



The Knowledge Center for Oracle Professionals

Product Review: HyperBac for Oracle

Jim Czuprynski, Jim.Czuprynski@us.fujitsu.com

Synopsis. Oracle 10g Release 2 (10gR2) offers significant enhancements to database backup and recovery with Recovery Manager (RMAN), including the ability to compress backup sets and encrypt backups for increased security. A relatively new product called HyperBac provides an excellent alternative to Oracle 10gR2 compressed backupset methodology without the need to modify any RMAN backup, restoration, and recovery scripts. In addition, HyperBac also provides the ability to compress and encrypt Oracle DataPump dumpsets. This article provides an overview of HyperBac features that nicely complement Oracle 10gR2's enhancements to database backups and exports.

When I first reviewed the new features that Oracle Database 10g Release 1 (10gR1) offered in the Recovery Manager (RMAN) arena, I recognized immediately that the capability to create *incrementally updateable image copy backups* was an immense improvement over prior releases. It offers DBAs the means to perform a single incremental level 0 image copy backup of the database's datafiles and then simply apply just the changed blocks in later incremental level 1 backup sets of those same datafiles to update the initial base backup. (For details on how to implement this feature, please see my [article](#) on this and other RMAN enhancements in Oracle 10g.)

Oracle 10gR1 also introduced a third way to create a database backup, the *compressed backup set*, to the RMAN tool set. Compressed backup sets still contain all the same block-level information as their uncompressed counterparts, but take up a lot less space – normally between 25% to 50% compression. For datafiles that contain predominantly character-based information, the compression ratio could be as great as 90%.

Oracle 10gR2 also added the ability to create *encrypted backup sets*. RMAN now can apply advanced encryption algorithms to any Oracle backup set via a password derived from one of three sources: *Transparent Data Encryption (TDE) only, password only, or either password or TDE*. Once encrypted, the backup sets cannot be decrypted without application of the corresponding password. (Please see my [article](#) on RMAN encrypted backups for a full explanation of these security features.)

These are impressive features, but I believe that a few capabilities are unfortunately still missing in Oracle Database 10g:

- **Compressed backup sets take much longer to create than uncompressed backup sets.** Based on my experiments and feedback from other DBAs, a compressed backup set can take *up to five times longer* to create than its uncompressed counterpart and its creation tends to consume large amounts of CPU resources. This is apparently due to an extremely aggressive implementation of the Lempel-Ziv compression algorithm during compressed backup set creation, and is unfortunately not tunable in Oracle Database 10g.
- **Encrypting backups with Transparent Data Encryption requires additional licensing costs.** The simplest way to enable both database encryption capabilities as well as encrypting Oracle backups is to use the Oracle 10gR2's Transparent Data Encryption (TDE) security features. However, TDE requires the purchase of Oracle Database Advanced Security, an extra licensing cost that may not be insignificant for small- and medium-sized Oracle Database shops.
- **Image copy backups cannot be encrypted.** As previously described, Oracle Database 10g's suggested backup strategy -- Incrementally Updateable Image Copies -- is the new standard backup method. Since text strings stored within a tablespace's datafile can indeed be read "in the clear" with even simple operating system functions, it's important that an image copy backup of any datafile can be secured as well. However, Oracle Database 10g prohibits the direct application of encryption methods to an image copy backup of a datafile. This is a true disappointment, since many shops simply copy their image copy backups from the Flash Recovery Area directly to tape media.

HyperBac Features Overview

Fortunately, I've found a potential solution to these shortfalls with an application named **HyperBac** from Xceleon Technologies, a technology partner with Oracle Corporation. The [HyperBac web site](#) offers plenty of information about its availability for several different RDBMSs and platforms besides Oracle.

The easiest way to understand how HyperBac works is to think of it as a *filter*. HyperBac simply *accepts input* from RMAN or DataPump, *filters* that input by applying either encryption or compression, and then *produces an output file* that's saved to disk. When restoring a datafile, or importing data back into an Oracle database, HyperBac *applies the filter in reverse*, either decrypting or decompressing the backup files in the backup set or export dumpset during the restoration or import operation. HyperBac offers the following features:

- **Backup Compression.** As noted previously, Oracle 10g does already offer the ability to create compressed backup sets, but the rather aggressive Lempel-Ziv compression methodology tends to result in noticeably longer elapsed time to create these types of RMAN backup sets. The good news here is that HyperBac can create compressed backup sets that, while not as aggressively compressed as in Oracle 10g, are considerably smaller than standard backup sets. Moreover, it's also possible to apply backup compression to an image copy backup *while RMAN is writing out the final output file*.

- **Backup Encryption.** Oracle 10g also already offers the ability to apply encryption to RMAN backup sets, but RMAN image copy backups cannot be encrypted. Once again, HyperBac comes to the rescue because I can simply apply the HyperBac encryption filter to any RMAN output dataset whether it's a backup set, compressed backup set, or even an image copy backup. HyperBac allows the DBA to select from either 128-bit, 192-bit, or 256-bit AES encryption algorithms. Best of all, HyperBac doesn't require any additional setup to enable backup decryption; essentially, HyperBac just applies its encryption algorithm in reverse to decrypt the incoming encrypted backup set as RMAN reads from it during a **RESTORE** operation.
- **Export File Compression.** HyperBac offers the ability to apply its powerful compression algorithms directly to either a DataPump Export dump set or a standard export file as it's being written to its destination directory. As you might guess, HyperBac simply applies the same decompression algorithm to uncompress an export file as it's being imported into an Oracle 10g database.
- **Export File Encryption.** Since it essentially functions as a filter, HyperBac also permits the encryption of a DataPump Export dump set file or export file as it's being written to its destination directory. And as you might guess, HyperBac simply applies the encryption as a filter in reverse when importing data from an encrypted DataPump Export dump set or export file. The same encryption algorithms available for RMAN encryption are also available for encryption of dump sets and export files.

Evaluation Environment Setup

For the record, I'm using a dual-core AMD Athlon 64-bit CPU (Winchester 4200) to run VMWare Server 1.0.3 to access a virtualized database server environment. I configured my VMWare virtual machine to fully utilize both of the dual-core CPUs but just 1024 MB of memory. For my VMWare virtual machine's OS, I used the Oracle Enterprise Linux (OEL) version that is equivalent to Red Hat Enterprise Linux 4.0 Update 4 (i.e. Linux kernel version *2.6.9-42.0.0.1.ELsmp*).

Once my VMWare virtual machine was configured, I installed the software for Oracle Database 10gR2 (10.2.0.1.0) and then constructed the standard 10gR2 "seed" database, including the standard sample schemas. I then used RMAN to create a series of baseline backup sets and image copy backups for later comparisons against those I'd be creating with HyperBac. [Listing 1](#) illustrates the creation of an RMAN *image copy full backup* of the database; [Listing 2](#) and [Listing 3](#) show the creation of full backups of the database using both the normal compressed backup set methods, respectively.

Installing HyperBac for Oracle

HyperBac is extremely easy to install. I simply downloaded the corresponding HyperBac demonstration software that matched my VMWare virtual machine's environment, obtained the appropriate key for the 30-day demonstration, and then installed the

software on my database server. Note that the installation of HyperBac for Oracle does require **root**-level access to the host:

```
$>./hyperbac-oracle-linux -i

Hyperbac for PRODUCTION PLATFORMS

END-USER LICENSE AGREEMENT

THIS IS A CONTRACT, PLEASE READ IT CAREFULLY, BY INSTALLING THIS
SOFTWARE YOU ACCEPT ALL THE TERMS AND CONDITIONS OF THIS AGREEMENT, YOU
WILL NOT BE ABLE TO USE THIS SOFTWARE UNLESS YOU ACCEPT THIS AGREEMENT.
BY USING THE SOFTWARE, YOU ARE AGREEING TO BE BOUND BY THE TERMS OF
THIS LICENSE. IF YOU DO NOT AGREE TO ANY TERM OR CONDITION, YOU MAY NOT
INSTALL OR RUN THIS SOFTWARE.

...
... (remainder of licensing not shown for sake of brevity)
...

2005 Xceleron Technologies. All Rights Reserved.

If you agree to the license agreement press 'Y' or 'N' to end the
installation: Yes

IMPORTANT: Product licensing message:

HyperBac has 30 evaluation days remaining. At the end of the evaluation
period the product will stop functioning.
To obtain a license file contact www.hyperbac.com with the system key:
<systemkey obscured>, or directly with the link:
http://www.hyperbac.com/obtainlicense.asp?systemkey=<systemkey obscured >

Press (N)ext to continue...
N

Installing hyperbac service files...OK
Before configuring the service to start at boot time, it is recommended
that you start the service now to ensure that it is configured
correctly.
Start service now (Y/N): Yes
Starting hyperbac: [ OK ]
Auto start service (Y/N): Yes

Congratulations on successfully installing HyperBac, for information on
configuring and using the HyperBac product, please refer to the
'Getting Started Guide for HyperBac'. This document is located at:
file:///etc/hyperbac/doc/GettingStarted.html
```

I used the default configuration for HyperBac for Oracle as defined by HyperBac's configuration file, aptly named **hyperbac.conf**, that's stored in the **/etc/hyperbac** directory. Here's an example of its contents:

```
IncludeList=*;[20,0]|*.hbc;[20,0]+*;[20,0]|*.hbe;[20,26128]
IndexPath=indexes
KeyPath=keys
IndexPath=indexes
LicencePath=license/hyperbac.lic
LogPath=logs
DebugFlags=1
```

The most important parameter in this configuration file is the one named **IncludeList**. It defines the *file extensions* that HyperBac will perform its operations against, as well as the operation *scope* – either simply *compression*, or both *compression and encryption* – which it will perform to any files that match the listed filename expressions listed. In this default configuration, for example, HyperBac for Oracle will apply its compression capability against any file that has a file extension of **hbc**, and it will apply its combined compression and encryption capability against any file that has an extension of **hbe**.

(See *Appendix C* of the *Getting Started with HyperBac for Linux Systems* guide for excellent documentation of the parameters that can be specified in **hyperbac.conf**.)

Compressing RMAN Backup Sets with HyperBac

HyperBac can compress the files that comprise an Oracle Database 10g RMAN backup set while the backup set is being created, and it makes no difference if it's a "normal" backup set or a compressed backup set. I've preserved the results of both RMAN operations in [Listing 4](#) and [Listing 5](#), respectively, and **Table 1** below shows the relative savings obtained by using HyperBac to compress both types of backup files:

Table 1. HyperBac Results: RMAN Backup Set Compression			
Backup Type	Resulting RMAN Backup Set Size	With HyperBac Compression	Savings
Normal Backup Set	597.50	124.60	79.1%
Compressed Backup Set	110.20	103.70	5.9%
<i>All file sizes are in MB unless otherwise noted.</i>			

Compressing RMAN Image Copy Backups with HyperBac

Even more impressive, HyperBac can compress the files that are created during an RMAN image copy backup operation *while the image copy files are being created*. I used the code shown in [Listing 6](#) to create an image copy backup of the seed database while routing the resulting backup files through the HyperBac filtering mechanism. **Table 2** illustrates the relative space savings – an average of better than 85% - when compared to a standard RMAN image copy of the same datafiles:

Table 2. HyperBac Results: RMAN Image Copy Backup Compression				
Datafile Name	Actual Size	RMAN Image Copy	HyperBac Image Copy	Relative Savings
SYSTEM01.DBF	480.00	480.00	89.70	81.3%
UNDOTBS01.DBF	40.00	40.00	6.20	84.5%
SYSAUX01.DBF	240.00	240.00	14.60	93.9%
USERS01.DBF	5.00	5.00	0.16	96.8%
EXAMPLE01.DBF	100.00	100.00	16.10	83.9%
Total:	865.00	865.00	126.76	85.3%
<i>All file sizes are in MB unless otherwise noted.</i>				

Encrypting RMAN Backup Sets with HyperBac

HyperBac can also simultaneously *compress and encrypt* RMAN backup sets with virtually no modification of the existing RMAN script that creates the backup files. The only change required is the addition of a **FORMAT** directive within the backup script so

that the file extension for each backup file created matches one of those named in the HyperBac configuration file.

To illustrate, I used the RMAN backup script shown in [Listing 7](#) to create a full database backup of my target Oracle database, producing a “normal” RMAN backup set. Note that the only modification was the addition of a **FORMAT** directive that names each image copy backup file so that its extension is **.hbe**. The resulting statistics for the creation of this backup set is shown in **Table 3** below. While the encrypted backup took no less time to create using HyperBac in this scenario, it did actually result in a smaller backup set “footprint” as well:

Table 3. HyperBac Results: RMAN Backup Set Encryption				
	Actual Size	RMAN Backup Set Copy	HyperBac Image Copy	Relative Savings
Total File Sizes:	865.00	597.50	133.69	77.6%
<i>All file sizes are in MB unless otherwise noted.</i>				

Encrypting RMAN Image Copy Backups with HyperBac

HyperBac can also simultaneously *compress and encrypt* RMAN image copy backups while they are created during an image copy backup operation. To illustrate, I used the RMAN backup script shown in [Listing 8](#) to create an image copy backup of all datafiles in my target Oracle database. Again, the only modification to my original image copy backup script was a **FORMAT** directive that names each image copy backup file so that its extension is **.hbe**. The resulting statistics for backing up the database’s datafiles are shown in **Table 4** below. Note the significant reduction in size of the backup files, both individually and cumulatively:

Table 4. HyperBac Results: RMAN Image Copy Backup Encryption			
Datafile Name	RMAN Image Copy	HyperBac Image Copy	Relative Savings
SYSTEM01.DBF	480.00	92.59	80.7%
UNDOTBS01.DBF	40.00	9.86	75.3%
SYSAUX01.DBF	240.00	17.13	92.9%
USERS01.DBF	5.00	0.19	96.2%
EXAMPLE01.DBF	100.00	16.17	83.8%
Total:	865.00	135.94	84.3%
<i>All file sizes are in MB unless otherwise noted.</i>			

To prove that RMAN can still restore and recover database components from a backup set or image copy that was produced with the help of HyperBac, I destroyed the datafile for my database’s EXAMPLE tablespace and then restored and recovered the tablespace. [Listing 9](#) shows the successful results of this test.

Compressing and Encrypting DataPump Exports with HyperBac

The final HyperBac features I'll review involve its ability to create either *compressed* or *compressed and encrypted* DataPump Export dumpsets. While it's possible to create an encrypted DataPump export in Oracle 10gR2, this capability again requires the licensing of Oracle Advanced Security features, while HyperBac requires no additional Oracle licensing to enable its capabilities.

As shown in [Listing 10](#), I created a DataPump Export dump set that comprised the entire contents of the standard "seed" database, and I used the results from this operation as a baseline for comparison to DataPump Export operations I performed in concert with HyperBac.

I then created a compressed DataPump Export dump set and an encrypted DataPump dump set as shown in [Listing 11](#) and [Listing 12](#), respectively. **Table 5** shows the end results: a *75% reduction* in the size of the resulting dump sets, and a 45% reduction in the amount of time it took to create the files:

DataPump Operation	Dump Set Size (MB)	Elapsed Time (seconds)
Baseline DataPump Export Dump Set	70.04	255
DataPump Export Dump Set – Compressed with HyperBac	17.04	140
DataPump Export Dump Set – Encrypted with HyperBac	17.06	124

And to verify that these DataPump dump sets were viable, I re-imported just one table from the SH schema (**SH.PRODUCTS**) into the HR schema (as **HR.PRODUCTS**) by using the **REMAP_SCHEMA** directive during the DataPump Import. As shown in [Listing 13](#) I used the encrypted DataPump Export created in Listing 9 as the target for the DataPump import operation.

Conclusion

HyperBac for Oracle offers excellent solutions for compression and encryption of Oracle RMAN backup sets, RMAN image copy backup files, DataPump Export dump sets, and Export files. It is simple to install, configure and customize for immediate use in concert with any Oracle 10g database. HyperBac for Oracle can be downloaded for a 30-day trial from the HyperBac web site at <http://www.hyperbac.com/>

References and Additional Reading

Even though I've hopefully provided enough technical information in this article to encourage you to explore with these features, I also strongly suggest that you first review the corresponding detailed Oracle documentation before proceeding with any

experiments. Actual implementation of these features should commence only after a crystal-clear understanding exists. The following Oracle 10gR2 documentation is extremely useful for understanding the deeper technical details of this article, especially the backup, restore, and recovery capabilities of Oracle 10gR2 Recovery Manager (RMAN) and the export and import capabilities of Oracle DataPump:

B14191-02 *Oracle Backup and Recovery Advanced User's Guide*

B14192-03 *Oracle Backup and Recovery Basics*

B14193-03 *Oracle Backup and Recovery Quick Start Guide*

B14194-03 *Oracle Backup and Recovery Reference*

B14215-01 *Oracle Utilities*