Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

SQL Server database backups behave before and after encryption, what changes in **backup size**, and how the **transfer rate** parameter is affected.

🧩 1. Backup Size Before and After Encryption

SQL Server offers **backup encryption** (introduced in SQL Server 2014) which encrypts the backup file at the time of creation using a certificate or asymmetric key.

🔍 What Changes After Encryption

Aspect	Before Encryption	After Encryption
Backup File Size	Normally smaller if using compression. Size depends on database data + compression ratio.	Slightly larger compared to compressed unencrypted backups (5-10% overhead) because encryption reduces compressibility.
Backup Compression	Fully effective. High compression ratio possible for text-heavy data.	Backup compression still works but achieves lower ratio (encrypted data is almost incompressible).
Security	Backup file can be restored by anyone who has the .bak file.	Requires certificate/private key used for encryption — improves security.

Key Point:

- If you use **Compression + Encryption** together → Compression happens **first**, then encryption.
- That means you still benefit from compression, but final .bak file will be slightly larger than the compressed one without encryption.

🧩 2. Transfer Rate Parameter in SQL Server Backups

SQL Server shows **backup throughput** as **MB/sec** in SSMS or in the output of BACKUP DATABASE.
This is the **Transfer Rate** parameter.

Example:

```
BACKUP DATABASE [MyDB]
TO DISK = 'D:\Backup\MyDB_Enc.bak'
WITH ENCRYPTION (ALGORITHM = AES_256, SERVER CERTIFICATE = MyBackupCert),
COMPRESSION, STATS=10;
```

Sample Output:

```
10 percent processed.
20 percent processed.
...
Processed 1200 pages for database 'MyDB', file 'MyDB_Data' on file 1.
Processed 3 pages for database 'MyDB', file 'MyDB_Log' on file 1.
BACKUP DATABASE successfully processed 1203 pages in 3.421 seconds (2.830 MB/sec).
```

Here 2.830 MB/sec is the **transfer rate**.

🧠 **Factors Affecting Transfer Rate**

Factor	Impact
Encryption Algorithm	AES_256 is most secure but slightly slower than AES_128.
CPU Performance	Encryption is CPU-intensive — weak CPUs reduce backup throughput.
Disk Speed (I/O)	Slow backup destination disk can become a bottleneck.
Compression	Can actually increase throughput (less data written to disk).
Network Speed (if backing up to network share)	Latency and bandwidth limit transfer rate.

🛠 3. Steps to Measure Backup Size & Transfer Rate (Before vs After Encryption)

Step 1: Run Backup Without Encryption

```
BACKUP DATABASE [MyDB]
TO DISK = 'D:\Backup\MyDB_NoEnc.bak'
WITH COMPRESSION, STATS=10;
```

- Note **final backup file size** from Windows Explorer or:

```
EXEC master.dbo.xp_fileexist 'D:\Backup\MyDB_NoEnc.bak';
```

- Check transfer rate in SQL output.

Step 2: Run Backup With Encryption

First, create a **Database Master Key** and **Certificate** (one-time setup):


```
USE master;
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'StrongPasswordHere';
CREATE CERTIFICATE MyBackupCert
WITH SUBJECT = 'Database Backup Encryption Certificate';
```

Then run encrypted backup:

```
BACKUP DATABASE [MyDB]
TO DISK = 'D:\Backup\MyDB_Enc.bak'
WITH ENCRYPTION (ALGORITHM = AES_256, SERVER CERTIFICATE = MyBackupCert),
COMPRESSION, STATS=10;
```

- Again, note backup size & transfer rate.

📊 **Expected Results**

Test	Backup Size	Transfer Rate
Before Encryption (Compressed)	Smallest size (depends on data compression ratio)	Highest MB/sec (less CPU cost)
After Encryption (Compressed)	Slightly larger (5–10% overhead)	Slightly lower MB/sec (depends on CPU load & encryption algorithm)
After Encryption (No Compression)	Significantly larger (because encryption makes compression impossible later)	Similar or slightly slower throughput

Praveen Madupu
Mb: +91 98661 30093
Sr. Database Administrator

🎯 **Best Practices**

- **Always use compression + encryption together** → best size & security tradeoff.
- **Monitor CPU usage** during encrypted backups — avoid running during peak load if CPU-bound.
- **Test different algorithms (AES_128, AES_256, TRIPLE_DES)** → choose the best balance between security & speed.
- **Measure transfer rate over several runs** → disk caching or network congestion can cause variations.

Here's a **practical sample comparison table** for a ~1 TB database backup showing **before vs after encryption** with real-world style numbers. This will help your DBA team understand the impact on **backup size** and **transfer rate** during benchmarking.

📊 **Sample Comparison: Backup Size & Transfer Rate**

Scenario	Backup File Size	Compression Ratio	Transfer Rate (MB/sec)	Notes
Uncompressed, No Encryption	**1,024 GB** (≈ 1 TB)	N/A	160 MB/sec	Fastest write but huge file size, no security.
Compressed, No Encryption	**270 GB**	~3.8x reduction	210 MB/sec	Compression reduces I/O → higher throughput.
Uncompressed + Encryption (AES_256)	**1,040 GB** (≈ +1.5%)	N/A	140 MB/sec	Encryption overhead slows backup slightly.
Compressed + Encryption (AES_256)	**285 GB** (≈ +5.5%)	~3.6x reduction	190 MB/sec	Best balance — slightly bigger than compressed-only but secured.

🧠 **Interpretation of Results**

- **Backup Size Impact:**
  - Encryption increases file size by ~5–10% when combined with compression.
  - Without compression, encrypted files are nearly same size as source data.
- **Transfer Rate Impact:**
  - Encryption adds CPU overhead → slightly reduces MB/sec (10–15% drop typical).
  - Compression usually offsets some performance hit (less data written).

🛠 **Benchmarking Tips**

Run Multiple Tests at different times of day to factor in CPU/Disk contention.
Capture Backup History from msdb.dbo.backupset:

```
SELECT
database_name, backup_start_date, backup_finish_date,
backup_size/1024/1024/1024 AS BackupSize_GB,
compressed_backup_size/1024/1024/1024 AS CompressedSize_GB,
 DATEDIFF(SECOND, backup_start_date, backup_finish_date) AS Duration_Seconds,
  (compressed_backup_size / 1024.0 / 1024.0) /
 NULLIF(DATEDIFF(SECOND, backup_start_date, backup_finish_date), 0) AS TransferRate_MBsec
 FROM msdb.dbo.backupset
WHERE database_name = 'MyDB'
ORDER BY backup_start_date DESC;
```

This gives **actual size, compression ratio, and transfer rate** for historical backups.

Monitor CPU Usage:

- Use sys.dm_exec_requests and sys.dm_os_performance_counters to see CPU pressure during backup.
- If CPU is saturated, consider **AES_128** (faster) instead of AES_256 for non-regulated environments.

Here's the **visual comparison bar chart** — showing backup size (GB) vs transfer rate (MB/sec) across the four scenarios. This makes it easy for your DBA team to present encryption impact on both size and speed.