LOGZILLA DOCUMENTATION

Data Archiving and Retention

Configure LogZilla online and archived storage retention, hourly chunk transitions, and compressed archive policies to control disk footprint

Administration · Generated May 3, 2026 · logzilla.ai/docs/administration/data-archiving-and-retention

© LogZilla Corporation · logzilla.ai

LogZilla provides automated data archiving to manage storage space while maintaining access to historical log data. The archiving system moves older data from active storage to compressed archive storage, where it remains searchable but with slower performance.

Important: Archive settings directly affect data retention and storage usage. Incorrect configuration can result in premature data loss. Always verify settings before making changes in production environments.

How Archiving Works

LogZilla manages data in two states:

- **Online Data**: Recent data stored in active indexes for fast searching
- **Archived Data**: Older data moved to compressed storage, still searchable but with reduced performance

Data is organized into hourly chunks and automatically transitioned based on configured retention policies.

Archive Configuration

Archive settings are managed through the storage configuration:

```
# View current archive settings
logzilla settings list storage

# Configure archive settings
logzilla settings update AUTO_ARCHIVE_ENABLED=true
logzilla settings update ARCHIVE_FLUSH_DAYS=365
logzilla settings update ARCHIVE_EXPIRE_DAYS=30
```

Archive Settings

Setting	Description	Default	Impact
`AUTO_ARCHIVE_ENABLED`	Enable automatic archiving	`true`	Disabling stops all automatic archiving
`ARCHIVE_FLUSH_DAYS`	Days before data is archived	`365`	Shorter = less online storage used
`ARCHIVE_EXPIRE_DAYS`	Days before archived data is deleted	`5`	**Critical**: Data permanently lost after this period

Setting	Description	Default	Impact
AUTOARCHIVE_CRON_HOUR	Daily archive job time (24-hour format)	5	Schedule during low-usage periods

Important: ARCHIVE_EXPIRE_DAYS determines when archived data is permanently deleted. Total data retention is ARCHIVE_FLUSH_DAYS

- ARCHIVE_EXPIRE_DAYS. Verify this meets the site's retention requirements.

Automatic Archiving

LogZilla automatically manages data lifecycle:

Daily Processing: Archive job runs at configured hour
Data Archiving: Online data older than ARCHIVE_FLUSH_DAYS is archived
Data Expiration: Archived data older than ARCHIVE_EXPIRE_DAYS is deleted
No Downtime: Process runs in background without service interruption

Manual Archive Management

Use the logzilla archives command for manual operations:

Archive Specific Date Ranges

```
# Archive data for specific period
logzilla archives archive --ts-from 2024-01-01 --ts-to 2024-02-01

# Archive data older than specified days
logzilla archives archive --expire-days 30

# Round timestamps to nearest hour (aligns with LogZilla's hourly chunks)
logzilla archives --round archive --ts-from 2024-01-01 --ts-to 2024-02-01
```

Remove Archived Data

```
# Permanently remove archived data for specific period
logzilla archives remove --ts-from 2023-01-01 --ts-to 2023-02-01
```

Warning: The `remove` command permanently deletes data. This action cannot be undone. Ensure a backup exists if the data might be needed later.

Automatic Archive Searching

LogZilla automatically determines whether to search archived data based on the date range specified:

Recent Data: Searches within the online data retention period use only active indexes for fast results
Historical Data: Searches with date ranges extending beyond online data automatically include archived data
Performance Impact: Archive search performance depends on the storage hardware used for archived data
Transparent Access: Users don't need to manually "restore" data - LogZilla handles archive access transparently

No Rehydration Required

Unlike traditional log management systems that require users to manually **rehydrate** archived data back to active storage, LogZilla provides direct access to archived logs without any manual intervention.

Traditional Rehydration Process (Not Required with LogZilla)

Most log management platforms require administrators to:

Identify archived data needed for analysis
Request rehydration of specific time ranges or data sets
Wait for rehydration to complete (often hours or days)
Pay additional costs for rehydration processing and temporary storage
Manage rehydrated data lifecycle and cleanup

LogZilla's Transparent Archive Access

LogZilla eliminates the rehydration bottleneck entirely:

- **No rehydration delays**: Archived data is immediately accessible
- **No rehydration costs**: No additional charges for accessing historical data
- **No rehydration management**: No need to track or cleanup rehydrated datasets
- **No rehydration planning**: No advance planning required for historical analysis
- **No rehydration limits**: Access any archived timeframe without restrictions

Benefits Over Rehydration-Based Systems

Immediate Access: Start analyzing historical data instantly without waiting for rehydration processes to complete.

Cost Efficiency: Eliminate rehydration fees and temporary storage costs associated with bringing archived data back online.

Operational Simplicity: Remove the complexity of managing rehydration workflows, scheduling, and cleanup processes.

Forensic Readiness: Respond to security incidents immediately without rehydration delays that could impact investigation timelines.

Compliance Reporting: Generate compliance reports spanning any timeframe without advance rehydration planning.

Competitive Advantage: LogZilla's architecture provides direct archive access capabilities that eliminate the operational overhead and costs associated with traditional rehydration workflows found in other log management platforms.

API Queries

Authentication Required: All LogZilla API requests require authentication. See Getting Started (https://www.logzilla.ai/docs/logzilla-api/getting-started) for API token creation and usage instructions.

LogZilla automatically determines whether to search archived data based on the date range specified in the query. When the time range extends beyond the online data retention period, archived data is automatically included.

Example: Recent Data Search (Online Only)

```
# List existing API tokens (requires admin access)
sudo logzilla authtoken list

# Use an existing USER token (replace with your actual token)
TOKEN="6160ce50a098067f39d1acc72396b31c5518a5ca7b178538"

# Or create a new USER token if needed (creates user-prefixed token)
# TOKEN=$(sudo logzilla authtoken create | tail -1)

# Search last hour (online data only - fast)
curl -X POST \
  -H "Authorization: token $TOKEN" \
  -H "Content-Type: application/json" \
  -d '{
    "type": "Search",
    "params": {
      "time_range": {"preset": "last_1_hours"},
      "filter": [{"field": "message", "value": "error"}],
      "page_size": 100
    }
  }' \
  "http://your-logzilla-server/api/query"
```

Example: Historical Data Search (Includes Archives)

```
# Search last 30 days (includes archived data - slower)
curl -X POST \
```

```
  -H "Authorization: token $TOKEN" \
  -H "Content-Type: application/json" \
  -d '{
    "type": "Search",
    "params": {
      "time_range": {"preset": "last_30_days"},
      "filter": [{"field": "message", "value": "error"}],
      "page_size": 100
    }
  }' \
  "http://your-logzilla-server/api/query"
```

Example: Specific Date Range Search

```
# Search specific date range (may include archives)
curl -X POST \
  -H "Authorization: token $TOKEN" \
  -H "Content-Type: application/json" \
  -d '{
    "type": "Search",
    "params": {
      "time_range": {
        "ts_from": 1609459200,
        "ts_to": 1609545600
      },
      "filter": [
        {"field": "host", "value": "web-server-01"},
        {"field": "severity", "op": "le", "value": 3}
      ],
      "sort": ["first_occurrence"],
      "page_size": 50
    }
  }' \
  "http://your-logzilla-server/api/query"
```

Performance Note: Queries spanning archived data will take longer to complete and may return status `202 ACCEPTED` for asynchronous processing. Use the returned `query_id` to check results or implement websocket subscriptions for real-time updates.

Monitoring and Logs

Archive Activity Logs

Monitor archive operations through LogZilla's system logs:

```
# View archive-related log entries
sudo grep -i archive /var/log/logzilla/logzilla.log
```

```
# Monitor real-time archive activity
sudo tail -f /var/log/logzilla/logzilla.log | grep -i archive
```

API-Based Archive Monitoring

```
# Use existing API token (replace with your actual token)
TOKEN="6160ce50a098067f39d1acc72396b31c5518a5ca7b178538"

# Or create new USER token if needed (creates user-prefixed token)
# TOKEN=$(sudo logzilla authtoken create | tail -1)

# Query archive-related system events
curl -X POST \
  -H "Authorization: token $TOKEN" \
  -H "Content-Type: application/json" \
  -d '{
    "type": "Search",
    "params": {
      "time_range": {"preset": "last_24_hours"},
      "filter": [
        {"field": "program", "value": "logzilla"},
        {"field": "message", "value": "archive"}
      ],
      "sort": ["-first_occurrence"],
      "page_size": 50
    }
  }' \
  "http://your-logzilla-server/api/query"
```

Archive Status

Monitor current archive status:

```
# View archived data chunks
 curl -H 'Authorization: token user-914012b8e78f8b305db63a6c04019d77655a9a3daa7badda' http://your-logzilla-server/api/archives
```

System Logs

Archive operations are logged to the main LogZilla log:

```
# Monitor archive operations
grep -i "archive" /var/log/logzilla/logzilla.log
```

Configuration Examples

Common Retention Scenarios

90 Days Online, 1 Year Total Retention:

```
logzilla settings update ARCHIVE_FLUSH_DAYS=90
logzilla settings update ARCHIVE_EXPIRE_DAYS=275
# Total retention: 90 + 275 = 365 days
```

30 Days Online, 3 Years Total Retention:

```
logzilla settings update ARCHIVE_FLUSH_DAYS=30
logzilla settings update ARCHIVE_EXPIRE_DAYS=1065
# Total retention: 30 + 1065 = 1095 days (3 years)
```

7 Days Online, 6 Months Total Retention:

```
logzilla settings update ARCHIVE_FLUSH_DAYS=7
logzilla settings update ARCHIVE_EXPIRE_DAYS=173
# Total retention: 7 + 173 = 180 days (6 months)
```

Best Practices

Storage Planning

- **Calculate Total Retention**: `ARCHIVE_FLUSH_DAYS` + `ARCHIVE_EXPIRE_DAYS`
- **Monitor Disk Usage**: Archive storage is located at `/var/lib/logzilla-archive/`
- **Plan for Growth**: Consider data volume increases over time
- **Use `--round` Flag**: When manually archiving, use `--round` to align with hourly chunks

Performance Optimization

- **Schedule Wisely**: Run archive jobs during low-usage periods
- **Limit Archive Searches**: Use archived data searches sparingly
- **Monitor Impact**: Archive operations can affect system performance

© LogZilla Corporation · logzilla.ai

Data Safety

- **Verify Settings**: Confirm retention periods meet compliance requirements
- **Test Procedures**: Validate archive and search functionality in test environments
- **Document Policies**: Maintain clear data retention documentation

Relocating Archive Storage

Use this procedure to move the archive volume to a new host directory (bind mount) when reallocating disk space.

Check Current Archive Size

```
# Inspect size of the archive volume (no host path assumptions)
docker run --rm -v lz_archive:/archive \
  logzilla/runtime sh -lc 'du -csh /archive'
```

Move the Archive to a New Host Directory

```
logzilla stop

# Ensure /new_archive_dir exists and has sufficient space
docker run --rm \
  -v /new_archive_dir:/new_archive_dir \
  -v lz_archive:/temp_archive \
  logzilla/runtime sh -lc 'mv /temp_archive/* /new_archive_dir/'

# Recreate the volume as a bind mount to the new directory
docker rm lz_watcher
docker volume rm lz_archive
docker volume create --opt type=none --opt o=bind \
  --opt device=/new_archive_dir lz_archive

logzilla start
```

Notes:

- Replace `/new_archive_dir` with the destination directory on the host. Create it before running the commands.
- The archive path inside the container remains `/var/lib/logzilla-archive/`. The `lz_archive` volume binds that path to the specified host directory.
- Ensure no containers are using the archive during the move; `logzilla stop` handles this.

Troubleshooting

Common Issues

Archive Job Failures: Check system logs and available disk space

Slow Archive Searches: Verify system resources and consider limiting search scope

Missing Archived Data: Confirm data hasn't exceeded `ARCHIVE_EXPIRE_DAYS`

Storage Space Issues: Monitor both online and archive storage locations

Recovery Procedures

For archive-related issues:

Check system logs for error messages
Verify storage configuration and available space
Contact LogZilla support for data recovery assistance
Maintain external backups for critical data protection