request@raidix.com
raidix.com

RAIDIX ERA 2.0 ADMINISTRATOR'S GUIDE

2019

BRIEF CONTENTS

INTRODUCTION

Intended Audience

This guide is intended for administrators and users of RAIDs based on the RAIDIX ERA 2.0 software.

The guide contains instructions on how to configure and manage RAIDs in RAIDIX ERA 2.0.

What is RAIDIX ERA 2.0?

RAIDIX ERA 2.0 is high-performance software RAID developed specifically for NVMe storage devices and new types of SAN networks. RAIDIX ERA 2.0 technologies use high potential of Flash devices (NVMe, SAS, SATA) to create a fast fault-tolerant RAID available as a local block device with opportunity of export via network by using auxiliary software. RAIDIX ERA 2.0 is a Linux kernel module and a management utility, which are built and configured for the most popular distributions, namely, Ubuntu 16.04 LTS, Ubuntu 18.04 LTS, CentOS 7.4, 7.5, 7.6. RAIDIX ERA 2.0 is installed on servers with slots for Flash memory devices or with connected JBOFs. RAIDIX ERA 2.0 enables you to combine drives into high-performance fault-tolerant RAIDs.

RAIDIX ERA 2.0 Specifications

Supported RAID levels	RAID 0, 1, 5, 6, and 7.3.
Maximum number of drives in a RAID	64.
Maximum number of drives in the system	Defined by hardware configuration.
Maximum number of RAIDs	128.
Maximum RAID size	Defined by drive sizes.
Space for RAID metadata storage	100 MiB from the beginning of each drive in RAID.

1. ERA RAID MANAGEMENT

Manage your software ERA RAID in Linux by using `eraraid` utility.

1.1 Command Line Interface (CLI) Description

In the console, enter commands in the following format:

```
eraraid {mode} {parameters}
```

The `{mode}` parameter has the following values:

create	Create a RAID.
show	Show information on a RAID.
restore	Restore a RAID.
destroy	Delete a RAID completely.
unload	Unload a RAID.
replace	Replace or delete a drive in a RAID.
recon	Start RAID reconstruction.
init	Start RAID initialization.
modify	Modify RAID parameters.
config	Manage a configuration file.
error-log	Show errors in ERA RAID.
license	License management in ERA RAID.
drive-clean	Clean a drive metadata.
import-show	Show possible RAIDs for import.
import-all	Import all possible RAIDs.
import	Import selected RAID from the config on selected drives.

To show the full list of commands, run:

```
eraraid -h
```

To show the ERA RAID version, run:

```
eraraid -v
```

Console syntax peculiarities:

1. Type the command parameters in one line.
2. The parameter in braces or with no parentheses is mandatory ({mode}).
3. The parameter in brackets is clarifying.
4. Commands parameters are separated by spaces.
5. Use short or long forms of command attributes.

   Example:

   ```
   eraraid create –n (or --name) [input RAID name] –l (or --level) [input RAID level] –d (or --drives) [input drive numbers]
   ```

6. To get the list of all methods and objects of lower levels, add attribute `–h`:

   ```
   eraraid {mode} –h
   ```

Detailed description of methods is presented below.

1.2 License Management

To start working in the system, add a valid license file on each node. To do so, you need the hardware key (hwkey) which can be found by running the command:

```
eraraid license --show
```

Command output example when no license added:

```
hwkey: C6C67AF003871B7C
license_key: null
version: 0
crypto_version: 0
created: 0-0-0
expired: 0-0-0
disks: 0
levels: 0
type: nvme
disks_in_use: 0
status: expired
```

Command output example when a license added:

```
hwkey: C6C67AF003871B7C
license_key: C443B932D9CA66AA550A81862F24E
6D98FB6BB7031CA1071FCED1B38A3593
version: 1
crypto_version: 0
created: 2019-3-27
expired: 2020-12-31
disks: 128
levels: 7
type: nvme
disks_in_use: 0
status: valid
```

Output contents:

hwkey	Hardware key.
license_key	License key.
version	Software version.
crypto_version	Version of crypto-API for the license generator.
created	The date when the license was created.
expired	License expire date.
disks	Maximum number of drives.
levels	Maximum RAID level.
type	Drive type.
disks in use	Number of used drives in the system.
status	License state.

You can save the command output as a text file by running the command:

```
eraraid license --show > license_request.txt
```

To get your license key, sent your hardware key to the RAIDIX support team by e-mail.

After you got your license file, go to the server and apply the license key by running the command:

```
eraraid license --update /root/license.txt
```

To check the applied license, run:

```
eraraid license --show
```

The full parameters command:

```
eraraid license [-h] (--show|--update {PATH}|--reset)
```

Attributes:

-h	--help	Show this help message and exit.
-s	--show	Show license information.
-u	--update	Update license from the specific path.
-r	--reset	Delete current license.

1.3 Create a RAID

To create a RAID, run:

```
eraraid create [-h] -n NAME -l {0,1,5,6,7} -ss
{16,32,64,128,256} -d
```

Required attributes:

-n	--name	RAID name.
-l	--level	RAID level: **0**, **1**, **5**, **6**, and **7**.
-d	--drives	Space-separated list of block devices.

Additional attributes[1]:

-ss	--strip-size	Strip size in KiB. Possible values: **16**, **32**, **64**, **128**, or **256**. The default is **16**.

[1] See Set up additional parameters. The default parameters.

-ip	--init-prio	Initialization priority. Possible values: from 0 to 100%. The default is 100.
-rp	--recon-prio	Reconstruction priority. Possible values: from 0 to 100%. The default is 100.
-me	--merge-enabled	Activate (1) or deactivate (0) the Merge parameter. The default is 0.
-se	--sched-enabled	Activate (1) or deactivate (0) the Scheduling parameter. The default is 0.
-ml	--memory-limit	RAM usage limit in MiB. The default is unlimited.
-rl	--request-limit	Activate (1) or deactivate (0) limitation of requests per RAID. The default is 0.

Example: Creation of a RAID5 named era5 consisting of 4 NVMe drives — nvme0n1, nvme1n1, nvme2n1, nvme3n1 with stripe size equal to 64 KiB.

```
eraraid create -n era5 -s 64 -l 5 -d /dev/nvme0n1
/dev/nvme1n1 /dev/nvme2n1 /dev/nvme3n1
```

To create a RAID of levels 5, 6 or 7, you need at least 4 drives; RAID0 requires at least 1 drive; RAID1 requires at least 2 drives.

RAM is limited from 1024 MiB to maximum system capacity.

Warning: Creating ERA RAID over ERA RAID devices is not recommended.

1.4 Show RAID State

To see information on RAID state, run:

```
eraraid show [-h] [-n NAME | -o] [-u {s,k,m,g}] [-f
{table,json,prettyjson}]
```

Attributes:

-h	--help	Show the help message and exit.
-n	--name	RAID name.
-o	--online	Show only assembled RAIDs.

-u	--units	Size units: **s** – sectors, **k** – kilobyte, **m** – megabyte, **g** – gigabyte.
-f	--format	Output format: • table – as a table; • json – json; • prettyjson json – human readable format.
-e	--extended	Show extended table

Example: Showing information on RAID **era5**:

```
eraraid show
```

```
RAIDs
 name | static                | state   | devices                    | param
------+-----------------------+---------+----------------------------+-----------------------
 era5 | size: 2682 GiB        | online  | 0 /dev/nvme0n1 online      | init_prio     : 100
      | level: 5              | initing | 1 /dev/nvme1n1 online      | init_progress : 0
      | stripe_size: 64       |         | 2 /dev/nvme2n1 online      | recon_prio    : 100
      | active: True          |         | 3 /dev/nvme3n1 online      | recon_progress: 0
      | config: True          |         |                            |
```

Output contents:

name	RAID name.

static	Static RAID parameters:
	• Size.
	• Level.
	• Stripe size.
	• Active:
	  ○ **True**, if the RAID block device is in the system.
	  ○ **False**, if:
	  1. The RAID is not loaded after reboot. Restore the RAID
	  2. The RAID is unloaded. Restore the RAID.
	  3. The raidix_nvme.ko driver is missing or not loaded; rpm package is missing.
	• Config:
	  ○ **True**, if the configuration file is in the system.

	○ False, if the file is missing. In this case, restore the config from the drives.
state	RAID state: • Online – the RAID is available and ready for work; • Initing – the RAID is initializing; • Degraded – the RAID is available and ready for work but some drives are missing or failed; • Reconstructing – the RAID is in the process of reconstruction; • Offline – the RAID is unavailable; • Need_recon – the RAID needs reconstruction; • Need_init – the RAID need initialization; • Read Only – the license is expired. The RAID is read-only.
devices	The list of devices included in the RAID and their current stated: • Online – the drive is active; • Offline – the drive is missing or unavailable; • Reconstructing – the drive is in process of reconstruction; • Need_recon – the drive needs reconstruction.
info	Modifiable RAID values: • init_progress (only for RAID levels 5, 6, and 7.3) – initialization progress. Possible values: from 0% to 100%. • recon_progress (only for RAID levels 1, 5, 6, and 7.3) – reconstruction progress. Possible values: from 0% to 100%. • memory_usage_mb – amount of usage RAM, if memory_limit=0 (is not limited) than memory_usage_mb is not displayed.
serials	Serial numbers of drives in RAID (displayed in extended version)
param	Dynamic RAID parameters: • init_prio – initialization priority: from 0% to 100%; • init_progress – initialization progress: from 0% to 100%; • recon_prio – reconstruction priority: from 0% to 100%; • recon_progress – reconstruction progress: from 0% to 100%.

1.5 Restore a RAID

RAIDs restores automatically after any failure. If any problems arise, restore a RAID manually by running the command:

```
eraraid restore [-h] (-n NAME | -a)
```

Attributes:

-h	--help	Show the help message and exit.
-n	--name	RAID name.
-a	--all	Restore all RAIDs.

Example: Restoring the RAID **era5**:

```
eraraid restore –n era5
```

1.6 Delete a RAID

To delete a RAID completely, run:

```
eraraid destroy [-h] (-n NAME | -a)
```

Attributes:

-h	--help	Show the help message and exit.
-n	--name	RAID name.
-a	--all	Destroy all RAIDs.

Example: Deleting the RAID **era5**:

```
eraraid destroy –n era5
```

1.7 Unload a RAID

Unloading (or deactivation) is a deletion of a RAID while keeping the configuration file. Unlike complete deletion, unloading enables you to restore the RAID later. To unload a RAID, run:

```
eraraid unload [-h] (-n NAME | -a)
```

Attributes:

-h	--help	Show the help message and exit.
-n	--name	RAID name.

-a	--all	Unload all RAIDs.

Example: Unloading the RAID era5:

```
eraraid unload –n era5
```

1.8 Replace or Exclude a Drive

To exclude or replace a drive in a RAID, run:

```
eraraid replace [-h] –n NAME –no NUMBER –d DRIVE
```

Attributes:

-h	--help	Show the help message and exit.
-n	--name	RAID name.
-no	--number	Drive number in the RAID. To see the drive number, use the command `eraraid show`.
-d	--drive	New drive. To exclude the drive or mark is as missing, set the path to the drive as "null".

Example: Replacing the drive 0 with the drive nvme4n1 in the RAID era5:

- Mark the drive 0 as missing:

```
eraraid replace –n era5 –no 0 –d null
```

- Replace the drive 0 with the drive nvme4n1:

```
eraraid replace –n era5 –no 0 –d /dev/nvme4n1
```

1.9 RAID Reconstruction

Reconstruction starts after replacing drives automatically for RAID levels 1, 5, 6, and 7.3. For manual start of reconstruction process, run:

```
eraraid recon [-h] –n NAME (--start | --stop)
```

Attributes:

-h	--help	Show the help message and exit.
-n	--name	RAID name.
	--start	Start reconstruction.
	--stop	Stop reconstruction.

Example: Starting RAID era5 reconstruction:

```
eraraid recon --start -n era5
```

To improve the system performance under the load, try decreasing reconstruction priority by changing the corresponding RAID parameter.

1.10 RAID Initialization

Initialization starts automatically after a RAID is created. To manage the process of initialization, run:

```
eraraid init [-h] -n NAME (--start | --stop)
```

Attributes:

-h	--help	Show the help message and exit.
-n	--name	RAID name.
	--start	Start initialization.
	--stop	Stop initialization.

Example: Starting initialization of the RAID era5:

```
eraraid init --start -n era5
```

To improve the system performance under the load, try decreasing initialization priority by changing the corresponding RAID parameter. The random write performance is higher on an initialized RAID.

1.11 Change RAID Parameters

To improve the system performance under the load, try decreasing initialization and reconstruction priorities. If the priority is equal to 0, reconstruction or initialization starts and continues only if there is no load. By default, both priorities are set to 100%, which stands for the highest possible rate of reconstruction (initialization) process. To increase user throughput-performance, reduce reconstruction and initialization priority. The `--merge-enabled` parameter reduces the amount of *read-modify-write*, increasing write performance, which is recommended for small blocks and intensive input stream. `--sched-enabled` parameter optimizes RAID for low-flow loads.

To change the RAID dynamic parameters, run:

```
eraraid modify [-h] -n NAME [-ip [0..100]] [-rp [0..100]]
[-me {0,1}] [-se {0,1}] [-ml MEMORY_LIMIT] [-rl {0,1}]
```

Attributes:

-h	--help	Show the help message and exit.
-n	--name	RAID name.
-ip	--init-prio	Initialization priority: from 0 to 100% (maximum rate of reconstruction).
-rp	--recon-prio	Reconstruction priority: from 0 to 100% (maximum rate of reconstruction).
-me	--merge-enabled	Activate (1) or deactivate (0) merge.
-se	--sched-enabled	Activate (1) or deactivate (0) scheduling.
-ml	--memory-limit	RAM usage limit in MiB.
-rl	--request-limit	Activate (1) or deactivate (0) requests limitation per RAID.

Example: Setting reconstruction priority for the RAID era5 equal to 50%:

```
eraraid modify -n era5 -rp 50
```

1.12 Config Management

To archive or restore (in case of its loss or damage) the configuration file by using metadata on the drives, run:

```
eraraid config [-h] (-r [RESTORE] | -a | -b | -d | -p)
```

Attributes:

-h	--help	Show the help message and exit.
-r	--restore	Restore the configuration from RESTORE file (default from backup).
-a	--apply	Apply the config by unloading and restoring all RAIDs.
-b	--backup	Copy the current config into the BACKUP file.
-d	--drives	Get the config from drives and restore it to the **/etc/eraraid.conf.drive** file.
-p	--print	Print the config from drives.

Example: Restoring the configuration file from the drives:

- To read configuration files from the drives, run:

```
eraraid config –d
```

- To restore the configuration file, run:

```
eraraid config -r /etc/eraraid.conf.drive
```

1.13 Delete a Drive Metadata

To remove metadata from the drives, run:

```
eraraid drive-clean [-h] -d DRIVE [DRIVE ...]
```

Attributes:

-h	--help	Show the help message and exit.
-d	--drives	List of drives to clean metadata.

1.14 Import RAID Operations

RAIDIX ERA 2.0 provides users the opportunity to import RAIDs that are presented in drives metadata but are not presented in the system configuration file.

Use this feature in cases when you want to unite several ERA RAID systems into one.

RAIDs with identical names cannot exist in one system. Therefore, RAID import can require renaming in case of identical names. If there is drive conflict with the RAID system, RAID imports in **degraded** mode without the conflicting drive. In case of drive conflict with imported RAID, conflict drive places in the RAID that was imported first.

Possible conflicts:

- name: Conflict with system raids;
- drives: Conflict with system raids;
- name: Conflict with import raids;
- drives: Conflict with import raids;
- name: Conflict with system and import raids;
- drives: Conflict with system and import raids.

Import-related commands:

```
eraraid import-show [-h] [-f {table,json,prettyjson}] [-d
DRIVES [DRIVES ...]]
```

Attributes:

-h	--help	Show the help message and exit.
-f	--format	Output format: • table – table format; • json – json format; • prettyjson – human-readable json.
-d	--drives	Show imported RAID from the drives.

```
eraraid import-all [-h] [-d DRIVES [DRIVES ...]]
```

Attributes:

-h	--help	Show the help message and exit.
-d	--drives	Import RAIDs from specified drives.

```
eraraid import [-h] -id UUID [-n NAME] [-d DRIVES [DRIVES
...]]
```

Attributes:

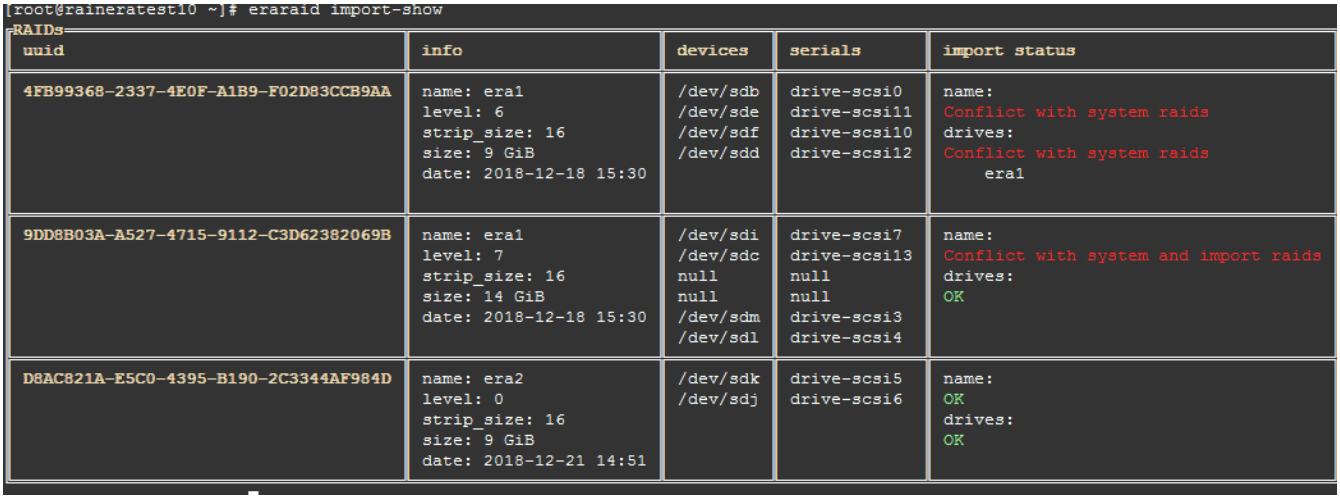
-h	--help	Show the help message and exit.
-id	--uuid	RAID UUID.
-n	--names	New RAID name.
-d	--drives	Import RAIDs from specified drives.

Example:

1. To execute the import-show command, run:

```
eraraid import-show
```

Utility will find and display information about founded RAIDs on drives that can be imported. On the figure below are three import available RAIDs:



Conflicts are highlighted in red color. The first RAID has a conflict with system RAIDs (present in the system) by name and drives. The second RAID conflicts with system and import RAIDs by name. The third RAID does not conflict with any RAIDs. To get information about system RAIDs, run:

```
eraraid show
```

```
[root@raineratest10 ~]# eraraid show
RAIDs
```

name	static	state	devices	info
era1	size: 4 GiB level: 1 strip_size: 16 active: True config: True	online	0 /dev/sdb online 1 /dev/sdn online 2 /dev/sdo online	recon_progress : 0 memory_usage_mb: -

2. To import all RAIDs without conflicts, run:

```
eraraid import-all
```

```
RAIDs
```

name	static	state	devices	info
era1	size: 4 GiB level: 1 strip_size: 16 active: True config: True	online	0 /dev/sdb online 1 /dev/sdn online 2 /dev/sdo online	recon_progress : 0 memory_usage_mb: -
era2	size: 9 GiB level: 0 strip_size: 16 active: True config: True	online	0 /dev/sdk online 1 /dev/sdj online	

3. To import the RAID with name and drives conflicts, run:

```
eraraid import -id 4FB99368-2337-4E0F-A1B9-F02D83CCB9AA -n
era3
```

The conflicting drive was replaced to a null drive. To get the RAID out of **degraded** mode, replace the null drive.

```
[root@raineratest10 ~]# eraraid show
RAIDs
```

name	static	state	devices	info
era1	size: 4 GiB level: 1 strip_size: 16 active: True config: True	online	0 /dev/sdb online 1 /dev/sdn online 2 /dev/sdo online	recon_progress : 0 memory_usage_mb: -
era2	size: 9 GiB level: 0 strip_size: 16 active: True config: True	online	0 /dev/sdk online 1 /dev/sdj online	
era3	size: 9 GiB level: 6 strip_size: 16 active: True config: True	online degraded initialized	0 null offline 1 /dev/sde online 2 /dev/sdf online 3 /dev/sdd online	init_progress : 100 recon_progress : 0 memory_usage_mb: 0

1.15Error Messages

To view the error log (default is 10 commands), run:

```
eraraid error-log [-h] [ -n [1..100]]
```

Attributes:

-h	--help	Show the help message and exit.
-n	--lines	Output the last N lines. Possible values are from 1 to 100.

Example: Displaying the last 20 error messages:

```
eraraid error-log -n 20
```

List of all displayed error messages:

1. Initialization errors:
    1.1. "RAID %s cannot be initialized in offline state".
    1.2. "RAID %s is already in init state".
    1.3. "RAID %s doesn't need initialization".

1.4. "RAID %s cannot be initialized: reconstruction is in progress".
1.5. "RAID %s cannot be initialized: need reconstruction".
1.6. "RAID %s cannot be initialized in degraded state".

2. License errors:

2.1. "Wrong hardware key: %08llX. System key: %08ll".
2.2. "Cannot allocate memory for license_key_txt buffer".
2.3. "Cannot allocate memory for license_key_bin buffer".
2.4. "Cannot allocate memory for decrypted_key buffer".
2.5. "Cannot parse license command".
2.6. "Cannot convert license key".
2.7. "Cannot decrypt license key".
2.8. "RAID level %d exceeds license level %d".
2.9. "Total used disks %d exceeds license disks %d".
2.10. "License expired".

3. Common errors:

3.1. "Cannot parse command".
3.2. "Cannot allocate the devices array".
3.3. "Cannot allocate the devices buffer".
3.4. "Cannot create RAID %s without disks".
3.5. "RAID %s with current name already exists".
3.6. "Cannot parse UUID for RAID %s".
3.7. "RAID %s with current UUID %pUB already exists.
3.8. "Cannot create RAID %s on current license".
3.9. "Cannot create RAID %s".
3.10. "Cannot find RAID %s".
3.11. "Too many disks - %d".
3.12. "Cannot parse recon command".
3.13. "Unsupported command".

4. RAID errors:

4.1. "Cannot create offline RAID %s".
4.2. "Wrong device number %d in RAID %s".
4.3. "Cannot open block device %s for replace in RAID %s".

4.4. "New device %s is too small for RAID %s: new size %llu < %llu sectors".

4.5. "Cannot allocate memory for device name in RAID %s".

5. Reconstruction errors:

5.1. "RAID %s cannot be reconstructed in offline state".

5.2. "RAID %s is already in reconstruction state".

5.3. "RAID %s cannot be reconstructed: no devices for".

2. ERA RAID CONFIGURATION RECOMMENDATIONS

2.1 RAID Creation

The appropriate RAID level depends on the required availability level.

Level of availability as high as 99.999% can be achieved by using RAID6 if the RAID consists of less than 20 drives. Use RAID7.3 with more than 20 drives.

The recommended stripe size for the ERA RAID is 16 KiB (set by default).

2.2 RAID Usage

Init-prio и Recon-prio.

The `{modify}` command allows to change initialization and reconstruction priorities. If the priority value of the corresponding function is zero, reconstruction or initialization starts and continues only if there is no workload. By default, both priorities are set to 100% that stands for the highest possible rate of reconstruction or initialization process. To improve the system workload performance, try decreasing initialization and reconstruction priorities.

Scheduling.

There are 2 possible ways of handling an incoming request:

- continue execution on the current CPU;
- transfer the request to the other CPU core and continue execution. Note that it takes time for the transferring.

If the access pattern uses less than half of the system CPU, it is efficient to use the `--sched-enabled` parameter. When a lot of requests are processed by the single CPU core, enabling scheduling allows to redistribute the workload equally between all system CPUs. On multithreading access patterns scheduling is inefficient, because useless transfer of requests from one CPU core to another wastes time.

Recommendation: Enable the `--sched-enabled` parameter when the access pattern is low threaded.

Merge.

The `--merge-enabled` parameter allows to improve the system workload performance when access pattern is sequential and high threaded, and the block sizes are small. This parameter sets a waiting time for all incoming requests in sequential areas. During waiting time, requests

to this area are not intentionally transferred to the drives. Instead of immediate data transfer, incoming requests are formed into a tree structure. At the end of waiting time, requests are merged together if possible. This function reduces the number of *read-modify-write* operations on syndrome RAIDs. Despite the extra waiting time, this function can improve the system workload performance. If the access pattern is mainly random or queue depth is small, the waiting time will not allow merging requests. In this case enabling `--merge-enabled` will decrease the system workload performance.

Recommendation: Use the `--merge-enabled` parameter when the access pattern is sequential and high threaded and the block sizes are small.

Request-limit.

This parameter limits the amount of incoming requests per RAID. For example, writing files on file system without synchronization.

Recommendation: Enable this parameter when you are working with file system and the writing is performed with the `direct` parameter equals to 0 to improve the system workload performance.

2.3 Set up RAID Parameters Recommendations

1. Syndrome RAID creation starts the initialization process automatically. During it, RAID is available for reading and writing operations. Since initialization priority by default is set to 100, you can wait until the initialization is finished, or if the access pattern is not *random write*, you can lower the initialization priority. Therefore, user I/O will be processed faster due to the reduction of initialization requests. If the initialization priority is set to 0, initialization requests are not created during user I/O.
2. The reconstruction process starts automatically. By default, reconstruction priority equals to 100, which means reconstruction has maximum priority among other processes. Setting the priority to 0 allows the user I/O processes running before the reconstruction process.
3. Recommended RAID strip size is 16 KiB.
4. Tuned-adm throughput-performance provides better performance on most of the tests:

```
tuned-adm profile throughput-performance
```

5. Current memory usage is being monitored and controlled to be within the limit. You can modify the `--memory-limit` parameter at any time. By default, memory usage is

unlimited. Deactivating monitoring of current memory usage and limitation control can improve system workload performance. Set `--memory-limit` to 0 to deactivate monitoring with the `modify` command.

NUMA

1. Create a RAID out of drives belonging to the same NUMA node, if your systems are multiprocessor. To figure out the NUMA node drive, run the command:

   ```
   cat /sys/block/nvme0n1/device/device/numa_node
   ```

   or via lspci:

   ```
   lspci -vvv
   ```

2. At creation of NVMeoF target for ERA RAID, you can use network adapter of the same NUMA node as NVMe drives.

Workload

In ERA RAID, user I/O tends to be executed on the same CPU on which the user sent them. However, for some access patterns, you can transfer I/O commands to other CPUs, so the commands will not idle. Starting from RAIDIX ERA 2.0, you can enable I/O scheduling to all system CPU using a parameter `--sched-enabled` (1 – activated, 0 – deactivated).

Activating and deactivating the scheduling parameter depending on the access pattern recommendations are provided below.

Merge

1. If the access pattern is sequential and block sizes are less than *stripe_data_len*, you can activate merge requests with small block size in order to avoid part of *read-modify-write* operations. To activate merge, use the `--merge-enabled` parameter (1 – activated, 0 – deactivated).

2. Deactivate merge when queue depth of user's workload is not enough to merge a full stripe. Activate merge if iodepth * block_size >= stripe_data_len where stripe_data_len = num_data_strips * stripe_size. For example, for RAID 6 out of 6 drives and strip_size =16 KiB, stripe_data_len = 4 * 16K = 64KiB.

NVMeoF

1. ERA RAID allows using NVMeoF devices to create a RAID. Set the `--ctrl-loss-tmo` parameter to 0 to prevent command freezing because of connection loss when using these devices. It is relevant to *nvme-cli* version >= 1.4.

```
nvme connect -t rdma -n nqn.raidix12_1 -a 10.30.0.12 -s
4420 --ctrl-loss-tmo=0
```

2. At the creation of NVMeoF target for ERA RAID, you can enable *merge* if the access pattern assumably will be sequential write. Linux Kernel and Mellanox NVMeoF targets split big requests to 32 KiB + remain. This kind of behavior leads to constant *read-modify-writes*. For a SPDK NVMeoF target, set the `InCapsuleDataSize` parameter denoting at by what value requests should be split.