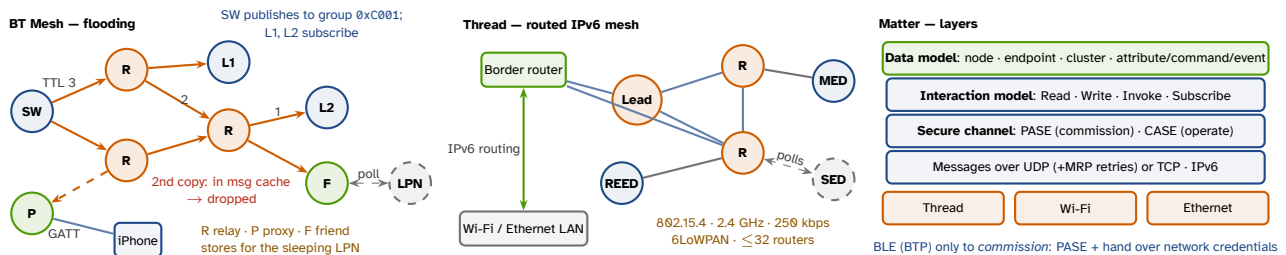


In one line: **Bluetooth Mesh** floods small messages over BLE *advertising* (no connections) with pub/sub group addresses; **Thread** is a **routed IPv6** mesh on 802.15.4 reaching the LAN through a **border router**; **Matter** is the **application layer** (data model + security + commissioning) that runs over Thread, Wi-Fi or Ethernet — commissioned over BLE, local-first, multi-admin.



Bluetooth Mesh (SIG, not LE pairing)

- Advertising bearer:** non-connectable adverts (AD 0x2A message · 0x2B beacon · 0x29 PB-ADV), nodes scan all the time. No central/peripheral, no per-link pairing.
- Managed flooding:** relays rebroadcast; **TTL** (7 bit, 0 = not relayed) caps hops; a **message cache** drops repeats. No routing tables — robust, airtime-hungry; more relays = coverage, not speed.
- Features:** relay · **proxy** (GATT ↔ adv, for phones) · **friend** (queues for its LPN) · **LPN** (radio mostly off, *polls* its friend).
- Addresses:** unicast 0x0001–0x7FFF (per element) · virtual 0x8000–0xBFFF (hashed 128-bit label) · group 0xC000–0xFEFF · fixed 0xFFFC proxies, FFFD friends, FFFE relays, FFFF all.
- Models:** node → elements → models (Generic OnOff server/client); a client *publishes* to an address, servers *subscribe* — bind by group, like MQTT topics.
- Provisioning** (PB-ADV or PB-GATT): Invite → Capabilities → ECDH P-256 → OOB-authenticated Confirmation/Random → Data: NetKey, unicast address, IV index. Then configure with the DevKey: bind AppKeys, set pub/sub.
- Keys:** **NetKey** (network layer — relays forward, can't read) · **AppKey** (payload, per function: lights vs locks) · **DevKey** (config). Replay: 24-bit SEQ + 32-bit IV index; receivers keep a replay list — persist SEQ.
- Tiny:** 11-byte access payload unsegmented, ≤ 384 B segmented — control, not bulk. Mesh 1.1 (2023) adds directed forwarding, remote provisioning, DFU.

Thread

- IEEE **802.15.4** (Zigbee's radio): channels 11–26, 250 kbps, 127-byte frames; **6LoWPAN** compresses IPv6 — every node has an IPv6 address.
- Roles:** **Router** (always on, routes) · **Leader** (a router that assigns router ids, re-elected) · **REED** (can promote) · **MED** (radio on, not routing) · **SED** (sleeps, polls its parent) · **Border router** (IPv6 to the LAN; none = isolated island). Self-healing.

Pick one

	BT Mesh	Thread	Matter
Layer	full stack on BLE	network (IPv6)	application
Forward	flooding + TTL	routing	— (below)
IP	no	yes	needs IP
Phone	via proxy (GATT)	via border router	native in iOS/Android
Best at	lighting, many nodes, no infra	low-power IP sensors	cross-ecosystem home

Remember

Mesh floods (TTL + cache), Thread routes (IPv6 + BR), Matter speaks (clusters), BLE commissions.

Matter

- CSA standard (ex-Project CHIP), 1.0 in Oct 2022. **Not a radio:** Matter over Thread, Wi-Fi or Ethernet; Zigbee/Z-Wave only via a **bridge** (endpoints per bridged device). **Local-first:** control on the LAN, cloud only for remote.
- Data model:** **node** (one operational identity) → **endpoints** (0 = root: basic info, commissioning, credentials, ACL) → **clusters** (OnOff 0x0006, LevelControl 0x0008) → attributes, commands, events. Subscribe = min/max interval reports.
- Commissioning:** QR (MT: ...) / NFC / 11-digit code = passcode + discriminator → find the device (BLE adv or DNS-SD _matterc._udp) → **PASE** (SPAKE2+) → **attestation:** DAC → PAI → PAA root listed in the CSA's DCL, + Certification Declaration → Wi-Fi/Thread credentials → **NOC** installed → **CASE** on the IP network (_matter._tcp).
- Fabric** = root CA + fabric id; each admin (Apple, Google, Alexa) installs its own NOC — **multi-admin** via a new commissioning window.

Apple's role

- Home app / HomeKit commission Matter accessories (iOS 16.1+); a **home hub** (HomePod mini, HomePod 2nd gen, Apple TV 4K with Thread) = Thread border router + remote access + automations.
- MatterSupport** framework: a third-party app commissions a device into *its own* ecosystem (MatterAddDeviceRequest); ThreadNetwork shares Thread credentials.
- No native Bluetooth Mesh** on iOS: the app implements the mesh stack (e.g. Nordic nRF Mesh) and talks to a **proxy node** over GATT (0x1828; provisioning 0x1827); never a relay.

Interview traps

- ! "Mesh = GATT links" — adverts; GATT only at the proxy.
- ! "Matter = Thread" — app layer vs one transport. No border router, no Internet.
- ! Relay privacy: NetKey forwards, AppKey reads — separate on purpose.

Likely questions

- iOS app for a BLE mesh? — own stack over CoreBluetooth via a proxy node.
- How does a Matter device join Google *and* Apple? — second fabric, own NOC.
- Why does an LPN need a friend? — flooding assumes listeners; the friend queues.