Subject: The Cat, BIP discussion.

Dear Bitcoin-dev,

Given recent discussions surrounding the malincentives of spam and the perceived futility in addressing these issues, I felt it necessary to propose a working solution. For those interested, I have written a brief history of the problem; please skip to “The BIP” below if you are only interested in the proposal.

Bitcoin is a peer to peer open source monetary network created to facilitate online payments between individuals without trust in a third party. By its nature as an open protocol, and its censorship-resistant design with no single authority maintaining its operation, we have to rely on incentives and culture for network stewardship. Historically, we saw many trends emerge which threatened Bitcoin’s primary use as a monetary network. They were successfully dealt with by introducing OP_Return with a small limit as a more efficient and limited way of directing non-monetary data to a provably non-spendable space that could more easily be pruned from nodes and, importantly, not pollute the UTXO set.

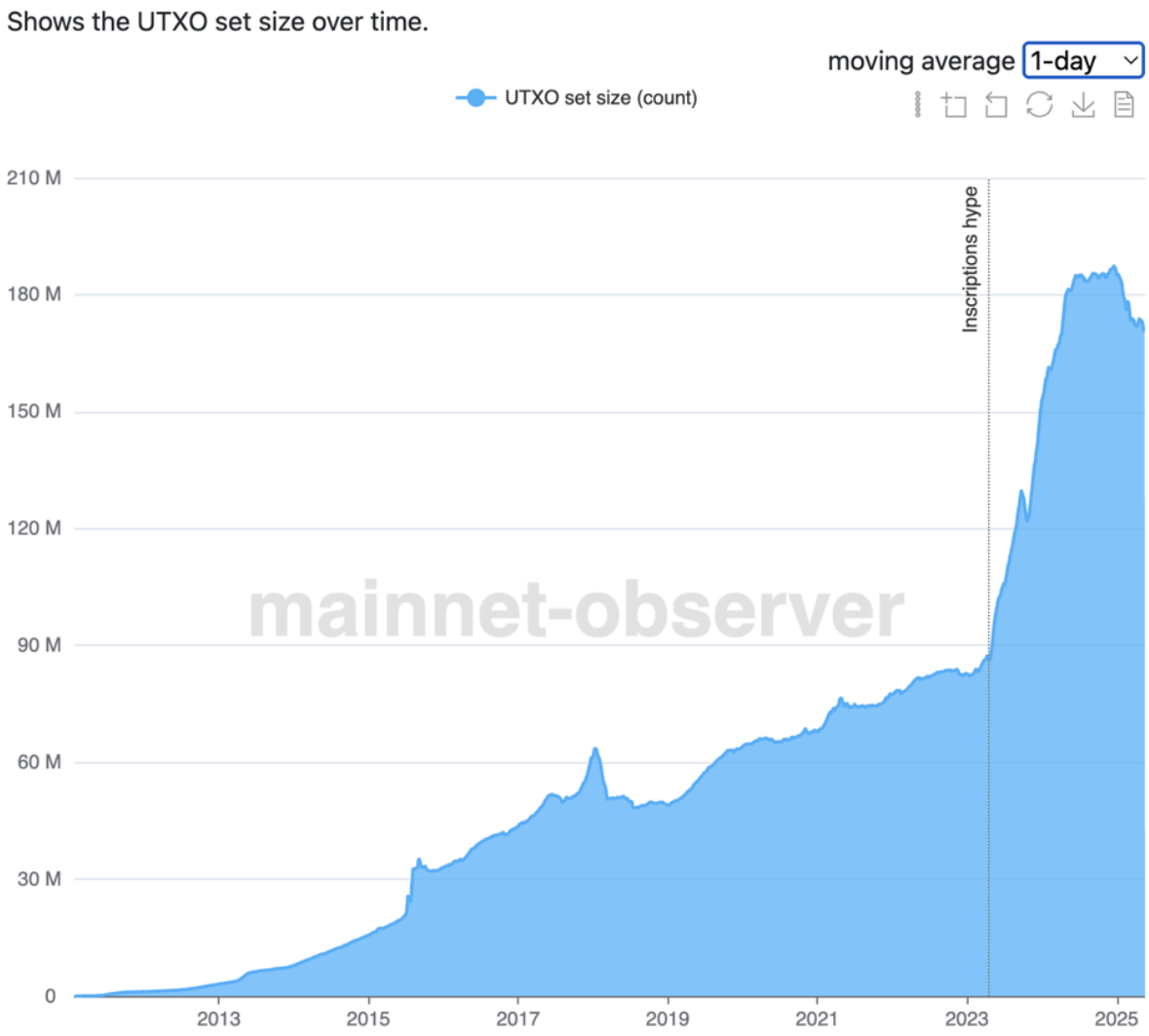
Legendary Bitcoin developer Gregory Maxwell said at the time, ***“Part of the idea here is shaping behavior towards conservative needs.”***

This was enforced through standardness filters, which are a way of discouraging certain types of transactions that, while technically consensus valid, we have agreed as a community are usually not used in standard practice and are potentially harmful.

In recent years, these trends have come back into focus with the creation of schemes like Ordinal theory and its associated inscriptions, and Bitcoin Stamps, which are techniques for embedding non-monetary data (like images or tokens) into Bitcoin transactions. Ordinals typically hide data in the Taproot witness field (benefiting from a weight “discount”), while Stamps encode data in unspendable outputs (e.g., fake bare multisig addresses). These practices turn the blockchain into a data storage system rather than purely a payments network. As anyone familiar with basic economics knows, what you subsidize, you get more of.

Critically, many Ordinal inscription and Stamp transactions create dust outputs (tiny UTXOs of only a few hundred sats) that remain unspent indefinitely, bloating the UTXO set. Because Stamp outputs are expected to remain unspent indefinitely, they persist in the UTXO set forever.

Veteran Bitcoin developer Mark “Murch” Erhardt has described the stamps UTXO issue as ***“probably, from a technical perspective, one of the more egregious uses of blockchain."***



Over roughly 14 years (2009-early 2023), Bitcoin's UTXO set grew gradually to around 80-90 million entries. Then, in less than a year, it doubled to more than 160 million by late 2023, a historic anomaly driven largely by the Ordinals and Stamps craze. Analyses suggest that by mid-2025, over 30% of all UTXOs were tied to Ordinal inscriptions. Nearly half of all UTXOs (around 49%) now contain less than 1,000 satoshis, strongly indicating they are spammy dust outputs used for data embedding rather than normal economic activity. Many of these are Taproot outputs or outputs sitting exactly at the standard dust threshold, consistent with algorithmic spam creation. In short, UTXO spam now accounts for tens of millions of entries and represents an unprecedented explosion of UTXO bloat.

The UTXO database (chainstate) has ballooned alongside this spam. Before 2023, the chainstate was on the order of 4-5 GB; by early 2024 it exceeded 11 GB, meaning the disk footprint required to hold the unspent set roughly doubled in about a year, in line with Ordinals and BRC-20 mania. Over the same period, the full blockchain grew by about 93 GB in 2023 alone, versus roughly 55 GB per year previously, largely due to inscription data filling blocks. While pruned nodes can discard old block data, they cannot discard UTXOs: every unspent

output, even spam, must remain in the chainstate until it is spent or explicitly removed by a protocol change. This makes UTXO spam a permanent, compounding burden on full nodes. As Bitcoin developers have noted, techniques that deliberately embed data in UTXOs are among the most egregious abuses of the system, precisely because they clutter the UTXO set with junk that cannot be pruned away under current rules.

Bitcoin's design includes some anti-spam measures, but spammers have continually evolved "cat-and-mouse" tactics to bypass them.

This is where our proposal comes in.

THE BIP: The Cat.

Recent attempts to reduce spam have focused on the supply side, trying to stop spam through policy and standardness rules. These efforts have had limited success. The Cat instead looks to economics, removing or reducing the financial incentive for creating spam outputs in the first place. It targets the demand side by making designated Non Monetary UTXOs (NMUs) permanently unspendable, which in turn reduces market demand for those assets.

The Cat defines a fixed, reproducible set of "NMUs" using established external indexers, and makes them permanently unspendable by consensus, so that such data cannot be monetized or transacted, only archived. Once rendered unspendable, we will be removing these UTXOs made unspendable by consensus under this proposal, from the UTXO set, materially reducing the current resource requirement for nodes (likely on the order of tens of millions of UTXOs, roughly ~30% of the set, subject to measurement). The NMU classification itself is encoded in a compact membership structure (a Binary Fuse Filter plus a false-positive exclusion list) that ships with the binary and does not require any node to reindex or re-download the chain.

Draft Specification

Status: Discussion draft. A formal BIP, compliant with BIP 0003 and the usual BIP process, will be prepared if there is interest in pursuing this direction.

1. Definitions

Non-Monetary UTXO (NMU):

A UTXO that is classified as containing inscription-style or stamp-style data according to the procedure in §2 and that satisfies the value and creation-height constraints below.

NMU bit:

A single Boolean flag (0 or 1) stored alongside each UTXO in the node's UTXO database:

- NMU = 1 -> UTXO is non-monetary and may not be spent.

- NMU = 0 -> UTXO is monetary (normal behavior).

Conceptually, NMU = 1 means “this outpoint is in NMUSet_snap as defined by this BIP.” Implementations MAY choose to compute this bit on the fly from the membership structure rather than store it explicitly; see §3.

NMU value threshold:

A consensus constant VALUE_MAX_NMU = 1000 satoshis. Only UTXOs with value strictly less than VALUE_MAX_NMU are eligible to be classified as NMU. UTXOs with value greater than or equal to VALUE_MAX_NMU MUST be treated as monetary (NMU = 0) for all time under this BIP, even if external tools associate inscriptions or stamps with some of their satoshis.

NMU creation-height window:

This proposal intentionally restricts NMU classification to UTXOs created between:

H_min_NMU: the earliest block height at which the reference indexers (Ord 0.24.0 and the specified Stamps version) first recognize any inscription-style or stamp-style NMU, and

H_snap: the snapshot height defined in §2.4.

Both H_min_NMU and H_snap are consensus constants fixed before activation. Any UTXO created at height < H_min_NMU or > H_snap MUST be treated as monetary (NMU = 0) regardless of filter results.

NMU membership structure (NMU_DATA):

A static, exact membership structure that encodes NMUSet_snap using:

- A Binary Fuse Filter (BFF-8) constructed from all eligible NMU outpoints; and

- A false positive exclusion list (FP_EXCLUDE) that contains all non-NMU outpoints at H_snap that would otherwise match the filter.

NMU_DATA is distributed as a single binary blob, together with a consensus constant NMU_DATA_HASH = SHA256d(NMU_DATA) which all compliant implementations MUST verify before using it for consensus validation (see §3.3).

Activation height (H_cat):

A consensus constant H_cat, the block height at which the NMU spending rule in §4 first becomes eligible for enforcement. The exact value and deployment mechanism for H_cat are out of scope for this draft and would be specified in a formal BIP.

2. NMU classification (reproducible list)

The normative consensus object introduced by this BIP is the set NMUSet_snap: a fixed set of outpoints classified as non-monetary at snapshot height H_snap on the best chain, after applying the value and height restrictions in §1 and §2.3.

Consensus enforcement of The Cat only depends on whether an outpoint is in NMUSet_snap or not; it does not depend on how any particular node obtained that set.

To define NMUSet_snap in a precise and reproducible way, this BIP specifies two reference classification rule sets that operate purely on Bitcoin chain data, as implemented in:

- Ord 0.24.0 at commit 5d2fbbe64b362cd6c30d6901e50cbe80084761f8

- Stamps - [exact version/commit to be pinned before any activation]

These reference implementations define the inscription-style and stamp-style classification rules in terms of Bitcoin chain data. They are cited here as normative descriptions of how to decide whether a given outpoint carries an inscription or a stamp at height H_snap.

Implementations that wish to independently derive or verify NMUSet_snap MAY:

- Execute these reference implementations from genesis up to H_snap, or

- Independently re-implement the same classification rules and confirm that the resulting set of outpoints matches the NMU set implied by NMU_DATA.

Running Ord or Stamps is not required for ordinary full-node operation under this BIP; they are simply the canonical specification of the classification logic.

This section uses the consensus constants H_cat, H_snap, and H_min_NMU as defined in §1.

2.1 Ord NMU set

According to the classification rules implemented in Ord 0.24.0 at commit 5d2fbbe64b362cd6c30d6901e50cbe80084761f8, witness data is parsed for “inscription envelopes” and each recognized inscription is associated with a particular outpoint (txid, vout) on the best chain at height H_snap.

Define OrdNMUSet as the set of all outpoints that, when those Ord 0.24.0 rules are applied to the best chain up to H_snap, are classified as carrying inscriptions at H_snap. Concretely, for each inscription that Ord 0.24.0 would associate with a specific outpoint (txid, vout) on that chain, that outpoint is included in OrdNMUSet.

This includes, for example, inscription UTXOs used by protocols such as BRC-20, but the BIP makes no protocol-level distinction; they are all treated as “NMU” if they meet the value and height thresholds in §2.3.

An implementation may obtain OrdNMUSet by actually running Ord 0.24.0, or by re-implementing its published classification rules and applying them to the chain.

2.2 Stamps NMU set

According to the classification rules implemented in the specified BTC Stamps indexer (version/commit TBD), Counterparty metadata and Bitcoin transactions are parsed to identify “stamp-style” embedded assets and associate them with one or more bare multisig outputs on the best chain at height H_snap.

Define StampsNMUSet as the set of all outpoints that, when those Stamps rules are applied to the best chain up to H_snap, are classified as containing a stamp at H_snap. For each stamp that the reference Stamps implementation would associate with a specific outpoint (txid, vout) on that chain, that outpoint is included in StampsNMUSet.

As with Ord, an implementation may obtain StampsNMUSet either by running the reference Stamps indexer or by independently implementing the same classification rules and applying them to the chain.

2.3 Combined NMU set + value & height thresholds

The initial raw NMU set at snapshot height is:

$$\text{NMUSet\_raw} = \text{OrdNMUSet} \cup \text{StampsNMUSet}$$

Apply both the NMU value threshold and the creation-height window:

$$\text{NMUSet\_snap} = \{ u \in \text{NMUSet\_raw} \mid \text{value}(u) < \text{VALUE\_MAX\_NMU} \text{ and } H_{min\_NMU} \leq \text{height}(u) \leq H_{snap} \}$$

where value(u) is the number of satoshis in UTXO u, and height(u) is the height of the block that created that UTXO.

Every UTXO in NMUSet_snap MUST be treated as having NMU = 1 under the new consensus rule (see §4). Nodes MAY realize this either by pre-seeding an NMU bit in their UTXO database or by computing it dynamically via the membership query in §3.3.

Notes:

Running Ord or Stamps is only required for those who wish to independently derive and verify NMUSet_snap; it is not a requirement for ordinary full node operation under The Cat.

The value threshold is intended to avoid classifying large, mixed-use UTXOs as NMUs when a small number of inscribed satoshis have been combined with otherwise monetary funds.

Restricting to height(u) ≥ H_min_NMU reflects the historical fact that these protocols appeared only after a certain point, and it trivially avoids misclassification of very old UTXOs.

This proposal intentionally does not specify any new in-client classification rule for discovering future NMU formats. It explicitly targets non-monetary UTXOs identifiable via Ord 0.24.0 and Stamps at the time of snapshot.

2.4 Snapshot block commitment and reorg behaviour

This BIP commits to a specific chain history at the snapshot height H_snap by introducing a new consensus constant:

SNAP_BLOCK_HASH: a 32-byte value equal to the block hash at height H_snap on the chain from which NMUSet_snap was originally derived.

A node MUST only enforce the NMU consensus rule from §4 if, for its current best chain:

There exists a block at height H_snap, and

The hash of that block is exactly equal to SNAP_BLOCK_HASH.

If, for the node's current best chain, the block at height H_snap has a different hash, the node MUST treat all UTXOs as monetary (NMU = 0) for the purposes of consensus validation, and MUST NOT reject any transaction or block on the basis of the NMU rule.

In particular, if a reorganization occurs that replaces the block at height H_snap with a different block (i.e., a reorg deeper than H_snap), enforcement of The Cat is automatically disabled until and unless the best chain once again has SNAP_BLOCK_HASH at height H_snap. Any desire to apply NMU classification to a different chain history would require a new consensus commitment (for example, a new SNAP_BLOCK_HASH defined by a subsequent BIP and activation process).

3. NMU bit storage, membership structure, and activation

3.1 NMU bit storage model

Consensus only cares about which UTXOs are NMU, not how the bit is stored or computed. Implementations MAY:

- Store the NMU bit as an extra bit in the UTXO database (e.g., in the Coin structure).
- Store NMUs in a separate structure keyed by outpoint.
- Compute NMU on the fly at validation time using the NMU_DATA membership structure from §3.3, optionally caching results.

The only consensus requirement is: given the same chain, compliant implementations must converge on the same NMU classification for all UTXOs. Conceptually, for a UTXO u with value(u) and height(u), NMU(u) is defined as:

- 1 if:
    - H_min_NMU ≤ height(u) ≤ H_snap, and
    - value(u) < VALUE_MAX_NMU, and
    - u ∈ NMUSet_snap (as tested via §3.3),
- 0 otherwise.

3.2 Activation

At activation height H_cat:

Nodes must have:

- Fully validated the chain up to at least H_snap.
- Loaded and verified NMU_DATA (see §3.3).

From the first block at or after H_cat for which the snapshot block commitment condition in §2.4 holds, nodes MUST enforce the consensus rule in §4, using the NMU classification defined above. No reindex or UTXO rewrite is required for activation; NMU classification can be applied as an overlay on top of the existing UTXO database.

Implementations MAY, as an optimization, perform a one-time pass over their UTXO set after activation to:

- Set Coin.NMU = 1 for all UTXOs that satisfy the NMU predicate; and/or

- Physically delete such UTXOs from the UTXO database (see §5).

Such a pass is a local optimization and not required for consensus, provided the on-the-fly NMU classification via NMU_DATA would yield the same results.

3.3 NMU membership structure: Binary Fuse Filter + exclusion list

This BIP does not require nodes to redo initial block download or reindex the chain in order to enforce NMU classification. Instead, it commits to a canonical membership structure NMU_DATA which encodes NMUSet_snap in a compact form.

3.3.1 Canonical outpoint encoding

For all purposes related to NMU_DATA (filter construction, querying, and the exclusion list), an outpoint (txid, vout) is encoded as a fixed 36-byte key.

txid_le (bytes 0–31): the 32-byte transaction ID in little-endian order, matching the internal uint256 representation used by Bitcoin Core and similar implementations.
vout_le (bytes 32–35): the 4-byte output index, serialized as a little-endian unsigned 32-bit integer (e.g. 0 -> 00 00 00 00, 1 -> 01 00 00 00, 256 -> 00 01 00 00).

This encoding is normative for NMU_DATA. All compliant implementations MUST use exactly this encoding when querying the filter or the exclusion list.

3.3.2 NMU_DATA serialization

NMU_DATA is a single binary blob. All integer fields are little-endian and appear in the following order:

- MAGIC (4 bytes): ASCII string "NMU1".
- VERSION (1 byte): format version (initially 0x01).
- SNAP_HEIGHT (4 bytes): block height H_snap.
- SNAP_HASH (32 bytes): block hash at H_snap (must match SNAP_BLOCK_HASH from section §2.4).
- FILTER_SEED (8 bytes): 64-bit seed used by the Binary Fuse Filter hash functions.
- FILTER_LEN (4 bytes): length in bytes of FILTER_DATA.
- FILTER_DATA (FILTER_LEN bytes): raw Binary Fuse Filter data.
- FP_COUNT (4 bytes): number of false-positive entries.

- FP_ENTRIES (FP_COUNT × 36 bytes): lexicographically sorted array of canonical 36-byte outpoint encodings that are not in NMUSet_snap but would otherwise match the filter.
- CHECKSUM (32 bytes): SHA256d of all preceding bytes from MAGIC through the last FP_ENTRY.

The SHA256d of the entire NMU_DATA blob is called NMU_DATA_HASH and is treated as a consensus constant compiled into node software.

A node MUST NOT enforce the NMU consensus rule unless NMU_DATA passes its internal CHECKSUM validation and SHA256d(NMU_DATA) equals NMU_DATA_HASH.

3.3.3 Binary Fuse Filter query

The filter is a static Binary Fuse Filter over 36-byte keys with 8-bit fingerprints, giving an approximate false-positive rate of about 1/256.

For a given key, a deterministic hash function derived from FILTER_SEED maps the 36-byte key to three byte positions inside FILTER_DATA and to an 8-bit fingerprint. A membership query reads the three bytes at those positions, XORs them, and compares the result to the fingerprint. If they match, the filter reports the key as “probably present”.

The exact mapping from a 36-byte key to three positions and an 8-bit fingerprint, and the interpretation of FILTER_DATA, MUST follow a single, fully specified algorithm (for example, a pinned “binary fuse filter” reference implementation). How FILTER_DATA was constructed is not consensus-critical; only the query algorithm and the fixed contents of NMU_DATA are consensus-critical.

3.3.4 False positive exclusion list

Because the Binary Fuse Filter is probabilistic, some UTXOs that are not in NMUSet_snap will still be reported as present at H_snap. During snapshot construction, the false positive exclusion list FP_EXCLUDE is computed by enumerating all UTXOs at H_snap that are not in NMUSet_snap, testing each with the filter, collecting those that match, and sorting their 36-byte keys lexicographically to form FP_ENTRIES.

At runtime, FP_EXCLUDE is treated as an override: if an outpoint’s canonical key appears in FP_EXCLUDE, that outpoint MUST be treated as non-NMU, regardless of the filter result.

3.3.5 Membership query (is_nmu)

Given a UTXO u at outpoint (txid, vout), with value(u) and height(u), nodes evaluate the NMU membership predicate is_nmu(u) as follows.

First, UTXOs outside the NMU window are never NMUs. If height(u) is less than H_min_NMU or greater than H_snap, is_nmu(u) is false.

Second, very large UTXOs are always treated as monetary. If value(u) is greater than or equal to VALUE_MAX_NMU, is_nmu(u) is false.

For the remaining UTXOs, the node computes key = canonical_encode(txid, vout) using the 36-byte encoding defined in section 3.3.1. If key appears in FP_EXCLUDE (using binary search over the sorted FP_ENTRIES array), is_nmu(u) is false. Otherwise, the node queries the Binary Fuse Filter with key. If and only if the filter reports the key as present, is_nmu(u) is true.

Within the domain H_min_NMU ≤ height(u) ≤ H_snap and value(u) < VALUE_MAX_NMU, this predicate exactly matches membership in NMUSet_snap: every UTXO in NMUSet_snap is reported as NMU, and every UTXO outside NMUSet_snap is reported as non-NMU, with no false positives and no false negatives.

4. New consensus rule

The rule in this section is enforced only when the snapshot block commitment condition from §2.4 holds and NMU_DATA has been successfully verified.

From the first block at or after H_cat that follows activation:

Rule: Forbidden spending of NMUs

A transaction is invalid if any of its inputs refers to an outpoint (txid, vout) such that the corresponding UTXO u satisfies is_nmu(u) = true as defined in §3.3.5 (equivalently, has NMU = 1).

Formally, when validating a transaction input:

- Let (txid, vout) be the input outpoint.
- Let u = Coin(txid, vout) be the referenced UTXO as seen by the node at the time of validation.
- If u does not exist: reject as usual (unchanged behavior).
- Otherwise, if is_nmu(u) = true, then the transaction MUST be rejected as invalid.

This rule applies:

- To mempool acceptance.

- To block validation.

- To all future reorg scenarios, provided the snapshot commitment in §2.4 remains satisfied.

5. UTXO set removal and pruning

Because NMUs are permanently unspendable after activation, they can be dropped from the spendable UTXO set.

5.1 Logical removal

Implementations MUST treat UTXOs with is_nmu(u) = true (or NMU = 1) as non-existent for the purpose of:

- Satisfying transaction inputs (they can never be used).

- Fee calculations, coin selection, wallet balances, etc.

5.2 Physical pruning (optional)

Nodes that operate with -prune=1 (or similar backwards-compatible pruning configuration) MAY:

- Omit NMUs from the on-disk UTXO set.

- Remove any previously persisted NMUs during a compaction / cleanup pass by running is_nmu(u) on each UTXO and deleting those that return true.

This proposal does not require pruned nodes to discard raw historical blocks. It only authorizes pruning of UTXO entries that are permanently unspendable due to the NMU rule.

Non-pruned nodes (full archival nodes) MAY continue to store historical blocks and full transaction data as usual. This preserves archival access to inscription / stamp data while still neutralizing it economically.

Rationale

This BIP makes minimal changes to the consensus surface. It adds no new opcodes and does not expand Bitcoin's scripting language or programmability. Instead, it introduces a single new consensus concept: a binary classification that marks some existing UTXOs as permanently unspendable Non-Monetary UTXOs (NMUs). In that sense, it is closer to a one-time reclassification of a subset of UTXOs than an ongoing change to Bitcoin's programming model.

External indexers, determinism, and reproducibility

Instead of re-implementing complex inscription and stamp classification rules in Bitcoin clients, this BIP relies on mature tools that are already used by the non-monetary data community, specifically Ord and Stamps. Exact versions of these indexers are specified and pinned by commit so that their behavior is deterministic. The NMU set is defined in terms of chain data as seen through these specified indexers, rather than as an arbitrary hard-coded list of UTXOs. This keeps the consensus rule grounded in chain-derived information while avoiding the need to embed inscription logic directly in Bitcoin client software.

Fixed snapshot vs future formats

By scoping the NMU classification to what Ord 0.24.0 and Stamps (at a pinned commit) can see at the snapshot height H_snap, this proposal neutralizes the existing inscription and stamp economy on day one without importing evolving, open-ended classification rules into consensus. It treats the snapshot as a fixed historical boundary and leaves future NMU formats and mitigation strategies as a separate question for policy or for future BIPs. If further action is desired on post-snapshot NMU formats, the community can consider additional updates or separate proposals without expanding the scope of this one.

UTXO set size and overhead

Because all NMUs become permanently unspendable under this proposal, they can be removed from the UTXO set, which reduces its size. The new storage overhead introduced by The Cat is dominated by the global membership structure NMU_DATA, which consists of a Binary Fuse Filter over NMUSet_snap plus a correction list of false positives. For a plausible NMU count on the order of 50 million, the filter occupies roughly 9 bits per element (≈ 56 MB), and the exclusion list adds around 15–20 MB before compression, for an uncompressed size on the order of 70–80 MB and an expected compressed size of roughly 40–50 MB.

This is small relative to typical UTXO set sizes and to the on-disk savings from pruning tens of millions of dust-sized NMUs. Per-UTXO storage overhead remains a single NMU bit if implementations choose to materialize it in the database; the rest of the classification data is stored once per node.

Censorship resistance and scope

Bitcoin’s censorship resistance is primarily concerned with the ability to move monetary value without trusted intermediaries. This proposal intentionally targets non-monetary uses of the chain (inscriptions, stamps, and similar schemes) and leaves ordinary monetary transfers untouched. Reasonable people may still describe permanently disabling some UTXOs as a form of censorship; the distinction this BIP draws is between money and arbitrary data storage.

The same type of mechanism could in principle be abused to target arbitrary UTXOs, which is why this proposal deliberately scopes itself to a one-time, transparently derived set of dust-sized

non-monetary outputs. Any attempt to apply similar techniques to ordinary monetary UTXOs would be highly contentious and would require explicit social consensus from users, miners, and economic nodes. In other words, this mechanism does not lower the technical or social barriers to censoring ordinary monetary activity; any such censorship would still require its own explicit, contentious change in consensus rules.

The classification rules on which The Cat relies are deliberately narrow and mechanical. An output is marked as an NMU only if it is (a) identified by the pinned Ord/Stamps rules as carrying a non-monetary artifact at H_snap, (b) dust-sized, below the VALUE_MAX_NMU threshold, and (c) within the defined height window. The intent is that no ordinary monetary UTXOs fall into NMUSet_snap; any such inclusion would be treated as an error in the snapshot construction and grounds to regenerate NMU_DATA before activation, not as an acceptable trade-off. The 1,000-sat value limit serves as an additional hard guardrail against accidentally classifying normally-sized wallet outputs.

Centralization and trust model for NMU generation

Anyone can independently generate the list of NMUs. It is incumbent on people to do so if they wish to minimize the trust required to run a fully validating node. The criteria for which outputs are designated as NMUs are necessarily known and established by virtue of how prevalent and open the targeted data protocols are.

It is valid to worry that fewer than 100% of node runners will regenerate the NMU list themselves, somewhat increasing the overall trust placed in third parties compared to today. However, this concern applies specifically to a “blacklist of UTXOs,” which is precisely the area where scrutiny is highest. That scrutiny should help ensure that enough independent parties index and attest to the contents of NMUSet_snap to exceed any reasonable threshold of credulity.

A useful analogy is the GUIX-based reproducible build process. Bitcoiners accept that many users will only run node binaries they download, rather than compiling from source and independently verifying the code. Compiled binaries must exist for Bitcoin to maintain decentralization in practice. This is an aspect of trust inherent in Bitcoin: we accept some technical limitations of users and employ a practical, trust-minimized workaround that, while not perfect, is better than the alternative.

If there were to be an issue with widely used binaries, it would become ubiquitously known very quickly. Similarly, if The Cat were flawed and began targeting monetary UTXOs, this would be detected well before activation, and the necessary fixes applied, with any distortion of its motivation exposed and corrected.

Backward compatibility

This proposal is a consensus-changing soft fork. Legacy nodes that do not implement The Cat will continue to accept blocks that spend NMUs as valid, while nodes that do implement The Cat will reject such blocks as invalid. As with any soft fork, activation requires clear opt-in from miners and from economic nodes. After activation, miners and other block proposers must avoid including transactions that spend NMUs, because such blocks will be rejected by upgraded nodes. Wallets and applications that track inscriptions, BRC-20 assets, and related schemes will observe that NMUs become permanently unspendable under The Cat and that balances associated with those UTXOs are no longer movable under the activated ruleset.

FAQ

Do node operators need to run Ord or Stamps?

No. The Cat does not require ordinary node operators to run Ord or Stamps, or to understand their rules in detail. Those indexers are cited as normative references for the inscription and stamp classification logic used to derive NMUSet_snap. In practice, nodes only need the canonical NMU_DATA blob and its hash NMU_DATA_HASH to enforce the consensus rule.

Anyone who wishes to independently verify NMUSet_snap can:

inspect the open-source Ord and Stamps code,

or re-implement equivalent classification rules,

and confirm that the derived set of NMU outpoints matches the set encoded in NMU_DATA (i.e., the blob whose hash is NMU_DATA_HASH).

What rules do the reference indexers actually use to decide which UTXOs are NMUs?

This BIP does not re-specify the full Ord and Stamps protocols, but in broad terms:

Ord 0.24.0 (inscriptions).
According to the classification rules implemented in Ord 0.24.0 at commit 5d2fbbe64b362cd6c30d6901e50cbe80084761f8, witness data is scanned for “inscription envelopes” in Taproot script paths. An inscription envelope is an OP_FALSE OP_IF … OP_ENDIF block containing data pushes that start with the ASCII tag ord and encode a content type and body. When those rules are applied to the best chain up to height H_snap, each recognized inscription is associated with a particular outpoint (txid, vout) (typically the first satoshi of a reveal transaction input, or as directed by the pointer field in the protocol). OrdNMUSet is defined as the set of outpoints that Ord 0.24.0 would classify as carrying inscriptions at H_snap under these rules.

Stamps (BTC Stamps).

According to the classification rules implemented in the pinned BTC Stamps indexer, Counterparty transactions are inspected for a STAMP:-prefixed base64 string in the description field, which encodes image data. The Stamps rules then map this metadata to specific dust-sized bare multisig outputs in the underlying Bitcoin transactions. When those rules are applied to the best chain up to height H_snap, each recognized stamp is associated with one or more outpoints (txid, vout). StampsNMUSet is defined as the set of outpoints that this reference Stamps implementation would classify as containing such stamp-style embedded assets at H_snap.

Won't spammers just make workarounds and spam anyway?

The Cat targets the financial incentive to create spam. It is almost impossible to stop a determined user from adding data to Bitcoin. The more realistic mitigation is to remove the ability to reliably trade these embedded assets as if they were durable, on-chain economic objects. Once the existing inscription and stamp economies are neutralized and their dust-sized carrier UTXOs are fenced off, the motivation to spam the chain in the same way is greatly reduced.

Can't users just spend their NMUs after the snapshot, removing them from the list?

Yes, a user can attempt to evade The Cat by spending an NMU after the snapshot. However, there are tens of millions of such outputs. It would be extremely costly, in aggregate fees, to move them all. If large-scale evasion occurs before activation, it is straightforward to take a new snapshot, forcing evaders to pay again if they want to continue. This is the nature of the cat-and-mouse game: attempts to pre-empt the snapshot by moving large numbers of NMUs impose substantial costs on those outputs, while the network can respond by choosing an appropriate snapshot height if necessary.

What if some of my UTXOs contain "inscribed sats"? Won't those get "Catted"?

In this BIP, "inscribed sats" can be understood as satoshis that Ord and similar indexers are treating as non-monetary artifacts. The Cat does not look at individual sat numbers or "rarity" directly; it only classifies whole UTXOs via the is_nmu(u) predicate, using NMUSet_snap and a simple value/height guardrail.

A UTXO can only be classified as an NMU if all of the following are true:

- it falls inside the NMU height window,
- its value is strictly below VALUE_MAX_NMU (currently 1,000 sats), and
- its outpoint appears in NMUSet_snap as identified by Ord/Stamps at the snapshot.

Anything at or above VALUE_MAX_NMU is always treated as a normal monetary output, regardless of what Ord says about the sats inside it.

In practice, if you once interacted with Ord and those “inscribed sats” now live inside a normal-sized wallet UTXO (1,000 sats or more), this proposal does not touch that UTXO at all. It remains fully spendable under the ordinary rules. The only “inscribed sats” that get Catted are those deliberately parked in tiny dust outputs below the VALUE_MAX_NMU threshold, and that Ord (or Stamps) is already treating as non-monetary artifacts at H_snap. Those dust UTXOs are precisely the objects this proposal is targeting to fence off from the monetary UTXO set.