Path Traversal Filter Bypass in Dataflow Module

Moderate **sreichel** published **GHSA-6vqf-6fhm-7rc6** 2 days ago

Package

php **openmage/magento-lts** (Composer)

Affected versions	Patched versions
< 20.17.0	20.17.0

Description

Source: https://hackerone.com/reports/3482926
Original GHSA: https://github.com/OpenMage/magento-lts/security/advisories/GHSA-9qv7-qm9f-2xfh

---

The Dataflow module in OpenMage LTS uses a weak blacklist filter (`str_replace('../', '', $input)`) to prevent path traversal attacks. This filter can be bypassed using patterns like `..././` or `..../`, which after the replacement still result in `../`. An authenticated administrator can exploit this to read arbitrary files from the server filesystem.

CVSS Vector

```
CVSS:3.1/AV:N/AC:L/PR:H/UI:N/S:U/C:H/I:N/A:N
```

Metric	Value	Justification
Attack Vector (AV)	Network	Exploitable via admin panel
Attack Complexity (AC)	Low	Simple bypass pattern
Privileges Required (PR)	High	Requires admin authentication
User Interaction (UI)	None	No additional user interaction needed

Metric	Value	Justification
Scope (S)	Unchanged	Impacts the vulnerable component
Confidentiality (C)	High	Can read sensitive system files
Integrity (I)	None	Read-only vulnerability
Availability (A)	None	No impact on availability

Affected Products

- OpenMage LTS versions < 20.16.1
- All versions derived from Magento 1.x with these code paths

Affected Files

File	Line	Vulnerable C
`app/code/core/Mage/Dataflow/Model/Convert/Parser/Csv.php`	67	`str_replace('', urldecode(`
`app/code/core/Mage/Dataflow/Model/Convert/Parser/Xml/Excel.php`	63	`str_replace('', urldecode(`

Vulnerability Details

The Dataflow module allows administrators to import data from files. The `files` parameter specifies which file to import from the `var/import/` directory. To prevent path traversal, the code uses `str_replace()` to remove `../` sequences:

```
$file = Mage::app()->getConfig()->getTempVarDir() . '/import/'
    . str_replace('../', '', urldecode(Mage::app()->getRequest()->getParam('files')));
```

However, `str_replace()` only performs a single pass, making it trivially bypassable:

Bypass Examples

Input	After `str_replace('../', '', ...)`	Result
`..././`	`../`	Bypass
`....//`	`../`	Bypass
`..././..././..././etc/passwd`	`../../../etc/passwd`	File read

Attack Scenario

1. Attacker gains admin access (via compromised credentials, social engineering, etc.)
2. Navigate to System > Import/Export > Dataflow Profiles
3. Create or modify an import profile
4. Set the `files` parameter to: `..././..././..././etc/passwd`
5. Run the profile to read the contents of `/etc/passwd`

Proof of Concept

```
# Request to Dataflow with bypass pattern
GET /admin/system_convert_gui/run/id/1/?files=..././..././..././etc/passwd

# The str_replace removes '../' leaving:
# ..././..././..././etc/passwd -> ../../../etc/passwd

# Final path resolves to:
# /var/www/html/var/import/../../../etc/passwd -> /etc/passwd
```

Remediation

Replace the weak `str_replace()` filter with `basename()` to extract only the filename:

```
// Before (vulnerable)
$file = Mage::app()->getConfig()->getTempVarDir() . '/import/'
    . str_replace('../', '', urldecode(Mage::app()->getRequest()->getParam('files')));

// After (fixed)
$file = Mage::app()->getConfig()->getTempVarDir() . '/import/'
    . basename(urldecode(Mage::app()->getRequest()->getParam('files')));
```

Using `basename()` ensures only the filename portion is used, completely preventing any path traversal regardless of the input pattern.

Workarounds

If immediate upgrade is not possible:

1. **Restrict admin access**: Limit Dataflow access to trusted administrators only
2. **Disable Dataflow**: If not in use, disable the Dataflow module entirely
3. **Web Application Firewall**: Block requests containing path traversal patterns
4. **File permissions**: Ensure the web server user has minimal filesystem permissions
5. **Monitor admin activity**: Alert on suspicious Dataflow profile execution

Impact

An attacker with admin access can read sensitive files including:

- `/etc/passwd` - System user information
- `app/etc/local.xml` - Database credentials
- `.env` files - Environment secrets
- Log files - Potentially sensitive application data
- Configuration files - Server and application configuration

References

- CWE-22: Path Traversal
- CWE-184: Incomplete List of Disallowed Inputs
- OWASP: Path Traversal

Credit

This vulnerability was discovered and responsibly disclosed by blackhat2013 through HackerOne.

Timeline

- **2025-12-31**: Vulnerability reported via HackerOne
- **2026-01-21**: Fix developed and tested
- **2026-XX-XX**: Security release v20.16.1 published
- **2026-XX-XX**: Public disclosure

Severity

Moderate 4.9 / 10

CVSS v3 base metrics

Attack vector	Network
Attack complexity	Low
Privileges required	High
User interaction	None
Scope	Unchanged
Confidentiality	High
Integrity	None

Availability	None

Learn more about base metrics

CVSS:3.1/AV:N/AC:L/PR:H/UI:N/S:U/C:H/I:N/A:N

CVE ID

CVE-2026-25525

Weaknesses

- CWE-22
- CWE-184