

CONTENTS

patchVantage API Reference	3
1.1 Introduction	3
1.2 Use Cases.....	4
1.3 Usage	4
1.4 Troubleshooting	4
1.5 Keys	5
1.6 Environments.....	5
1.7 Conflict Management	5
1.8 Sample Output	5
1.9 stop-environments	6
1.10 start-environments	6
1.11 view-jobs	7
1.12 backup-databases	8
1.13 describe-environments	9
1.14 describe-group-headers.....	10
1.15 describe-snap-configurations.....	11
1.16 describe-data-mask-headers	12
1.17 describe-data-mask-headers	13
1.18 describe-server-releases	14
1.19 create-snap-clone	15
1.20 snap-sga-size.....	15
1.21 apply-security-patches	16
1.22 apply-patches	17
1.23 apply-patches-bulk	18
1.24 apply-data-mask.....	19
1.25 sqlserver-version	19

1.26	mysql-version	19
1.27	upgrade-server	20
1.28	send-message	20
1.29	is-patch-applied	21
1.30	get-latest-cpu	21
1.31	get-cpu-list	21
1.32	is-aws-credentials-configured	21
1.33	ec2-instance-id	22
1.34	ec2-instance-type	22
1.35	ec2-instance-vcpu	22
1.36	ec2-instance-memory	23

patchVantage API Reference

Web Services

1.1 Introduction

The patchVantage CPI is a python program that executes all the DBaaS functions in a single line command. It uses web services to connect with the server and completely script complex upgrades and other DBA tasks.

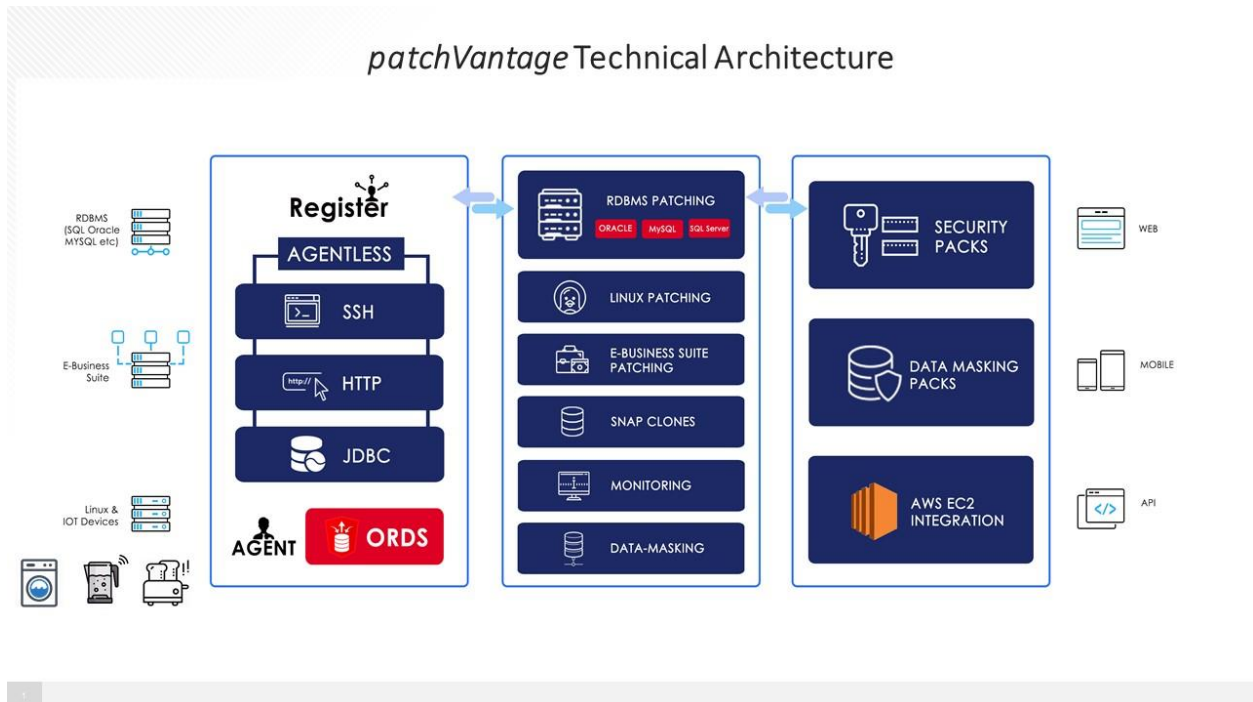






Figure 1 Layered Architecture of Product

1.2 Use Cases

The CLI can be used in multiple ways. One example is to apply the latest Oracle security patches. The API finds the latest Patch Groups, determines whether patches need to be applied ,performs the operations and then sends an exception report. It can be called externally from another program when patching is deemed to be suitable without predetermined scheduling

	Install and start using the API in seconds
	Save time by repeating important functions with scripts or a third-party e.g. Ansible
	Perform heavy duty automation
	All functions available and more

1.3 Usage

Many parameters used in patchVantage are simple string or numeric values, such as Environment in the example below.

pv is-tier –environment ip-172-31-0-45.324.CRM

Optionally, you can optionally separate the parameter name from the value with an equal's sign (=) instead of a space. This is typically necessary only if the value of the parameter starts with a hyphen.

Most output is either text when a single response is expected(in the example above) or json. The API has a return code of 0 when it executes successfully and non-zero otherwise.

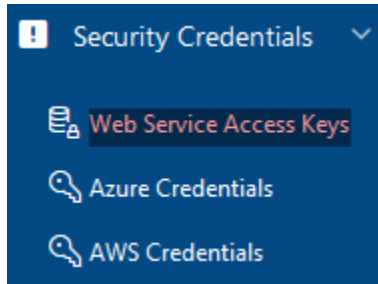
1.4 Troubleshooting

Setting the environment variable export PV_DEBUG=true will product output. If you require a log file, then then export PV_CONSOLE=false will write to a logfile called pv.log

1.5 Keys

The keys can be obtained from logging into the application.

Navigate to Security Credentials->Web Service Access Keys and click [Display](#)



In the same way that AWS applies keys these are set as environment variables :

```
export PV_CLIENT_ID=<string>
```

```
export PV_CLIENT_KEY=<string>
```

1.6 Environments

Most operations are performed against environments . These are of three target types, Server(HOST),Databases(DB) or Application(APPS)

1.7 Conflict Management

Where a job conflicts with another environment the API will abort with an error message

For example, it is not possible to patch the database and backup at the same time

1.8 Sample Output

As described the output is normally json (which can be parsed in bash) or a single text value

Output :

```
{
  "control_environment":[
    {"JOB_NAME":"PV5299"
    ,"DESCRIPTION":"Control RDBMS Environment[REST]"
    ,"ENVIRONMENT":"ip-172-31-0-45.324.CRM"
    ,"STATUS":"RUNNING"}}]
```

1.9 stop-environments

Synopsis : `stop-environments --environment_names [String – wait yes|no`

`--environment-names (string)`

A list of environment names to stopped. They can be Databases or applications

Can be a comma delimited String

`--wait (string)`

Wait for process to finish. The default is **no**

Examples : `pv stop-environments --environment-names \
ip-172-31-0-45.324.CRM --wait no`

json parse : `control_environment`

1.10 start-environments

Synopsis : `start-environments --environment_names [String] – wait yes|no`

`--environment-names (comma delimited string)`

A list of environment names to stopped. They can be Databases or applications

Can be a comma delimited String

`--wait (string)`

Wait for process to finish. The default is **no**

Examples : `pv start-environments --environment-names \
ip-172-31-0-45.324.CRM --wait no`

json parse : `control_environment`

1.11 view-jobs

View details of jobs in process or completed.

Synopsis : `view-jobs --query [Column1=String,Value1=String]`

`-- Columns [Column1=String]`

`-- job_names [String]`

`--query (string)`

List of Column ,Value Tuples

`--columns (string)`

List of Columns

`--job_names`

Comma delimited of Jobs to be viewed

Supported Columns

`job_name status environment description message`

`start_date scheduled_date elapsed_time pv_user`

Examples : `pv view-jobs --job-names [PV5299]`

`pv view-jobs --query [Column=Status,Value=RUNNING]`

`pv view-jobs --job-names [PV5299] --columns [job_name,status]`

json parse : entities

1.12 backup-databases

Synopsis : backup-databases --database-names [String]

--type hot | cold

--stage [Value]

--archive-mode yes | no

--s3 yes | no

--wait yes | no

--database-names(comma delimited string)

Comma delimited list of Environment Names, Oracle with Tier=DB

--Type (string)

Specify hot or cold backup (default=cold)

--stage

Filesystem on server where unique backup will be created

--archive-mode(default=no)

If using hot backup, you can turn on archiving

--s3

Send backup to AWS (default=no)

* AWS Credentials must have been configured

--delay

Delay backup in hours

--wait

Wait for Backup to Finish

json parse : backup_database

Examples : pv backup-databases \

--database-names ip-172-31-0-45.324.CRM \

--type cold \

--stage /u01/db.backup --s3 yes

1.13 describe-environments

View details of Servers, Databases and Applications

Synopsis : describe-environments --query [Column1=String,Value1=String]

-- Columns [Column1=String1]

-- environments-names [String]

--query (string)

List of Column ,Value Tuples

--columns (string)

List of Columns

--environment-names

Comma delimited of Environments to be viewed

Supported Columns

environment tier status description appl_top db_size platform creation date port
hostname container mysql_version sqlserver_version linux_version

Examples : pv describe-environments

pv describe-environments \

--environment-names [ip-172-31-0-45.324.CRM]

pv describe-environments \

--environment-names [ip-172-31-0-45.324.CRM] \

--query [Column=Tier, Value=DB]

json parse : entities

1.14 describe-group-headers

View details of patch groups (e.g. upgrades)

Synopsis : describe-group-headers --query [Column1=String,Value1=String]

-- Columns [Column1=String1]

-- group-names [String]

--query (string)

List of Column ,Value Tuples

--columns (string)

List of Columns

--group-names

Comma delimited of Groups to be viewed

Supported Columns

group_name tier description product version platform sr

Examples : pv describe-group-headers \

--group-names apr_cpu_2019_rdbms_18

pv describe-group-headers \

--group-names apr_cpu_2019_rdbms_18 \

--query [Column=Product, Value=RDBMS]

json parse : entities

1.15 describe-snap-configurations

View details SNAP Clone Setup's

Synopsis : describe-snap-configurations --query [Column1=String,Value1=String]

-- Columns [Column1=String1]

-- snap-names [String]

--query (string)

List of Column ,Value Tuples

--columns (string)

List of Columns

--snap-names

Comma delimited of Setup to be viewed

Supported Columns

configuration_name	host	data	oracle_home	
snap_id	description	applied	stage	data_mask

Examples : pv describe-snap-configurations

pv describe-snap-configurations \

-- snap-names [SNAP_Setup_21] \

--query [Column=Data_Mask,Value=PAYMENT_CONFIDENTIAL]

json parse : entities

1.16 describe-data-mask-headers

View summary information on data masks available

Synopsis : describe-data-mask-headers --query [Column1=String,Value1=String]

--Columns [Column1=String1]

--data-mask-names [String]

--query (string)

List of Column ,Value Tuples

--columns (string)

List of Columns

--data-mask-names

Comma delimited of Environments to be viewed

Supported Columns

policy_name	description	masking_type
-------------	-------------	--------------

Examples : pv describe-data-mask-headers

pv describe-data-mask-headers \

--query [Column=Masking_Type,Value=REDACTION]

json parse : entities

1.17 describe-data-mask-headers

View summary information on data masks available

Synopsis : describe-data-mask-headers --query [Column1=String,Value1=String]

--Columns [Column1=String1]

--data-mask-names [String]

--query (string)

List of Column ,Value Tuples

--columns (string)

List of Columns

--data-mask-names

Comma delimited of Environments to be viewed

Supported Columns

policy_name	description	masking_type
-------------	-------------	--------------

Examples : pv describe-data-mask-headers

pv describe-data-mask-headers \

--query [Column=Masking_Type,Value=REDACTION]

json parse : entities

1.18 describe-server-releases

View information on Linux, MySQL and SQL Server. It can also be used to view supported versions ,including future upgrade versions.

Synopsis : describe-server-releases --environment-names[String]

--release[String] current | supported

--product[String] sqlserver | mysql

--product(string)

Either sqlserver or mysql

--release (string)

Either current or supported

--environment-names

Comma delimited of Environments to be viewed

Examples : pv describe-server-releases \

--environment-names [HOST_EC2_AMAZON_ORACLE_1]

--release supported

--product SQLServer

json parse : entities

1.19 create-snap-clone

Create a copy of an Oracle Database in real-time using a preset Snap Configuration

Synopsis : create-snap-clone --snap-configurations[String]

Examples : create-snap-clone --snap-configurations[SNAP_Setup_21]

json parse : create_snap

1.20 snap-sga-size

Determine the amount of memory used by the snap Clone. This is a prerequisite to creation

Synopsis : snap-sga-size--snap-names[String]

Examples : snap-sga-size --snap-names [SNAP_Setup_21]

json parse : snap_sga_size

1.21 apply-security-patches

This will apply the latest security patches

`--environment-names[String]`

Comma delimited list of Environments to be patched

`--force yes | no`

Force the application of patches , even if they have been applied

`--prereq yes | no`

Option to apply or not apply pre-processing checks(default=no)

`--delay[Number]`

Delay in hours before patching begins

`--control yes | no (default=yes)`

Stop Environment before patch and Start Environment After

Examples : `pv apply-security-patches --environment-names Insurance-RDBMS`

`pv apply-security-patches --environment-names Insurance-RDBMS \
--delay 1`

json parse : `apply_security_patches`

1.22 apply-patches

Apply patches to Oracle Database or Application

`--environment-names[String]`

Comma delimited list of Environments

`--phase apply | rollback (* rollback is not applicable to E-Business Suite)`

`--patches`

Comma delimited list of patch numbers

`--wait yes | no`

Wait for patch to complete

`--force yes | no`

Force application of patch even if its already applied

`--conflict yes | no`

Abort if it conflicts with another job which accesses the environment

`--output html | json`

Output can be HTML file or JSON

Examples : `pv apply-patches --environment-names [ip-172-31-0-45.324.CRM] \`

`--phase apply --patches 27131551 --wait yes \`

`--output json --force no`

json parse : `apply_patches`

1.23 apply-patches-bulk

Apply patches in bulk to synchronize a master database with many others

--gold-environment[String]

The Reference Database

--environment-names[String]

Comma delimited list of Environments to be patched relative to Reference

--patches

RESTRICT the patches to this comma delimited list of patches
(default leave this blank)

--operation apply_ missing | rollback_applied

Apply the missing patches or rollback the applied patches

Examples : pv apply-patches-bulk \

--gold-environment [EC2_EXEC1_DB] \

--environment-names [EC2_DEMO_DB] \

--patches [25095982] \

--operation apply_missing

json parse : apply_patches_bulk

1.24 apply-data-mask

This will apply a pre-configured data mask

`--environment-names[String]`

Comma delimited list of Environments to be patched

`--mask_name[String]`

The name of the Data Mask

Examples : `apply-data-mask --environment-names ip-172-31-0-45.324.CRM`

json parse : `apply_data_mask`

1.25 sqlserver-version

Obtain the current SQL Server version of this host

`--environment[String]`

SQL Server Host as patch target

Examples : `sqlserver-version--environment-name ip-172-31-0-45.324`

json parse : `sqlserver_version`

1.26 mysql-version

Obtain the current MySQL version of this host

`--environment[String]`

MySQL Server Host as patch target

Examples : `mysql-version--environment-name ip-172-31-0-45.324`

json parse : `mysql_version`

1.27 upgrade-server

Upgrade SQL Server , MySQL Servers or Linux servers

--version[String] latest | <SQL Server Version>

--environment-names[String]

Comma delimited list of Environments to be upgraded

--product mysql | sqlserver | linux

Product to be upgraded

--wait yes | no

Wait for patch to complete

Examples : upgrade-server --environment-names [HOST_EBS122A_APPLMGR]

--product mySQL

--version 8.0

json parse : upgrade_server

1.28 send-message

Send a message/notification to a patchVantage user . It will send an email and mobile text message

--user[String]

Existing user on the system

--message[String]

Any quoted string up to 150 characters

Examples : send-message --user dba1 --message "Patch Applied"

1.29 is-patch-applied

Determine if a patch has been applied to an environment – RDBMS or Application

`--environment[String]`

The Database or Application environment

`--patch`

The patch number

Examples : `is-patch-applied --environment ip-172-31-0-45.324.CRM --patch 27131551`

1.30 get-latest-cpu

Obtain the latest security patch group for a Database or Application

`--environment[String]`

The Database or Application environment

Examples : `get-latest-cpu --environment ip-172-31-0-45.324.CRM`

1.31 get-cpu-list

Obtain a list of security patch groups for a Database or Application

`--environment[String]`

The Database or Application environment

Examples : `get-cpu-list --environment ip-172-31-0-45.324.CRM`

json parse : `get_cpu_list`

1.32 is-aws-credentials-configured

Determine whether your AWS credentials are configured. This is necessary for other API's to work

1.33 ec2-instance-id

Obtain the AWS InstanceID for a given Environment.

`--environment[String]`

The Host, Database or Application environment

Examples : `ec2-instance-id --environment ip-172-31-0-45.324.CRM`

1.34 ec2-instance-type

Obtain the AWS Instance Type for a given Environment.

`--environment[String]`

The Host, Database or Application environment

Examples : `ec2-instance-type --environment ip-172-31-0-45.324.CRM`

1.35 ec2-instance-vcpu

Obtain the AWS Instance Virtual CPU's for a given Environment.

`--environment[String]`

The Host, Database or Application environment

Examples : `ec2-instance-vcpu --environment ip-172-31-0-45.324.CRM`

1.36 ec2-instance-memory

Obtain the AWS Instance Memory for a given Environment.

`--environment[String]`

The Host, Database or Application environment

Examples : `ec2-instance-vcpu --environment ip-172-31-0-45.324.CRM`