Nova LayerZero Implementation

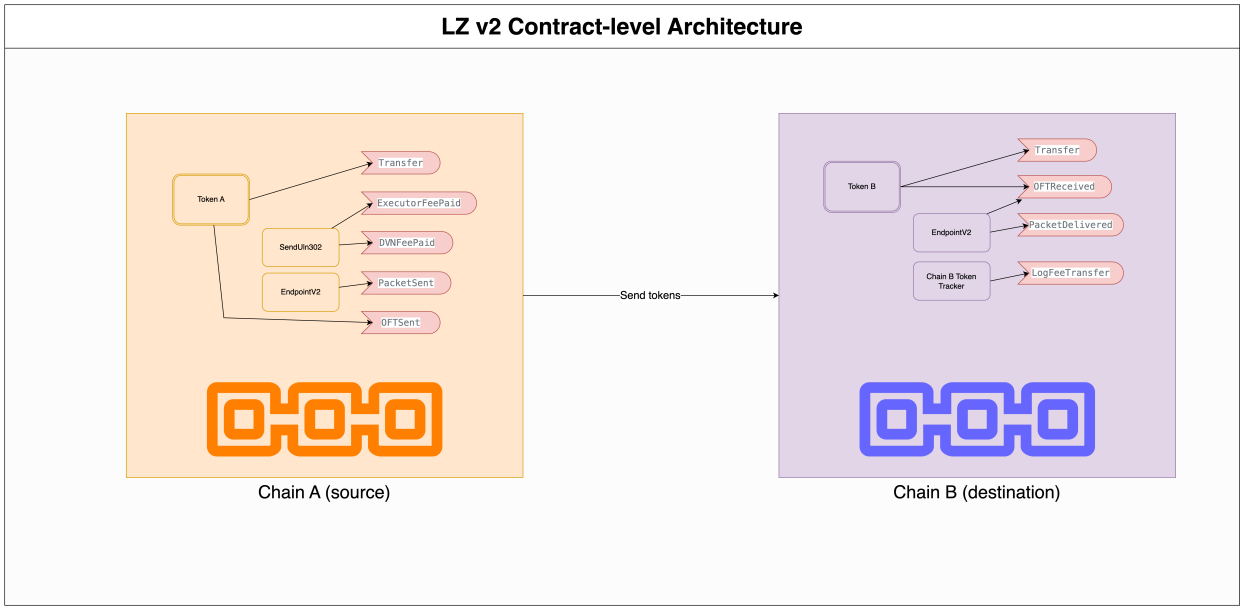
Description

LayerZero is a cross-chain protocol that is based on contract-level design pattern. This means there is no separate blockchain running, where the message packet is validated although there is a system here where there could be multiple validators.

> Throughout we would be referring to v2 of LayerZero, not v1.

In terms of contracts, there are mainly 3 types:

1. Omnichain App (OApp)
2. Omnichain Fungible Token (OFT)
3. Omnichain Non-Fungible Token (ONFT)



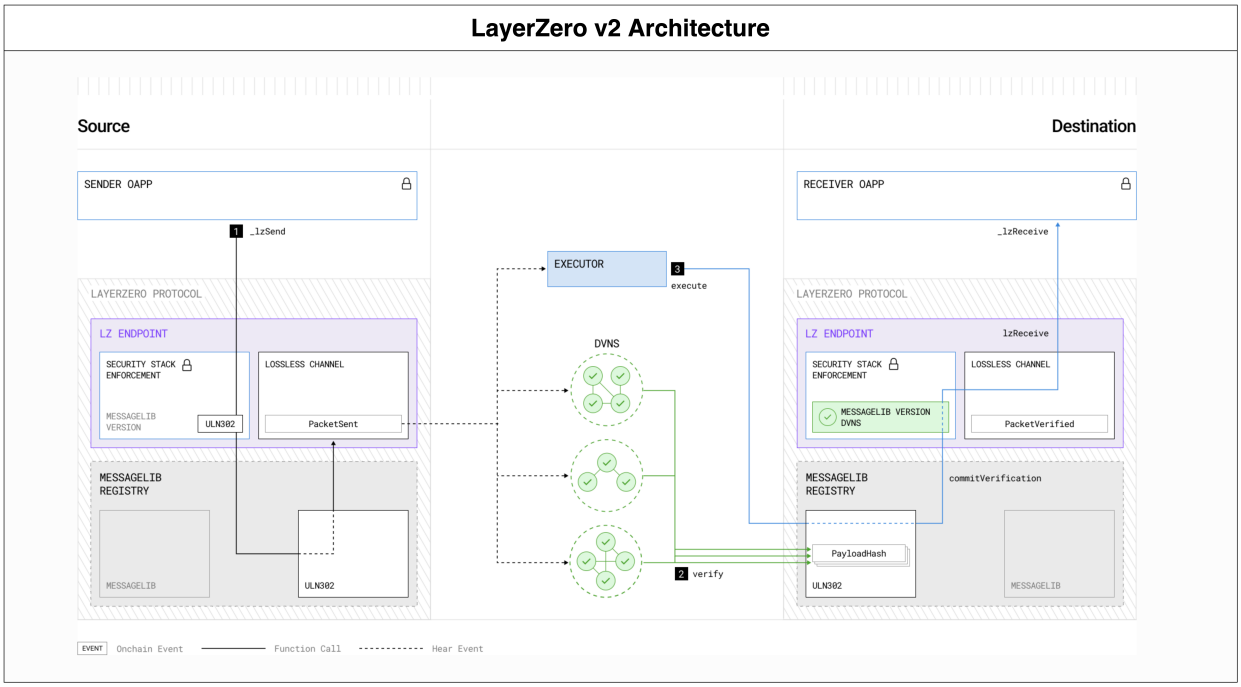
In the diagram above, there are many events emitted on both the source and destination chains. Based on the emitted events from source chain, the bridge takes care of both verification and execution by DVN and Executor respectively. DVN does its job if it detects `DVNFeePaid` event and Executor does its job if it detects `ExecutorFeePaid` event, but only after the verification is done.

> 💡 DVN and Executor need not be 2 separate entities. It could be one single entity that does both the jobs.

When a transaction is sent/confirmed on source chain, the transaction is converted into a message packet like this:

```
/// MESSAGE PACKET

struct Packet {
    uint64 nonce; // the nonce of the message in the pathway
    uint32 srcEid; // the source endpoint ID
    address sender; // the sender address
    uint32 dstEid; // the destination endpoint ID
    bytes32 receiver; // the receiving address
    bytes32 guid; // a global unique identifier
    bytes32 message; // the message payload
}
```



All the messages (sent 📤 from source chain) are verified and executed on the 📥 destination chain. Here:

- the *verification* could be done by one or more entities. This depends on the security stack configured by the delegate of OApp/OFT/ONFT on a per-chain pathway basis. More than 1 verification is possible and it can be enough depending on the confirmations set in the config for sender contract. In AutoBridge, the no. of confirmations is set to 1. So, 1 confirmation would be enough before execution.

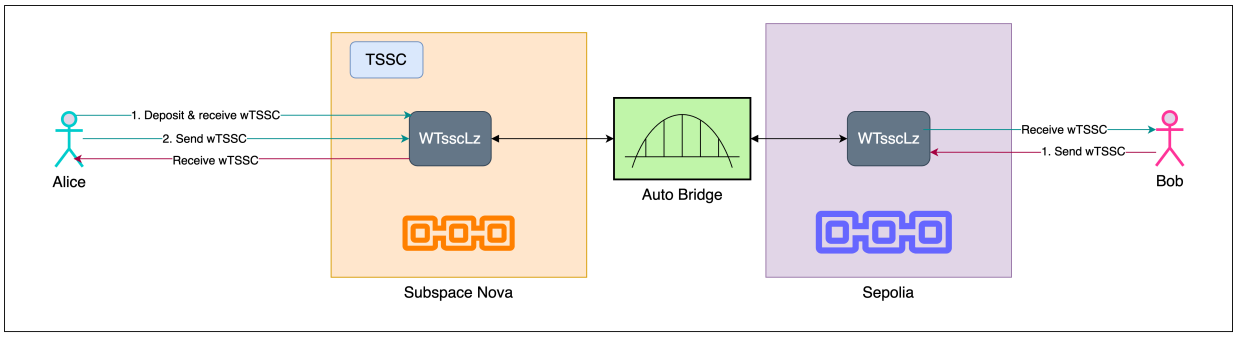
> When configuring DVN for a contract, make sure the DVN supports both the source and destination chains.

- the *execution* could be done by an individual EOA. Only one execution is enough. Execution essentially means minting equivalent amount of tokens on the destination chain to the receiver in terms of LZ OFT.

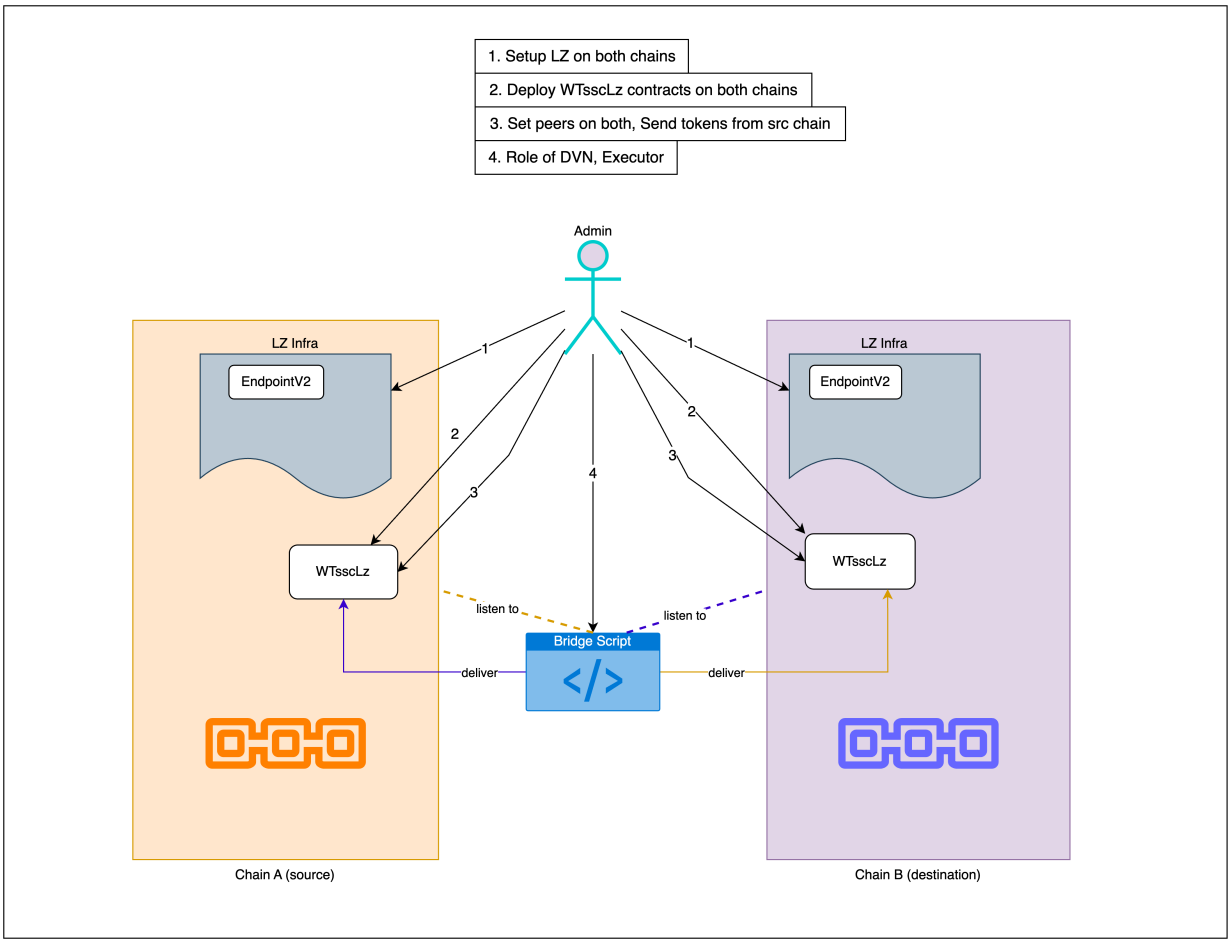
Development

The development is done in 2 parts:

1. Create the "**LZ-compatible Wrapped TSSC contract**" . This means additionally it has both `lzSend` and `lzReceive` functions as well to send tokens to & fro between chains.
2. Build the native bridge between the chains - *Subspace Nova* and *Ethereum Sepolia*. In the image below, note the two actions (`deposit` if they fall short of wTSSC balance) are required on Nova chain in contrary to one action on Sepolia.



Setup



Following are the steps followed as shown in the image above:

1. Make EVM chain LZ compatible: Setup LZ using script for both:
    a. Nova with respective local and remote Endpoint IDs. Here, remote EID refers to that of Sepolia.
    b. Sepolia with respective local and remote Endpoint IDs. Here, remote EID refers to that of Nova.

▼ Illustration ℹ️

> 💡 Just imagine if a entity decides to support multiple chains, then there needs to be so much of setup required with multiple combinations of pairs with local and remote EIDs.

> LZ suite of contracts remains common for each chain. During the setup, the EIDs values has to be set with different pairs.

The setup script has 2 parts in terms of using EIDs. One part uses local EID and the second part uses remote EID. So, if the contracts are deployed on one chain during LZ setup, then it's not required to redeploy, but instead set the remote EIDs for the same set of contracts using the second part of the script, just so that the message if sent from the sender contract on that chain is able to get delivered on the other chain with the set remote EID.

2. Contracts Deployment: Deploy Wrapped TSSC LZ compatible (WTsscLz) contract on both the source and destination chains using respective local EndpointV2 addresses and admins.

> 💡 The contract has both `send` and `receive` functions inside. This means we don't have to deploy 2 different contracts on either side for 2 purposes, but one on either side. Admins/delegates of both the contracts could be same or different.

3. Set peers: Admin of both contracts set peer with remote EID and remote WTsscLz address via this TS script. E.g. set peer of `WTsscLz` contract (on Nova) with Sepolia's EID and `WTsscLz` contract address.

4. Deposit TSSC & Send wTSSC: The `WTsscLz` contract has functions:
    a. `deposit`: Anyone can deposit TSSC and instead they receive same amount of WTSSC.
    b. `send`: Anyone can send their WTSSC from Nova to Sepolia.

These two functions are used here.

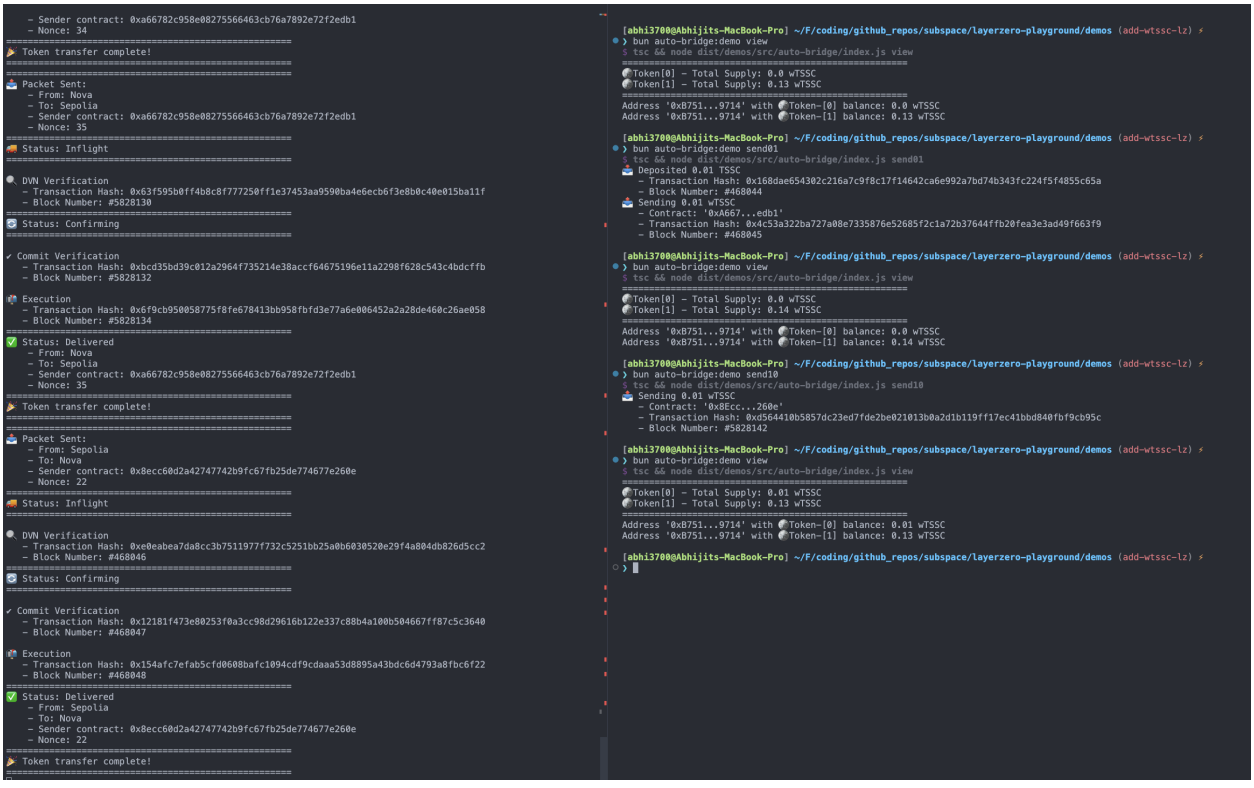
5. Host Bridge script: This bridge script is written in TypeScript that listens to any event `PacketSent` sent from `WTsscLz` contract. Post receiving, it does 2 main jobs:
   a. Verification
   b. Execution

   In summary, these are transactions performed on the destination chain.

Demo

Watch the demo videos 🎬:

- Send tokens from Nova to Sepolia.
- Send tokens from Sepolia to Nova. Add-on



▼ Illustration ℹ️

- Left side: bridge script is running and listening to emitted event `PacketSent` by the endpointV2 contract on source chain. And it verifies and executes a message and then on execution, packet's nonce is matched on destination chain as on source chain.
- Right side: run-1 is viewing the token balances and total supplies on both chains. In run-2, `0.01 TSSC` is deposited and instead `0.01 wTSSC` is minted, followed by `0.01 wTSSC` is sent. Actually, the same amount is burned to zero address on Nova chain.
  And the run-3 is same as run-1 after the message packet is verified and executed on destination chain as seen on left hand side of CLI. run-4 is about sending
  `0.01 wTSSC` from Sepolia to Nova followed by viewing balance after message's verification and execution.

Caveats 👀⚠️

- Sequential Execution: Multiple incoming messages from same contract can be verified by the DVNs/bridge in any order (although sequential is recommended), but then has to be executed sequentially because of its incremental nonces. The sequence is ensured/checked by `WTsscLz` contract by looking at their nonce decoded from the encoded packet when detected by the bridge script.
- Failure of successive message(s): Suppose, message-1 is sent from a contract A on Nova at block #1000 and on Nova it takes 1 day as confirmation period. Now, message-2 from the same contract A is sent from Nova at block #1001. Now after confirmation period is over, it's found that the message-1 is not added to block as it was fraudulent, but message-2 is added to block.

  ▼ NOTE:

  > Here, bridge when detects the message sent, looking at its transaction status (added/confirmed), it could be added to a queue, if waiting for confirmation. Another BE service running as CRON job can periodically check the

> queue's pending messages & verify on destination chain in sequence thereby executing.

In this case, message-2 can't be executed by the bridge on destination chain as message-1 with less nonce got rejected on Nova chain level itself.

- Backdoor Execution Security bug: This is a security bug observed on AutoBridge as per LZ's contract-level architecture. Here is the video recorded to showcase the issue.

  Watch video 🎬

  In this video, it can be viewed that there is a backdoor execution possible on destination chain without the message actually sent from the source chain. And this is possible by a trusted bridge when an OApp add it to its DVN security stack.

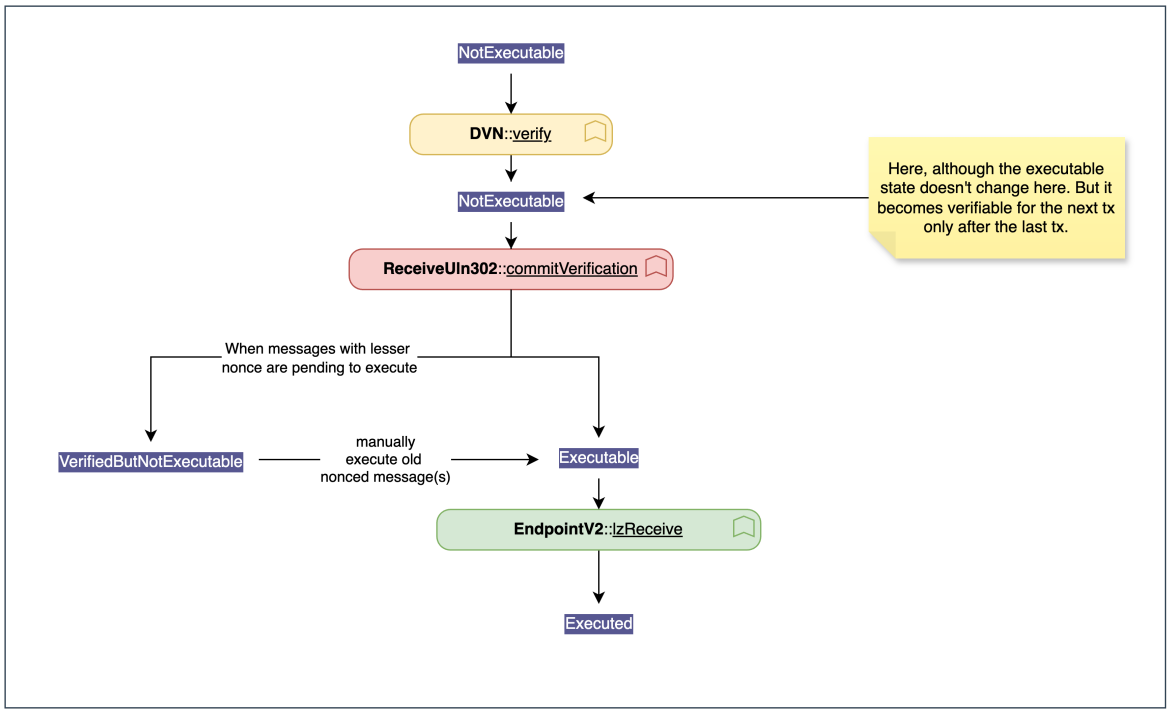
  Security Bug Issue.

  ▼ Summary

  Currently, there are 2 main issues/disclaimer for developers using LZ approach as cross-chain solution:

  1. LZ is (kind of) centralized with few DVNs available for message verification before its execution.
  2. An LZ OApp developer needs to apply due diligence before setting its DVN security stack.

- Currently, the bridge script is watching the events emitted from contracts - `WTsscLz` contract, LZ contracts suite on both source and destination chains.
- Message status: Currently, the bridge script runs based on both the message's verifiability and execution states:

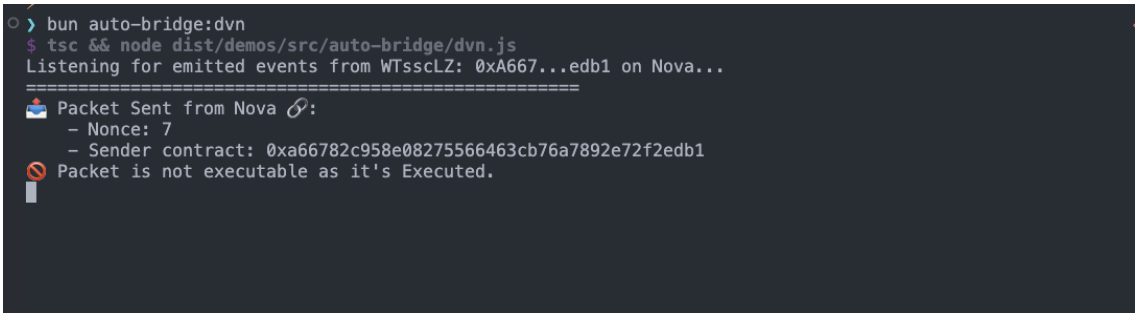


In the image above, the execution states (`NotExecutable`, ...) are shown in purple color.

▼ Illustration ℹ️

And based on the incoming messages's status (verifiability & executable) fetched from `EndpointV2View` contract on destination chain, the DVN verification, Verification commitment and Execution are done as transactions.

In the image below, the packet is blocked as it's already executed (already shown in the image previously). This can happen if the bridge script is rebooted, then it automatically detects the last sent transaction which was already executed before. It happens especially when the RPC is running on localhost.



Future

At this point, we have a complete PoC of AutoBridge running that is able to send transactions to & fro between Subspace Nova & ETH Sepolia chains.

But for mainnet, we seek support from external DVNs like **LayerZero Labs, Google Cloud, Polyhedra zkLightClient.** Once the message is verified, it can be executed either by them or anyone.

List of public DVNs

- ▼ Q. Can external entity provide support as DVN/Executor or both for the Subspace Nova (EVM) chain for delivering cross-chain message transfers to/from EVM-compatible chains like Ethereum and Polygon?

  We have a Wrapped TSSC token contract that adheres to the LayerZero OFT standard, and we are able to deploy this contract to other EVM chains, including Ethereum and Polygon. However, as there is no offchain layer supporting the our EVM chain, so we are specifically seeking support from the external DVNs for our Nova (EVM) chain.

- ▼ Q. Need to ask them if they can do both verification & execution?

  If not, then we have to run a BE service that executes or handover the responsibility to domain operators may be.