skynew 4LAN ファンレス小型PC IN-1｜Ubuntu ServerとDocker導入および基本動作試験報告書

ファンレス小型PC 4LAN 2.5G対応試験日 2026年10月1日

判定

インストール検証 PASS。Dockerのhello-world実行、3コンテナの起動、基本動作13項目すべての合格を確認した。本体再起動後もアプリが復帰し、進捗・共有ファイル・フォルダーが保持され、MQTTの更新が再開した。

試験環境

項目	記録
対象	SKYNEW IN-1 ファンレス小型PC
ネットワーク仕様	4LAN 2.5G対応
OS	Ubuntu Server 24.04.5 LTS amd64
CPU	Intel Pentium J3710 4コア 1.60GHz
メモリ	約4GB OS記録3824MB
ストレージ	KingFast 64GB級 SSD OS表示59.6GiB
root領域	LVM論理ボリューム28.3GiB
Docker Engine / Compose	29.8.2 / 5.5.1
アプリ	Node-RED 4.1系 / Mosquitto 2.1.2 / Flask 3.1.3
通信構成	有線NIC4個認識 enp1s0接続 同一LAN内HTTP
実施方法	インストールログ・ブラウザー操作・JSON検証・再起動後確認

検証時刻は基本動作が10:54:00 JST、再起動後確認が10:56:54 JST。原ログの01:54:00Zと01:56:54Zを日本時間へ換算した。試験時間や長時間連続稼働時間を、この2点から推定していない。

基本動作13項目の結果

No	確認項目	結果
1	operatorでログイン	PASS
2	viewerでログイン	PASS
3	未認証での状態取得を拒否	PASS
4	MQTTサンプル受信	PASS
5	模擬生産カウンター増加	PASS
6	viewerによる進捗変更を拒否	PASS
7	viewerによるアップロードを拒否	PASS
8	CSRFトークンなしの更新を拒否	PASS
9	模擬異常通知を表示	PASS
10	別クライアントから保存進捗を確認	PASS
11	ダウンロードとアップロードのSHA256一致	PASS
12	作成フォルダーを一覧で確認	PASS
13	viewer画面のアップロード操作を無効化	PASS

出典 基本動作確認.json checked_utc 2026-10-01T01:54:00Z。13件のpassはいずれもtrue。MQTT受信確認時のサンプル数は23件。

ファイル一致の証拠

対象ファイル 業務デモ資料.txt

SHA256

0989be34a8ae9bbf4c870cdbff3bdfc0f4a7d494afa7b1f67403168930ea9ff1

アップロード前後および再起動後の記録で同じハッシュを確認した。共有ファイルの内容保持を裏付ける結果である。

再起動後の復帰と観測値

確認内容	記録
ログイン	true
進捗の保持	true
フォルダーの保持	true
ファイル内容	SHA256一致
MQTTの再開	true
カウンター更新再開	true

観測値	基本動作確認時	再起動後
時刻 JST	10:54:00	10:56:54
利用可能メモリ	3080MB	3240MB
保存領域の空き	19.49GB	19.47GB
1分ロード平均	1.03	0.14

上記は別時点の瞬間値であり、再起動による性能改善や長時間安定性を示す比較実験ではない。CPU温度の連続記録は本試験資料に含まれない。



図 生産進捗の操作画面 実機アプリで保存と別クライアントへの反映を確認

Ubuntu ServerとDockerのインストール

1 準備とOSの確認

Ubuntu Server 24.04 LTSのamd64版、インターネット接続、sudo権限のあるユーザー、同一LANの操作PCを用意します。OSを新規導入する場合は、必要なデータを退避してから公式ISOでUSBインストールメディアを作成し、Boot Menuから起動します。インストール先のディスクを確認し、遠隔操作する場合はOpenSSH Serverを有効にします。

cat /etc/os-release

ip -br address

lsblk -f

df -h /

sudo vgs

sudo lvs

有線LANのIPv4アドレスを確認し、ルーターのDHCP予約などで固定します。Dockerの初回導入・イメージ取得にはインターネット接続が必要です。既存Docker環境がある場合は、公式手順で競合パッケージと既存データを確認してから進めます。

2 Docker公式APTリポジトリから導入

sudo apt update

sudo apt install -y ca-certificates curl

sudo install -m 0755 -d /etc/apt/keyrings

sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg ¥

-o /etc/apt/keyrings/docker.asc

sudo chmod a+r /etc/apt/keyrings/docker.asc

. /etc/os-release

sudo tee /etc/apt/sources.list.d/docker.sources >/dev/null <<EOF

Types: deb

URIs: https://download.docker.com/linux/ubuntu

Suites: ${UBUNTU_CODENAME:-$VERSION_CODENAME}

Components: stable

Architectures: $(dpkg --print-architecture)

Signed-By: /etc/apt/keyrings/docker.asc

EOF

sudo apt update

sudo apt install -y docker-ce docker-ce-cli containerd.io ¥

docker-buildx-plugin docker-compose-plugin

sudo systemctl enable --now docker

sudo docker version

sudo docker compose version

sudo docker run --rm hello-world

hello-worldの成功メッセージを確認できれば、Dockerの導入確認は完了です。同梱のinstall_docker.shには同等のコマンドがあり、内容確認後にsudo bash install_docker.shで実行できます。新規インストール時のバージョンは取得時点で変わるため、表示結果を記録します。

3つの業務アプリを起動する

3 同梱設定をIN-1へ配置

配布パッケージの08_導入手順・設定ファイル/in1-demoフォルダーをSCPやUSBでIN-1へ転送します。以下はその親フォルダー内で実行する例です。/opt/in1-demoが既にある場合は上書きせず、既存構成を先に確認してください。

sudo cp -a in1-demo /opt/in1-demo

cd /opt/in1-demo

sudo cp .env.example .env

sudo nano .env

.envのIN1_BIND_IPを、このIN-1に割り当てられた実際の有線LAN IPv4アドレスへ変更します。例の192.168.1.50をそのまま使わないでください。

IN1_BIND_IP=192.168.1.50

4 初期アカウントとコンテナを作成

sudo python3 setup.py

sudo docker compose config --quiet

sudo docker compose up -d --build

sudo docker compose ps

sudo docker compose logs --tail 100

setup.pyはoperatorとviewerのパスワードを初回にだけ表示し、認証情報を生成します。パスワードは安全な場所に保管してください。既存accounts.jsonがあると停止するため、再起動のたびに初期化し直す必要はありません。mqtt・nodered・webの3サービスが稼働していることを確認します。

5 ブラウザー操作と管理

同一LANのPCからhttp://IN-1のIPv4アドレス:8080へアクセスします。operatorは進捗変更、模擬警告切替、ファイル登録を操作できます。viewerは閲覧とダウンロード用です。共有デモは1ファイル最大10MBです。大容量ディスクを接続しても、このアップロード上限は自動では増えません。

Run on the management PC; replace USER and IP. #ファンレス小型PC #4LANPC

ssh -L 1880:127.0.0.1:1880 USER@192.168.1.50

SSH接続中に操作PCでhttp://127.0.0.1:1880を開くとNode-REDを管理できます。Node-REDの1880番は本体のloopbackに限定し、MQTTの1883番はDocker内部だけで使います。Webの8080番も管理されたLAN内に限定します。

6 動作確認と停止 再開

数値更新、警告の表示と解除、進捗の別ブラウザー反映、アップロードとダウンロード、viewerの更新拒否を順に確認します。メンテナンス時間に再起動し、保存した進捗とファイルが残り、MQTT更新が再開することまで確認してください。

sudo docker compose stop

sudo docker compose start

Reboot only after saving work and arranging downtime. #ファンレス小型PC #4LANPC

sudo reboot

容量を増やす方法とUSBストレージの設定

保存量が増える運用では、導入前にmSATA SSDの大容量品への交換を検討してください。用途に応じてUSB大容量ストレージやNASも選べます。まずOSの割り当て状況を確認すると、不要な交換を避けられます。

方法	主な用途	確認事項
mSATA交換	OS・Docker・データをまとめて増量	mSATAの規格・寸法・互換性、バックアップ、移行手順
USB SSDまたはHDD	共有資料やバックアップの保存	電源、固定マウント、常時接続、起動時の確認
NAS	ネットワーク上の共有資料とバックアップ	共有権限、通信経路、接続失敗時の停止設計

LVMの割り当てを先に確認

試験機はSSD全体が59.6GiB、LVM内のroot論理ボリュームが28.3GiBでした。rootの空き約19.5GBだけで物理ディスク全体の残量は判断できません。sudo vgsのVFreeを確認し、空きがある場合はバックアップ後に管理者が拡張を検討します。

sudo vgs

sudo lvs

Only if this LV is correct and unused VG space is available: #ファンレス小型PC #4LANPC

sudo lvextend -r -l +100%FREE /dev/ubuntu-vg/ubuntu-lv

このコマンドはVGの未割り当て領域をすべてrootへ割り当てます。別の論理ボリューム用に領域を残す場合は、容量を個別に指定します。mSATAを交換する場合は、電源を切って作業し、OSを再導入するか、適切な移行手順で復元した後に起動と保存を確認します。M.2 SSDとは規格が異なります。

7 USBを共有ファイルの保存先にする

以下は、既にext4で使用可能なUSBストレージを用意した例です。接続後にlsblk -fでUUIDを確認します。既存データがあるディスクを、この手順で初期化する必要はありません。

sudo apt install -y rsync

sudo mkdir -p /srv/in1-storage

sudo nano /etc/fstab

/etc/fstabへ次の1行を追加します。実際のUUIDに置き換えてください。

UUID=REPLACE_WITH_UUID /srv/in1-storage ext4 defaults,nofail 0 2

sudo systemctl daemon-reload

sudo mount /srv/in1-storage

findmnt /srv/in1-storage

sudo mkdir -p /srv/in1-storage/files

以降は後述の共通手順で/data/filesだけを移します。OS・Docker本体やアプリイメージは内蔵側に残るため、USB追加だけでrootの空きが増えるわけではありません。

NASへの保存とマウント後の起動管理

8 NASのSMB共有を使う場合

USBの代わりにNASを使う場合の例です。NASに専用共有フォルダーと読み書き用アカウントを作成し、USBと同時に同じ場所へマウントしないでください。SQLiteのbusiness.db、認証情報、セッション鍵は内蔵ディスクに残し、共有ファイルまたはバックアップだけをNASに保存します。

sudo apt install -y cifs-utils rsync

sudo mkdir -p /srv/in1-storage

sudo install -m 600 /dev/null /root/.in1-nas

sudo nano /root/.in1-nas

資格情報ファイルへNAS専用アカウントを記入します。パスワードを公開記事やコマンド履歴へ書かないでください。

username=NAS_ACCOUNT

password=NAS_PASSWORD

/etc/fstabには次の内容を1行で記入します。NAS_IP、SHAREは実環境へ変更します。

//NAS_IP/SHARE /srv/in1-storage cifs credentials=/root/.in1-nas,vers=3.0,_netdev,nofail,uid=0,gid=0,file_mode=0660,dir_mode=0770 0 0

sudo systemctl daemon-reload

sudo mount /srv/in1-storage

findmnt /srv/in1-storage

sudo mkdir -p /srv/in1-storage/files

9 USBとNASに共通する移行と起動手順

マウントが成功したことを確認してから実行します。共有フォルダーは本デモ専用とし、シンボリックリンクなどを外部から混在させないでください。

cd /opt/in1-demo

sudo docker compose stop web

sudo mountpoint -q /srv/in1-storage

sudo rsync -rt ./data/files/ /srv/in1-storage/files/

sudo cp compose.storage.example.yaml compose.storage.yaml

sudo cp in1-files.service.example /etc/systemd/system/in1-files.service

sudo systemctl daemon-reload

sudo systemctl enable --now in1-files.service

sudo systemctl status in1-files.service --no-pager

同梱の追加設定は/data/filesだけを切り替え、マウント前の誤起動を防ぐためWebのDocker自動再起動を止め、systemdでマウント確認後に起動します。この構成では、通常の起動・停止もsudo systemctl start in1-files.serviceまたはstopを使います。導入後は試験ファイルの登録・取得と本体再起動を確認してください。

保存先を増やしてもバックアップは必要

NASへの保存は、それだけではバックアップになりません。別世代のバックアップと復元確認を用意します。SQLiteをコピーするときはアプリを停止するか適切なバックアップ方法を使います。Dockerのイメージ領域まで移す場合は、/var/lib/dockerに加えてcontainerd側の保存場所も確認し、単なる共有ファイル移行とは分けて設計します。

確認範囲と導入前の確認事項

今回の結論は、SKYNEW IN-1でUbuntu Server上のDocker導入、3コンテナの起動、基本機能13項目、再起動後の復帰とデータ保持を確認できたことです。インストール検証はPASSです。

本試験は2026年10月1日の短時間の導入・基本動作確認です。長時間安定性、同時接続上限、CPU温度の連続推移、実センサー接続、病院での実運用、2.5G通信の実効速度、4LAN同時通信・ルーティング性能は検証対象ではありません。4つの有線NICを認識し、通信にはenp1s0のみを使いました。業種の動画と看板は活用イメージです。

ONLYOFFICEは文書共同編集の拡張候補で、今回の3コンテナには含まれません。公式Docker版の目安はメモリ4GB以上、空き容量40GB以上、SWAP4GB以上などです。試験機のroot空き約19.5GBと他アプリとの同居を考慮し、容量の増強だけでなくCPU・メモリも含めて検討してください。NASや別サーバーへ処理を分ける方法もあります。

USB・NAS・mSATAの拡張手順は導入設計例で、今回の基本動作試験とは別に確認が必要です。同梱アプリはLAN内HTTPデモです。外部公開にはTLS、認証強化、バックアップ、ネットワーク制御を別途設計してください。Docker公開ポートはUFWだけで制限できるとは限りません。

同梱の汎用配置用setup.pyは、原構成の保存先を/opt/in1-demoへ変更し、公開IPを.envで指定できるよう整理したものです。Python構文・設定構造の確認と、元ログが示す実機試験を区別しています。配布用構成を対象機へ再インストールした試験は、本報告には追加していません。

参照資料

Docker Engine のUbuntu導入https://docs.docker.com/engine/install/ubuntu/

Node-RED のDocker導入https://nodered.org/docs/getting-started/docker

ONLYOFFICE Docs Docker版の要件https://helpcenter.onlyoffice.com/docs/installation/docs-community-sys-reqs-docker.aspx

UbuntuのNAS共有マウントhttps://ubuntu.com/server/docs/how-to/samba/mount-cifs-shares-permanently/

Docker保存領域https://docs.docker.com/engine/daemon/

SQLiteのネットワーク保存に関する注意https://www.sqlite.org/useovernet.html

参照日 2026年10月1日。商標は各権利者に帰属します。

検索キーワード：skynew, IN-1, ファンレス小型PC, 4 LAN PC, 4LAN