What is Compression on Solana?

The most cost-efficient way of minting large numbers of digital assets in crypto.

What is NFT compression?

Compressed NFTs on Solana are the most cost-efficient way of minting large numbers of digital assets in all of crypto. TL;DR – Instead of storing data in accounts (expensive), compression lets you store it on the Solana ledger (cheap).

Overview

NFT compression allows developers to mint large amounts of NFTs for a fraction of the cost. This is achieved by storing the NFT properties on the Solana ledger instead of accounts. Traditionally, developers would need to allocate an account for each NFT in a collection. To read more about accounts, see this guide. While allocating a single account is cheap, the costs add up for large numbers of NFTs as they each require an account. With compression, you store the NFTs on the Solana ledger (via transactions) via a Merkle tree. The validity of the Merkle tree can be checked by looking at the "root hash" – derived by iteratively hashing together the contents of the entire tree. The root hash is stored in an account. Modifications to a compressed NFT require a "proof" to ensure that the NFT cannot be maliciously modified. The data returned by an indexer can also be verified by comparing the root hash with what is stored in the root account. For a deeper dive into how this works, please read the Solana documentation and check out our blog post explainer. Helius tracks compressed NFT state to facilitate faster look-ups and simplify your application code. The data provided by Helius includes a cryptographic proof that guarantees data integrity.

How are compressed NFTs different?

* Compressed NFTs are not native Solana tokens. They do not have a token account, mint account, or metadata. * One account exists per Merkle tree and each tree can hold millions of NFTs. * One collection is able to use multiple Merkle trees (recommended for larger collections). * A Merkle tree account can also hold multiple collections (not recommended). * A DAS API call is required to read any information about a compressed NFT (e.g. attributes, collection information, etc). This would have affect on Solana dApps loading your assets, etc. * All NFT modifications must happen through the * Bubblegum program * . * A compressed NFT can be converted to a regular NFT (but not the other way around). For practical reasons it is recommended to keep the tree size to 1 million or less. This is because the proof path will begin to exceed the Solana transaction account limit.

What modifications are possible?

At the time of writing, the following methods are supported by Bubblegum: * Mint * - Mint compressed NFT * Transfer * - Transfer compressed NFT * Burn * - Burn compressed NFT * Delegate, CancelDelegate * - Delegate authority of an NFT to a different wallet/Cancel the redemption of an NFT (Put the NFT back into the Merkle tree) * Redeem, CancelRedeem * - Redeem an NFT (remove from tree and store in a voucher PDA)/ * * Cancel the redemption of an NFT (Put the NFT back into the Merkle tree). * Decompress * - Decompress an NFT into an uncompressed Metaplex NFT. This will cost the rent for the token-metadata and master edition accounts that will need to be created. * VerifyCreator, SetAndVerifyCreator * - Verify and un-verify a creator that exists in the NFT's creator array. * VerifyCollection, SetAndVerifyCollection * - Verify or un-verify an NFT as a member of a Metaplex Certified collection when the collection is already set in the Metadata. Or set a new collection in the metadata and verify the NFT as a member of the new collection. You can find out more from the Bubblegum program docs here .

How does the indexer work?

Compressed NFT information is not stored in a traditional Solana account. Instead all the metadata is stored on a ledger and needs the help of indexing services to quickly fetch the metadata required. Note that you can still derive the current state by replaying the relevant transactions, but providers like Helius do this for you for convenience. The indexer listens to all Bubblegum transactions, parses them, and updates its state. For example, when a compressed NFT is minted the indexer will parse that transaction and extract all of the NFT info (name, collection, owner). If a tree was never seen before or it is missing an update, the indexer will fetch the tree's transaction history and reconstruct the state. The indexer code can be found here .

Examples

You can get started with the following examples: * * Helius Compression Examples * * *Metaplex Compression Examples *

Further Reading

* * What is Compression, a Twitter Thread * * *Metaplex Compression Overview * * *Metaplex Compression Documentation Hub * * *Solana Account Compression Repo * * *Exploring Compression on Solana * If you're more of a

visual learner, here's a crash course on it:

Solana RPC Nodes -Previous (Alpha) Priority Fee API Next- Compression & DAS API Digital Asset Standard (DAS) API

Last modified1mo ago