This system is mainly used for registration and identity verification of smart home devices, so as to alleviate the performance problems caused by the centralization of the Internet of Things system, and effectively prevent illegal devices from being connected to the system.

In this system, the smart device manufacturer generates a unique private ID for the SRAM-based PUFs after producing the smart device, and uses this as one of the subsequent registration information. ID SRAM PUF technology provides every IoT device with a strong and device-unique cryptographic key pair, which forms the basis of the device's digital identity. In order to register the device, the manufacturer (producer of device D) produces a device (e.g., D1), and then challenges the PUF of the device to obtain its response and regards it as the private ID of the device (e.g., IDD1). It then computes a cryptographically secure hash (e.g., HIDD1).

The device needs to be registered in the local blockchain to which the user belongs. and then it can be connected to the IoT system. Devices connected to the IoT will be regularly scanned and verified to prevent illegal devices from joining. Each local area will have a local administrator.

Although the devices that entered the IoT network have been registered and screened, there may still be rogue employees in the system who can replace the real devices with cloned peer devices. Therefore, it is necessary to verify the identity of all edge devices to verify their authenticity regularly. In the local blockchain a local administrator will set up which is mainly responsible for registering devices to the local blockchain data set, and regularly check the device in the IoT system. If there are any illegal devices found in the system, the device information will be sent to the user.

The device registration process is as follows:

1) The local administrator first verifies whether the device already exists in the local blockchain data set, and if so, terminates the following operation.

2) The administrator generates a random key (e.g., KD3) and programs the key (e.g., KD3) into the device's one-time programmable (OPT) memory, which will prevent the device's ID from being directly accessed, thereby preventing the ID from being copied by illegal persons. Note that OTPs are resistant to tampering, and the contents remain unchanged once programmed. A one-time programming capability is provided primarily to prevent an attack arising when an adversary may gain direct access to an edge device.

3) The local administrator sends the random key (e.g., KD3) to the gateway (e.g., G1) of the local IoT system. The gateway (e.g., G1) encrypts the key (e.g., KD2) through asymmetric encryption and returns the encrypted key (e.g., EKD3) to the administrator.

4) The administrator then uploads the device ID, the hash value of the device's private ID, and the encrypted key {D3, HIDD3, EKD3} returned by the gateway to the local blockchain to complete the registration of the device in the local blockchain. At this point, a gateway will be able to verify all the devices it communicates with. Due to the above encryption, even if the gateway is attacked, the attacker can only see the hash value of the private ID and the encrypted key, thereby increasing the difficulty of cloning the device. During the normal operation of the gateway, all devices that communicate with it are periodically authenticated.

The verification process can be described as follows:

1) The gateway queries the data set of the local blockchain to obtain the public ID (e.g., D3) of the device, and the local blockchain returns the hash value of the private ID (e.g., HIDD3) of the device and the encrypted key (e.g., EKD3) to the gateway.

2) After receiving the return value, the gateway decrypts the encrypted key (e.g., EKD3) with its own secret key (e.g., EG1), obtains the key (e.g., KD3).

3) The gateway reads the random key (e.g., K\*D3) from the device stored in the device previously. And compares it with the KD3 which obtained in step 2. If they are equal, the device passes verification, otherwise, it is set to "Failed device".