Google News RSS Collector - Complete Setup Guide

Table of Contents

Introduction

This guide will help you set up an automated Google News RSS collector on a Red Hat Enterprise Linux (RHEL) virtual machine. The system will automatically fetch news articles based on keywords you specify and store them in organized files for later use with AI language models.

Important: This guide assumes you have no programming experience and provides detailed, step-by-step instructions.

What This System Does

- **Automatically collects news**: Fetches news articles from Google News based on keywords
- **Runs on schedule**: Automatically runs twice daily (5 AM and 2 PM)
- **Removes duplicates**: Ensures the same article isn't stored multiple times
- **Organizes data**: Stores articles in daily files with proper formatting
- **Works offline**: Only connects to the internet to fetch news
- **Logs everything**: Keeps detailed logs of all activities

Prerequisites

- A fresh RHEL VM with internet access
- Root or sudo access to the VM
- Basic command line knowledge (we'll guide you through everything)

Step 1: Initial System Setup

1.1 Connect to Your RHEL VM

Open a terminal or SSH into your RHEL VM:

```bash
# If connecting via SSH
ssh username@your-vm-ip-address

# If already on the VM, open terminal
# (Usually Ctrl+Alt+T or search for "Terminal")
```

1.2 Update the System

First, let's make sure your system is up to date:

```bash
# Update all packages
sudo dnf update -y

# Install basic development tools
sudo dnf groupinstall "Development Tools" -y

# Install essential packages
sudo dnf install -y git curl wget vim nano
```

1.3 Create a Working Directory

```bash
# Create a directory for our RSS collector
sudo mkdir -p /opt/rss_collector
sudo chown $USER:$USER /opt/rss_collector
cd /opt/rss_collector
```

Step 2: Python Installation

2.1 Install Python 3.10+

bash

```bash
# Check if Python 3.10+ is already installed
python3 --version

# If Python version is less than 3.10, install it
sudo dnf install -y python3.11 python3.11-pip python3.11-venv

# Create a symbolic link for easier access
sudo ln -sf /usr/bin/python3.11 /usr/local/bin/python3
```

2.2 Create Virtual Environment

bash

```bash
# Create a virtual environment
python3 -m venv rss_env

# Activate the virtual environment
source rss_env/bin/activate

# Upgrade pip
pip install --upgrade pip
```

Important: Always activate the virtual environment before running the RSS collector:

bash

```bash
cd /opt/rss_collector
source rss_env/bin/activate
```

Step 3: Download and Setup the RSS Collector

3.1 Create Directory Structure

bash

```bash
# Create all necessary directories
mkdir -p {config,logs,feeds,src/utils,tests/fixtures}

# Create empty __init__.py files for Python modules
touch src/__init__.py
touch src/utils/__init__.py
touch tests/__init__.py

# Create .gitkeep files for empty directories
touch logs/.gitkeep
touch feeds/.gitkeep
```

3.2 Create Configuration Files

Create the main configuration file:

bash

```bash
cat > config/feeds.json << 'EOF'
{
  "keywords": [
    "cryptocurrency bitcoin",
    "artificial intelligence",
    "climate change",
    "economy inflation",
    "technology news"
  ]
}
EOF
```

Create the settings file:

bash

```bash
cat > config/settings.json << 'EOF'
{
  "proxy": {
    "enabled": true,
    "port": 8081,
    "host": "127.0.0.1"
  },
  "schedule": {
    "times": ["05:00", "14:00"],
    "timezone": "Asia/Kolkata"
  },
  "fetcher": {
    "timeout": 30,
    "retry_attempts": 3,
    "retry_delay": 5,
    "delay_between_keywords": 300
  },
  "storage": {
    "feeds_dir": "feeds",
    "logs_dir": "logs",
    "file_format": "json",
    "indent": 2,
    "encoding": "utf-8"
  },
  "rss": {
    "base_url": "https://news.google.com/rss/search",
    "parameters": {
      "hl": "en-IN",
      "gl": "IN",
      "ceid": "IN:en"
    }
  }
}
EOF
```

3.3 Create Environment File

bash

```bash
cat > .env << 'EOF'
# Proxy settings
HTTP_PROXY=http://127.0.0.1:8081
HTTPS_PROXY=http://127.0.0.1:8081

# Logging level (DEBUG, INFO, WARNING, ERROR)
LOG_LEVEL=INFO

# Data retention (days)
RETENTION_DAYS=30
EOF
```

Step 4: Install Dependencies

4.1 Create Requirements File

bash

```bash
cat > requirements.txt << 'EOF'
httpx==0.25.2
feedparser==6.0.10
APScheduler==3.10.4
python-dotenv==1.0.0
pytz==2023.3
lxml==4.9.3
pytest==7.4.3
pytest-cov==4.1.0
pytest-mock==3.12.0
requests==2.31.0
EOF
```

4.2 Install Dependencies

bash

```bash
# Make sure virtual environment is activated
source rss_env/bin/activate

# Install all dependencies
pip install -r requirements.txt

# Verify installation
pip list
```

Step 5: Configure the System

5.1 System User Setup

```bash
# Create a dedicated user for the RSS collector (optional but recommended)
sudo useradd -r -s /bin/false -d /opt/rss_collector rss_collector
sudo chown -R rss_collector:rss_collector /opt/rss_collector
```

5.2 File Permissions

```bash
# Set appropriate permissions
find /opt/rss_collector -type f -exec chmod 644 {} \;
find /opt/rss_collector -type d -exec chmod 755 {} \;
chmod +x /opt/rss_collector/run.py
```

Step 6: Setup Network and Proxy

6.1 Install and Configure Proxy

bash

```bash
# Install tinyproxy
sudo dnf install -y epel-release
sudo dnf install -y tinyproxy

# Configure tinyproxy
sudo cp /etc/tinyproxy/tinyproxy.conf /etc/tinyproxy/tinyproxy.conf.backup

# Create new tinyproxy configuration
sudo tee /etc/tinyproxy/tinyproxy.conf > /dev/null << 'EOF'
Port 8081
Listen 127.0.0.1
Timeout 600
DefaultErrorFile "/usr/share/tinyproxy/default.html"
StatFile "/usr/share/tinyproxy/stats.html"
LogLevel Info
PidFile "/var/run/tinyproxy/tinyproxy.pid"
LogFile "/var/log/tinyproxy/tinyproxy.log"
MaxClients 100
MinSpareServers 5
MaxSpareServers 20
StartServers 10
MaxRequestsPerChild 0
ViaProxyName "RSS-Collector-Proxy"
ConnectPort 443
ConnectPort 80
EOF

# Start and enable tinyproxy
sudo systemctl start tinyproxy
sudo systemctl enable tinyproxy

# Check status
sudo systemctl status tinyproxy
```

6.2 Configure Firewall

bash

```bash
# Check firewall status
sudo firewall-cmd --state

# If firewall is running, open the proxy port for localhost
sudo firewall-cmd --zone=trusted --add-source=127.0.0.1/32 --permanent
sudo firewall-cmd --reload
```

Step 7: Test the Installation

7.1 Manual Test Run

```bash
# Navigate to the RSS collector directory
cd /opt/rss_collector

# Activate virtual environment
source rss_env/bin/activate

# Test the installation (dry run)
python3 run.py --test

# If test is successful, run a single fetch
python3 run.py --once
```

7.2 Check Output

```bash
# Check if feeds were created
ls -la feeds/

# Check logs
ls -la logs/

# View the latest log
tail -f logs/$(date +%Y-%m-%d).log
```

7.3 Verify Data Format

```bash
# Check the structure of fetched data
cat feeds/$(date +%Y-%m-%d).json | head -50
```

Step 8: Setup Automatic Scheduling

8.1 Create Systemd Service

bash

```bash
# Create systemd service file
sudo tee /etc/systemd/system/rss-collector.service > /dev/null << 'EOF'
[Unit]
Description=RSS Collector Service
After=network.target

[Service]
Type=forking
User=rss_collector
WorkingDirectory=/opt/rss_collector
Environment=PATH=/opt/rss_collector/rss_env/bin
ExecStart=/opt/rss_collector/rss_env/bin/python /opt/rss_collector/run.py --daemon
Restart=always
RestartSec=10

[Install]
WantedBy=multi-user.target
EOF

# Reload systemd and enable service
sudo systemctl daemon-reload
sudo systemctl enable rss-collector
sudo systemctl start rss-collector

# Check service status
sudo systemctl status rss-collector
```

8.2 Alternative: Cron Setup

If you prefer using cron instead of systemd:

bash

```bash
# Edit crontab for the rss_collector user
sudo crontab -u rss_collector -e

# Add these lines (adjust paths as necessary):
# Run at 5:00 AM IST
0 5 * * * cd /opt/rss_collector && /opt/rss_collector/rss_env/bin/python run.py --once

# Run at 2:00 PM IST
0 14 * * * cd /opt/rss_collector && /opt/rss_collector/rss_env/bin/python run.py --onc
```

Step 9: Monitor and Manage

9.1 Monitoring Commands

```bash
# Check service status
sudo systemctl status rss-collector

# View recent logs
journalctl -u rss-collector -f

# Check application logs
tail -f /opt/rss_collector/logs/$(date +%Y-%m-%d).log

# Check feed files
ls -la /opt/rss_collector/feeds/

# Check disk usage
du -sh /opt/rss_collector/feeds/
```

9.2 Maintenance Tasks

```bash
# Restart the service
sudo systemctl restart rss-collector

# Stop the service
sudo systemctl stop rss-collector

# Update keywords (edit the configuration)
nano /opt/rss_collector/config/feeds.json

# After editing config, restart service
sudo systemctl restart rss-collector
```

9.3 Log Rotation Setup

```bash
# Create logrotate configuration
sudo tee /etc/logrotate.d/rss-collector > /dev/null << 'EOF'
/opt/rss_collector/logs/*.log {
    weekly
    missingok
    rotate 12
    compress
    notifempty
    create 644 rss_collector rss_collector
    postrotate
        systemctl reload rss-collector > /dev/null 2>&1 || true
    endscript
}
EOF
```

Troubleshooting

Common Issues and Solutions

1. Python Import Errors

```bash
# Error: ModuleNotFoundError
# Solution: Ensure virtual environment is activated
cd /opt/rss_collector
source rss_env/bin/activate
pip install -r requirements.txt
```

2. Permission Denied Errors

```bash
# Fix permissions
sudo chown -R rss_collector:rss_collector /opt/rss_collector
sudo chmod -R 755 /opt/rss_collector
```

3. Network Connection Issues

```bash
# Test proxy connectivity
curl -x http://127.0.0.1:8081 http://google.com

# Check proxy status
sudo systemctl status tinyproxy

# Restart proxy if needed
sudo systemctl restart tinyproxy
```

4. Service Won't Start

```bash
# Check system logs
journalctl -u rss-collector -n 50

# Run manually to see errors
cd /opt/rss_collector
source rss_env/bin/activate
python3 run.py --once
```

5. No Data Being Collected

```bash
# Check feeds.json format
cat config/feeds.json | python3 -m json.tool

# Check internet connectivity through proxy
curl -x http://127.0.0.1:8081 "https://news.google.com/rss/search?q=test&hl=en-IN&gl=I
```

Diagnostic Commands

```bash
# System information
hostnamectl
cat /etc/redhat-release

# Python environment check
python3 --version
pip --version
which python3

# Network check
ss -tlnp | grep 8081
netstat -tlnp | grep 8081

# Disk space check
df -h /opt/rss_collector

# Process check
ps aux | grep rss
```

Missing Files Reference

Based on the file structure provided, here are the files that still need to be created:

1. Core Python Modules

src/main.py

- Main entry point for the RSS collector
- Coordinates all other modules
- Handles command-line arguments

src/config_manager.py

- Loads and validates configuration files
- Manages settings.json and feeds.json

src/rss_fetcher.py

- Handles HTTP requests to Google News RSS
- Implements retry logic and proxy support

src/rss_parser.py

- Parses RSS XML content

- Normalizes data into standard format

src/storage_manager.py

- Manages file I/O operations
- Implements deduplication logic
- Handles JSON formatting

src/scheduler.py

- Manages scheduled execution
- Implements APScheduler integration

src/utils/logging_utils.py

- Centralized logging configuration
- Log formatting and file management

src/utils/proxy_utils.py

- Proxy configuration and validation
- Network routing management

src/utils/helpers.py

- Miscellaneous utility functions
- Date/time helpers, string formatting

2. Test Files

tests/test_config_manager.py

- Unit tests for configuration loading

tests/test_rss_fetcher.py

- Tests for HTTP client functionality

tests/test_rss_parser.py

- RSS parsing validation tests

tests/test_storage_manager.py

- File operations and deduplication tests

tests/test_scheduler.py

- Scheduling logic tests

tests/fixtures/sample_rss.xml

- Sample RSS feed for testing

tests/fixtures/sample_feeds.json

- Sample configuration for testing

3. Setup and Documentation

run.py

- Main execution script
- Command-line interface

setup.py

- Package installation script
- Dependencies and entry points

README.md

- Project documentation
- Quick start guide

System Architecture Overview

```
┌─────────────────────────────────────────────┐
│                RSS Collector                │
├─────────────────────────────────────────────┤
│  Scheduler (APScheduler)                    │
│  ├── 5:00 AM Trigger                       │
│  └── 2:00 PM Trigger                       │
├─────────────────────────────────────────────┤
│  RSS Fetcher (httpx)                       │
│  ├── Proxy Support (tinyproxy:8081)        │
│  ├── Retry Logic (3 attempts)              │
│  └── Rate Limiting (5min between keywords) │
├─────────────────────────────────────────────┤
│  RSS Parser (feedparser)                   │
│  ├── XML to JSON conversion                │
│  └── Data normalization                    │
├─────────────────────────────────────────────┤
│  Storage Manager                            │
│  ├── Deduplication (by link/hash)          │
│  ├── Daily files (YYYY-MM-DD.json)         │
│  └── UTF-8 formatting                      │
├─────────────────────────────────────────────┤
│  Logging System                             │
│  ├── Daily logs (YYYY-MM-DD.log)           │
│  ├── Error tracking                        │
│  └── Performance metrics                   │
└─────────────────────────────────────────────┘
```

Configuration Management

Adding New Keywords

1. Edit the configuration file:

   bash

   ```bash
   nano /opt/rss_collector/config/feeds.json
   ```

2. Add your keywords to the array:

```json
{
  "keywords": [
    "your new keyword",
    "another keyword phrase",
    "existing keyword"
  ]
}
```

3. Restart the service:

```bash
sudo systemctl restart rss-collector
```

Changing Schedule

1. Edit settings file:

```bash
nano /opt/rss_collector/config/settings.json
```

2. Modify the schedule times:

```json
{
  "schedule": {
    "times": ["06:00", "15:00", "22:00"],
    "timezone": "Asia/Kolkata"
  }
}
```

3. Restart the service:

```bash
sudo systemctl restart rss-collector
```

Data Format Examples

feeds.json (input)

```json
{
  "keywords": [
    "artificial intelligence",
    "machine learning",
    "data science"
  ]
}
```

YYYY-MM-DD.json (output)

```json
[
  {
    "fetched_at": "2025-05-18T05:00:00Z",
    "query": "artificial intelligence",
    "source_url": "https://news.google.com/rss/search?q=artificial%20intelligence&hl=e
    "articles": [
      {
        "title": "AI Breakthrough in Healthcare",
        "link": "https://example.com/ai-healthcare",
        "published": "2025-05-18T04:30:00Z",
        "source": "Tech News Daily",
        "snippet": "New AI system shows promising results..."
      }
    ]
  }
]
```

Security Considerations

File Permissions

- All configuration files should be readable only by the rss_collector user
- Log files should be protected from unauthorized access
- Feed data should be accessible to authorized users only

Network Security

- Only outbound connections through the specified proxy port
- No inbound network connections required
- Proxy logs all external requests for audit purposes

System Security

- Run as dedicated user with minimal privileges
- Use systemd for secure service management
- Regular security updates through dnf/yum

Performance Optimization

System Resources

- Memory usage: ~50-100MB during normal operation
- CPU usage: Minimal, mostly I/O bound
- Disk space: ~1MB per day of feeds (varies by keyword count)

Optimization Tips

1. Adjust retry attempts based on network reliability
2. Increase delay between keywords if rate limiting occurs
3. Implement data archiving for long-term storage management
4. Monitor log file sizes and implement rotation

Backup and Recovery

Backup Important Files

bash

```bash
# Create backup script
cat > /opt/rss_collector/backup.sh << 'EOF'
#!/bin/bash
DATE=$(date +%Y%m%d)
BACKUP_DIR="/opt/backups/rss_collector"
mkdir -p $BACKUP_DIR

# Backup configuration
tar -czf $BACKUP_DIR/config_$DATE.tar.gz config/

# Backup recent feeds (last 7 days)
find feeds/ -name "*.json" -mtime -7 -exec tar -czf $BACKUP_DIR/feeds_$DATE.tar.gz {}

# Backup logs (last 30 days)
find logs/ -name "*.log" -mtime -30 -exec tar -czf $BACKUP_DIR/logs_$DATE.tar.gz {} +

echo "Backup completed: $DATE"
EOF

chmod +x /opt/rss_collector/backup.sh
```

Recovery Procedure

1. Restore configuration files from backup
2. Restart services
3. Verify functionality with test run
4. Resume normal operations

Support and Maintenance

Regular Maintenance Tasks

Weekly:

- Check log files for errors
- Verify feed data is being collected
- Monitor disk space usage

Monthly:

- Update system packages
- Backup configuration and recent data

- Review and clean old log files

Quarterly:

- Review and update keywords
- Update Python dependencies
- Performance optimization review

This completes the comprehensive setup guide for the Google News RSS Collector system. The system is designed to be robust, maintainable, and suitable for production use on a RHEL environment.