

# MySQL: Error "No space left on device" en /tmp

## Síntomas del problema

Los usuarios reportan errores similares a estos en cPanel, phpMyAdmin y otras aplicaciones:

```
Can't create/write to file '/tmp/#sql_911_0.MAI' (Errcode: 28 "No space left on device")
Se experimentó un error mientras se obtenían los datos: Can't create/write to file
'/tmp/#sql_1234_0.MAI'
```

## Diagnóstico paso a paso

### Verificar espacio en disco

```
df -h

Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1        20G   8.5G   10G   46% /
tmpfs            2.0G   144K   2.0G    1% /tmp
```

Salida para indicar que todo es normal

### Verificar inodos (la causa real)

```
df -i

Filesystem      Inodes   IUsed   IFree IUse% Mounted on
/dev/sda1      1310720  145230 1165490   11% /
tmpfs           524288   524271     17  100% /tmp
```

**Observación:** La salida indica el agotamiento al 100% de los inodos.

### Identificar los archivos problemáticos

Al intentar listar `/tmp` con `ls`, obtendremos:

```
ls /tmp
-bash: /bin/ls: Argument list too long
```

Por eso utilizamos `find`:

```
find /tmp -maxdepth 1 -name 'sess*' | wc -l
```

Resultado atípico: `288789` (cientos de miles de archivos)

## Causa raíz

Acumulación masiva de archivos de sesión PHP (`sess_*`) que no se eliminan automáticamente, causando agotamiento de inodos en `/tmp`.

## Solución inmediata

```
find /tmp -maxdepth 1 -type f -name 'sess*' -delete
```

**¿Por qué usar find?** Porque `rm sess*` falla con "Argument list too long" cuando hay miles de archivos.

## Prevención

## Configuración automática de limpieza

Añadir al crontab de root:

```
# Limpieza diaria de archivos temporales antiguos (>24h)
0 3 * * * find /tmp -type f -name "sess_*" -mtime +1 -delete >/dev/null 2>&1
```

## Script para aplicar cambios PHP en todas las versiones de cPanel

**Contexto:** En cPanel cada versión PHP tiene su propio archivo de configuración:

```
/opt/cpanel/ea-php81/root/etc/php.ini
/opt/cpanel/ea-php82/root/etc/php.ini
/opt/cpanel/ea-php83/root/etc/php.ini
```

## Script automático para todas las versiones:

```
#!/bin/bash
# fix_php_gc_all_versions.sh
# Script para corregir configuración de garbage collector en todas las versiones PHP de cPanel

echo "=== Corrigiendo configuración PHP Garbage Collector ==="
echo "Fecha: $(date)"
echo

# Buscar todas las versiones de PHP instaladas de forma eficiente
PHP_VERSIONS=$(find /opt/cpanel -maxdepth 1 -name "ea-php*" -type d | grep -E "ea-php[0-9]+" |
sort -V)

if [ -z "$PHP_VERSIONS" ]; then
    echo "❌ No se encontraron versiones de PHP en cPanel (/opt/cpanel/ea-php*)"
    exit 1
fi

echo "Versiones de PHP encontradas:"
echo "$PHP_VERSIONS" | sed 's/.*ea-php/- PHP /' | sed 's/$//'
echo

for php_dir in $PHP_VERSIONS; do
    php_ini="$php_dir/root/etc/php.ini"
    version=$(basename "$php_dir" | sed 's/ea-php//')

    echo "--- Procesando PHP $version ---"

    if [ ! -f "$php_ini" ]; then
        echo "⚠️ No existe: $php_ini"
        continue
    fi

    # Crear backup
    cp "$php_ini" "$php_ini.backup.$(date +%Y%m%d_%H%M%S)"
```

```

echo "📁 Backup creado"

# Mostrar configuración actual
current_prob=$(grep -E "^session\.gc_probability\s*=" "$php_ini" | cut -d=' ' -f2 | tr -d '
')
current_div=$(grep -E "^session\.gc_divisor\s*=" "$php_ini" | cut -d=' ' -f2 | tr -d ' ')

echo "   Configuración actual: gc_probability=$current_prob, gc_divisor=$current_div"

# Aplicar cambios
sed -i 's/^session\.gc_probability\s*=.*\/session.gc_probability = 1/' "$php_ini"
sed -i 's/^session\.gc_divisor\s*=.*\/session.gc_divisor = 1000/' "$php_ini"

# Verificar cambios
new_prob=$(grep -E "^session\.gc_probability\s*=" "$php_ini" | cut -d=' ' -f2 | tr -d ' ')
new_div=$(grep -E "^session\.gc_divisor\s*=" "$php_ini" | cut -d=' ' -f2 | tr -d ' ')

echo "   Nueva configuración: gc_probability=$new_prob, gc_divisor=$new_div"
echo "   📁 PHP $version actualizado"
echo
done

echo "=== Resumen ==="
echo "📁 Cambios aplicados a todas las versiones PHP"
echo "⚠️ Los cambios se aplicarán en nuevas sesiones PHP"
echo

# Comando de verificación
echo "Verificar cambios:"
echo "grep -E 'session\.gc_(probability|divisor)' /opt/cpanel/ea-php*/root/etc/php.ini"

```

**Version actualizada** del script se mantiene en [Git Lab Castris - Utilidades](#)

### Uso del script:

```

chmod +x fix_php_gc_all_versions.sh
./fix_php_gc_all_versions.sh

```

# Análisis de la configuración PHP problemática

## Configuración actual encontrada en servidores cPanel:

```
session.gc_probability = 0      ; ▲ PROBLEMA: Numerador en 0
session.gc_divisor = 0          ; ▲ PROBLEMA: Denominador en 0
session.gc_maxlifetime = 1440  ; ☐ Correcto: 24 minutos (1440 segundos)
```

**Explicación del problema:** El garbage collector de PHP funciona con esta fórmula:

- Probabilidad de limpieza =  $gc\_probability / gc\_divisor$

Con la configuración actual:  $0 / 0 = \text{indefinido}$  → **Garbage collector completamente desactivado**

## Qué significa cada parámetro:

- `gc_probability`: Numerador de la probabilidad (cuántas veces de cada X)
- `gc_divisor`: Denominador de la probabilidad (el total X)
- `gc_maxlifetime`: Tiempo en segundos después del cual una sesión se considera "basura"

## Ejemplos de probabilidades:

- $1/100 = 1\%$  → 1 de cada 100 peticiones ejecuta limpieza
- $1/1000 = 0.1\%$  → 1 de cada 1000 peticiones ejecuta limpieza
- $0/0 = \text{desactivado}$  → Nunca se ejecuta limpieza automática

# Configuración PHP corregida

## Para servidores de producción (recomendado):

```
session.gc_probability = 1      ; Cambiar de 0 a 1
session.gc_divisor = 1000       ; Cambiar de 0 a 1000 (0.1% probabilidad)
session.gc_maxlifetime = 1440  ; Mantener actual
```

## Para servidores de desarrollo/bajo tráfico:

```
session.gc_probability = 1      ; Cambiar de 0 a 1
session.gc_divisor = 100        ; Cambiar de 0 a 100 (1% probabilidad)
session.gc_maxlifetime = 1440  ; Mantener actual
```

## Justificación de los valores:

- **Producción (1/1000):** Menor impacto en rendimiento, suficiente con alto tráfico
- **Desarrollo (1/100):** Mayor frecuencia de limpieza necesaria con poco tráfico

# Monitoreo de inodos

Script para alertas:

```
#!/bin/bash
#
# check_tmp_inodes.sh
# Checks the inode usage on /tmp and sends an alert if it exceeds a threshold.
#

# Ensure the script stops if any command fails
set -euo pipefail

# --- CONFIGURATION ---
# Path to the main configuration file
# Check repo for crons/.check_tmp_inodes.cfg file demo
# Not edit this file, copy the crons/.check_tmp_inodes.cfg file to /root/.check_tmp_inodes.cfg
CONFIG_FILE="/root/.check_tmp_inodes.cfg"

# --- FUNCTIONS ---

# Function to log errors and exit
# Usage: error_exit "Error message"
error_exit() {
    echo "ERROR: $1" >&2
    exit 1
}

# --- MAIN SCRIPT ---

# 1. Check if the configuration file exists and load it
if [ ! -f "$CONFIG_FILE" ]; then
    error_exit "Configuration file not found at $CONFIG_FILE"
fi
```

```

# Load variables from the configuration file
# shellcheck source=/dev/null
source "$CONFIG_FILE"

# 2. Verify that the necessary variables are defined
if [ -z "${EMAIL_TO-}" ] || [ -z "${THRESHOLD-}" ]; then
    error_exit "EMAIL_TO and THRESHOLD variables must be defined in $CONFIG_FILE"
fi

# 3. Get the current inode usage
# 'tail -n1' is used to be more robust on systems where 'df' might have more than 2 lines
USAGE=$(df -i /tmp | tail -n1 | awk '{print $5}' | sed 's/%//')

if ! [[ "$USAGE" =~ ^[0-9]+$ ]]; then
    error_exit "Could not retrieve a numeric value for inode usage. Got: '$USAGE'"
fi

# 4. Compare usage with the threshold and send an alert if necessary
echo "INFO: Current inode usage on /tmp: ${USAGE}% (Alert threshold: >${THRESHOLD}%)"

if [ "$USAGE" -gt "$THRESHOLD" ]; then
    # Build the message body
    HOSTNAME=$(hostname -f)
    BODY="ALERT on server: $HOSTNAME

Inode usage in the /tmp directory has exceeded the ${THRESHOLD}% threshold.

Current usage: ${USAGE}%

It is recommended to check the contents of /tmp to free up inode space.
"
    # Send the email
    echo "$BODY" | mail -s "$EMAIL_SUBJECT" "$EMAIL_TO"

    echo "ALERT: Threshold exceeded. Notification email sent to $EMAIL_TO."
else
    echo "INFO: Inode usage is within normal limits."
fi

exit 0

```

# Configuración de directorios temporales seguros

## Problema con la configuración actual

- `/home/user/tmp` → enlace simbólico a `/tmp` (cPanel)
- Todos los usuarios comparten el mismo espacio de inodos

## Solución recomendada

Configurar directorios temporales individuales:

```
// En configuración PHP por usuario
session.save_path = "/home/USERNAME/temporal"
upload_tmp_dir = "/home/USERNAME/temporal"
```

## Monitoreo continuo

## Comando para verificar estado actual

```
echo "Espacio: $(df -h /tmp | awk 'NR==2 {print $5}')"
echo "Inodos: $(df -i /tmp | awk 'NR==2 {print $5}')"
echo "Archivos sess: $(find /tmp -maxdepth 1 -name 'sess*' 2>/dev/null | wc -l)"
```

## Señales de alerta temprana

- Uso de inodos > 80% en `/tmp`
- Más de 10,000 archivos `sess_*`
- Quejas de usuarios sobre lentitud en aplicaciones web

---

**Severidad:** Crítica - Afecta disponibilidad de MySQL y aplicaciones web

**Impacto:** Todos los usuarios del servidor

**Tiempo de resolución:** 2-5 minutos con la solución inmediata



## Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de [Castris](#)

Si necesitas soporte profesional puedes contratar con Castris [soporte profesional](#).

---

Revision #1

Created 27 June 2025 16:18:55 by Abkrim

Updated 27 June 2025 16:19:10 by Abkrim