; The additional fee required to overcome the increase in penalty P_T

For the case B = 0 this reduces to $F_T = R_{Base}B_T^2$

$M_F = M_L$; Median for minimum fee calculation

To calculate the minimum fee we consider a transaction of weight T_R at the start of the penalty, B = 0 with $M_N = M_F$ 100% of the fee required to pay the penalty incurred is the minimum fee.

$B_{RL} = T_R / M_F$;

$F_{RL} = R_{Base}B_{RL}^2$; Fee required to pay the penalty incurred

$f_{RL} = R_{Base}B_{RL}/M_F$; Fee required to pay the penalty incurred per byte for a given M_F

$f_{IL} = f_{RL}$; Minimum fee per byte

Note 1: Transitional Hard Fork (Implications)

T_R = 2500 bytes or 5000 byes if note 2 below is chosen for FCMP++.

Note 2: (Optional) Double fees

Increase T_R to 20000 bytes.

Note 3: (Optional): Additional, minimum fee, per byte for large transactions.

$f_{IN} = 4f_{IL}$, If a transaction has a weight T_T > 20000 bytes and / or more than 8 inputs.

Changes

1) T_R is increased from 3000 bytes to 10000 bytes. For clarity T_R is defined outside of consensus.
2) There is no reduction to 95% of the minimum fee. This is all now handled on the wallet side.

2b) Wallet Fees (Old)

For the calculation of wallet fees we assume that the next 10 blocks have no transactions, other than the coinbase transaction, the empty blocks, We then calculate M_{LW} and M_{SW} by following the calculation of M_L and M_S at this future point. We use the previous 99990 blocks and the future 10 empty blocks (100000 blocks) for M_L and the previous 90 blocks and future 10 blocks (100) blocks for M_S.

Define

$M_{BW} = M_B$ for the last 99990 blocks
$M_{BW} = 0$ for the future 10 blocks; A value of 0 bytes can be used for the empty blocks for the purposes of calculating M_{LW}.

M_{LW} = The median over the last 99990 blocks and future 10 blocks (100000 blocks) of max((min (M_{BW} , $2M_L$), Z_M , $M_L/2$) ; The current value of M_L from consensus is used; Effective long term median for wallet fees

M_{LW} ; Penalty free zone for wallet fees

M_{SW} = the median over the last 90 blocks and future 10 blocks (100 blocks) of max(M_{BW} , M_{LW}); Effective short term median for wallet fees

$M_{NW} = \min(M_{SW}, 50M_{LW})$

$M_{FW} = M_{LW}$; Median for wallet fee calculation

$B_{RLW} = T_R / M_{FW}$; Used for the low and normal fees

$B_R = T_R / Z_M$; Used for the medium and high fees

$F_L = R_{Base}B_{RLW}^2$; Low transaction fee for reference transaction

$f_L = R_{Base}B_{RLW}/M_{FW}$; Low transaction fee per byte for a given M_{FW}

$f_N = 4f_L$;Normal transaction fee per byte for a given M_{FW}

$f_M = 16\, R_{Base}B_R/M_{FW}$; Medium Transaction fee per byte for a given M_{FW}

$f_P = 2R_{Base}/M_{NW} = f_M M_{FW}/(8B_R M_{NW})$; Maximum Penalty (B =1) Transaction fee per byte for a given M_{NW}

$f_H = 4f_M \max(1, M_{FW}/(32B_R M_{NW}))$; High Transaction fee per byte

2b) Wallet Fees (New)

For the calculation of wallet fees we assume that the next 1000 blocks have no transactions, other than the coinbase transaction, the empty blocks, We then calculate M_{LW} by following the calculation of M_L at this future point. We use the previous 99000 blocks and the future 1000 empty blocks (100000 blocks) for M_L.

Define

$M_{BW} = M_B$ for the last 99000 blocks
$M_{BW} = 0$ for the future 1000 blocks; A value of 0 bytes can be used for the empty blocks for the purposes of calculating M_{LW}.

M_{LW} = The median over the last 99000 blocks and future 1000 blocks (100000 blocks) of max((min (M_{BW} , $2M_L$), Z_M , $M_L/2$) ; The current value of M_L from consensus is used; Effective long term median for wallet fees

M_{LW} ; Penalty free zone for wallet fees

$M_{FW} = M_{LW}$; Median for wallet fee calculation

$B_{RLW} = T_R / M_{FW}$; Used for the low, normal and medium fees

$B_R = T_R / Z_M$; Used for the high and maximum fees

$F_L = R_{Base}B_{RLW}^2$; Low transaction fee for reference transaction

$f_L = R_{Base}B_{RLW}/M_{FW}$; Low transaction fee per byte for a given M_{FW}

$f_N = 4f_L$; Normal transaction fee per byte for a given M_{FW}

$f_M = 16f_L$; Medium Transaction fee per byte for a given M_{FW}

$f_H = 64\, R_{Base}B_R/M_{FW}$; High Transaction fee per byte for a given M_{FW}

$f_X = \max(2R_{Base}/M_{FW}, 4f_H)$ Maimum transaction fee per byte. This is greater than or equal to Maximum Penalty (B =1) Transaction fee per byte for a given M_{FW}

Changes

1) All fees including the high fee are now based upon M_{LW} with the ratio between fees constant for a given M_{LW}.
2) The grace period is increased to 1000 blocks.
3) 5 fee levels with at least a 4x factor between fees.

3) Transitional considerations for Minimum penalty free zone, Z_M, and Median calculations after the fork.

Define:

Z_{MOld} = 300000 bytes (Z_M before hard fork)

M_{BOld} = Block Weight in bytes (before hard fork)

3) Transitional considerations for Minimum penalty free zone, Z_M, and Median calculations after the fork.

Calculation of M_L, M_S, M_{LW} and M_{SW} where blocks from a previous the Monero version are included in a calculation after the fork. M_B is modified as follows:

Define

Z_M = 1000000 bytes

For blocks before the hard fork
$M_B = M_{BOld} (Z_M / Z_{MOld})$

The medians are then calculated normally.

Note 1: Transitional Hard Fork (Implications)

Apply the transitional considerations for each hard fork:

1) Z_{MOld} = 300000 bytes, Z_M = 250000 bytes
2) Z_{MOld} = 250000 bytes, Z_M = 1000000 bytes

4) Wallet Fee Rounding

Wallet fees, f_N, f_L, f_M, and f_H are rounded up to the desired number of significant digits in the significant

Wallet Fee Rounding Examples

Two significant digits

27810	Rounded to : 28000
37.94	Rounded to : 38
0.5555	Rounded to : 0.56
0.002342	Rounded to : 0.0024

4) Wallet Fee Rounding

Wallet fees, f_N, f_L, f_M, f_H, and f_X are rounded to 2 significant digits in the significant

Wallet Fee Rounding Examples

Two significant digits

27810	Rounded to : 28000
37.94	Rounded to : 38
0.5555	Rounded to : 0.56
0.002342	Rounded to : 0.0023

5) Transaction Weights

Transaction weighs are used to account for the different growth rate the output proof verification time with the number of outputs. This is done at consensus level and can lead to double charging if the fee per byte rate is increased because of an increase in the transaction weight.

5) Transaction Weights (Proposed)

Breakdown of the transaction weights as follows:

Use a standard weights roughly based upon the current size in bytes. Calculation based upon the number of inputs and 2 outputs. Then three additional weights are added to the 2 output weights for outputs greater than 2 as follows:

a) 3 or 4 outputs: The additional weight added is based upon the average increase in size over inputs of 4 outputs over 2 outputs.

b) 5, 6, 7 or 8 outputs: The additional weight added is based upon the average increase in size over inputs of 8 outputs over 2 outputs.

c) 9, 10, 11, 12, 13, 14, 15, 16 outputs: The additional weight added is the average increase in size over inputs of 16 outputs over 2 outputs.

A standard weight for the fee is implicit. This is minimal and will avoid the need to do recursive wallet fee calclations.

d) TX extra will have a weight equal to its size, that is added on.

The weights will be Itemized in a spreadsheet to follow.