Get List of Normal Backups

```
GET /virtual_machines/:virtual_machine_id/backups/images.xml
GET /virtual_machines/:virtual_machine_id/backups/images.json
```

XML Request example

```
curl -i -X GET -H 'Accept: application/xml' -H 'Content-type: application/xml' -u user:userpass --url
http://onapp.test/virtual_machines/:virtual_machine_id/backups/images.xml
```

JSON Request example

```
curl -i -X GET -H 'Accept: application/json' -H 'Content-type: application/json' -u user:userpass --url
http://onapp.test/virtual_machines/:virtual_machine_id/backups/images.json
```

An array of backups is returned. If there are no backups, an empty array is returned.

XML Output example

```
<?xml version="1.0" encoding="UTF-8"?>
<backups type="array">
  <backup>
    <allow_resize_without_reboot type="boolean">false</allow_resize_without_reboot>
    <allowed_hot_migrate type="boolean">true</allowed_hot_migrate>
    <allowed_swap type="boolean">true</allowed_swap>
    <backup_server_id type="integer">1</backup_server_id>
    <backup_size type="integer">310896</backup_size>
    <built type="boolean">true</built>
    <built_at type="datetime">2013-12-24T14:34:06+03:00</built_at>
    <created_at type="datetime">2013-12-24T14:31:20+03:00</created_at>
    <data_store_type>lvm</data_store_type>
    <id type="integer">1951</id>
    <identifier>uml64qyvbzv1kb</identifier>
    <image_type nil="true"/>
    <initiated>days</initiated>
    <iqn nil="true"/>
    <locked type="boolean">false</locked>
    <marked_for_delete type="boolean">false</marked_for_delete>
    <min_disk_size type="integer">5</min_disk_size>
    <min_memory_size type="integer">128</min_memory_size>
    <note nil="true"/>
    <operating_system>linux</operating_system>
    <operating_system_distro>ubuntu</operating_system_distro>
    <target_id type="integer">11860</target_id>
    <target_type>Disk</target_type>
    <template_id type="integer">897</template_id>
    <updated_at type="datetime">2013-12-24T14:34:06+03:00</updated_at>
    <user_id type="integer">1875</user_id>
    <volume_id nil="true"/>
    <backup_type>normal</backup_type>
    <disk_id type="integer">11860</disk_id>
  </backup>
</backups>
```

Explanation of the data returned:

allowed_resize_without_reboot – true if resizing CPU & RAM is allowed without restarting the storage server backed up

allowed_hot_migrate – true if hot migration is allowed for the storage server backed up

allowed_swap – true if swap disk is allowed for storage server backed up, otherwise false

backup_server_id – the ID of the backup server on which the backup is stored

backup_size – the disk space taken by this backup in kB

backup_type – normal or incremental

built – true if the storage server backed up has been built

built_at – the date when the disk backup was built

created_at – the date when the record in the database was created

updated_at – the date when this record in database was updated

data_store_type - data store type: lvm, vmware or solidfire

id – the ID of this backup

identifier - disk identifier

image_type - backup type (currently only *tar* is available)

initiated - period when backup is initiated: days, weeks, months, or years

locked – true if the storage server backed up has been locked

marked_for_delete – the backup is marked for deletion (for auto-backups)

min_disk_size – the minimum disk size

operating_system_distro – the OS distribution of the storage server backed up

operating_system – the OS of the storage server backed up

target_id - ID of a backup target

target_type - target for which the backup was taken; For normal backups it is a disk. For incremental backups it's virtual server.

template_id – the ID of a template from which the storage server backed up was built

user_id - the ID of a user the storage server belongs to

volume_id - data store ID

SolidFire - related parameters:

iqn - volume ISCSI qualified name