OpenVMS Technical Journal V7

# Using VMS_CHECK to Collect OpenVMS Configuration Data

Kostas G. Gavrielidis, Master Technologist HP Services

**Overview**

You can choose from several layered products and utilities for collecting operating system configuration and performance data, along with the layered products configuration and performance for the HP OpenVMS operating system. Because OpenVMS runs on three different hardware architectures (VAX, Alpha and Itanium), you have to choose the right tool. This article presents the `VMS_Check` utility, which I developed for collecting OpenVMS configuration information. `VMS_Check` is written entirely in the DIGITAL Command Language (DCL). DCL is similar to any of the UNIX shells, such as the Bourne shell (sh), the C shell (csh), and the Korn shell (ksh); it is a command language interpreter that parses commands and passes control to the programs that make up the OpenVMS operating system. While programs developed on any one of the OpenVMS compilers such as, C/C++, Pascal, BLISS, FORTRAN, COBOL, and so forth, they may not run unchanged or without relinking on all the three architectures; DCL procedures work without changes.

**How VMS_Check Works**

VMS_Check functions like the `sys_check` and `cfg2html` tools, which run on UNIX systems:

- The `sys_check` tool provides configuration and analysis of information gathered on the system. It is useful for debugging or diagnosing system problems. The `sys_check` tool gathers information on over 60 components and subsystems, and performs over 200 analysis operations. It gathers this information into easy to browse and transportable files. These files are sent to support engineering when escalating IPMT cases. It runs on all supported version of the Tru64 UNIX operating system and is included in the Tru64 UNIX operating system and the patch kits.
- The `cfg2html` tool is a UNIX shell script that creates system documentation for HP-UX 10+11, AIX, SCO-UX, SunOS and Linux systems in HTML and ASCII formats. Plugins for SAP, Oracle, Informix, MC/SG, FibreChannel, TIP/ix, Mass Storage like XP48/128/256/512/1024/12000,

EVA3000/EVA5000, Network Node Manager, and OmniBack/DataProtector, and so forth, are included.

VMS_Check is a DCL procedure that runs on all the three OpenVMS architectures and is extendable -- you can include in it any series of OpenVMS commands as if you were entering them at the OpenVMS operating system command prompt (the $). The current version of the VMS_Check tool collects data from any system, standalone or in an OpenVMS Cluster, and presents it in both its original form and with HTML wrappers. The main report is an HTML file named of VMS_Check-<*nodename*>-<*ddmonyy-hhmm*>.HTML. For example:

```
VMS_Check-OWL-14MAR2005-1516.HTML
```

This main file is supported by several text and HTML files, which contain the actual data that make up the complete system report.

### The Purpose of VMS_Check

The primary goal in devoloping the VMS_Check tool was to collect the data on a customer's configuration. It started out as a small procedure with the goal to collect database related configuration information. Slowly it grew to a large DCL command procedure that now includes operating system and storage configuration information.

All of the VMS_Check report sections include information in tables or plain text which can easily be used elsewhere, such as in any of the Microsoft tools Word, Excel, etc.

**Table 1** shows an example of a table generated on an OpenVMS Cluster system, including information about each node, its version, node name, current date and time, and system uptime.

| BBCX Cluster Nodes Table | | | |
|---|---|---|---|
| **OpenVMS Version** | **Node Name** | **Current Date and Time** | **Uptime** |
| OpenVMS V7.3-2 | BBC200 | 25-FEB-2005 11:40:47.70 | 27 10:33:44 |
| OpenVMS V7.3-2 | BBC202 | 25-FEB-2005 11:40:47.73 | 27 09:51:18 |
| OpenVMS V7.3-2 | BBC204 | 25-FEB-2005 11:40:47.78 | 9 12:45:43 |
| ... | ... | ... | ... |
| OpenVMS V7.3-2 | BBC309 | 25-FEB-2005 11:40:48.12 | 19 12:44:58 |
| OpenVMS V7.3-2 | BBC311 | 25-FEB-2005 11:40:48.15 | 2 12:05:09 |

**Table 1 Cluster Nodes Table**

### Data Collected by VMS_Check

The VMS_Check tool collects setup and configuration information for databases and associated layered products, such as Oracle, Rdb, ACMS, Ingres, and so forth, on OpenVMS platforms.

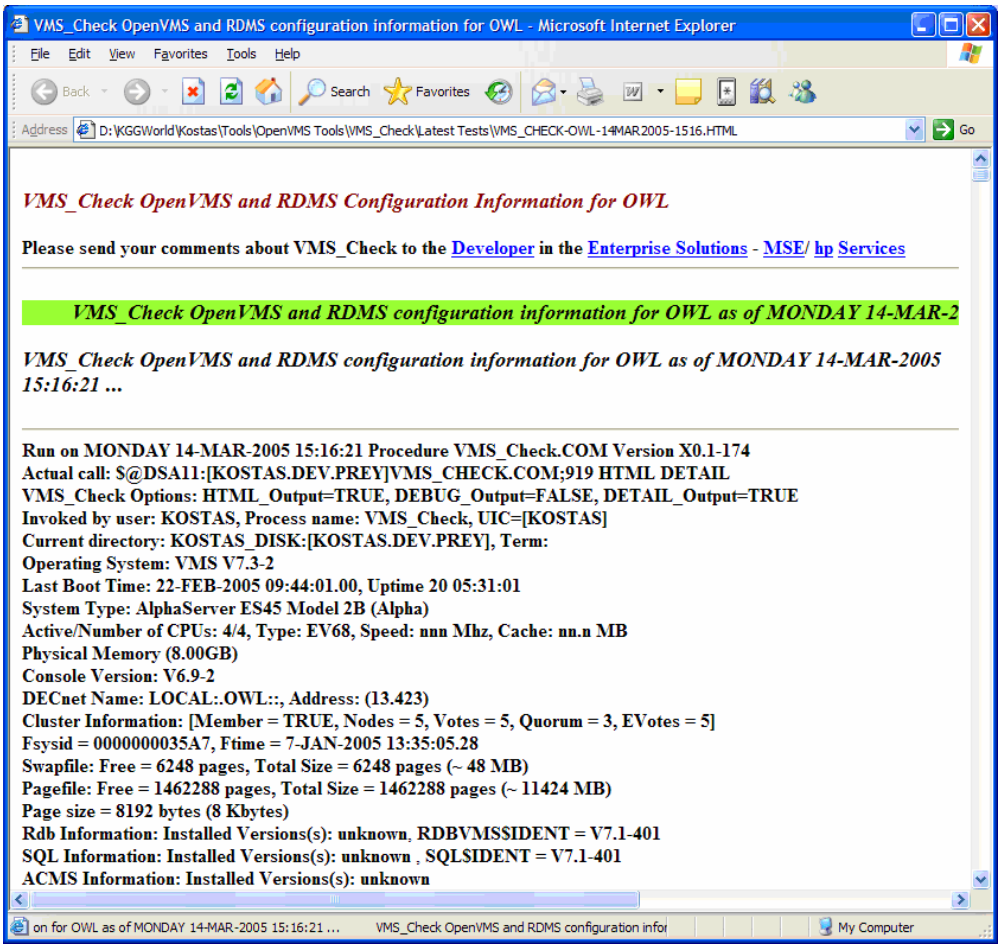


**Figure 1- VMS_Check Report in the MS IE Browser**

### Sections of VMS_Check Reports

As in most generated HTML reports, VMS_Check creates a table of contents at the beginning of the report. The following example shows the table of contents generated by the VMS_Check tool:

**Table of Contents**



**Goto:** Top|Contents|Bottom

### Navigating the VMS_Check Report

You can jump to different sections of interest in the report from inside the main HTML report file, and under each section of the report. The following menu appears:

> **Goto:** Top | Contents | Bottom
> **Section:** System | Storage | Network | Database | Layered Products | Performance | Interactive

The **Goto** references jump to the Top, Table of Contents and Bottom sections of the report.

The **Section** references jump to the System, Storage, Network, Database, Layered Products, Performance and Interactive sections of the report.

### Internal References to Other Files

The main VMS_Check report file contains internal references to other text and HTML files generated by VMS_Check. These file references are described in **Table 2**. This example was generated for the OpenVMS system named OWL.

| Internal References generated for OWL | |
|---|---|
| **Reference** | **Description** |
| Devices_Mounted-OWL.txt | All devices mounted |
| Devices_DU-OWL.txt | All DU devices |
| Devices_HSJ-OWL.txt | All HSJ devices |
| SDA-of-running-OWL.txt | Analyze System |
| SPL-of-running-OWL.txt | Spinlock Information |
| … | … |
| PSA-Brief-OWL.TXT | Performance Analysis (Brief) |
| PSA-Full-OWL.TXT | Performance Analysis (Full) |
| PSA-Perf-OWL.TXT | Performance Evaluation |
| MON-ALL-SUM-OWL.TXT | Monitor all classes (Summary) |
| MON-ALL-AVE-OWL.TXT | Monitor all classes (Average) |

**Table 2 - Internal References**

### Information about Mounted Devices

Table 3 shows the table of all mounted devices that is generated by VMS_Check in the Storage section of the report. It include information about each device, including the device name, the volume name, the device type, the total blocks, the free blocks, percent of blocks free, fragmentation index, and the fragmentation report.

| Devices on OWL | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Device** | **Volume** | **Type** | **Total blocks** | **Free blocks** | **%Free** | **Frag Index** | **Frag Report** |
| _DSA2: | ALP_SITEB | DGX00 | 68251131 | 24772725 | 36 | 44.1 | frpt |
| _DSA11: | ALPHA_USER | DGX00 | 71112778 | 24899409 | 35 | 5.2 | frpt |
| _DSA12: | DATABASE | DGX00 | 213291762 | 38709535 | 18 | 2.4 | frpt |
| _DSA500: | OWL_PAGE | DKX00 | 35565080 | 20564320 | 57 | 30.2 | frpt |

**Table 3 - Mounted Devices Table**

VMS_Check generates a bar graph for the percent of free disk space and the fragmentation index for all devices. **Figure 2** shows the pie chart graph of the total disk capacity, which includes the total MB's used and free disk space.

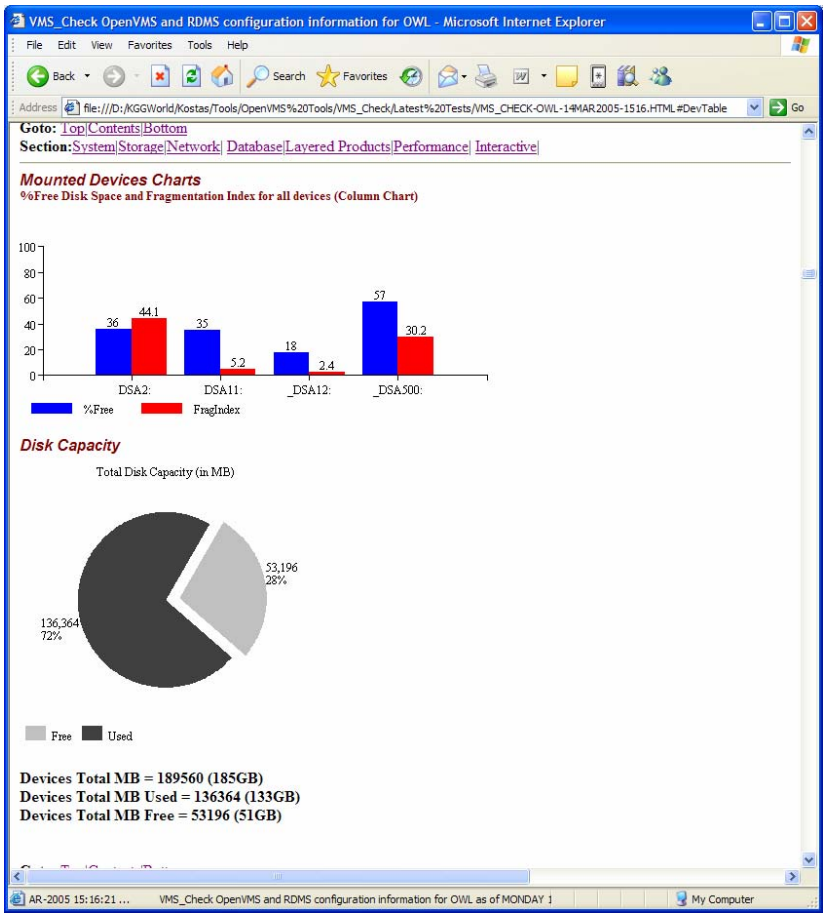


**Figure 2 - Mounted Devices Charts**

### Console Environment Variables

**Table 4** describes some of the information that can be collected from the console.

| Console Variables for OWL | | | |
|---|---|---|---|
| **Name** | **V/N** | **Value** | **Function** |
| auto_action | N | RESTART | Specifies the action the console will take following an error, halt or power-up. Values are: restart, boot and halt |
| auto_fault_restart | | UNDEFINED | Controls whether the SCM will restart when a fault is encountered. |
| Boot_dev | N | SCSI3 0 10 0 3 0 0 0 @wwid0,SCSI3 0 10 0 4 0 0 0 @wwid0,SCSI3 0 8 0 1 0 0 0 @wwid0,SCSI3 0 8 0 2 0 0 0 @wwid0 | Defines the default device or device list from which booting is attempted when no device name is specified by the boot command. |
| Bootdef_dev | N | SCSI3 0 10 0 3 0 0 0 @wwid0,SCSI3 0 10 0 4 0 0 0 @wwid0,SCSI3 0 8 0 1 0 0 0 @wwid0,SCSI3 0 8 0 2 0 0 0 @wwid0 | Defines the default device or device list from which booting is attempted when no device name is specified by the boot command. |
| … | | … | … |

**Table 4 - Console Environment Variables**

### GETSYI Information

The GETSYI informatoin for a single node is shown in **Table 5**.

| GETSYI Information for EMPIRE | | |
|---|---|---|
| **Type** | **Value** | **Description** |
| ACTIVECPU_CNT | 2 | Count of the CPUs actively participating in the current boot of the symmetric multiprocessing (SMP) system. |
| AVAILCPU_CNT | 2 | Number of CPUs available in the current boot of the SMP system. |
| ARCHFLAG | 245760 | Architecture flags for the system |
| ARCH_NAME | Alpha | Name of the CPU architecture on which the process is executing |
| BOOTTIME | 24-SEP-2004 14:11:24.00 | The time when the node was booted. |
| … | | |
| VP_MASK | 0 | Longword mask, the bits of which, when set, indicate which processors in the system have vector coprocessors. |
| VP_NUMBER | 0 | Unsigned longword containing the number of vector processors in the system. |

**Table 5 - GETSYI Information From a Single Node**

GETSYI clusterwide information is shown in **Table 6**

| GETSYI Information for all VMSCluster Nodes | | | | |
|---|---|---|---|---|
| **Item** | **EMPIRE Value** | **AIRTRN Value** | **…** | **STORM Value** |
| CLUSTER_EVOTES | 11 | 11 | | 11 |
| CLUSTER_FSYSID | 0000000034AD | 0000000034AD | | 0000000034AD |
| CLUSTER_MEMBER | TRUE | TRUE | | TRUE |
| CLUSTER_NODES | 11 | 11 | | 11 |
| DECNET_FULLNAME | LOCAL:.EMPIRE:: | LOCAL:.AIRTRN:: | | LOCAL:.STORM:: |
| HW_MODEL | 1976 | 1962 | | 2030 |
| HW_NAME | AlphaServer ES45 Model 2 | AlphaServer DS10L 617 MHz | | hp AlphaServer ES47 7/1000 |
| NODENAME | EMPIRE | AIRTRN | | STORM |
| … | … | … | … | … |
| NODE_SYSTEMID | 0000000035A5 | 0000000034B0 | | 00000000343A |
| NODE_VOTES | 1 | 0 | | 1 |

**Table 6 - Clusterwide GETSYI Information**

### How to Use VMS_Check

You can invoke the VMS_Check tool with the `VMS_Check-detail.COM` command procedure, which is included with the software kit. This command procedure invokes the VMS_Check command procedure by setting the appropriate privileges and flags for VMS_Check to collect configuration information about the RDBMSs and the OpenVMS environment. For example:

```
$ @VMS_Check-Detail
```

This DCL command procedure actually performs the following steps:

```
$! Determine output file name
$!
$ nodename = f$getsyi("nodename")
$ systime = f$edit(''f$time()', "TRIM")
$ date = f$element(0, " ", systime)
$ time = f$element(1, " ", systime)
$ day  = f$element(0, "-", date)
$ mon  = f$element(1, "-", date)
$ year = f$element(2, "-", date)
$ hour = f$element(0, ":", time)
$ min  = f$element(1, ":", time)
$ if f$length(day)  .eq. 1 then day = "0"+day
$ filename = "VMS_Check-"+nodename+"-"+day+mon+year+"-
"+hour+min+".HTML"
$! show symbol filename
$ SET PROC/PRIV=(ALL,NOBYPASS)
$ exec := SPAWN/NOWAIT/INPUT=NL:/OUTPUT='filename'
```

```
$! show symbol exec
$ EXEC @VMS_Check.COM HTML DETAIL
```

To invoke VMS_Check in a single system environment, enter the following commands:

```
$ SET PROC/PRIV=(ALL,NOBYPASS)
$ EXEC:= SPAWN/NOWAIT/INPUT=NL: -
 /PROCESS=VMS_Check -
 /OUTPUT=VMS_Check-'F$getsyi("NODENAME")'.HTML
```

To generate HTML output and to get a DETAIL description of the current environment, enter the following command:

```
$ EXEC @VMS_Check.COM HTML DETAIL
```

To debug VMS_Check, enter the following command:

```
$ EXEC @VMS_Check.COM DEBUG NODETAIL
```

The DEBUG option creates the VMS_Check.DEBUG file in the current directory. This file has a record of all actions performed by VMS_Check preceded by a time stamp for the start of each action.

To generate HTML output from all the nodes on the VMS Cluster, enter the following commands:

```
$ MCR SYSMAN
SYSMAN> SET ENV/CLUSTER
SYSMAN> DO -
_SYSMAN> SPAWN
_SYSMAN> /INPUT=NL:/OUTPUT=VMS_Check_'F$getsyi("NODENAME")'.HTML -
_SYSMAN> /PROCESS=VMS_Check -
_SYSMAN> @DSA110:[KOSTAS.DEV]VMS_Check.COM HTML NODETAIL
SYSMAN> EXIT
```

### Completing the Report Generation

The completion state of the report is at the end of the generated HTML file. Figure 3 shows an example of successful report generation:

```
*------------------------------------------------------------------------*
*
*  VMS_Check has successfully completed.                                 *
*
*------------------------------------------------------------------------*
```

---

*Generated by VMS_Check.COM X0.1-177 on 16-MAY-2005 14:13:31.10*

**Figure 3 - Successful Report Generation**

### How to Review the VMS_Check Report

To review the report and associated generated files, transfer them from the OpenVMS environment to another environment, such as a Windows laptop, Use FTP to transfer all the files generated by VMS_Check, in ASCII mode. For example, you can use the following FTP commands to transfer the VMS_Check files to your laptop for review:

```
ftp> open a.b.c.d.com
Connected to a.b.c.d.com.
220 a.b.c.d.com FTP Server (Version 5.4) Ready.
User (a.b.c.d.com:(none)): kostas
331 Username kostas requires a Password
```

```
Password:
230 User logged in.
ftp> cd [kostas.dev.prey]
250-CWD command successful.
250 New default directory is DSA11:[KOSTAS.DEV.PREY]
ftp> hash
Hash mark printing On  ftp: (2048 bytes/hash mark) .
ftp> prompt
Interactive mode Off .
ftp> mget *
...
ftp> quit
```

## For more information

For informaton about `cfg2html`, go to http://come.to/cfg2html.

For information about `sys_check`, go to: http://h30097.www3.hp.com/sys_check/

To get a copy of the VMS_CHECK tool please download:
http://h71000.www7.hp.com/openvms/journal/v7/vms_check.zip