Installation and first activation

For the server administrator · Fresh installation on Linux x86_64 with Docker Compose. One activation covers the installation and all its users.

1 Before running the installer

Open START-HIER.html. Find the archive and its .sha256 file in 00_Installation. Verify the checksum and prepare Linux, DNS, HTTPS certificates and credentials according to the installation manual. Have the Medlix purchase code issued for this installation ready.

2 Complete installation

After configure and install, ACTIVATION_REQUIRED is an expected result. Open the configured Uptime HTTPS address in your browser. The business application starts after the license passes verification.



3 With internet access

Enter the purchase code in Kaufcode and choose Online aktivieren (activate online). The server needs HTTPS access to lizenz.medlix.de. Wait for confirmation and application startup, then sign in with the account created during setup and configure MFA.

First activation on an isolated network

The Uptime server stays on the internal network. Only the second device needs internet access. Activation requires neither a file transfer nor terminal commands.

1 Open the request in Uptime

Enter the purchase code and select Mit QR-Code aktivieren (activate using QR). Treat the QR code as confidential. It contains the purchase code and the signed request for this installation.



Illustration with fictitious data. Scan the QR code on your own server screen.

2 Confirm on the second device

Scan with the camera, check that the HTTPS address is lizenz.medlix.de and confirm activation on the Medlix page. Copy or transcribe the entire grouped confirmation code.

3 Apply the grant locally

On the original Uptime screen enter it in Freischaltcode and select Freischaltung bestätigen (confirm activation). Sign in once Uptime is ready.

Operations, recovery and troubleshooting

Activation and user accounts

The grant belongs to the server installation. Create accounts and assign permissions as described in the administrator manual. Business screens, APIs and background processing are locked before activation. Provision integration API access separately if required.

Restart and backup

An activated perpetual license works after a restart while the license service is offline. Preserve the persistent /keys directory when recreating the container. A complete Uptime backup includes the installation identity. Encrypt backups, restrict access and verify restoration on an isolated target.

Occupied seat or invalid confirmation code

Confirm that the purchase code was issued for this Uptime installation. For a replacement server, check the existing backup and coordinate the transfer with Medlix. For an invalid offline confirmation, check the complete code and use the installation that created the request. Keep purchase codes, QR codes and private keys out of support messages.

Acceptance and existing installations

After activation, verify MFA login, permissions and a synthetic sample report. Use an empty target for a fresh install. Coordinate preactivation and cutover with Medlix for existing installations. Replacing a single application container takes restart time.

This edition combines the R3 business application with a license gate. Earlier statements about image identity with the previous server and immediate first login refer to the previous delivery. Use the release identity and checksums supplied in this package.

OPERATE. UNDERSTAND. VERIFY.

Installation and operations manual

Uptime Control 02.15.1

Detailed procedures, clear responsibilities and traceable checks.

Edition 6 November 2026
Linux x86_64 + Docker | Browser application

No real customer data in illustrations. Usage rights follow the separate agreement.

Contents

01 Installing this edition 3

02 Delivery baseline and package boundaries 3

03 Purpose and architecture 3

04 Platform standard and sizing 4

05 DNS, network and TLS 4

06 Prepare the host 4

07 Verify and extract the package 5

08 Prepare DNS, certificate files and initial configuration 6

09 Customer configuration 6

10 Installation and result 7

11 First login and business configuration 7

12 Files, permissions and credentials 7

13 Status, start, stop and logs 8

14 Complete backup 8

15 Restore to an empty target 9

16 Certificates and updates 9

17 Troubleshooting and fallback 9

18 Acceptance, handover and retirement 10

19 Vendor references and verification limits 10

20 Authorise additional application servers 11

21 Current acceptance and custom configuration 11

01 Installing this edition

Installer: Uptime-Control-02.15.1-linux-amd64.tar.gz in 00_Installation. The archive contains all three tested Linux-amd64 images, setup tools and current installation manuals. No C# application source or access to the previous server is required. It contains no real customer data or default login passwords.

This is a fresh-install package for an empty target. Do not install it over an existing Uptime deployment. The service/ticket module is enabled; the Integration API remains disabled until explicitly configured. Additional-server expansion is separate.

Prepare Linux x86_64, a local Docker daemon with the Compose plugin, Python3, OpenSSL, adequate capacity and two DNS names covered by a valid TLS certificate. Sizing guidance below is operational planning, not a guaranteed workload ceiling.

02 Delivery baseline and package boundaries

This package installs version 02.15.1 on a new Linux x86_64 system with Docker. All three container image IDs match the running server captured on 6 November 2026 at 12:27:25 UTC. Customer configuration is generated afresh.

This is a fresh installation without existing customer or operational data. The internal server backup is separate and cannot be consumed by this customer installer’s restore-new command because its structure differs.

The package includes the compiled application, database, identity service and installation helpers. No C# application source or development environment is required. Necessary browser files and readable installation scripts remain part of operation.

The installer is a Linux tar archive with install.sh, not a Windows EXE. A root-authorized Linux administrator provides DNS, certificates, firewall and backup storage.

03 Purpose and architecture

Complete fresh installation of Uptime Control02.15.1 on a dedicated Linux server with Docker. Linux/Docker administration skills and an authorised maintenance task are required. This is not an updater or migration tool for an existing customer installation.

Components: unchanged Uptime application image; PostgreSQL18.6 with separate uptime and identity databases; production-mode Keycloak 26.8.2; a one-time setup helper for schema, initial administrator and restricted runtime grants. The application runs without root privileges or automatic migrations. No sample business data is created.

Data flow: browser to HTTPS application on443; login to HTTPS identity on8443; both services to the internal database. PostgreSQL publishes no host port. An internal database network and a separate frontend network belong exclusively to this Compose project.

The operator owns the host, DNS, certificates, firewall, time synchronisation, accounts, backups and acceptance. Verify departures from this single-server architecture before production use. These manuals do not establish additional maintenance/hosting commitments.

04 Platform standard and sizing

Target platform: Linux x86_64; deployment reference Ubuntu Server 24.04 LTS with approved security updates. Docker Engine with Linux containers and the Docker Compose plugin is required. Command form: docker compose. Host tools: Python 3.12 or later, OpenSSL, tar and sha256sum. No .NET SDK or application source is required.

Initial planning allowance:4 vCPU,8 GB RAM,100 GB SSD plus separate backup storage. This is not a guaranteed user/load limit. Import archives and reports determine actual storage use. Allow additional space for the archive and uncompressed images during extraction/docker load.

Local verification used Linux containers with Docker Engine29.1.3 and Compose2.40.3. A complete Ubuntu customer host was not deployed here. Host permissions, hardening, actual firewall, reboot and capacity require customer acceptance.

Before starting, record server/IP, owners, maintenance window, DNS names, certificate expiry, storage/backup destination, acceptable data loss (RPO), recovery target (RTO), user/role model. Connect further application instances only after password-protected server authorisation and separate infrastructure acceptance; see the server-expansion chapter.

05 DNS, network and TLS

Point two distinct DNS names to the same server IP, for example uptime.example.com and login.example.com. Default ports:443 and8443. Both must be reachable from intended workstations. Restrict SSH22 to the administration network. Do not publish PostgreSQL5432, internal KeycloakHTTP8080 or management ports.

Permit DNS/NTP and necessary outbound update/certificate traffic under the operator policy. Verify Docker-published ports at the actual host/network firewall; ordinary firewall status alone does not prove isolation. Resolve occupied443/8443 ports or other service conflicts first.

Provide fullchain.pem containing the server certificate/intermediates and privkey.pem containing an unencrypted but permission-protected PEM key for unattended startup. The certificate must cover both DNS names in SAN entries. Disk encryption and restricted root access provide further protection.

Public CA: leave trust_ca empty. Internal CA: supply a PEM trust chain and separately establish trust on host/workstations. The installer checks key/certificate, SAN, chain and at least seven remaining validity days. Provide adequate validity and a renewal process for handover. Never bypass certificate failures by disabling checks.

06 Prepare the host

On a new dedicated Ubuntu 24.04 amd64 host without an existing Docker installation, run these commands in order. Use the appropriate operator procedure for other distributions or an existing container host. Do not duplicate an existing repository.

```
sudo apt-get update

sudo apt-get install ca-certificates curl python3 openssl

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

sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc

sudo chmod 0644 /etc/apt/keyrings/docker.asc
```

```
echo 'deb [arch=amd64 signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu
noble stable' | sudo tee /etc/apt/sources.list.d/docker.list

sudo apt-get update

sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin

sudo systemctl enable --now docker

sudo docker version

sudo docker compose version
```

Reference: https://docs.docker.com/engine/install/ubuntu/ (6 November 2026). Record installed versions.

Install Docker from the vendor repository following the official Ubuntu instructions. Do not execute unreviewed third-party scripts as root. Vendor instructions govern current repository keys/package names. Required: Engine/CLI, containerd and the Compose plugin. Docker access grants extensive host authority and is restricted to technical administrators.

Check prerequisites:

```
uname -m

python3 --version

openssl version

docker version

docker compose version

docker info

df -h

timedatectl status
```

Expected: x86_64, working local Linux daemon, sufficient storage and synchronised time. The installer rejects remote Docker contexts.

The installer does not automatically configure external backups, firewall rules or monitoring. Assign these responsibilities before operational approval. A shared host requires a separate conflict/permission review.

07 Verify and extract the package

Securely copy the archive and.sha256 file from 00_Installation to the new server. Compare the checksum through an independent agreed channel. Compiled container images are included; docker load does not require registry access. Host/Docker setup and certificates may still need network access.

From the transfer directory:

```
sha256sum -c Uptime-Control-02.15.1-linux-amd64.tar.gz.sha256

sudo install -d -m 0755 /opt/uptime-control-release

sudo tar -xzf Uptime-Control-02.15.1-linux-amd64.tar.gz -C /opt/uptime-control-release

cd /opt/uptime-control-release

sudo sh install.sh check
```

Expected: PACKAGE_AND_LOCAL_DOCKER_OK. Do not proceed after checksum failure. release.json contains file hashes and exact image IDs; obtain both package and manifest from a trusted source. SHA256 is not a digital vendor signature.

The future runtime directory /opt/uptime-control must be new and empty. Keep package/runtime directories separate. The archive contains no customer data, private certificates or predefined production passwords.

08 Prepare DNS, certificate files and initial configuration

1. Have the operator point two DNS names at the same server address. Both names must also resolve and be reachable from browser clients. Defaults: application 443, login 8443.

2. Obtain a PEM certificate with intermediate certificates and matching private key covering both names from the responsible CA. For an internal CA, supply the trust chain to clients and the installer. Do not blindly use a self-signed test certificate in production.

3. Store files securely on the host, for example /etc/uptime-tls/fullchain.pem and /etc/uptime-tls/privkey.pem. Restrict the private key to technical administrators; the installer creates its runtime copies with required permissions.

4. Account for Docker port publishing in the network firewall; a simple UFW rule alone may be insufficient. Test from permitted and denied networks. Do not publish database port 5432 externally.

5. Copy customer.example.json to customer.json, edit it and check all eleven fields. Example: sudo cp customer.example.json customer.json; then sudo nano customer.json.

6. app_host and identity_host contain names only, without https://. certificate and private_key are absolute Linux paths. Leave trust_ca empty for a public CA; otherwise supply the CA-file path. admin_email identifies the responsible person.

7. Wait for CONFIGURED after configure. The personal initial password is requested privately at that point. Do not use a supplier activation password as a user password.

09 Customer configuration

Copy customer.example.json to customer.json. Replace all example values. Do not put passwords into the file.

app_host/identity_host: distinct DNS names without scheme/path. app_port=443 and identity_port=8443: default ports. bind_address=0.0.0.0: accessible to intended workstations, protected by firewall.127.0.0.1 is for local testing only.

admin_username: personal/accountable initial account. admin_name/admin_email: actual responsible person. certificate/private_key: absolute host paths to PEM files. trust_ca: empty for a public CA, otherwise an absolute PEM trust-chain path.

```
sudo sh install.sh configure --directory /opt/uptime-control --config customer.json
```

Enter the initial password twice without echo,12 to128 characters. Store an individually generated long value in a password manager and transfer separately. Technical passwords/OAuth secret are randomly generated and not printed.

CONFIGURED confirms preparation, not a running application. state.json holds technical secrets in the protected runtime directory. Never copy it into support tickets or PDFs.

10 Installation and result

Before starting: target belongs to this fresh installation, DNS/certificates are correct, ports are free, root operation is authorised and storage is sufficient.

```
sudo sh install.sh install --directory /opt/uptime-control
```

Sequence: load exact images; create PostgreSQL and two empty databases/roles; start production Keycloak; import new realm; configure password/OTP evidence; migrate Uptime schema; create initial administrator; grant restricted application permissions; start application; verify HTTPS readiness with certificate validation.

Success: UPTIME_READY and INSTALLATION_READY. installed.json records technical startup only. The initial password is removed from the realm import after success; technical secrets remain protected. Repeating install on a completed installation is refused. Use start for ordinary startup.

On interruption, do not delete unidentified directories/volumes. Inspect status/logs. No general automatic rollback is provided. Before installed.json exists, a repeated run may continue the incomplete setup; inspect cause and existing state first. A full retry is limited to a positively identified isolated installation following a backup decision.

11 First login and business configuration

Open the configured HTTPS address and check redirection to the separate identity service. Sign in with the initial account, enrol OTP in the personal authenticator, confirm and sign in again. Do not photograph/send the QR secret. On code errors synchronise time and wait for the next code.

Verify correct person, administrative access, demo mode off, service/ticket module enabled, separate server expansion not implicitly enabled and absence of sample airports/machines. Create a synthetic test account through the application and verify actual identity provisioning. uptime.provisioner is a technical account without a business role; do not disable it.

Business setup order: sender/company; airports with country/IANA timezone; areas; machines with codes/serial numbers; contract versions with validity, operating windows, allowance and targets; personal users with correct assignments.

With approved synthetic files, check preview/import, authorised service correction, reports, PDF/CSV and denied auditor changes. Do not leave test records as production performance data. Import real files only after recorded approval.

12 Files, permissions and credentials

state.json: parameters, project identifier and technical secrets. compose.json: derived container configuration including secrets. tls/: certificates/private key. keys/: Data Protection keys and any protected activation receipts. init.sql: DB roles. realm.json: realm/client settings. setup/: compiled initialisation/check helper.

The uniquely named PostgreSQL volume contains uptime and identity. A volume is not a backup. release.json records image IDs/package state. Do not distribute runtime files with ordinary customer documents.

Linux permissions: top runtime directory0700; keys owned byUID/GID1000; TLS directory permits container group1000 read access. Verify ownership/modes after moving; never use chmod777. Strictly limit Docker/root authority.

Transfer separately and securely: Linux/SSH, initial application administrator, Keycloak emergency administrator, backup/password-manager access. Record channel/receipt, not secrets, in acceptance forms. Do not change technical passwords in state.json alone: database/identity and runtime configuration must remain consistent.

13 Status, start, stop and logs

Run from the retained package directory:

```
sudo sh install.sh status --directory /opt/uptime-control
```

```
sudo sh install.sh stop --directory /opt/uptime-control
```

```
sudo sh install.sh start --directory /opt/uptime-control
```

status shows only this project and checks HTTPS readiness. stop cleanly stops application, identity and DB without deleting volumes. start starts these services. Restart via stop/start during maintenance.

Targeted docker compose logs need the project identifier from state.json and the actual compose.json. Do not paste complete resolved configuration into tickets. Logs rotate at10MB/five files per service; this does not replace central long-term logging.

Restart policy unless-stopped: enable Docker at boot and test an actual host reboot with the customer. Intentionally stopped services remain stopped according to Docker semantics. Monitor readiness, login, storage, certificate expiry and backup success. Define thresholds/alert recipients.

Do not run installation, backup and restore commands concurrently. One responsible operator coordinates the maintenance window.

14 Complete backup

The tool backs up both databases, application keys, TLS material and required configuration. A UI application-backup download does not replace this complete operational backup.

Create a destination outside the runtime directory. Backup pauses application and identity; schedule maintenance. PostgreSQL remains available for consistent dumps. Restart is attempted after success/failure. The final archive name is created only after all components succeed.

```
sudo install -d -m 0700 /var/backups/uptime-control
```

```
sudo sh install.sh backup --directory /opt/uptime-control --backup-dir /var/backups/uptime-control
```

Expected: BACKUP_COMPLETE with path and readiness. Each.tar.gz receives a.sha256 file. The archive contains highly sensitive keys, passwords and identities. It is NOT automatically encrypted. Encrypt using the agreed backup system before off-host storage and keep recovery keys separately.

A same-server copy does not protect against server loss. Document off-host copying, integrity checks, retention, restore exercises and alerts. No timer/cloud/NAS access is automatically configured. Test recovery after the first backup and relevant changes; rotate only after a verified new backup.

15 Restore to an empty target

Required: matching package/images, complete backup/checksum, TLS/CA/DNS access, free space and a new empty runtime directory. restore-new does not overwrite an existing installation. Version must match the backup.

Before switching, stop the old instance and preserve fallback. Identical ports cannot be occupied simultaneously on one host. On replacement hardware, change DNS only after successful acceptance; never accidentally provide two writable instances.

```
sha256sum -c uptime-backup-TIMESTAMP.tar.gz.sha256
```

```
sudo sh install.sh restore-new --directory /opt/uptime-control-restored --archive
/secure/path/uptime-backup-TIMESTAMP.tar.gz
```

Replace example path/TIMESTAMP. The tool verifies internal hashes, creates a new Compose project/DB, restores both databases and keys/configuration, then starts services. RESTORE_READY indicates technical readiness.

Sign in using an existing account and the same authenticator; verify master data, roles, import archives, report revisions, exports and module locks. Compare record counts/business samples against backup state. Release users only afterwards. On failure isolate the target, retain old installation/backup and do not present an empty replacement DB as successful recovery.

16 Certificates and updates

Monitor expiry in advance. Validate replacement names, chain, matching key and validity. Take a complete backup. During maintenance stop services, replace TLS files while preserving permissions/ownership, restart and test HTTPS/login from a normal client without certificate exceptions.

Domain/port/CA changes jointly affect OAuth redirects, Keycloak hostname, trust and application settings. Use a separately verified change plan; editing a file is not automatic migration.

This package has no general existing-system updater. Future updates require a defined starting state, verified package, backup, migration rehearsal, fallback and acceptance. Never set images to latest or replace individual components without review.

The operator maintains OS/Docker security updates; afterwards verify readiness, login, roles, import/export and backup. Version contractual changes separately. Customer source changes leave the verified delivery state.

17 Troubleshooting and fallback

Checksum error: check transfer/reference; do not rewrite the manifest. Remote Docker: deliberately select local context. Target not empty: choose a new directory or treat it as an existing installation; do not indiscriminately delete.

Identity not ready: inspect DB, storage, TLS permissions, hostnames and logs. Readiness/TLS error: inspect chain, SAN, CA trust and app/DB connection; do not disable security validation.

Login without access: inspect account and active scope. OTP rejected: synchronise time/wait for a new code. Account creation error: inspect technical provisioning account, complete realm profile and HTTPS connection. Unexpectedly locked service/ticket module: check delivered configuration and role permissions. Only the additional-server expansion requires its separate authorization.

Backup failure: inspect status/space and repeat a complete backup. Restore failure: preserve unchanged old instance/backup. Do not apply generic manual SQL fixes to business data. Support evidence: time/timezone, version, phase and sanitised error/reference. Never send state.json, private keys or dumps unprotected.

18 Acceptance, handover and retirement

Complete the separate acceptance form: valid TLS, URLs, external DB isolation, OTP, users/roles, master data/rules, import/correction/export, module locks, complete backup with encrypted off-host copy, recovery including identities/keys, host reboot and named operational owners.

Retain package checksum, image IDs, actual non-secret configuration, acceptance date and contacts. Keep passwords/recovery material in a separate vault. Treat RPO/RTO/capacity as commitments only after explicit agreement and verification.

Retirement: agree retention/export, verify full backup and revoke access in a controlled manner. stop deletes nothing. Deleting volumes/identities/keys is a separate irreversible action outside installation.

Supply source only if agreed. It is not required for binary operation. Preserve third-party notices and the component inventory.

19 Vendor references and verification limits

Vendor references, accessed 6 November 2026:

https://docs.docker.com/engine/install/ubuntu/

https://docs.docker.com/compose/install/

https://www.keycloak.org/server/enabletls

https://www.keycloak.org/server/configuration-production

https://www.keycloak.org/server/db

Actual package configuration and installation evidence define the verified state. Vendor information may change; recheck before later platform updates.

Fresh-install rehearsal and earlier source/business verification are separate. New Linux containers and synthetic accounts were used locally; no production data in this fresh-install test. Physical Ubuntu host, public DNS/CA, customer firewall, external NAS/cloud backups and monitoring remain customer acceptance items.

Customer-specific settings and certificates are required installation inputs. The operator records final operational approval after verifying the actual environment.

20 Authorise additional application servers

Server expansion uses its own supplier password. Initial installation creates a new installation ID and an empty node registry. No additional server is automatically started. The primary application may remain in Single mode.

In the extracted package directory on the primary Linux host:

```
sudo python3 modules.py status --directory /opt/uptime-control
```

Create an individual permit; the password is prompted without echo:

```
sudo python3 modules.py server-release --directory /opt/uptime-control --node node2 --output
/secure/node2.zip
```

The protected destination directory must exist and the output file must not exist. Names use 1-48 letters, digits or hyphens; node1 is reserved. Wrong passwords, duplicate names, concurrent releases and existing output files are rejected. The local primary container must run the matching image.

If authorisation succeeded but export was interrupted:

```
sudo python3 modules.py server-export --directory /opt/uptime-control --node node2 --output
/secure/node2.zip
```

The confidential ZIP contains the individual permit, registry, image identity and administrator template. It contains neither owner.json nor an activation password. It is not a complete second-host installer. Use separate names and permits for further nodes.

Connect the additional host

First prepare private networking, the existing shared PostgreSQL database and Keycloak, persistently shared keys, correct TLS certificates, load balancing and connection budgets. Do not create a new empty database or identity realm.

Load the identical application image from the installation archive; INSTALLATION.json records its exact ID. Use compose.additional.yml and a protected runtime.env containing the existing application/identity/database settings. Mount only the individual permit and registry read-only. Keep owner.json on the primary management host. UID 1000 requires access to shared keys and certificates. The additional host must resolve the HTTPS identity endpoint and validate its certificate.

If the ticket module is active, share its protected receipt and keys. A one-time key copy is insufficient because keys rotate. Distribute updated registries to existing nodes in a controlled way after changes.

Accept login/session switching, permissions, imports, persistence, reports, tickets, node failure and recovery on the real infrastructure. Local checks for this edition are not a physical multi-host test or capacity commitment.

21 Current acceptance and custom configuration

For 02.15.1, clean installation, real browser MFA login, user provisioning, full backup/restore with 86 identical application tables and post-restore login passed. The API additionally passed 17 checks using actual Keycloak tokens. This does not pre-approve the customer environment.

Important: separately protect and back up custom Compose/API configuration and test its recovery. restore-new regenerates the standard configuration from saved installation state; arbitrary later manual Compose changes are not promised to be restored automatically. Standard-backup acceptance used API-disabled configuration.

Use valid certificates for real access. Inside a VPN, both application and identity names must resolve and be reachable. Check Docker-published ports independently of host-firewall assumptions. Do not publish database ports.

Official host preparation: https://docs.docker.com/engine/install/ubuntu/ and https://docs.docker.com/compose/install/linux/. Sources checked 6 November 2026. Do not remove unrelated containers or Docker packages from an existing customer host without review.