

Readiness report — Home Assistant + ESPHome Sensor Node

Checked 2026-09-03 15:50 · 3 searches, 9 pages fetched

Readiness: Home Assistant + ESPHome Sensor Node

Verdict: Ready with caveats · 2026-09-03 · est. parts cost \$40-\$85 per station (host board + one ESP node + cable) **Cybersecurity — software & network: Review before networking**

This is the cold check for this lab — there is no previous report, so it establishes the baseline every future check measures against. Sourcing itself is easy: this bill of materials has no PCB components and no manufacturer part numbers, just four commercial off-the-shelf lines, and all three that the lab actually names are cheap and stocked by many sellers today. Two things keep it off a clean "Ready". First, the lab's own sensor modules are never identified in any uploaded file, so one line of the bill of materials cannot be ordered as written. Second, the host board is no longer a free choice: Home Assistant Supervisor 2026.03.0 dropped the 32-bit machine types, and the board-support page the lab points instructors at has not caught up.

Action required

Part	Problem	Recommended action
Sensor modules and peripheral electronics	The lab's materials never identify them — no part numbers, interfaces or quantities in any uploaded file, and no ESPHome YAML config was included	Pick the sensors and add them to the bill of materials before ordering. Anything chosen must run at 3.3 V logic levels. Nothing could be priced for this line this run.
Host computing platform (Home Assistant gateway)	Home Assistant Supervisor 2026.03.0 removed the armhf/armv7/i386 architectures and the machine types <code>raspberrypi</code> , <code>raspberrypi2</code> , <code>raspberrypi3</code> , <code>raspberrypi4</code> , <code>qemuarm</code> , <code>qemux86</code> , <code>odroid-xu</code> and <code>tinker</code> — but the Board support overview page the lab's README links to still lists Pi2, Pi3, 32-bit Pi4, ODROID-XU4 and Tinker Board	Buy a Raspberry Pi 4 Model B with at least 2 GB (or a Pi 5), an ODROID-N2+, or an x86-64 UEFI machine, and install the 64-bit image — the dropped <code>raspberrypi4</code> is the 32-bit build, not the board. Do not buy an ODROID-XU4 or Asus Tinker Board, and do not reuse an old Pi 2 or Pi 3. Also budget for the power supply, microSD card and Ethernet connection the installation page requires; none of the three is in this bill of materials and none was priced this run.

Cybersecurity — software & network

Review before networking — Every pinned toolchain dependency checked clean except an Intel-macOS-only cryptography pin, but ESPHome's documented defaults leave a node's control, OTA and fallback-AP surfaces open unless the instructor configures authentication and encryption keys, and the lab ships no YAML to show which defaults apply.

Component	Version	How known	Support status
Home Assistant Operating System (HAOS)	not determined	not determined (haos_README.md names Buildroot LTS, Docker Engine, GRUB/U-Boot, RAUC and AppArmor as the platform but gives no release tag)	—
ESPHome (Python CLI/compiler and generated firmware)	not determined	not determined (no ESPHome release version appears in any file uploaded with the lab; requirements.txt pins its dependencies only)	—
cryptography (non-Intel-macOS build hosts)	50.0.1	read from project (fetched/01-requirements.txt (esphome requirements.txt, identical to the copy uploaded with the lab))	—
cryptography (Intel macOS build hosts)	48.0.1	read from project (fetched/01-requirements.txt, second cryptography line, environment marker platform_system == "Darwin" and platform_machine == "x86_64")	—
aioesphomeapi (ESPHome native API client)	46.3.0	read from project (fetched/01-requirements.txt)	—
paho-mqtt (MQTT client)	1.6.1	read from project (fetched/01-requirements.txt)	—
zeroconf (mDNS discovery)	0.151.3	read from project (fetched/01-requirements.txt)	—
bleak (Bluetooth LE)	3.0.2	read from project (fetched/01-requirements.txt)	—
esptool (flashing)	5.3.1	read from project (fetched/01-requirements.txt)	—
platformio (build driver)	6.1.19	read from project (fetched/01-requirements.txt)	—
requests	2.34.2	read from project (fetched/01-requirements.txt)	—
jinja2	3.1.6	read from project (fetched/01-requirements.txt)	—
PyYAML	6.0.3	read from project (fetched/01-requirements.txt)	—
ESP-IDF	not determined	unpinned (a comment in fetched/01-requirements.txt states "esp-idf >= 5.0 requires this"; the build pulls whatever the toolchain resolves, so no exact version exists to check)	—
ESP8266 Arduino core	not determined	not determined (snapshot records it as the toolchain driver invoked via ninja; no version in any provided file)	—

Component	Version	How known	Support status
tzdata / pyparsing / argcomplete	not determined	unpinned (fetched/01-requirements.txt declares minimum-version ranges only (tzdata >=2026.3, pyparsing >=3.3.2, argcomplete >=3.7.2))	—

Component	Finding	Severity	Fix available?
cryptography 48.0.1 (Intel macOS build hosts only)	On an Intel Mac used as the build host, the pinned cryptography 48.0.1 carries three known flaws (a padding-oracle in PKCS#7 decryption, an exponential certificate path-building slowdown, and a verifier that accepts wildcard names it should reject); every other build host installs 50.0.1, which the same check found clean. — GHSA-g6cj-pr64-35w5 , GHSA-jwv3-5hgf-82ww , GHSA-m2h6-j472-rp4c	high	No — project pins cryptography==48.0.1 for Intel macOS because 49+ ships no Intel macOS wheels; build on Linux or Apple silicon instead and the pin becomes 50.0.1.
ESPHome captive_portal fallback AP	If a node cannot reach Wi-Fi, captive_portal raises the node's own fallback access point and auto-loads the web OTA /update endpoint, so a node that loses the classroom network becomes an open flashing target for anyone in radio range.	posture	Mitigation only — omit captive_portal: from classroom configs, or pair it with web_server auth: and an OTA password
ESPHome native API and OTA encryption	The native API link and OTA uploads are encrypted and authenticated only when the config supplies a Noise NNpsk0 encryption key or a password; the documented default on a node with no encryption: block is plaintext OTA, authenticated at most by an OTA password.	posture	Mitigation only — generate a per-device API encryption key and an OTA key or password before the nodes go on a shared network
ESPHome web OTA endpoint (/update)	The web OTA firmware-upload endpoint / update is not covered by the Origin check at all, so wherever web OTA is enabled without web_server auth: anyone who can reach the node can flash their own firmware onto it.	posture	Mitigation only — set web_server auth: and an OTA password, or leave the web_server OTA platform out of the config
ESPHome web_server component	Where a node's YAML enables web_server without an auth: block, every entity view and control endpoint is reachable by anyone who can reach the node, and the Origin header check that blocks browsers is bypassed by design by any non-browser client such as curl or a script.	posture	Mitigation only — set web_server auth: credentials and keep the nodes on an isolated VLAN or SSID

What this means for your lab

Every pinned Python package in the ESPHome toolchain was checked this run and only one came back with known problems, and that one only bites if you build on an Intel Mac. The real risk in this lab is not a vulnerable library, it is what the sensor nodes do by default.

ESPHome's own threat model says plainly that a node's web interface is an open control surface unless the YAML sets a password: with it enabled and no auth, anyone else on the same Wi-Fi can read your students' sensor readings and flip any switch the node controls, and if web OTA or the captive-portal fallback is on, they can upload their own firmware to the device. The lab ships no YAML configs, so which of these apply depends entirely on the configs your students write. Before these nodes go on a campus network: give each device an API encryption key and an OTA password, set `web_server` auth: if you enable the web interface, leave `captive_portal` out of classroom configs, and ask campus IT for an isolated VLAN or SSID for the devices. Nothing here says the Home Assistant hub itself is safe or unsafe — the lab never states which HAOS release it uses, so that side could not be checked at all.

Not checked

- Home Assistant Operating System (HAOS) and its Buildroot/Docker/GRUB/U-Boot/RAUC/AppArmor stack: no release version in the lab's materials, so neither a support-status nor an advisory lookup was possible
- ESPHome itself (compiler and firmware): no release version in the lab's materials; only its Python dependencies could be checked
- ESP-IDF: unpinned range (≥ 5.0) — no exact version to check
- ESP8266 Arduino core: version not stated in any provided file
- `tzdata` / `pyparsing` / `argcomplete`: unpinned minimum-version constraints in `requirements.txt` — no exact version to check
- ESP32/ESP8266 firmware libraries pulled by PlatformIO at build time: no vulnerability database covers vendor embedded firmware libraries; absence of advisories here is not evidence of safety
- MQTT broker: the materials name `paho-mqtt` as the client but no broker, port or credentials, so there is nothing to check

Parts

Bill of Materials and Hardware Requirements

Part	Spec/Package	Status	Stock	Price	Source
Host Computing Platform (Home Assistant gateway/server)	SBC or x86-64 UEFI PC; 5 V DC or 100–240 V AC; min. 2 GB RAM per the HA installation page	readily available	multiple sellers	\$35.00 (RAM not stated in listing); \$55.00 2GB; \$61.05 4GB starter kit	Seeed Studio / PiShop USA / Vilros (Google Shopping, 2026-09-03)
Embedded Sensor Node Microcontroller Development Board	ESP32/ESP8266 dev board, DIP pin headers; 5 V USB in, 3.3 V logic	readily available	multiple sellers	\$8.00 (ESP32-DevKitM-1); range \$3.90–\$14.99 across marketplace sellers	Mouser via Google Shopping, 2026-09-03
Host-to-Node Serial Programming Cable	USB-A to micro-USB (or USB-C), data-capable, USB 2.0	readily available	multiple sellers	\$2.37 ; range \$0.80–\$8.22	Monoprice via Grainger (Google Shopping, 2026-09-03)
Sensor Modules and Peripheral Electronics	Not stated — must interface at 3.3 V logic	not checked	—	—	—

Notes on the rows above:

- **Host board.** The \$35.00 listing does not state its RAM; the Home Assistant installation page fetched this run gives the requirement as "Raspberry Pi 4 or 5 with power supply (minimum 2 GB RAM), microSD memory card, Ethernet connection", so confirm the variant on the product page before ordering. The ODROID-N2+ and x86-64 mini-PC options the snapshot also allows were not priced this run.
- **ESP board.** The lab pins no specific board — "any standard ESP8266 or ESP32-family development board supported by PlatformIO and ESPHome". The priced ESP32-DevKitM-1 uses the ESP32-MINI-1 module rather than the WROOM-32 the snapshot names as its example; both are ESP32-family boards on DIP headers, which is all the lab requires. Other listings this run: ESP32-DevKitC WROOM-32D \$3.90 (eBay), Inland ESP32-WROOM-32D \$7.99 (Micro Center), KUONGSHUN ESP32-DevKitC \$7.99 (Hand Held Legend), ShillehTek ESP-WROOM-32 CP2102 USB-C \$8.99, ESP32-DevKitC WROOM-32U \$14.99 (Walmart seller).
- **Cable.** The lab's one hard requirement for this line is that the cable carries data, not power only — and none of the listings states that in its search result, so check the product page before ordering. Only micro-USB was priced, per the lab's own sourcing plan; if the ESP board you buy has a USB-C port, you need a USB-C data cable instead, which was not priced this run.

- **Sensors.** Nothing was checked for this line because nothing was specified — see Action required.

Substitutes

None needed.

Other dependencies

- **Board fabrication: none.** No Gerbers, board dimensions or layer counts exist in the lab's materials; every piece of hardware is commercial off-the-shelf.
- **External pages checked this run — all reachable (2026-09-03):** [ESPHome security best practices](#), [Home Assistant installation](#), [Home Assistant getting started](#), [Board support overview](#), and the [HAOS development-builds index](#) (reachable; its listing is rendered by JavaScript, so only the page title came back as text).
- **One of those pages is stale.** The Board support overview still lists Pi2, Pi3B 32/64-bit, Pi4B 32-bit, ODROID-XU4 and Tinker Board in its board table, which Supervisor 2026.03.0 no longer supports. Treat it as a starting point, not a shopping list — see Action required.
- **Accessories the lab omits.** The Home Assistant installation page requires a power supply, a microSD card and an Ethernet connection for a Raspberry Pi host. None appears in the bill of materials, and none was priced this run.
- **Not checked this run:** the remaining links in the lab's materials — [esphome.io](#), the ESPHome GitHub issues and discussions pages, the Device Builder threat model, [home-assistant.io](#) and its /apps/ page, the developer-docs root, ADR-0015 and ADR-0017, and the six upstream project sites (Buildroot, GRUB, U-Boot, Docker Engine, RAUC, AppArmor). No evidence was gathered on them.

Watch list

- **Sensor modules** — the open line. Until the lab names its sensors, no future check can price or verify them, and no cost estimate here is complete.
- **Raspberry Pi machine types.** The 32-bit removal has already landed; the next thing to watch is whether `raspberrypi3-64` and the older ODROID types follow, and whether the developer-docs board table is corrected. Re-read the Supervisor release notes before each cohort orders hardware.
- **The \$35.00 Raspberry Pi listing.** It sits well below every other Pi 4 offer priced this run and does not state its RAM; it may be a low-memory variant that no longer meets the 2 GB minimum. Worth re-pricing next check.
- **ESP dev-board connector drift.** Newer ESP32 boards ship USB-C rather than micro-USB. The cable line was priced as micro-USB only, so a board change silently invalidates it.

- **Raspberry Pi 4 supply generally.** Prices this run spread from \$35.00 to \$249.18 for nominally the same board, which usually signals thin authorized stock and marketplace markup. If the low end disappears, the per-station cost roughly doubles.