Version 2026.11.09

Hofstetter Computer Consulting
Deployment, configuration, and security guidance
Build: 45f3fb6ea56f

Table of Contents

Introduction 4
Prerequisites 4
System Requirements 5
Release Bundle Contents 5
Prepare the Environment 6
GitHub Container Registry Authentication 6
Generate Secure Values 6
Retrieve the Cognos CA Certificate 7
Generate a Local HTTPS Certificate for NGINX 8
Environment Variables Explained 8
Required Secrets 8
Initial Admin Account 9
Cognos Configuration 10
NGINX Configuration 11
PostgreSQL Settings 11
Django & Proxy Settings 12
Application Runtime Settings 12
Security Variables with Safe Defaults 13
Start SimpliLoad 13
Start the Trial Version 13
Start the Full Licensed Version 14
Upgrade SimpliLoad 14
Stop SimpliLoad 15
Remove SimpliLoad 15
Security Overview 16
Key Management 16
Key Rotation Procedure 16
Threat Model Summary 17
Data Handling Statement 17

Operations 18
Architecture Overview 18
Operational Considerations 19
Versioning Policy 19
Licensing 19
Customer Responsibilities 20
Support 20
Troubleshooting 20

Introduction

SimpliLoad is a lightweight, containerized execution and load-testing tool designed for IBM Cognos Analytics environments. It provides a simple, reliable way to run Cognos reports, measure performance, and automate repeatable workloads without requiring any changes to your Cognos configuration.

SimpliLoad requires an IBM Cognos Analytics environment running version 12.1.3 or later, as earlier versions do not provide the REST API endpoints used for report execution.

This deployment guide explains how to install, configure, and operate SimpliLoad using Docker and Docker Compose. It covers environment preparation, secure configuration, runtime behavior, and operational considerations for both trial and full-license installations.

SimpliLoad enforces secure defaults, including strict HTTPS usage, certificate validation, and hardened Django settings.

The guide is intended for administrators responsible for deploying SimpliLoad in on-premises or cloud environments. Familiarity with Docker, Cognos Analytics, and basic Linux or Windows command-line usage is recommended.

Before beginning, review the prerequisites and ensure your environment meets the system requirements described in the next section.

Prerequisites

Before deploying SimpliLoad, ensure that you have a supported operating system, Docker installed, and access to an IBM Cognos Analytics environment with appropriate credentials and certificates.

System Requirements

- **Operating System:** Linux, Windows 10/11, or macOS with Docker Desktop.
- **Docker:** Recent version of Docker Engine or Docker Desktop with Docker Compose.
- **Cognos Environment:** An IBM Cognos Analytics environment running version 12.1.3 or later is required. Earlier versions do not support the REST API endpoints used by SimpliLoad.
- **Processor:**Minimum 1 vCPU for trial mode; 2-4 vCPU recommended for load testing workloads.
- **Memory:** At least 8 GB RAM recommended for running Cognos load tests.
- **Disk:** Sufficient space for Docker images, containers, and PostgreSQL data volume.
- **GitHub PAT (Full Version Only):** A GitHub Personal Access Token with `read:packages` permission is required to pull licensed SimpliLoad images from GHCR.

Release Bundle Contents

The SimpliLoad release bundle contains all files required to deploy the application using Docker Compose.

- **docker-compose.release.yml:** Compose file for the full licensed version.
- **docker-compose.trial.yml:** Compose file for the trial version.
- **.env.example:** Template environment file used to create your `.env`.
- **nginx.conf:** Example NGINX configuration used by the containerized reverse proxy.
- **SimpliLoad Deployment Guide.pdf:** These deployment instructions for SimpliLoad.
- **SimpliLoad EULA.pdf:** End-User License Agreement.
- **SimpliLoad Privacy Statement.pdf:** Privacy policy for SimpliLoad.
- **SimpliLoad Support Policy.pdf:** Support terms and coverage.

Prepare the Environment

Before starting SimpliLoad, prepare your deployment directory and environment configuration. Create a `.env` file based on the provided `.env.example` template and update the values according to your environment.

All configuration values described in the following sections must be defined in `.env` before starting the container.

GitHub Container Registry Authentication (Full Version Only)

Licensed SimpliLoad images are distributed through the GitHub Container Registry (GHCR). To pull these images, you must authenticate Docker using a GitHub Personal Access Token (PAT) with the `read:packages` permission. Trial images are public and do not require authentication.

Generate Secure Values

SimpliLoad requires several cryptographically secure values for authentication, encryption, and database access. Generate these values using the commands below and insert them into your `.env` file.

For PowerShell

```
# DJANGO_SECRET_KEY (50-character random string)
-join ((65..90) + (97..122) + (48..57) | Get-Random -Count 50 |
ForEach-Object { [char]$_ })

# SIMPLILOAD_MASTER_KEY (32-byte base64 key)
$bytes = New-Object byte[] 32
(New-Object
Security.Cryptography.RNGCryptoServiceProvider).GetBytes($bytes)
[Convert]::ToBase64String($bytes)

# POSTGRES_PASSWORD (24-character random password)
-join ((65..90) + (97..122) + (48..57) | Get-Random -Count 24 |
ForEach-Object { [char]$_ })

# DJANGO_SUPERUSER_PASSWORD (24-character random password)
-join ((65..90) + (97..122) + (48..57) | Get-Random -Count 24 |
ForEach-Object { [char]$_ })
```

For Linux

```
# DJANGO_SECRET_KEY
openssl rand -base64 38 | tr -d '\n'

# SIMPLILOAD_MASTER_KEY
openssl rand -base64 32

# POSTGRES_PASSWORD
openssl rand -base64 18

# DJANGO_SUPERUSER_PASSWORD
openssl rand -base64 18
```

Retrieve the Cognos CA Certificate

SimpliLoad must validate the HTTPS certificate presented by Cognos Analytics. Export the Root CA certificate that signed the Cognos server certificate and save it as `cognos.pem` inside `deploy/certs/`.

> **Important:** The certificate must be in PEM format. DER or PFX files are not accepted.

Generate a Local HTTPS Certificate for NGINX

If you do not have an internal CA-issued certificate, generate a self-signed certificate for local access to the SimpliLoad web interface.

> **Browser Warning:** Self-signed certificates are not trusted by browsers. Users will see a security warning and must manually proceed or import the certificate into their trusted store.

Generate Certificate Using Docker (Recommended)

```
docker run --rm -v "$PWD:/certs" alpine \
  sh -c "apk add openssl >/dev/null && \
         openssl req -x509 -nodes -days 730 \
         -newkey rsa:2048 \
         -keyout /certs/simpliload.key \
         -out /certs/simpliload.crt \
         -subj '/CN=simpliload.local'"
```

Environment Variables Explained

All configuration for SimpliLoad is performed through environment variables. Each variable is documented below with a description, recommended values, and any relevant warnings.

Required Secrets

DJANGO_SECRET_KEY

A long, random string used by Django to sign sessions and internal cryptographic operations. This value must remain secret and should never be reused across environments.

```
DJANGO_SECRET_KEY=replace-with-a-generated-secret
```

> **Warning:** Changing this value after deployment will invalidate all active sessions.

SIMPLILOAD_MASTER_KEY

A 32-byte base64-encoded key used to encrypt sensitive data stored in PostgreSQL, including Cognos API keys. This key must be backed up securely.

```
SIMPLILOAD_MASTER_KEY=replace-with-a-base64-master-key
```

> **Critical:** If this key is lost, encrypted credentials cannot be recovered.

Initial Admin Account

DJANGO_SUPERUSER_USERNAME

Username for the initial administrative account created during first startup.

```
DJANGO_SUPERUSER_USERNAME=admin
```

DJANGO_SUPERUSER_EMAIL

Email address associated with the initial administrative account.

```
DJANGO_SUPERUSER_EMAIL=admin@example.com
```

DJANGO_SUPERUSER_PASSWORD

Password for the initial administrative account. This value is used only during first startup.

```
DJANGO_SUPERUSER_PASSWORD=replace-with-a-superuser-password
```

Cognos Configuration

COGNOS_CERT_PATH

Internal container path where the Cognos CA certificate is mounted. This path is fixed and must not be changed.

```
COGNOS_CERT_PATH=/run/secrets/cognos.pem
```

> **Do Not Modify:** To change the host-side certificate location, update `COGNOS_CERT_HOST_PATH` instead.

COGNOS_CERT_HOST_PATH

Host-side path to the Cognos CA certificate. Update this if your certificate is stored in a different location.

```
COGNOS_CERT_HOST_PATH=./deploy/certs/cognos.pem
```

COGNOS_VERIFY_HOSTNAME

Enables TLS hostname verification when connecting to Cognos. This should remain enabled unless using a development certificate.

```
COGNOS_VERIFY_HOSTNAME=true
```

COGNOS_ALLOW_HTTP

Allows HTTP connections to Cognos. This should remain disabled to prevent accidental downgrade to insecure transport.

```
COGNOS_ALLOW_HTTP=false
```

COGNOS_ALLOWED_HOSTS

Comma-separated list of allowed Cognos hostnames. This prevents accidental connections to unauthorized servers.

```
COGNOS_ALLOWED_HOSTS=cognos.example.com
```

NGINX Configuration

SSL_CERT_HOST_PATH

Host-side path to the TLS certificate used by NGINX for HTTPS.

```
SSL_CERT_HOST_PATH=./deploy/certs/server.crt
```

SSL_KEY_HOST_PATH

Host-side path to the private key associated with the TLS certificate.

```
SSL_KEY_HOST_PATH=./deploy/certs/server.key
```

PostgreSQL Settings

POSTGRES_DB

Name of the PostgreSQL database used by SimpliLoad.

```
POSTGRES_DB=simpliload
```

POSTGRES_USER

Username for the PostgreSQL database user.

```
POSTGRES_USER=simpliload
```

POSTGRES_PASSWORD

Password for the PostgreSQL user. Required for database initialization.

```
POSTGRES_PASSWORD=replace-with-a-database-password
```

Django & Proxy Settings

ALLOWED_HOSTS

Comma-separated list of hostnames allowed to serve the SimpliLoad web interface.

```
ALLOWED_HOSTS=localhost,127.0.0.1
```

CSRF_TRUSTED_ORIGINS

Full scheme + hostname values trusted for CSRF validation.

```
CSRF_TRUSTED_ORIGINS=https://localhost
```

Application Runtime Settings

ASGI_THREADS

Number of worker threads used by the ASGI server for concurrent Cognos requests. Increasing this value improves parallel execution but increases CPU usage.

```
ASGI_THREADS=20
```

LOG_LEVEL

Controls the verbosity of SimpliLoad logging. Higher levels (DEBUG) include detailed Cognos REST payloads, response previews, status codes, and async polling loop diagnostics.

```
LOG_LEVEL=INFO
```

SIMPLILOAD_RUN_TIMEOUT_SECONDS

Maximum allowed execution time for Cognos reports. This prevents stalled or long-running reports from blocking the queue.

```
SIMPLILOAD_RUN_TIMEOUT_SECONDS=300
```

Security Variables with Safe Defaults

SimpliLoad enforces secure defaults for all deployments. These values should not be changed unless required by your environment.

SECURE_SSL_REDIRECT

```
SECURE_SSL_REDIRECT=1
```

SECURE_HSTS_SECONDS

```
SECURE_HSTS_SECONDS=31536000
```

SECURE_HSTS_INCLUDE_SUBDOMAINS

```
SECURE_HSTS_INCLUDE_SUBDOMAINS=false
```

SECURE_HSTS_PRELOAD

```
SECURE_HSTS_PRELOAD=false
```

Start SimpliLoad

Start the Trial Version

The SimpliLoad trial image is publicly available on GHCR and does not require authentication. Pull the image and start the application using the trial compose file:

```
docker compose -f docker-compose.trial.yml pull
docker compose -f docker-compose.trial.yml up -d
```

Start the Full Licensed Version

Licensed SimpliLoad images are private and require authentication before pulling. Authenticate Docker using your GitHub Personal Access Token (PAT) with `read:packages` permission:

```
echo YOUR_PAT | docker login ghcr.io -u YOUR_GITHUB_USERNAME --
password-stdin
```

After authentication, pull the licensed image and start SimpliLoad:

```
docker compose -f docker-compose.release.yml pull
docker compose -f docker-compose.release.yml up -d
```

Upgrade SimpliLoad

Pull the latest image and recreate containers while preserving persistent data:

```
# Full Version
docker compose -f docker-compose.release.yml pull
docker compose -f docker-compose.release.yml up -d --force-recreate

# Trial Version
docker compose -f docker-compose.trial.yml pull
docker compose -f docker-compose.trial.yml up -d --force-recreate
```

> **Note:** Licensed SimpliLoad images are private. If you have not already authenticated Docker to GHCR using your GitHub Personal Access Token (PAT), you must do so before running the upgrade commands. Trial images are public and do not require authentication.

Stop SimpliLoad

```
# Full Version
docker compose -f docker-compose.release.yml down

# Trial Version
docker compose -f docker-compose.trial.yml down
```

Remove SimpliLoad

Removing SimpliLoad consists of stopping the containers, deleting the Docker resources, and optionally removing the PostgreSQL data volume and deployment files.

1. Stop and Remove Containers

This command stops all SimpliLoad services and removes the containers, but preserves the PostgreSQL data volume:

```
# Full Version
docker compose -f docker-compose.release.yml down

# Trial Version
docker compose -f docker-compose.trial.yml down
```

2. (Optional) Remove the PostgreSQL Data Volume

To fully remove all stored data—including encrypted Cognos API keys, execution history, and user accounts—delete the Docker volume:

```
docker volume rm simpliload_postgres_data
```

> **Warning:** Removing `simpliload_postgres_data` permanently deletes all SimpliLoad data. This action cannot be undone.

3. (Optional) Remove Deployment Files

If you no longer need the SimpliLoad deployment directory, you may delete the release bundle files, certificates, and compose files. This step is optional and depends on your retention policies and environment management practices.

After these steps, no SimpliLoad components remain on the system.

Security Overview

SimpliLoad is designed with a security-first posture. All communication with Cognos Analytics is performed over HTTPS, certificate validation is enforced, and sensitive data is encrypted at rest using a dedicated master key. The following sections summarize the security model and provide guidance for key management and operational safety.

Key Management

SimpliLoad uses two primary cryptographic keys:

- **DJANGO_SECRET_KEY** — used for session signing and internal Django cryptography.
- **SIMPLILOAD_MASTER_KEY** — used to encrypt Cognos API keys and other sensitive values stored in PostgreSQL.

Both keys must be generated using a secure random source and stored in a protected location. The master key must be backed up securely; without it, encrypted data cannot be recovered.

> **Critical:** Loss of `SIMPLILOAD_MASTER_KEY` permanently prevents decryption of stored Cognos API keys. Always maintain an offline backup.

Key Rotation Procedure

Key rotation is supported for both Django and SimpliLoad keys. Rotation should be performed during a maintenance window to avoid session invalidation or temporary authentication failures.

To rotate keys:

1. Generate new secure values using the commands in the “Generate Secure Values” section.
2. Update `.env` with the new keys.
3. Restart SimpliLoad using the appropriate Docker Compose file.

> **Note:** Rotating `DJANGO_SECRET_KEY` will invalidate all active sessions. Users will need to sign in again.

Threat Model Summary

SimpliLoad's threat model focuses on protecting stored credentials, ensuring secure transport, and preventing unauthorized access to Cognos Analytics. Key considerations include:

- **Transport Security:** All Cognos communication requires HTTPS with certificate validation.
- **Credential Protection:** Cognos API keys are encrypted using the master key before being stored in PostgreSQL.
- **Container Isolation:** Application and database containers run with minimal privileges and isolated volumes.
- **Host Integrity:** Sensitive files such as certificates and environment variables must be protected by host-level access controls.

Data Handling Statement

SimpliLoad does not transmit data to external services. All Cognos interactions occur directly between SimpliLoad and your Cognos Analytics environment. No report content, credentials, or execution metadata leaves your infrastructure.

Stored data includes:

- Encrypted Cognos API keys
- Execution history and timing metrics
- User account information for the SimpliLoad web interface

All sensitive values are encrypted at rest and protected by strict access controls.

Operations

This section provides an overview of SimpliLoad's architecture, operational behavior, versioning model, licensing structure, and support expectations. These details help administrators maintain a stable and predictable deployment.

Architecture Overview

SimpliLoad is deployed as a set of Docker containers orchestrated through Docker Compose. The architecture consists of three primary components:

- **Application Container:** Runs the Django web interface, ASGI server, Cognos execution engine, and REST client.
- **PostgreSQL Container:** Stores encrypted credentials, execution history, and application metadata.
- **NGINX Container:** Provides HTTPS termination and reverse proxy routing to the application container.

All components communicate over internal Docker networks, and persistent data is stored in a dedicated PostgreSQL volume.

Operational Considerations

SimpliLoad is designed for continuous operation with minimal maintenance. Administrators should consider the following:

- **Resource Allocation:** Increase `ASGI_THREADS` for higher concurrency if CPU resources allow.
- **Certificate Rotation:** Renew NGINX and Cognos certificates before expiration to avoid service interruptions.
- **Database Growth:** Execution history may grow over time; periodic cleanup is recommended for long-running deployments.
- **Logging:** Use `LOG_LEVEL=INFO` for normal operation and `LOG_LEVEL=DEBUG` only for troubleshooting.
- **PostgreSQL Volume Location:** All persistent data is stored in the Docker-managed volume `simpliload_postgres_data`. On Linux hosts, this volume resides at `/var/lib/docker/volumes/simpliload_postgres_data/_data/`. On Windows and macOS, Docker Desktop stores the volume inside its internal VM. Removing this volume permanently deletes all SimpliLoad data.

Versioning Policy

SimpliLoad follows a semantic versioning model:

- **Major versions** introduce significant changes or new capabilities.
- **Minor versions** add features while maintaining backward compatibility.
- **Patch versions** include bug fixes and small improvements.

Upgrades are performed by pulling the latest image and recreating containers using the commands in the “Upgrade SimpliLoad” section.

Licensing

SimpliLoad is available in two licensing tiers:

- **Trial Version:** Limited functionality intended for evaluation and testing.
- **Full Version:** Complete feature set including advanced execution controls, extended history retention, and enterprise support.

Each release bundle includes the appropriate Docker Compose file for your license tier.

Customer Responsibilities

Administrators deploying SimpliLoad are responsible for:

- Protecting environment variables and certificates on the host system.
- Maintaining backups of `SIMPLILOAD_MASTER_KEY` and PostgreSQL data.
- Ensuring Cognos Analytics is reachable over HTTPS.
- Managing user accounts within the SimpliLoad web interface.
- Applying updates and security patches in a timely manner.

Support

Support is available for full-license customers. Assistance includes installation guidance, troubleshooting, and help with configuration or performance tuning. Trial users may reference this deployment guide and community resources.

Troubleshooting

This section provides practical guidance for diagnosing and resolving common issues encountered during installation or operation. Each item includes likely causes, recommended fixes, and steps for reviewing logs.

Invalid Certificate Errors

Common Causes:

- The Cognos CA certificate is missing or not in PEM format.
- The certificate path in `.env` is incorrect.
- Hostname verification is enabled but the Cognos certificate CN/SAN does not match.
- The NGINX certificate or key is unreadable by the container.

Fixes:

- Verify `cognos.pem` exists in `deploy/certs/` and is valid PEM.
- Confirm `COGNOS_CERT_HOST_PATH` points to the correct file.
- Ensure the Cognos server certificate's CN or SAN matches the hostname in `COGNOS_ALLOWED_HOSTS`.
- Check file permissions for `server.crt` and `server.key`.

Connection Failures to Cognos

Common Causes:

- Cognos is not reachable from the host system.
- Firewall rules or network ACLs block outbound HTTPS traffic.
- Incorrect Cognos hostname or port in `COGNOS_ALLOWED_HOSTS`.
- HTTP is disabled (`COGNOS_ALLOW_HTTP=false`) but Cognos is not serving HTTPS.

Fixes:

- Ping the Cognos server from the host to confirm reachability.
- Verify HTTPS access using a browser or `curl -v https://your-cognos-host`.
- Update `COGNOS_ALLOWED_HOSTS` to match the actual Cognos hostname.
- Ensure Cognos Analytics is configured for HTTPS.

Authentication Issues

Common Causes:

- The Cognos API key was copied incorrectly into the SimpliLoad Settings page (extra spaces, line breaks, hidden characters).
- The API key has expired based on the expiration date set in Cognos Analytics.
- The API key has been **locked** in Cognos, preventing all REST API authentication.
- The API key was deleted or regenerated in Cognos but the new key was not updated in the SimpliLoad Settings page.
- The API key belongs to a Cognos user who does not have permission to access the target folders or run the selected reports.
- The Cognos user account associated with the API key is disabled or suspended.

Fixes:

- Open Cognos Analytics → *Manage* → *Accounts* → *(User)* → *API Keys* and verify the key is:
  - active
  - not expired
  - not locked
  - not deleted
- Re-copy the API key carefully and paste it into the SimpliLoad Settings page, ensuring no trailing spaces or newline characters.
- If the key is locked, unlock it or create a new API key.
- If the key has expired, create a new API key with an appropriate expiration window.
- Ensure the Cognos user associated with the API key has access to the folders and reports SimpliLoad will execute.
- If the user account is disabled, re-enable it or create a new API key under a valid user.

> **Important:** If an API key is locked, expired, or deleted in Cognos Analytics, it cannot be recovered. A new key must be generated and updated in the SimpliLoad Settings page.

Slow or Stalled Report Execution

Common Causes:

- Insufficient ASGI worker threads for concurrent workloads.
- Long-running Cognos reports exceeding the timeout limit.
- High CPU usage on the host system.
- Network latency between SimpliLoad and Cognos.

Fixes:

- Increase `ASGI_THREADS` to improve concurrency.
- Raise `SIMPLILOAD_RUN_TIMEOUT_SECONDS` for complex reports.
- Review Cognos server performance and dispatcher load.
- Ensure SimpliLoad is deployed close to Cognos to minimize latency.

Container Startup Errors

Common Causes:

- Missing required environment variables.
- Incorrect file paths for certificates or secrets.
- PostgreSQL container failed to initialize due to invalid credentials.
- Volume permissions preventing PostgreSQL from writing data.

Fixes:

- Verify all required variables are present in `.env`.
- Confirm certificate paths exist and are readable by Docker.
- Ensure `POSTGRES_PASSWORD` matches the expected value.
- Check host filesystem permissions for the PostgreSQL volume.

How to Read SimpliLoad Logs

Container names vary depending on the directory name used when running Docker Compose. Docker automatically prefixes each container with the project name (the folder name), followed by the service name and an index. To determine the exact container names on your system, run:

```
docker ps --format "table {{.Names}}\t{{.Image}}\t{{.Status}}"
```

A typical deployment from a folder named `simpliload` will produce containers similar to:

- `simpliload-web-1` — Django application, ASGI server, Cognos REST client
- `simpliload-nginx-1` — HTTPS termination and reverse proxy
- `simpliload-db-1` — PostgreSQL database

Once you identify the correct container names, use the following commands to view logs:

```
# Application logs (Django, ASGI, Cognos REST calls)
docker compose logs -f simpliload-web-1

# NGINX logs (HTTPS routing, certificate issues)
docker compose logs -f simpliload-nginx-1

# PostgreSQL logs (database initialization, storage issues)
docker compose logs -f simpliload-db-1
```

Recommended Diagnostic Workflow:

1. Start with `docker compose logs -f simpliload-web-1` to identify application-level errors, including Cognos API key issues, REST failures, and execution errors.
2. Check `docker compose logs -f simpliload-nginx-1` for certificate validation failures, hostname mismatches, or routing problems.
3. Review `docker compose logs -f simpliload-db-1` if you suspect database initialization, storage, or encryption issues.
4. Temporarily set `LOG_LEVEL=DEBUG` in the Settings page for deeper insight into Cognos REST interactions. Return to `INFO` after troubleshooting.

> **Note:** Container names may differ if the deployment directory name changes. Always use `docker ps` to confirm the exact names before reading logs.

> **Note:** Always return `LOG_LEVEL` to `INFO` after troubleshooting to avoid excessive log output.