How does Open GSN Work?

There are several cogs in an Open GSN network:

* Relay Server: A server that listens for and processes meta-transaction requests from users. It forwards the requests to the Relay Hub, signs them, and pays the gas fees on behalf of users.
* Relay Hub: A smart contract deployed on the Ethereum blockchain that acts as a central point for managing relays, forwarding meta-transactions, and settling payments between users and relayers.
* Relay Manager: An entity (or contract) responsible for managing and maintaining a group of Relay Workers, ensuring their availability and efficient operation.
* Relay Worker: An account controlled by the Relay Manager, responsible for executing meta-transactions and paying the gas fees. Each Relay Worker has a unique Ethereum address and is funded with Ether to cover the gas costs.
* Paymaster: A smart contract that manages the payment for gas fees on behalf of users. It defines the rules for accepting or rejecting meta-transaction requests and compensates the relayers for their services.
* Forwarder: A smart contract responsible for verifying and forwarding meta-transaction requests from users to the intended contract recipient. It checks the validity of the user's signature and ensures the correct nonce is used.
* Contract Recipient (Our contract): The target smart contract that receives and processes the meta-transaction after being forwarded by the Forwarder. This is the smart contract that users interact with to perform desired actions.

Also, there are several concepts which one should familiarize with before trying to grasp and understand this technology, some of them are;

* Meta-transaction: A transaction that allows users to interact with smart contracts without holding Ether for gas fees. Instead, the gas fees are paid by a third party called a relayer. Meta-transactions include a user's signature and other necessary data for the relayer to execute the transaction on their behalf.
* Relayers: Entities that facilitate meta-transactions by receiving, processing, and executing them on the Ethereum blockchain. They pay the gas fees for the transactions and get compensated for their services.
* Signatures: Cryptographic proofs that allow users to authenticate and authorize transactions without revealing their private keys. Signatures are used in meta-transactions to ensure that the user has approved the transaction, and it has not been tampered with during transmission.

Who interact in order to allow gasless transactions. Here, a step by step of how a gasless transaction occurs, from user call to that call being executed on the recipient contract, and the Dapp reacting to it:

1. User Interaction: The user initiates a transaction to interact with an OpenGSN-enabled contract through a DApp (decentralized application) or a user interface.
2. Request Generation: The DApp constructs a metatransaction request containing the user's signature, destination contract, function call data, and the user's desired gas price.
3. Relay Server Selection: The DApp queries the relay hub to discover available relay servers and selects one based on its preferences, such as the lowest gas price or fastest response time.
4. Metatransaction Submission: The DApp submits the signed metatransaction request to the selected relay server.
5. Relay Server Validation: The relay server validates the received metatransaction request, ensuring that the user's signature and transaction data are correct.
6. Relay Manager: The relay server communicates with the relay manager, which is responsible for managing the pool of relay workers.
7. Relay Worker Selection: The relay manager selects a relay worker based on availability and gas price.
8. Transaction Signing: The relay worker signs the metatransaction with its own private key, indicating its commitment to pay the gas fees.
9. Relay Hub Interaction: The relay server submits the signed metatransaction to the relay hub, which verifies that the relay worker is authorized and properly staked. If the verification is successful, the relay hub allows the transaction to proceed.
10. Transaction Submission: The relay worker submits the signed transaction to the Ethereum network for inclusion in a block.
11. Paymaster Interaction: The relay hub checks with the Paymaster to confirm if it is willing to cover the gas fees for the metatransaction. The Paymaster verifies the transaction and signals its approval.
12. preRelayedCall: The Paymaster executes the preRelayedCall function to perform any necessary checks or preparations before the metatransaction is executed. This can include verifying the user's eligibility for fee coverage or setting up required state changes.
13. Forwarder: The forwarder contract verifies the user's signature and processes the metatransaction. If the signature is valid, the forwarder calls the target contract's function with the original user as the sender.
14. Contract Execution: The target contract executes the user's desired function call, triggered by the relay worker's transaction through the forwarder.
15. postRelayedCall: After the contract execution, the Paymaster's postRelayedCall function is called to finalize any state changes, record the transaction's outcome, or perform any other necessary actions.
16. Post-Execution Relay Hub Interaction: Once the transaction is included in a block and confirmed, the relay hub handles the accounting for gas fees, distributing payments from the contract to the relay worker, and updating its internal state.
17. Event Emission: The contract emits an event containing information about the successful metatransaction, such as the user's address and the transaction's details.
18. DApp Notification: The DApp listens for the emitted event and updates the user interface or performs additional actions based on the contract's response.