



Linux driver - Binary rpm/source rpm user guide and known limitation

TABLE OF CONTENTS

1. OVERVIEW	02
2. DRIVER COMPILATION STEPS	02
3. RELEASE CONTENTS	02
4. ERRATA/NOTES AND KNOWN LIMITATIONS	03
5. INSTALL/REMOVE/UPGRADE DRIVER PACKAGE	04
6. STEPS TO GENERATE BINARY RPM FROM SOURCE RPM	05
7. HOW TO USE DKMS SOURCE RPM/DEB FOR UBUNTU	08
8. SOURCE RPM SIGNING	10
9. SIGNED BINARIES FOR RHEL AND SLES	11

1. OVERVIEW

This README covers Avago's MegaRAID Linux driver-specific limitation and information. For any OS Distro specific limitation and information please check with OS Vendor support.

2. DRIVER COMPILATION STEPS

Driver source code is placed inside the released driver package. And Driver source tarball should be with below name:

megaraid_sas-<driver_version>-src.tar.gz

1) Untar driver source tarball:

```
# tar -zxvf megaraid_sas-<driver_version>-src.tar.gz
```

2) Go to driver source directory:

```
# cd megaraid_sas-<driver_version>/
```

3) To compile the driver for distro use the helper script compile "compile.sh" bundled inside source code:

```
# ./compile.sh
```

3. RELEASE CONTENTS

For any queries on supported OS matrix, please refer "OS_Support_list.txt" text file bundled with each release. OS Support list file list out test coverage executed by Avago. Megaraid_sas driver is GPLv2 open-source driver and source code level support is possible for many Linux kernel versions. If you do not find binary level support for your distribution in release contents, please use the source rpm method.

For any distribution or supported kernel version, there can be three possible packages:

- Driver update disk (Available under folder disks-xx)
- Precompiled binary (kmod/kmp/rpms).(Available under folder rpms-xx)
- Source rpm. (Available under folder rpms-xx)

4. ERRATA/NOTES AND KNOWN LIMITATIONS

a. Oracle Linux Installation errata:

Installing Driver during Installing for UEK from CD:

For UEK kernels, only the KMODs RPMs are provided, not the DUDs. The reasoning behind this is the OEL installation is using the native Red Hat kernels, not UEK. UEK kernel RPMs can be installed after the basic installation is completed. Please check with Oracle support team w.r.t UEK kernel installation process, limitation and other technical queries which is more of generic and not related to MR Drivers.

b. RPM install dependency issues:

If driver RPM installation fails with kABI checks dependency failure message, installing RPM package the user will need to use the "--nodeps" switch when installing the binary."

Example: `rpm -ivh --nodeps kmod-megaraid_sas-vxxxxxx_UEK.xxx.rpm`

"If "rpm -ivh throw any dependency warning/error"

RPM uses KMOD packaging dependency data to ensure the dependencies are met before installing the binary RPM.

Red Hat maintains a whitelist of kernel symbols which RPM uses to validate against the KMOD binaries. Some symbols may be in the kernel but not on the whitelist which results in a failed binary RPM install.

User can use the "--nodeps" switch when installing the binary to skip those whitelist symbol checks or any other dependency."

c. Xen Server Driver update note:

Known Issue:

In case of Citrix driver update disk use, where system has multiple MegaRAID controllers (at least one MR controller which is supported by Inbox Citrix<megaraid_sas> driver), user will not see storage behind controller which is not supported by Inbox Driver (normally this was not expected since user expected driver update disk will work well and it will detect all controllers).

Reason for the issue:

Since Xen Server load megaraid_sas driver from the inbox, before it asks for Driver update disk, user can see "No Storage Disk" on some servers as explained above.

Workaround:

If you are booting from CD, when presented with the 'boot:' prompt you want to type 'shell', where you can then rmmmod your driver before continuing with the installation.

5. INSTALL/REMOVE/UPGRADE DRIVER PACKAGE

Below are two commonly used packaging formats by various Linux distributions:

- 1) RPM(RHEL/SLES/Fedora/OEL uses RPM package)
 - 2) DEB(Ubuntu/Debian uses DEB package) in which driver binary support is provided.
- For More information refer below link:

https://en.wikipedia.org/wiki/Deb_file_format

Please note that after install/remove driver package, the system needs to be rebooted to get intended driver loaded or manually remove the module and insert (read man page “rmmmod” and “modprobe” for more info)

Whenever driver package is installed/uninstalled/upgraded, check the output of below command which should always have an updated driver version.

```
# modinfo megaraid_sas
```

Steps for Driver install/remove/upgrade for .rpm package

1. To install driver RPM, type below command:

```
# rpm -ivh <DRIVER_PACKAGE>.rpm
```
2. To uninstall driver RPM:
Type below command to check the installed driver package name:

```
# rpm -qa | grep megaraid
```

- The output will give installed megaraid_sas RPM packages.

Now type below command to uninstall the required RPM package:

```
# rpm -e <package to be uninstalled>
```

3. To upgrade driver RPM, type below command:

```
# rpm -Uvh <DRIVER_PACKAGE>.rpm
```

Steps for Driver install/remove/upgrade for .deb package

1. To install .deb package, execute following command:
`# dpkg -i <DRIVER_PACKAGE>.deb`
2. To verify the status of installed packages then type the following command:
`# dpkg -s megaraid-sas`
3. After installing megaraid_sas driver, type below command to get the currently installed megaraid_sas driver:
`# modinfo megaraid_sas`
4. To use installed DEB driver loaded, restart the machine and type following command to get currently loaded driver version:
`# cat /sys/modules/megaraid_sas/version`
 - This version should be the same as the driver version of the installed driver DEB package.
5. To uninstall megaraid_sas package, type below command:
`# dpkg -r megaraid-sas`
6. Verify "modinfo megaraid_sas" megaraid_sas version should be in-box version.

6. STEPS TO GENERATE BINARY RPM FROM SOURCE RPM

There are three variants of source rpms available in this package:

- a. Source rpm which use kmodtool interface (RHEL based)
<http://rpmfusion.org/Packaging/KernelModules/Kmods2>
- b. Source rpm which use kmp build interface (SLES based)
https://en.opensuse.org/Kernel_Module_Packages
- c. Source rpm which use generic build interface (Create initramfs internally and does not depend upon any external tool)

If a user doesn't know which source rpm is better for their environment, we recommend trying #c (for any other distro other than Redhat/Novell)

A quick search of “rpm -qa | grep kmod” can provide a hint, if kmod tool support is available or not.

To generate binary rpm from source rpm, a user should have a compilation/build environment to create kernel module, utilities to build RPM(e.g. rpmbuild..). E.a system where a user has yum repo configured, use below yum command to install development tools.

yum groupinstall “Development Tools”

E.a on Ubuntu user can try installing below missing components.

```
apt-get install rpm
apt-get install make
apt-get install gcc
apt-get install alien
```

Exact commands to create a build environment would be different across distros so this document is not the right place to cover those details. Please refer OS vendor document if needed how to create build/compilation environment for specific OS distro.

Below are steps to generate binary RPM from source RPM:

1. Install source RPM like below:

```
# rpm -ivvh megaraid_sas-<driver_version>.src.rpm
```

Installing above RPM will copy driver SPEC file to specific location (configured as part of rpm package. This path can be different for each OS distribution.)

To locate SPEC file, check output logs of above source RPM installation (see blue marked gives SPEC file location).

e.g.

```
[root@localhost tmp]# rpm -ivvh megaraid_sas-06.810.00.02-98.src.rpm
D: ===== megaraid_sas-06.810.00.02-98.src.rpm
..
Updating / installing...
 1:megaraid_sas-06.810.00.02-98  ##### [100%]
D: ===== Directories not explicitly included in package:
D:      0 /root/rpmbuild/SOURCES/
D:      1 /root/rpmbuild/SPECS/
D: =====
```

```

D: unknown 100755 1 ( 0, 0) 25
/root/rpmbuild/SOURCES/Module.supported;55a756c8
D: unknown 100644 1 ( 0, 0) 120552 /root/rpmbuild/SOURCES/megaraid_sas-
06.810.00.02.tar.gz;55a756c8
D: unknown 100644 1 ( 0, 0) 6783
/root/rpmbuild/SPECS/megaraid_sas.spec;55a756c8
GZDIO: 17 reads, 127888 total bytes in 0.000585 secs
D: closed db index /var/lib/rpm/Name
D: closed db index /var/lib/rpm/Packages
D: closed db environment /var/lib/rpm

```

2. Go to the directory where driver SPEC file is copied as part of #1. There must be SPEC file.
e.g. megaraid_sas.spec/lsi-megaraid_sas or megaraid_sas.spec(driver SPEC file name could be different for different distros, so check SPEC file with megaraid string in its name). Check spec file name in the #1 command output. It provides the location and spec filename.
3. Build binary RPM from source RPM. Run the following command:
rpmbuild -ba <DRIVER_SPEC_FILE>
4. Binary RPMs will be available if #3 exits without any error. Go to the directory where new binary RPM is generated.

E.a Snippet of the working case:

```

--
Wrote: /root/rpmbuild/SRPMS/megaraid_sas-06.810.00.02-98.src.rpm
Wrote: /root/rpmbuild/RPMS/x86_64/megaraid_sas-06.810.00.02-98.x86_64.rpm
Wrote: /root/rpmbuild/RPMS/x86_64/megaraid_sas-debuginfo-06.810.00.02-98.x86_64.rpm
Executing(%clean): /bin/sh -e /var/tmp/rpm-tmp.ZbHbmH
+ umask 022
+ cd /root/rpmbuild/BUILD
+ cd megaraid_sas-06.810.00.02
+ /usr/bin/rm -rf /root/rpmbuild/BUILDROOT/megaraid_sas-06.810.00.02-98.x86_64
+ exit 0
--

```

Step #5 is applicable only for deb package based OS distribution (Ubuntu/Debian).

5. Create driver .deb package from binary RPM, Execute below command:
alien -k --to-deb --scripts <GENERATED_DRIVER_RPM>

Now, Install generated binary rpms using steps mentioned at “**Install/Remove/Upgrade driver binary package**”

It is always recommended to verify the initramfs image to confirm updated megaraid_sas driver is packaged correctly before the system reboot. Do not assume that the latest driver is loaded after installing rpm generated from source rpm. See OS distribution-specific documentation on how to verify initramfs image. E.a FC21 user guide link for reference.

https://docs.fedoraproject.org/en-US/Fedora/21/html/System_Administrators_Guide/sec-Verifying_the_Initial_RAM_Disk_Image.html#bh-Verifying_the_Initial_RAM_Disk_Image_and_Kernel_on_IBM_eServer_System_i

7. HOW TO USE DKMS SOURCE RPM/DEB FOR UBUNTU

Canonical recommends for Ubuntu, driver support should be provided through DKMS based driver source deb package. We provide single driver source Deb package which should work for all Ubuntu versions supported by us.

We have captured here how to use DKMS enabled driver module.

To install/upgrade DKMS from Ubuntu repo, use below command:

```
# apt-get install dkms
```

In case you get DKMS Deb/RPM package, then install it using below commands:

For Deb package:

```
# dpkg -i dkms-<version>-<release>.noarch.deb
```

For RPM package:

```
# rpm -ivh dkms-<version>-<release>.noarch.rpm
```

This is a prerequisite to installing DKMS-enabled module RPMs.

To install a DKMS enabled module Deb/RPM, you simply install it like any other Deb/RPM:

For Deb package:

```
# dpkg -i <module>-<version>-<rpmversion>.noarch.deb
```

For RPM package:

```
# rpm -ivh <module>-<version>-<rpmversion>.noarch.rpm
```

With a DKMS enabled module Deb/RPM, most of the installation work done by the RPM is handed off to DKMS within the RPM/Deb. Generally, it does the following:

1. Installs module source into /usr/src/<module>-<moduleversion>/
2. Places a dkms.conf file into /usr/src/<module>-<moduleversion>/
3. Runs: # dkms add -m <module> -v <version>
4. Runs: # dkms build -m <module> -v <version>
5. Runs: # dkms install -m <module> -v <version>

Once the RPM/Deb installation is complete, you can use DKMS to understand which module and which module version is installed on which kernels. This can be accomplished via the command:

```
# dkms status
```

From here, you can then use the various DKMS commands (eg. add, build, install, uninstall) to load that module onto other kernels.

For more details, please refer URL: <https://github.com/dell/dkms>

Additional Note on compiling source rpm with retpoline support for Spectre V2 mitigation:

Driver RPMs with retpoline support can be built using the source RPM method described above. Please refer the documentation from respective OS vendors on additional information on Spectre V2 mitigation with retpoline.

1. This would require an updated GCC package with support for retpoline options. Refer the latest GCC version released by OSV that supports retpoline options: “-mindirect-branch” and “-mindirect-branch-register”.

2. The RPM SPEC file provided in the source RPM contains conditional build section that can be enabled by passing “--with retpoline” option to the rpmbuild command mentioned in step #3 above.
Eg: `#rpmbuild -ba --with retpoline <DRIVER_SPEC_FILE>`
3. Passing this flag through rpmbuild command will enable the below additional GCC flags required to compile the driver source with retpoline support:
`KCFLAGS='-mindirect-branch=thunk-inline -mindirect-branch-register'`
4. Driver module compiled this way will expose “retpoline: Y” in modinfo output.

8. SOURCE RPM SIGNING

All Source RPMs are signed/encrypted with GnuPG public-private encryption scheme to ensure the authenticity of the source RPMs. GPG public-private key pair is a 4k bit long with RSA encryption algorithm and all SRPMS are signed with the same. The private key is only held by Broadcom and associated public key is kept on the Broadcom website.

Below is the verification guide link which contains the detailed steps to verify the source RPM and public key link to download the public key file.

Verification guide link:

[VERIFICATION-GUIDE](#)

Public key link:

[PUBLIC-KEY](#)

Note: Verification guide also contains the public key link.

9. SIGNED BINARIES FOR RHEL AND SLES

In addition to the unsigned binaries, Broadcom provides OpenSSL signed binaries for RHEL and SLES OS for UEFI secure boot environment. Binaries are signed with OpenSSL public-private key pair. The private key is only held by Broadcom and the associated public key will be provided as part of the driver release package named DCSG00411462_selfcet.der.

The release package has folders prefixed with signed for signed driver binaries

Ex.

sles15: unsigned sles15 binaries

rhel8_oel8_centos8: unsigned rhel8 binaries

signed_sles15: signed sles15 binaries

signed_rhel8_oel8_centos8: signed rhel8 binaries

Please refer following vendor documentation for key enrollment:

https://drivers.suse.com/doc/Usage/Secure_Boot_Certificate.html#secure-boot-certificate