HashPack Wallet Connection Troubleshooting Guide

Author: Manus AI
Date: August 5, 2025

1. Introduction

This document provides a comprehensive guide to troubleshooting common issues encountered when connecting decentralized applications (dApps) with the HashPack wallet on the Hedera network. It aims to assist developers and users in successfully establishing and maintaining a stable connection, leveraging the latest integration methods and best practices. A smooth wallet connection is paramount for a seamless user experience within any Hedera dApp.

2. Understanding the Connection Mechanism

HashPack integrates with dApps primarily through WalletConnect, an open-source protocol that facilitates secure communication between dApps and mobile or desktop wallets. This protocol enables dApps to send transaction requests to wallets, and wallets to return signed transactions, all without exposing private keys to the dApp. The process typically involves a dApp initiating a connection request, the user approving this request within their HashPack wallet, and then a secure session being established.

3. Prerequisites for Successful Connection

Before troubleshooting, ensure the following prerequisites are met:

- **HashPack Wallet Installed and Updated:**
    - **Browser Extension:** Ensure the HashPack browser extension is installed and updated to the latest version. Regularly check the official HashPack website or your browser's extension store for updates.

    - **Mobile App:** If connecting via mobile, ensure the HashPack mobile application is installed and updated to its latest version from the respective app store (iOS App Store or Google Play Store).

- **Stable Internet Connection:** A reliable internet connection is crucial for both the dApp and the HashPack wallet to communicate with the Hedera network and WalletConnect relay servers.

- **WalletConnect Project ID:** For developers, a valid `projectId` obtained from WalletConnect Cloud is essential for initiating WalletConnect sessions. Without this, the connection will fail.

- **Correct Network Selection:** Ensure your dApp is configured to connect to the correct Hedera network (Mainnet or Testnet) and that the user's HashPack wallet is also set to the same network.

- **Browser Compatibility:** While WalletConnect is broadly compatible, ensure you are using a modern, up-to-date web browser (e.g., Chrome, Firefox, Brave, Edge) that fully supports Web3 functionalities.

4. Common Connection Issues and Solutions

This section details frequently encountered issues and provides actionable solutions.

4.1 HashPack Not Detected or Responding

Issue Description: The dApp fails to detect the HashPack wallet, or the WalletConnect modal does not appear/respond after clicking the connect button.

Possible Causes & Solutions:

1. **HashPack Extension Not Active/Enabled:**
    - **Solution:** Verify that the HashPack browser extension is installed and enabled in your browser. Check your browser's extension management page. Sometimes, extensions can be accidentally disabled.
2. **Browser Permissions:**

- **Solution:** Ensure HashPack has the necessary permissions to interact with the current website. In your browser's extension settings, check if HashPack has

access to all sites or specifically to the dApp's domain. Some browsers might require explicit permission for extensions to interact with certain pages.

3. **Conflicting Extensions:**

* **Solution:** Other browser extensions, especially other wallet extensions or privacy-focused extensions, might interfere with HashPack. Try disabling other extensions one by one to identify any conflicts.

4. **Browser Cache and Cookies:**

* **Solution:** Clear your browser's cache and cookies. Sometimes, stale data can cause unexpected behavior. After clearing, restart your browser and try again.

5. **HashPack Not Running (Mobile):**

* **Solution:** For mobile connections, ensure the HashPack app is open and running in the background or foreground when attempting to connect. Deep links might require the app to be active.

6. **Outdated HashPack Version:**

* **Solution:** Always use the latest version of HashPack. Older versions might have bugs or lack compatibility with newer WalletConnect protocols or Hedera network updates. Update HashPack via its official website or app store.

7. **Incorrect WalletConnect Library Integration (Developer Specific):**

* **Solution:** Double-check your dApp's code to ensure the `@hashgraph/hedera-wallet-connect` library (or Reown's AppKit) is correctly initialized with the proper `projectId`, `metadata`, and `network` configurations. Refer to the previous FrD for detailed integration steps.

4.2 Connection Rejected or Fails to Establish

Issue Description: The WalletConnect modal appears, but the connection is rejected by HashPack, or the connection process times out without establishing a session.

Possible Causes & Solutions:

1. **User Rejection:**

- **Solution:** The most common reason is that the user explicitly rejected the connection request within their HashPack wallet. Ensure the user understands the

connection request and approves it.

2. **Incorrect Network:**
    - **Solution:** The dApp might be requesting a connection to a different Hedera network (e.g., Mainnet) than the one the user's HashPack wallet is currently configured for (e.g., Testnet). Ensure both are aligned. The dApp should ideally prompt the user to switch networks if a mismatch is detected.
3. **Expired WalletConnect Session:**
    - **Solution:** WalletConnect sessions can expire. If a QR code or deep link is left open for too long, it might become invalid. Refresh the dApp page and try initiating the connection again.
4. **Firewall or VPN Interference:**
    - **Solution:** Network restrictions, firewalls, or VPNs can sometimes block communication with WalletConnect relay servers. Try temporarily disabling them to see if it resolves the issue. If it does, configure your firewall/VPN to allow WalletConnect traffic.
5. **HashPack Wallet Locked:**
    - **Solution:** Ensure the HashPack wallet is unlocked and accessible. If it's locked, the connection request cannot be processed. Unlock your HashPack wallet before attempting to connect.
6. **Insufficient Funds for Transaction Fees (for certain operations):**
    - **Solution:** While initial connection doesn't require HBAR, some subsequent operations (like associating tokens) do. Ensure the connected Hedera account has sufficient HBAR to cover potential transaction fees if the connection is part of a transaction flow.
7. **Outdated WalletConnect Libraries (Developer Specific):**
    - **Solution:** Ensure your dApp is using the latest versions of `@walletconnect/modal` and `@walletconnect/universal-provider` (if using Reown's AppKit). Outdated libraries can

lead to compatibility issues.

4.3 Transactions Not Being Signed or Approved

Issue Description: The wallet connects successfully, but when the dApp sends a transaction request, HashPack does not prompt for approval, or the transaction fails to sign.

Possible Causes & Solutions:

1. **HashPack Not Active/Focused:**
    - **Solution:** Ensure the HashPack extension or mobile app is active and in focus when a transaction request is sent. Some browsers might minimize the extension or send it to the background, preventing the approval prompt from appearing.
2. **Permissions Not Granted for Specific Operations:**
    - **Solution:** While the initial connection grants basic access, some dApps might require additional permissions for specific operations (e.g., token transfers, smart contract calls). Ensure the user has granted all necessary permissions within HashPack.
3. **Invalid Transaction Payload (Developer Specific):**
    - **Solution:** Double-check the transaction payload being sent to the wallet. Ensure it conforms to the Hedera SDK standards and the expected JSON-RPC method. Any malformed payload can cause the wallet to reject it silently or with an error.
4. **Network Mismatch for Transaction:**
    - **Solution:** Verify that the transaction is being sent to the same Hedera network (Mainnet/Testnet) that the user's HashPack wallet is currently connected to. A mismatch will result in transaction failure.
5. **Insufficient HBAR for Transaction Fees:**
    - **Solution:** Every transaction on Hedera requires a small HBAR fee. Ensure the connected account has enough HBAR to cover the transaction fee. If not, the transaction will fail.
6. **HashPack Internal Error:**

- **Solution:** Occasionally, HashPack itself might encounter an internal error. Try restarting the HashPack extension/app or reinstalling it as a last resort. Check HashPack's official support channels for known issues.

5. Best Practices for Developers

To minimize connection issues and provide a robust user experience, developers should adhere to the following best practices:

- **Always Use Latest Libraries:** Regularly update `@hashgraph/hedera-wallet-connect`, `@hashgraph/sdk`, and WalletConnect libraries to their latest stable versions. This ensures compatibility and access to the newest features and bug fixes.
- **Implement Comprehensive Error Handling:** Catch and handle all potential errors during the connection and transaction signing process. Provide clear, user-friendly error messages that guide the user on how to resolve the issue.
- **Provide Clear User Instructions:** Guide users through the connection process with clear on-screen instructions. Explain what to expect when clicking the connect button and what actions they need to take in their HashPack wallet.
- **Test Across Environments:** Thoroughly test your dApp's connection flow across different browsers, operating systems, and both desktop and mobile environments to identify and address any platform-specific issues.
- **Monitor WalletConnect Events:** Actively listen for `accountsChanged` and `chainChanged` events to keep your dApp's state synchronized with the user's wallet. This allows for dynamic updates if the user switches accounts or networks.
- **Implement Session Persistence:** For a better user experience, implement logic to persist WalletConnect sessions. This avoids requiring users to reconnect their wallet every time they visit your dApp.
- **Utilize HashPack Profile API:** Once connected, leverage the HashPack Profile API to fetch and display user profile information. This enhances personalization and provides a more engaging experience.

- **Provide a Disconnect Option:** Always offer a clear and easily accessible

disconnect option within your dApp. This gives users control over their connection.

6. Conclusion

Successfully integrating and maintaining a connection with the HashPack wallet is crucial for any dApp on the Hedera network. By understanding the underlying WalletConnect protocol, adhering to best practices, and proactively addressing common issues, developers can ensure a robust, secure, and user-friendly experience. This guide serves as a valuable resource for navigating the complexities of HashPack integration and resolving connection challenges efficiently.

7. References

[1] HashPack. (n.d.). *Help & Support*. Retrieved from https://www.hashpack.app/support

[2] HashPack. (n.d.). *Developers*. Retrieved from https://www.hashpack.app/developers

[3] WalletConnect. (n.d.). *Documentation*. Retrieved from https://docs.walletconnect.com/

[4] Hedera. (n.d.). *Hedera SDKs and APIs*. Retrieved from https://docs.hedera.com/hedera/sdks-and-apis/sdks

[5] GitHub. (n.d.). *hashgraph/hedera-wallet-connect issues*. Retrieved from https://github.com/hashgraph/hedera-wallet-connect/issues