Automated Malware and Exploit Data Collection Tool Documentation

version 1.0

Software Practicum Team 3

May 14th, 2020

Contents

Welcome to Automated Malware and Exploit Data Collection Tool’s documentation!

src

CeleryApp module

`CeleryApp.`**`createApp`** ()
Creates a Celery app :return: Celery’s app

CeleryWorker module

Entities package

Submodules

Entities.Entity module

class `Entities.Entity.`**`Entity`**
Bases: **`abc.ABC`**

abstract **`dictionary`** **()**
Converts an entity object to a dictionary.

abstract **`objectFromDictionary`** **(**dict**)**
Creates an object from a dictionary. :param dict: Dictionary containing the object information

Entities.MinionConfigFile module

class `Entities.MinionConfigFile.`**`MinionConfigFile`**
Bases: **`object`**

`generateMinionConfigFile` **(**conf_path**,** minion_id**)**
Generates the minion config file. :param conf_path: Path where the minion config file is saved :param minion_id: Minion id string :return: String containing the minion config file

Entities.Response module

class `Entities.Response.`**`Response`** (response='', reason='', status='', task_id='', body='')
Bases: **`Entities.Entity.Entity`**

`dictionary` **()**
Generates a dictionary for the Response object :return: A dictionary with Response’s data

`objectFromDictionary` **(**dict**)**
Creates a Response object from a dictionary. :param dict: A dictionary containing the Response data :return: A Response object

`setBody` **(**body**)**
Sets the body of a request. :param: String containing the request’s body

`setReason` **(**reason**)**

Sets success/fail reason of a request. :param reason: String containing the request's reason

setResponse (response)
Sets the response of a request. :param response: String containing the request's response

setStatus (status)
Set the status of a request. :param status: String containing the request's status

setTaskID (task_id)
Sets the task ID of a request. :param task_id: String containing the request's task id

Entities.VagrantFile module

class Entities.VagrantFile.**VagrantFile**
Bases: **object**

generateVagrantFile (machine, machine_path, minion_id)
Creates a vagrant file for this machine :param machine: Object which carries the virtual machine data :param machine_path: Folder path to this machine :param minion_id: Minion ID assigned to this virtual machine :return: String used to write the vagrant file

Module contents

Managers package

Submodules

Managers.ConfigManager module

class Managers.ConfigManager.**ConfigManager**
Bases: **object**

mongoURL ()
Sets the machine's URL running the MongoDB database. :return: MongoDB's URL string

redisURL ()
Sets the machine's URL running Redis. :return: Redis' URL string

uploadURL ()
Sets the machine's URL running the upload cloud. :return: Upload's URL string

vagrantURL ()
Sets the machine's url running vagrant. :return: Vagrant's URL string

Managers.ConsoleManager module

class Managers.ConsoleManager.**ConsoleManager**
Bases: **object**

printBlue (command)
Prints shell command in blue color. :param command: Shell command to be printed :return: None

printGreen (command)
Prints shell command in green color. :param command: Shell command to be printed :return: None

printRed (command)

Prints shell command in red color. :param command: Shell command to be printed :return: None

runCommandFromShell (command)
Executes a command on the host machines' command console. :param command: Shell command to be executed :return: None

Managers.DatabaseManager module

class Managers.DatabaseManager.**DatabaseManager**
Bases: **object**

deleteExploit (exploit_name)
Deletes a exploit from the database. :param exploit_name: Exploit's name string :return: Deleted document's id

deleteScenario (scenario_name)
Deletes a scenario from the database. :param scenario_name: Scenario's name string :return: Deleted document's id

deleteVulnerability (vulnerability_name)
Deletes a vulnerability from the database. :param vulnerability_name: Vulnerability's name string :return: Deleted document's id

editExploit (exploit_json)
Edits an exploit in the database. :param exploit_json: JSON file containing the exploit's data :return: Modified document's id

editScenario (scenario_json)
Edits a scenario in the database. :param scenario_json: JSON file containing scenario's data :return: Modified document's id

editVulnerability (vulnerability_json)
Edits a vulnerability in the database. :param vulnerability_json: JSON file containing the vulnerability's data :return: Modified document's id

getExploit (exploit_name)
Gets an exploit from the database. :param exploit_name: Exploit's name string :return: A list containing the exploit data

getExploitNames ()
Gets the exploits names from the database. :return: A list containing the exploits' names

getExploits ()
Gets the exploits stored in the database. :return: A list containing the stored exploits' data

getScenario (scenario_name)
Gets a specific scenario from the database. :param scenario_name: Scenario's name string :return: A list containing the scenario retrieved from the database

getScenarioNames ()
Gets the scenario's names. :return: A list containing the scenarios names

getScenarios ()
Gets the scenarios from the databases. :return: A list containing scenarios in the database

getVulnerabilities ()
Gets the vulnerabilities stored in the database. :return: A list containing the stored vulnerabilities' data

getVulnerability (vulnerability_name)

Gets a vulnerability from the database. :param vulnerability_name: Vulnerability's name string :return: A list containing the vulnerability's data

getVulnerabilityNames ()
Gets the vulnerabilities names from the database. :return: A list containing the vulnerability's names

insertExploit (exploit_json**)**
Inserts a scenario in the database. :param exploit_json: JSON file containing the exploit's data :return: Inserted document's id

insertScenario (scenario_json**)**
Inserts a scenario into the database. :param scenario_json: Scenario's JSON file to be inserted :return: Inserted document's id

insertVulnerability (vulnerability_json**)**
Inserts a vulnerability in the database. :param vulnerability_json: JSON file containing the vulnerability's data :return: Inserted document's id

Managers.FileManager module

class Managers.FileManager.**FileManager**
Bases: **object**

createMachineFolders (scenario_json**)**
Creates a folder for each machine in the scenario :param scenario_json: String with the scenario name :return: True if machine folders are created successfully

createSaltFiles (scenario_json**)**
Creates the salt files per each machine in a scenario. :param scenario_json: JSON containing the scenario data :return: Response object containing request info

createSaltStackFolder (scenario_json**)**
Creates a folder for each machine in the scenario :param scenario_json: String with the scenario name :return: True if machine folders are created successfully

createScenarioFolders (scenario_json**)**
Creates a scenario folder with the JSON, Exploit, Vulnerability and Machines subfolders :param scenario_json: String with the scenario name :return: True if the scenario is created successfully

createSharedFolders (scenario_json**)**
Creates the shared folder within a scenario. :param scenario_json: JSON containing the scenario data :return: Response object containing request info

createVagrantFiles (scenario_json**)**
Creates a vagrant file per machine in a scenario. :param scenario_json: JSON containing the scenario data :return: Response object containing request info

deleteScenariosFolder (scenario_name**)**
Deletes not used scenario folders. :param scenario_name: Scenario name as a string :return: None

getCurrentPath ()
test Gets the project folder path :return: String with the project path

getExploitJSONPath (exploit_name**)**
Gets exploit JSON path. :param exploit_name: Exploit name to search :return: Exploit path

getExploitsPath ()
Gets the exploits folder path :return: String with the exploit project path

getScenarioJSONPath (scenario_name)
Gets scenario JSON path inside a machine. :param scenario_name: Scenario name string :return: JSON path folder

getScenariosPath ()
Gets the scenarios folder path :return: String with the scenarios project path

getVulnerabilityJSONPath (vulnerability_name)
Gets vulnerability JSON path. :param vulnerability_name: Vulnerability name to search :return: Vulnerability path

purgeMachines (scenario_name, safe_machines)
Deletes machines that no longer exist within the scenario. :param scenario_name: Scenario name as a string :param safe_machines: Collection containing machines that must remain within the scenario :return: None

Managers.SaltManager module

class Managers.SaltManager.**SaltManager**
Bases: **object**

acceptKeys (minion_id)
Accepts the public keys for a specific minion. :param minion_id: The minion's id :return: None

copyingBeatsConfigFiles (minion_id)
Copies the beats config files into :param minion_id: The minion's id :return: None

generateKeys (keys_path, minion_id)
Generates the keys for a specific minion. :param keys_path: Path where the keys will be stored :param minion_id: The minion's id :return: None

generateMinionConfigFile (conf_path, minion_id)
Generates a minion config file. :param conf_path: Path where the config file will be stored. :param minion_id: The minion's id :return: String with the config minion file data

generateMinionID (machine_name)
Generates a minion id. :param machine_name: Machine's name string :return: A minion's id

runSaltHighstate (minion_id)
Runs the high state on a specific minion. :param minion_id: The minion's id :return: None

testPing (minion_id)
Pings a minion id to check connection. :param minion_id: The minion's id :return: None

Managers.VagrantManager module

class Managers.VagrantManager.**VagrantManager**
Bases: **object**

addBoxByName = <*@task: VagrantManager.addBoxByName of CeleryApp*>

addBoxByOVAFile = <*@task: VagrantManager.addBoxByOVAFile of CeleryApp*>

getAvailableBoxes ()
Gets the available boxes in the Vagrant context :return: A list of string with the available boxes

getSystemInfo ()
Gets the system info. :return: Response object containing request info

removeBoxByName = <*@task: VagrantManager.removeBoxByName of CeleryApp*>

runVagrantUp = <*@task: VagrantManager.runVagrantUp of CeleryApp*>

sendCommand (scenario_name**,** machine_name**,** command**,** default_timeout=5**,** show_output=True**)**
Sends a command to a virtual machine. :param scenario_name: Scenario's name string :param machine_name: Machine's name string :param command: Command to be executed :param default_timeout: Timeout for executing the command :param show_output: Boolean to show an output :return: Response object containing the status of the request

testNetworkPing (scenario_name**,** machine_name**,** destination_machine_name**,** count=1**)**
Tests connection between the host and a virtual machine. :param scenario_name: Scenario's name string :param machine_name: Machine's name string :param destination_machine_name: Machine's to be pinged :param count: Counter used in the ping command :return: Response object containing the status of the request

vagrantMachineCommand (scenario_name**,** machine_name**,** command**)**
Runs the given vagrant command on the desired machine, if allowed. :param scenario_name: Name of scenario containing the machine :param machine_name: String with the machine name :return: Response object containing the status of the request

static **vagrantStatus (**machine_name**,** machine_path**)**
Determines the status of the given vm :param machine_name: String with the machine name :param machine_path: Path to the given machine :return: String representing the status of the machine, False if machine not present

Module contents

VagrantServer module

VagrantServer.**addBoxByName** ()
Adds a box by name inside Vagrant. :return: Response object containing the status of the request

VagrantServer.**addBoxByOVAFile** ()
Adds a box by using an OVA file. :return: Response object containing the status of the request

VagrantServer.**getAvailableBoxes** ()
Gets the available boxes in the Vagrant context. :return: A list of string with the available boxes

VagrantServer.**removeBoxByName** ()
Removes a box by name from Vagrant. :return: Response object containing the status of the request

VagrantServer.**runVagrantUp** (scenario_name)
Executes the vagrant up command for each machine in the scenario. :param scenario_name: String with the scenario name :return: True if the vagrant up commands were successfully executed

VagrantServer.**taskstatus** (task_id)
Requests the status of a Celery task. :param task_id: Task's id to be requested. :return: Response object containing the status of the request

VagrantServer.**testPing** (scenario_name, source, destination)
Tests network connectivity between two virtual machines. :param scenario_name: String with the scenario name :param source: Source virtual machine :param destination: Destination virtual machine :return: Response object containing the status of the request

VagrantServer.**vagrantCommand** (scenario_name, machine_name, command)
Sends a command from the host machine to a virtual machine. :param scenario_name: String with the scenario name :param machine_name: Machine's name :param command: Vagrant command to be executed :return: Response object containing the status of the request

Indices and tables

- **genindex**
- **modindex**
- **search**

Index

S

T

U

V

Python Module Index

c

e

m

v