SEEDCARD Generator TL (TimeLock) Overview

Q: How do you generate a Timelocked Bitcoin Wallet?

Generating a Bitcoin wallet with a **timelock** (e.g., 21 years locktime) involves several key steps. These steps ensure that **all** addresses generated by the wallet are timelocked and funds cannot be spent until a specified block height or timestamp is reached.

The redeem script plays a critical role in ensuring this functionality.

---

1. Wallet Creation and Private Key Generation

The first step is the creation of the wallet itself, which includes generating a **private key**. The private key will be the primary piece of information required to sign and authorize transactions when spending funds from the generated addresses.

- **Generate the Private Key**: This can be done using any standard Bitcoin wallet generation method (e.g., using a cryptographic library such as **bitcoinjs-lib** for JavaScript or **bitcoincore-lib** for Python).
- **Generate the Public Key**: The public key is derived from the private key and will be used to create the Bitcoin address. This is a key step because the public key (or its hash) is included in the **redeem script**.

---

2. Create the Timelocked Redeem Script

Next, you create a **redeem script** that embeds the **timelock condition**. The redeem script is a piece of Bitcoin script that contains the locktime condition (using **OP_CHECKLOCKTIMEVERIFY (CLTV)**) and specifies the conditions under which funds can be spent.

Redeem Script Example:

```
<locktime> OP_CHECKLOCKTIMEVERIFY OP_DROP OP_DUP
OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
```

- **<locktime>**: This is the block height or timestamp at which funds can be spent. For example, if you want the locktime to be 21 years from the current block, you will calculate the block height or timestamp that represents 21 years and use it as the locktime value.

- **OP_CHECKLOCKTIMEVERIFY**: This opcode ensures that the transaction cannot be confirmed until the specified locktime is reached.
- **OP_DROP**: This removes the locktime value from the stack.
- **OP_DUP and OP_HASH160**: These operations duplicate the public key and hash it (to create a public key hash).
- **<pubKeyHash>**: This is the **public key hash** derived from the public key.
- **OP_EQUALVERIFY and OP_CHECKSIG**: These operations verify that the provided signature matches the public key and validates the transaction.
- The **redeem script** ensures that the funds are locked until the locktime is met. It defines the spending conditions, including the locktime enforced by CLTV.

---

3. Generate the Address

Once the redeem script is ready, you create the Bitcoin address. There are two common types of addresses for this:

- **P2SH (Pay-to-Script-Hash)**: The redeem script is hashed, and the resulting hash is used to create the address.
- **P2WSH (Pay-to-Witness-Script-Hash)**: The redeem script is hashed, but it's used in SegWit transactions, which offer lower fees and better efficiency.

To generate the address:

- **Hash the redeem script** using **RIPEMD-160(SHA-256(redeem_script))** to get the script hash.
- **Create the Bitcoin address** by encoding the script hash into the standard **P2SH or P2WSH** format (Base58 or Bech32, respectively).

This address is what you will use to **receive funds**.

---

4. Store the Redeem Script Securely

The redeem script is an essential part of the wallet. You will need to store the redeem script securely because it's required for spending funds after the locktime expires. There are several ways to store it:

- **Backup the redeem script** in a secure location (e.g., encrypted file, hardware wallet, paper backup, QR code).

- You **must store** the redeem script separately from the private key because, without it, the locktime condition cannot be satisfied, and the funds will be inaccessible after the timelock expires.

Tip: It's a good practice to generate the redeem script **before** generating the address, to ensure that everything is set up properly. However, if you lose access to the redeem script, you will lose access to the funds.

---

5. User Receives Funds

Once the wallet is set up and an address is generated, the user can receive funds. The funds will be locked and cannot be spent until the **locktime** (21 years in this case) has passed.

---

6. Spending Funds After Locktime Expiry

When the specified locktime has passed (e.g., 21 years later), the wallet can spend the funds. To do so:

- **Sign the transaction** with the private key.
- The **redeem script** is provided as part of the spending transaction to ensure that the locktime condition is satisfied.
- The transaction will include the **nLockTime** field, which ensures that it cannot be broadcast before the locktime is met.

Q: Is It Possible to Regenerate the Wallet with the Private Key and Redeem Script?

Yes, it is possible to regenerate the wallet as long as you have the **private key** and the **redeem script**. Here's how this would work:

1. **Regenerate the Private Key**: You can regenerate the private key from the original wallet seed or directly from the private key itself if you have it backed up.
2. **Regenerate the Public Key**: The public key is derived from the private key, which is used to create the Bitcoin address.
3. **Rebuild the Redeem Script**: With the **public key** (or its hash), you can rebuild the **redeem script** that enforces the locktime (using the same **locktime** value as originally set).
4. **Regenerate the Address**: Using the redeemed public key and the redeem script, you can regenerate the Bitcoin address.
5. **Sign Transactions**: Once the locktime has expired, you can use the **private key** to sign and broadcast the transaction to spend the funds.

Q: When Can the Redeem Script Be Output?

The redeem script can be **generated and output at the time of wallet creation**, and should be stored securely. The wallet software will generate this script when creating a new address, and it can be exported as a QR code or backed up for later use. Once the redeem script is created, you can safely store it alongside the private key (but separately, to prevent both being lost together).

Q: What happens if the redeem script is lost/hacked?

If the holder of a timelocked wallet only has the **private key** but does not have access to the **redeem script**, the following would happen:

---

1. The Holder Cannot Spend Funds Without the Redeem Script

The **redeem script** is a critical component of the locking mechanism for the wallet. It defines the conditions under which funds can be spent, including the **locktime** enforced by the `OP_CHECKLOCKTIMEVERIFY (CLTV)` opcode.

When attempting to spend funds from a **P2SH (Pay-to-Script-Hash)** or **P2WSH (Pay-to-Witness-Script-Hash)** address, the following process occurs:

- The transaction must provide the redeem script in order to prove that the spending conditions are met.
- The script is evaluated during transaction validation, and if the conditions are not satisfied (e.g., the locktime has not expired or the redeem script is missing), the transaction will be rejected.

If the **private key** is available but the **redeem script** is missing, the transaction cannot include the necessary conditions for spending the funds. Therefore, the funds will remain locked and inaccessible.

---

2. Why the Redeem Script Is Needed

In a timelocked wallet:

- The redeem script contains the **locktime condition** and other details needed to spend the funds.

- For a **P2SH address**, the redeem script is hashed and embedded in the address. However, the full script is not stored on the blockchain until it is revealed in a spending transaction.
- Without the full redeem script, the wallet software or node validating the transaction cannot verify that the spending conditions are met.

---

3. Comparison with Standard Single-Signature Wallets

In a standard single-signature wallet:

- The private key is sufficient to spend funds, as the locking script typically checks only the public key and a valid signature.
- There is no redeem script, so the private key alone provides full control of the funds.

In a timelocked wallet:

- The redeem script is essential to spending, even with the private key. The private key proves ownership, but the redeem script enforces the timelock condition.

---

4. What Happens Without the Redeem Script

- **Before Locktime Expires**: The holder cannot spend the funds because the locktime condition is not met. The private key is useless without the redeem script to specify the spending conditions.
- **After Locktime Expires**: Even though the locktime condition has been satisfied, the transaction will still require the redeem script to validate the timelock and allow the funds to be spent. Without the redeem script, the transaction will fail.

---

5. Options for Recovery Without the Redeem Script

If the redeem script is lost and the user has only the private key, recovery of the funds is nearly impossible. Here's why:

- The **script hash** stored in the P2SH address is only a hash of the redeem script. It is computationally infeasible to reverse-engineer the original redeem script from the hash alone.
- Without the original redeem script, the locktime and other conditions embedded in it are unknown and cannot be reproduced.

Q: What happens if the private key and redeem script is lost/hacked?

If the holder of the wallet has both the **private key** and the **redeem script**, but the **timelock has not expired**, the funds will still remain **locked**. Here's why:

Why Funds Remain Locked Before the Timelock Expires

The timelock condition is enforced by the Bitcoin network's consensus rules. Specifically:

1. **OP_CHECKLOCKTIMEVERIFY (CLTV) Enforces the Lock**:
   - The redeem script contains the `OP_CHECKLOCKTIMEVERIFY` opcode, which enforces the timelock condition.
   - The Bitcoin network will reject any transaction attempting to spend funds from this script before the specified block height or timestamp has been reached.
2. **Immutable Script Locking Conditions**:
   - Once funds are sent to a **Pay-to-Script-Hash (P2SH)** or **Pay-to-Witness-Script-Hash (P2WSH)** address, the script's locking conditions (defined by the redeem script) are effectively **immutable**. The script hash, embedded in the address, ensures that only a transaction satisfying the original script can spend the funds.
   - Even if the user modifies the redeem script to remove or change the timelock condition, the modified script will produce a different hash. This means it will not match the hash embedded in the P2SH or P2WSH address, and the transaction will be invalid.
   -
3. **Consensus Rules Apply Globally**:
   - Bitcoin nodes and miners validate transactions against the locking script and consensus rules. Any transaction that does not meet these conditions (e.g., attempting to bypass the timelock) will be rejected by the network.

Why the Holder Cannot Change the Script to Release Funds Earlier

- The funds are locked to the **script hash** derived from the original redeem script. For the network to validate a transaction, the original redeem script must be provided, and it must satisfy the conditions in the script.
- If the holder tries to modify the redeem script to remove the timelock:
  - The new script's hash will not match the one embedded in the P2SH or P2WSH address.

    - The transaction attempting to spend the funds will fail validation.

What Happens After the Timelock Expires?

Once the specified block height or timestamp is reached:

- The redeem script can be provided along with a valid signature derived from the private key to spend the funds.
- At this point, the transaction will pass validation, as the locktime condition has been satisfied.

Key Points to Remember

1. **Timelock Is Enforced by the Script**: The locking conditions in the script cannot be bypassed or altered once funds have been sent to the address derived from the script hash.
2. **Private Key Alone Cannot Override the Lock**: While the private key proves ownership, it cannot override the timelock enforced by the redeem script.
3. **Redeem Script Integrity Is Crucial**: The script hash embedded in the address ensures that only transactions satisfying the original script can unlock the funds.

Conclusion

If the holder of the wallet has only the **private key** but not the **redeem script**, the funds will be inaccessible. The redeem script is essential for satisfying the timelock condition and unlocking the funds, even after the locktime has expired. The redeem script should be treated as critical as the private key itself and securely backed up to avoid this risk.

If the holder has both the **private key** and the **redeem script**, the funds cannot be spent until the timelock condition has expired. Modifying the redeem script to bypass the timelock would invalidate the address derived from the script hash, making the funds inaccessible until the original script's conditions are met. The timelock is an immutable constraint enforced by Bitcoin's consensus rules.