Simple Step-by-Step Guide: How SurePack Certificate System Works

Overview

SurePack is like a digital ID card system where users can get unique, secure identities for encrypted communication. Think of it as getting a special passport that only you can use.

🔐 Creating Your Digital Identity (Client Side)

Step 1: User Starts the Process

- User runs a simple command: `surepack create`
- Optional: Can add their email address for recovery: `surepack create -e john@example.com -t 123456`
- User enters a password (like a master key for their digital safe)

Step 2: Creating Your Keys (Like Making a Lock and Key)

The computer creates THREE different types of locks and keys:

1. **Regular Key (RSA)** - Like a traditional house key
2. **Future-Proof Key #1 (Kyber)** - Protected against future quantum computers (for encryption)
3. **Future-Proof Key #2 (Dilithium)** - Protected against future quantum computers (for signatures)

Think of it as having three different locks on your door - even if someone figures out how to pick one, they still can't get in.

Step 3: Sending Your Request

- The computer keeps all the PRIVATE keys (like keeping your house keys)
- It sends only the PUBLIC keys to the server (like giving out copies of your locks)
- If you provided an email, it includes the verification code

Step 4: Receiving Your Identity

The server sends back:

- **Your unique name**: Three random words like `happy-cloud-tree.example.com`
- **Your certificate**: Like an official ID card signed by the server

Step 5: Verifying Everything is Legitimate

- The computer checks that the certificate is real (like checking a driver's license hologram)
- It saves a "fingerprint" of the server's signature for future verification

Step 6: Storing Everything Safely

Your computer saves:

- Your certificate (public - like your ID card)
- Your three private keys (encrypted with your password - like keys in a safe)
- The server's fingerprint (to verify future communications)

Everything is stored in a special folder on your computer, protected by your password.

💻 How the Server Creates Your Identity

Step 1: Receiving the Request

The server gets:

- Your public keys (the "locks")
- Optional: Your email and verification code

Step 2: Checking Your Identity (If Email Provided)

If you provided an email:

- Server checks if you're allowed to use that email domain
- Verifies your code matches what was sent to your email
- Ensures the code hasn't expired (1-hour time limit)
- Deletes the code after successful verification

Step 3: Creating Your Unique Name

The server:

- Randomly picks three words from a dictionary of 10,000 common words
- Creates combinations like `happy-cloud-tree`
- Checks if this name is already taken
- Tries up to 10 times to find a unique name

Fun fact: Some combinations are rarer than others:

- **Legendary**: All three words the same (like `love-love-love`)
- **Epic**: Two words match (like `happy-happy-tree`)
- **Common**: All different words

Step 4: Creating Your Certificate

The server creates an official certificate that includes:

- **Your name**: The three-word alias
- **Valid for**: 397 days (about 13 months)
- **Your public key**: So others can send you encrypted messages
- **Your email**: If provided, for account recovery
- **Special data**: Your quantum-resistant public keys
- **Server's signature**: Proving it's authentic

Step 5: Signing with the Master Key

- The server unlocks its master signing key (kept encrypted)
- Signs your certificate (like a notary stamping a document)
- This signature proves the certificate is genuine

Step 6: Storing and Returning

The server:

- Saves your certificate in two places:
  - Main storage: Under your three-word name
  - Email storage: Under your email (if provided)
- Sends the certificate back to you

Why HTTPS/SSL is Critical

The server MUST have a valid HTTPS certificate (the green padlock in your browser) because:

1. **Initial Trust**: Like checking the bank's official seal before opening an account
2. **Secure Communication**: All data is encrypted during transmission
3. **Server Authentication**: Proves you're talking to the real server, not an imposter
4. **Certificate Verification**: When checking if other users are legitimate

Without HTTPS, it would be like:

- Sending your passport application through regular mail instead of certified mail
- Anyone could intercept and create fake IDs
- You couldn't trust any certificates you receive

Simple Analogy

Think of the whole system like getting a passport:

1. **You apply** (create command) with your information
2. **You keep your private documents** (private keys) in your safe at home
3. **You send copies of public documents** (public keys) to the passport office
4. **The passport office verifies your identity** (email verification)
5. **They create your unique passport** (certificate with three-word name)
6. **They stamp it with their official seal** (digital signature)
7. **They send it back to you** (return certificate)
8. **You store it safely** (encrypted storage)

The HTTPS certificate is like the official government building - you know you're in the right place because you can see the official signs and security guards. Without it, you might accidentally give your information to scammers in a fake office.

What You End Up With

After the process, you have:

- A unique three-word identity (like `happy-cloud-tree.example.com`)
- Three types of private keys (locked with your password)
- A certificate that proves your identity
- The ability to:
    - Receive encrypted messages that only you can read
    - Send signed messages that others can verify came from you
    - Communicate securely even if quantum computers are invented

All of this happens in seconds, and you only need to remember your password!