FlexiCopy Documentation

Production-ready SFTP engine converted from HTML to PDF, with internal jump links preserved for navigation.

Dynamic multi-job SFTP copy engine **Atomic .tmp protection** **Resume-capable** **Integrity-focused**

Contents

Introduction

FlexiCopy is a dynamic multi-job SFTP copy engine built for production. It safely transfers files from remote servers to local or intermediary destinations; multithreaded, resume-capable, and paranoid about integrity.

Remote folder pattern
`base_path/YYYY/YYYYMM` (year/month)

Zero partial files
Atomic `.tmp` rename

Features

Platform Independent
Identical core logic and performance across Linux, Windows, and macOS environments.

Multi-job SFTP
Simultaneous jobs, independent configs, multiple servers.

Local/Remote destinations
Direct to disk or via mediator to other servers.

DONE folder management
MOVE, DELETE, KEEP actions plus optional zip.

TMP protection
`.tmp` prefix avoids incomplete overwrites.

File age skip
Skip files younger than N minutes (partial writes).

Integrity validation
Checksum (shasum) to ensure perfect copy.

Size comparison
Keep larger local files, replace only smaller.

Large file resume
Auto resume after network dropout.

Logging & rotation
Date-based, 30-day retention.

Multithreaded safe copy
Parallel transfers without file conflicts.

Benefits

- Fast and reliable for any file size, from KB to dumps.
- Automatic resume after network or server hiccup.
- Integrity with TMP + checksum (optional).
- Full audit trail with daily logs.
- Checksum validation ensures bit-perfect transfers.
- Granular configuration per server, pattern, destination.
- SFTP automation built for production scale.

Why should I use FlexiCopy?

FlexiCopy eliminates the pain of SFTP automation in enterprise environments:

Speed & efficiency
Multithreading + resume.

Data integrity
TMP + checksum guards.

Automatic resume
No manual restarts.

Flexible configuration
Per-job SFTP, paths, patterns.

Full visibility
Rotated logs, 30-day retention.

Production ready
Scales with workflow.

Dependencies

Python 3.10+ **paramiko** **PyYAML**

Installation & usage

1. Place app folder anywhere.
2. Create `requirements.txt` with:

```
paramiko==3.1.0
PyYAML==6.0
bcrypt==4.0.1
colorama==0.4.6
tqdm==4.65.0
```

paramiko 3.1.0 | PyYAML 6.0 | bcrypt 4.0.1 | colorama 0.4.6 | tqdm 4.65.0

1. Install: `pip install -r requirements.txt`
2. Configure `jobs.yaml` (example below).
3. Run: `python sftp_flexicopy.py EPSILON 2023`

jobs.yaml snippet

```
EPSILON:

  name: "EPSILON"                       # Job identifier
  description: "SDR EPSILON FILES"      # Description of the job

  source:
    type: SFTP                           # LOCAL | SFTP
    host: "11.2.3.0"                     # SFTP server IP/hostname
    port: 22                             # Port number
    user: "user21"                       # SFTP username
    password: "pass232_#"                # SFTP password
    base_path: "/F:/FROM_251/EPSILON/SDR"  # Remote base path

  destination:
    type: LOCAL                          # LOCAL | SFTP
    path: "/CDR/SBC/EPSILON"             # Destination path
    # Example SFTP destination configuration:
    # type: SFTP
    # host: "192.168.1.10"
    # port: 22
    # user: "destuser"
    # password: "destpass"
    # base_path: "/remote/path/for/destination"

  file_pattern: "sdr.bn4k.BorderNet-SBC.itsegh2vsbc"  # Regex pattern for files

  transfer:
    large_file_threshold_mb: 500         # Files above this size may be treated
differently
    chunk_size_mb: 32                    # Read/write chunk size in MB
    resume_enabled: true                 # true | false
    verify_integrity: false              # true | false (currently unused)
    show_progress: true                  # true | false
    retry_attempts: 3                    # Number of retry attempts
    retry_delay_seconds: 10              # Delay between retries in seconds

  done:
    action: MOVE                         # MOVE | DELETE | KEEP
```

```
    zip_before_move: true                # true | false (compress file before move)

min_file_age_minutes: 30                 # Minimum file age to process
max_workers: 5                           # Number of threads for parallel processing
```

User guide

- Update `jobs.yaml` with your sources and destinations.
- Remote folders must follow `YYYY/YYYYMM`.
- Execute: `python sftp_flexicopy.py <JOB> <YEAR>`
- Monitor terminal or logs in `logs/`.
- Large files auto-resume after interruption.

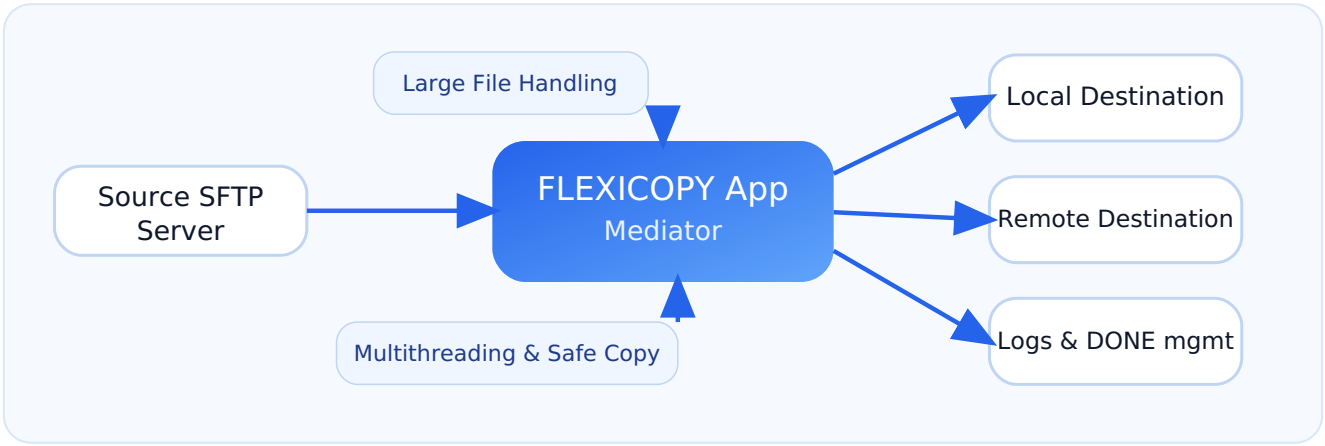
App structure

Component	Description
`sftp_flexicopy.py`	Main application entry point.
`jobs.yaml`	Job and connection configuration.
`logs/`	Runtime logs and rotation output.
`requirements.txt`	Dependency list.

Log file

Item	Value
Log Path	`{path_defined_in_jobs}/logs/`
Log File name	`sftp_flexicopy_main_{job_name}_{year}.log`

Architecture diagram



Additional information

Robust, production-ready, flexible. With TMP-prefix, multithreading, integrity checks and resume, FlexiCopy minimizes data loss and automates SFTP workflows confidently.

Developed by Emma | Supervised by Bertie