

NFT Team JMeter Plugin - Setup and Configuration Guide



NFT Team Performance Testing Solution

This custom JMeter plugin developed by the NFT Team provides three essential performance testing capabilities:

- **Global Setup**
- **STS (Simple Table Server)**
- **Pacing.**

The plugin is designed to streamline test execution, enable dynamic data management, and provide consistent timing control across test scenarios by directly extending the tools functionality.

Value Added Benefits

Centralised Framework Management

- **No more individual repo updates:** Framework components managed in a single central image repository (compared to 20+ previously)
- **Instant updates:** All test plans benefit from improvements without individual repository modifications
- **Consistent versioning:** Everyone uses the same tested and verified plugin versions

Enhanced Script Preparation

- **GUI-based configuration:** Intuitive UI objects replace complex scripting requirements
- **Reduced preparation time:** Visual configuration saves time and eliminates scripting errors
- **Lower complexity:** Point-and-click setup instead of code development and debugging

Streamlined Execution Pipeline

- **Simplified deployments:** No need to checkout common scripts or external components
- **FASTER pipeline execution:** Fewer dependencies and external file operations
- **Reduced failure points:** Less moving parts means more reliable test execution

Optimised IT Resource Usage

- **Memory efficiency:** No external scripts loaded into memory, reducing memory requirements and leak risks
- **FASTER container startup:** JMeter loads more quickly without external script dependencies
- **Compiled performance:** Native Java plugin code runs faster and more efficiently than interpreted scripts

Improved Reliability & Maintainability

- **Better error handling:** GUI elements provide immediate feedback and validation
- **Enhanced debugging:** Visual status indicators and built-in logging capabilities
- **Consistent behaviour:** Standardised plugin logic eliminates script variations across team

Team Productivity

- **Reduced onboarding time:** New team members can use GUI elements without learning custom scripting
- **Better collaboration:** Standardised approach improves knowledge sharing across the team
- **Enhanced CI/CD integration:** Self-contained plugins work seamlessly in automated pipelines

Table of Contents

- [Quick Start](#)
 - [Core Components](#)
 - [1. Global Setup \(Automatic\)](#)
 - [2. STS \(Simple Table Server\)](#)
 - [3. Pacing Configuration](#)
 - [Installation](#)
 - [Configuration](#)
 - [Usage Examples](#)
 - [Troubleshooting](#)
 - [Migration Notes](#)
-

Quick Start

Prerequisites

- JMeter 5.x or later
- Java 8 or later

Essential Files

- **JAR:** `lib/ext/dcu-sts-utils.jar` - Core plugin functionality
- **Configuration:** `user.properties` or `jmeter.properties`
- **GUI Elements:** Available in JMeter's Add menu under Config Elements

Immediate Use





The plugin is **ready to use immediately** after installation. Global Setup works automatically, whilst STS and Pacing provide GUI elements for easy configuration in test plans.

Core Components

1. Global Setup (Automatic)

Purpose: Automatically handles test plan initialisation tasks including environment detection, directory setup, template processing, and Git repository discovery.

Key Benefits

-  **Zero Test Plan Modifications:** Works with existing test plans automatically
-  **Centralised Management:** All configuration in JMeter installation
-  **Consistent Execution:** Runs automatically on every test start
-  **Property-Driven:** Easy environment-specific configuration

Features

Feature	Description	Properties Set
Template Processing	Converts <code>template.*</code> properties to regular properties	<code>template.PROPERTY_NAME</code> → <code>PROPERTY_NAME</code>
Directory Setup	Establishes absolute paths for test data	<code>JMX_DIR</code> , <code>DATA</code>
Hostname Detection	Detects system hostname and extracts test names	<code>hostname</code> , <code>testname</code> (from jm-gen patterns)
Environment Detection	Auto-configures central STS hosts based on environment	<code>IS_LOCAL_ENVIRONMENT</code> , <code>V_STS_HOST</code>

Feature	Description	Properties Set
Git Detection	Finds Git repository information	<code>GIT_REPO_PATH</code> , <code>GIT_REPO_NAME</code> , <code>GIT_REPO_URL</code>

Configuration

Add to your `user.properties`:

```
# Core Features (all default to true)
global.setup.template.processing=true
global.setup.directory.setup=true
global.setup.hostname.detection=true
global.setup.environment.detection=true
global.setup.git.detection=true





# Logging
global.setup.logging.verbose=false

# Central STS hosts (internal and external access)
V_STS_LOCAL_HOST=simple-table-server-ho-hmpo-ew2tlg1n11-i-team-nft-i-docker-
tlg1.service.np.ebsa.local
V_STS_HOST_DEFAULT=simple-table-server-ho-hmpo-ew2tlg1n11-i-team-nft-i-docker-
tlg1.np.ebsa.homeoffice.gov.uk
```

2. STS (Simple Table Server)

Purpose: HTTP server for managing CSV data files to enable dynamic test data sharing across multiple JMeter instances and test scenarios.

✓ Key Benefits


-  **Dynamic Data Management:** Share CSV data across test instances
-  **Real-time Operations:** KEEP, DELETE, ADD operations on CSV files
-  **Multi-instance Support:** Multiple JMeter instances can share data
-  **Thread-safe Operations:** Concurrent access handled safely

Core Operations

Operation	GUI Configuration	Behaviour
KEEP	Select file from dropdown → map to variables	Reads first row, moves it to end, stores values in variables
DEL	Select file from dropdown → map to variables	Reads and deletes first row, stores values in variables

Operation	GUI Configuration	Behaviour
ADDFIRST	Select file from dropdown → specify values	Adds new row to beginning of file
ADDLAST	Select file from dropdown → specify values	Adds new row to end of file

Files are automatically discovered from `/data/` folder and populated in the dropdown.

 **Migration Tip:** If migrating from script-based STS, simply copy the variable names from your existing `STS.groovy` script parameters and paste them into the UI variable fields. For example:

- Script: `KEEP,users.csv,USER_ID,PASSWORD,EMAIL`
- UI: Variables field: `USER_ID,PASSWORD,EMAIL`

STS Configuration

Add to your `user.properties`:

```
# Central STS Server Configuration (via ingress)
# SSL is required but certificates are automatically ignored (like curl -k)
sts.use.https=true

# Central STS Hosts (internal and external access)
V_STS_LOCAL_HOST=simple-table-server-ho-hmpo-ew2tlg1n11-i-team-nft-i-docker-
tlg1.service.np.ebsa.local
V_STS_HOST_DEFAULT=simple-table-server-ho-hmpo-ew2tlg1n11-i-team-nft-i-docker-
tlg1.np.ebsa.homeoffice.gov.uk

# STS Path Configuration
STS_PATH_PREFIX=/sts
```

Usage in Test Plans

Add STS Configuration elements to your test plan:

1. **Right-click** Test Plan or Thread Group
2. **Add** → **Config Element** → **STS Configuration**
3. **Add multiple rows** (one for each CSV file operation):
 - **Click "Add"** to add new rows
 - **Select** CSV file from dropdown (automatically populated from `/data/` folder)
 - **Configure** operation type (KEEP, DEL, ADDFIRST, ADDLAST)
 - **Set** variable mappings through the GUI
4. **Run** test - all STS operations execute automatically in order

 **Pro Tip:** Copy variable names from existing `STS.groovy` script parameters and paste directly into the UI variable fields.

File Structure Requirements

Repository Structure (must start with jm_ prefix):

```
jm_projectname/           # Repository name: jm_ + single continuous string
├─ testplan.jmx           # Your test plan file (in repository root)
├─ data/                  # CSV files directory (relative to JMX file)
│   ├─ users.csv
│   ├─ accounts.csv
│   └─ transactions.csv
```

Important:





- **Repository Naming:** Must start with `jm_` followed by a single continuous string (no spaces, underscores, or special characters). Examples: `jm_dcs`, `jm_dap`, `jm_myproject`
- **File Location:** All CSV files must be in the `/data/` folder relative to your JMX test plan file
- **Auto-Detection:** The STS UI element automatically discovers and populates CSV files from the data folder

Note: Script-based STS calls are still supported for advanced users, but the GUI approach is recommended for ease of use.

3. Pacing Configuration

Purpose: Controls timing between test iterations to maintain consistent load patterns and realistic user behaviour simulation.

Key Benefits

-  **Precise Timing Control:** Single values or random ranges
-  **Realistic Load Simulation:** Maintains consistent pacing across iterations
-  **Flexible Configuration:** Simple parameter-based setup
-  **Performance Monitoring:** Integration with pacing logs

Pacing Options

Format	Example	Result
Fixed Pacing	75	Exactly 75 seconds between iterations
Random Range	60, 80	Random value between 60-80 seconds (inclusive)

Configuration

Add Pacing Configuration elements to your test plan:

1. **Right-click** Thread Group
2. **Add** → **Config Element** → **Pacing Configuration**
3. **Set** pacing values:

- **Fixed:** Enter single value (e.g., 20 for 20 seconds)
 - **Random:** Enter range (e.g., 15,25 for 15-25 seconds)
4. **Run** test - pacing applied automatically between iterations

Pacing Workflow

1. **First Iteration:** No wait applied (immediate start)
2. **Subsequent Iterations:** Waits for specified pacing time from GUI configuration
3. **Automatic Management:** Plugin handles timing calculations and waits
4. **Visual Feedback:** Pacing status visible in GUI element

Note: Script-based pacing is still available for advanced scenarios, but the GUI configuration provides easier management.

Installation

1. Quick Setup - Clone NFT Team JMeter Repository

```
# Clone the complete NFT Team JMeter installation (recommended)
git clone ssh://git@bitbucket.bics-collaboration.homeoffice.gov.uk/hmponft/jmeter-and-plugins.git nft-jmeter
cd nft-jmeter

# This provides you with:
# - Latest JMeter installation
# - All NFT Team plugins pre-installed
# - Configured properties files
# - Ready to use immediately

export JMETER_HOME=$(pwd)
```

2. Manual Plugin Installation (if not using full clone)

```
# Copy the plugin JAR to JMeter extensions
cp dcu-sts-utils.jar $JMETER_HOME/lib/ext/

# Ensure GlobalSetupListener is registered (should already be configured)
# In jmeter.properties:
global.setup.listener=com.company.jmeter.setup.GlobalSetupListener
```

3. Central STS Configuration

```
# No local STS installation required - uses central STS via HTTPS ingress
# SSL is required but certificates are automatically ignored by the plugin
# STS operations will connect to the central server automatically
# Ensure network connectivity to:
# - simple-table-server-ho-hmpo-ew2tlg1n11-i-team-nft-i-docker-tlg1.service.np.ebsa.local
  (internal)
# - simple-table-server-ho-hmpo-ew2tlg1n11-i-team-nft-i-docker-
  tlg1.np.ebsa.homeoffice.gov.uk (external)
```

4. Verify Installation

```
# Check that GUI elements are available:
# 1. Start JMeter GUI (./bin/jmeter or ./bin/jmeter.sh)
# 2. Right-click Test Plan
# 3. Check Add → Config Element menu for:
#   - Global Setup Configuration (deprecated - automatic version preferred)
#   - STS Configuration
#   - Pacing Configuration
```

Configuration

Complete user.properties Configuration

```
#-----
# Global Template Variables
#-----
template.V_SCHEME=https
template.V_HOST=your-target-server.com
template.V_PORT=443
template.V_STS_HOST=simple-table-server-ho-hmpo-ew2tlg1n11-i-team-nft-i-docker-
tlg1.np.ebsa.homeoffice.gov.uk
template.DATA=/data/

#-----
# Automatic Global Setup Configuration
#-----
global.setup.template.processing=true
global.setup.directory.setup=true
global.setup.hostname.detection=true
global.setup.environment.detection=true
global.setup.git.detection=true
global.setup.logging.verbose=false

#-----
# Central STS Configuration
#-----
sts.use.https=true
```



```
V_STS_LOCAL_HOST=simple-table-server-ho-hmpo-ew2tlg1n11-i-team-nft-i-docker-
tlg1.service.np.ebsa.local
V_STS_HOST_DEFAULT=simple-table-server-ho-hmpo-ew2tlg1n11-i-team-nft-i-docker-
tlg1.np.ebsa.homeoffice.gov.uk
STS_PATH_PREFIX=/sts

#-----
# SSL Configuration
#-----
httpClient4.validate_after_inactivity=0
httpClient4.time_to_live=60000
httpClient4.idletimeout=60000
```



Usage Examples

Example 1: Basic Test Plan with All Components

```
Test Plan Structure:
TestPlan
├─ Global Variables (automatic via Global Setup)
├─ Thread Group
│   ├─ Pacing Configuration
│   │   └─ Pacing: 30,45 (random between 30-45 seconds)
│   ├─ STS Configuration (Data Retrieval - Start of iteration)
│   │   ├─ Row 1: Operation: KEEP, File: users.csv, Variables: USER_ID,PASSWORD
│   │   └─ Row 2: Operation: DEL, File: accounts.csv, Variables: ACCOUNT_ID,BALANCE
│   ├─ HTTP Requests (using ${USER_ID}, ${PASSWORD}, ${ACCOUNT_ID}, ${BALANCE})
│   └─ STS Configuration (Data Recording - End of iteration)
│       ├─ Row 1: Operation: ADDFIRST, File: processed.csv, Values:
${USER_ID},${__time(yyyy-MM-dd)},SUCCESS
│       └─ Row 2: Operation: ADDLAST, File: audit.csv, Values:
${ACCOUNT_ID},${BALANCE},TESTED
└─ [Pacing applied automatically between iterations]
```

Note: Multiple STS Configuration elements can be added, each with multiple rows for different CSV files.

Example 2: Environment-Specific Configuration

```
# Development Environment
template.V_HOST=dev-server.company.com
V_STS_HOST_DEFAULT=simple-table-server-ho-hmpo-ew2tlg1n11-i-team-nft-i-docker-
tlg1.np.ebsa.homeoffice.gov.uk
global.setup.logging.verbose=true

# Production Environment
template.V_HOST=prod-server.company.com
V_STS_LOCAL_HOST=simple-table-server-ho-hmpo-ew2tlg1n11-i-team-nft-i-docker-
tlg1.service.np.ebsa.local
global.setup.logging.verbose=false
```

Example 3: STS Data Management Workflow

GUI Configuration Steps:

Step 1: Add STS Configuration (Data Retrieval)

- └─ Operation: KEEP
- └─ Filename: accounts.csv
- └─ Variables: ACCOUNT_ID,ACCOUNT_TYPE,BALANCE

Step 2: Use retrieved data in HTTP Requests

- └─ URL: /api/account/\${ACCOUNT_ID}
- └─ Headers: Account-Type: \${ACCOUNT_TYPE}

Step 3: Add STS Configuration (Data Processing)

- └─ Operation: ADDFIRST
- └─ Filename: processed_accounts.csv
- └─ Values: \${ACCOUNT_ID},PROCESSED,\${__time(yyyy-MM-dd)}

Troubleshooting

Global Setup Issues

```
# Enable verbose logging
global.setup.logging.verbose=true
```

Check JMeter logs for messages like:

```
INFO [AutoGlobalSetup] Starting Automatic Global Setup...
INFO [AutoGlobalSetup] Template processing: 5 properties processed
INFO [AutoGlobalSetup] Directory setup: JMX_DIR = /path/to/test
```

STS Connectivity Issues

```
# Test central STS connectivity (external)
curl -X GET "https://simple-table-server-ho-hmpo-ew2t1g1n11-i-team-nft-i-docker-t1g1.np.ebsa.homeoffice.gov.uk/sts/READ?FILENAME=test.csv"

# Test central STS connectivity (internal - if on internal network)
curl -X GET "https://simple-table-server-ho-hmpo-ew2t1g1n11-i-team-nft-i-docker-t1g1.service.np.ebsa.local/sts/READ?FILENAME=test.csv"

# Note: Central STS server is managed centrally - no local server installation required
```

SSL Certificate Issues

```
# SSL is required for central STS - certificates are automatically ignored
# Do NOT disable HTTPS for central STS
sts.use.https=true

# SSL debugging (if needed for troubleshooting)
javax.net.debug=ssl,handshake
```

Note: The plugin automatically bypasses SSL certificate validation (equivalent to `curl -k`), so SSL certificate errors should not occur with the central STS.

Common Property Issues

Issue	Solution
Template properties not processing	Check <code>global.setup.template.processing=true</code>
STS host not resolving	Verify <code>V_STS_HOST_DEFAULT</code> or <code>V_STS_LOCAL_HOST</code>
Pacing not working	Verify Pacing Configuration element is added to Thread Group
Directory paths incorrect	Check <code>global.setup.directory.setup=true</code>

Migration Notes

From Common Scripts (Critical - Tests Will Fail Without This)

1. **Remove ALL references** to "common scripts" from every test element in your JMeter test plan
2. **Delete** any JSR223 samplers that reference:
 - `${__P(C_SCRIPTS)}*`
 - `scripts/common/*`
 - `global_setup.groovy`
 - `group_init.groovy`
 - `group_end.groovy`
 - `STS.groovy`
 - `DCU.groovy`
 - `pacing.groovy`
 - Any other common script references
3. **Replace** with appropriate GUI elements:
 - Global setup → Automatic (no action needed)
 - STS operations → STS Configuration elements
 - Pacing → Pacing Configuration elements

4. **Test thoroughly** - any remaining script references will cause failures in the streamlined pipeline

From Manual Pacing Scripts

1. **Replace** custom pacing logic with Pacing Configuration GUI elements
2. **Configure** pacing values through GUI (single values or ranges)
3. **Remove** manual pacing scripts - plugin handles timing automatically



Additional Resources

- **Detailed Documentation:** See individual README files for each component
- **Template Test Plans:** Available in `bin/templates/Template.jmx`
- **GUI Element Reference:** Access via JMeter's Add → Config Element menu
- **Property Reference:** See `auto-global-setup.properties` for complete options



Best Practices

1. **Start Small:** Enable one component at a time for initial testing
2. **Use Verbose Logging:** Enable during setup and debugging phases
3. **Environment Separation:** Use different property files per environment
4. **Version Control:** Keep property files in version control
5. **Monitor Performance:** Watch for impact of Git detection in large repositories
6. **Test Connectivity:** Verify STS connectivity before load tests
7. **Document Changes:** Record any custom configurations for team knowledge



Developed by the NFT Team

This plugin follows UK English spelling conventions and modular design principles for maximum reusability and maintainability.

Author & Contact:

- **David Campbell**, NFT Innovation Lead
- **Email:** david.campbell@digital.homeoffice.gov.uk
- **Phone:** 07958 562 789
- **NFT Team Email:** NFTHMPO@digital.homeoffice.gov.uk

For support or questions, please contact David Campbell or the NFT Team.