



Infection Monkey

Information Security Inc.

Contents

- About Infection Monkey
- Architecture
- Requirements
- Testing Setup
- Installing Infection Monkey
- Running Infection Monkey
- References

About Infection Monkey

- The Infection Monkey is an open source security tool for testing a data center's resiliency to perimeter breaches and internal server infection
- The Monkey uses various methods to self propagate across a data center and reports success to a centralized Command and Control(C&C) server



Architecture

- Monkey - A tool which infects other machines and propagates to them
- Monkey Island - A C&C server with a dedicated UI to visualize the Chaos Monkey's progress inside the data center



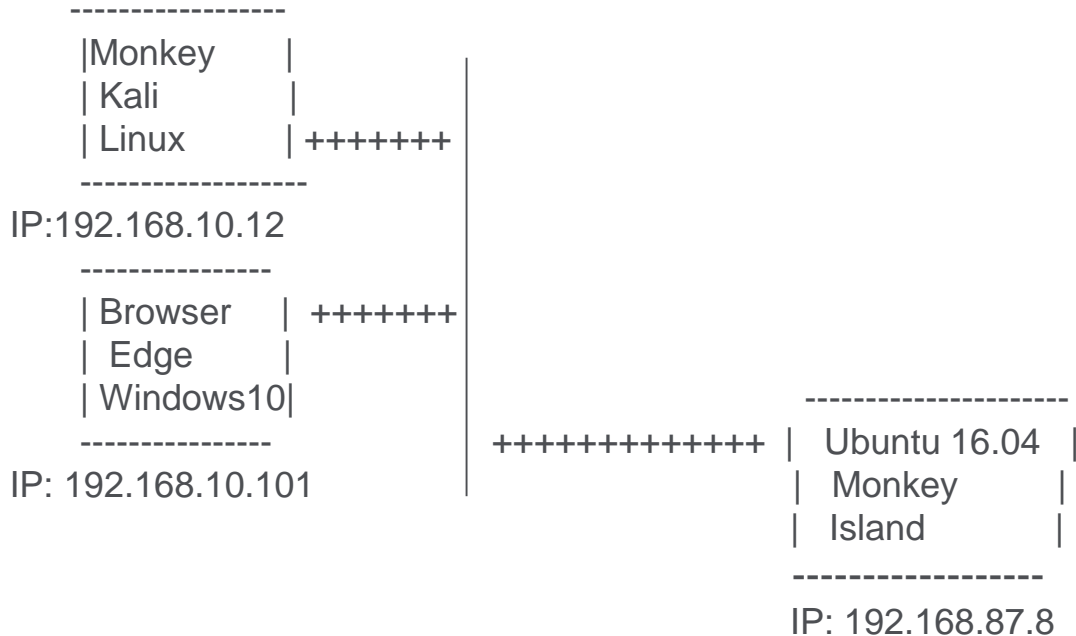
Architecture

- The monkey is composed of three separate parts
- * The Infection Monkey itself - PyInstaller compressed python archives
- * Sambacry binaries - Two linux binaries, 32/64 bit
- * Mimikatz binaries - Two windows binaries, 32/64 bit

Requirements

- The C&C Server has been tested on Ubuntu 14.04,15.04 and 16.04
- The Monkey itself has been tested on Windows XP, 7, 8.1 and 10
- The Linux build has been tested on Ubuntu server (multiple versions)

Testing Setup



Installing Infection Monkey

- Installing the InfectionMonkey on Linux
- Downloading Infection Monkey



GuardiCore

PRODUCT USE CASES PARTNERS COMPANY RESOURCES BLOG SUPPORT LABS

Get a Demo

How Resilient is Your Data Center to Advanced Threats?

With data center breaches in the headlines on a daily basis, ensuring the security and resiliency of your most important assets has never been a higher priority. And the first step in any advanced protection program is to identify potential risk to targeted attacks.

To help security professionals better test and assess potential vulnerabilities in their systems, Guardicore has developed a new open source tool, called the Infection Monkey, that allows pen testers and other security assessment pros to test their data center systems for potential security risks. The Infection Monkey is a free, modern tool that helps verify the effectiveness of security deployments and shed light on the weaker parts of the security chain.

Download Now



**INFECTION
MONKEY**

Installing Infection Monkey

- Installing the InfectionMonkey on Linux
- Unpacking the tarball

```
~/Monkey# tar -xf infection_monkey.tgz
~/Monkey# ls -lah
total 112M
drwxr-xr-x  2 root root  4.0K Feb 12 19:26 .
drwx----- 57 root root  16K Feb 12 19:26 ..
-rwxr-xr-x  1 root root  2.5K Sep  7  2016 example.conf
-rw-r--r--  1 root root  56M Feb 12 19:26 infection_monkey.tgz
-rw-r--r--  1 root root  56M Sep  7  2016 monkey_island.deb
-rwxr-xr-x  1 root root  11K Sep  7  2016 README.md
```

Installing Infection Monkey

- Installing the InfectionMonkey on Linux
- Installing Flask-pymongo

```
:/var# pip install flask-pymongo
Collecting flask-pymongo
  Downloading Flask_PyMongo-0.5.1-py3-none-any.whl
Collecting PyMongo>=2.5 (from flask-pymongo)
  Downloading pymongo-3.6.0-cp35-cp35m-manylinux1_x86_64.whl (378kB)
  100% |#####| 378kB 1.8MB/s
Collecting Flask>=0.8 (from flask-pymongo)
  Downloading Flask-0.12.2-py2.py3-none-any.whl (83kB)
  100% |#####| 92kB 6.1MB/s
Collecting itsdangerous>=0.21 (from Flask>=0.8->flask-pymongo)
  Downloading itsdangerous-0.24.tar.gz (46kB)
  100% |#####| 51kB 5.4MB/s
Collecting click>=2.0 (from Flask>=0.8->flask-pymongo)
  Downloading click-6.7-py2.py3-none-any.whl (71kB)
  100% |#####| 71kB 4.7MB/s
Collecting Jinja2>=2.4 (from Flask>=0.8->flask-pymongo)
  Downloading Jinja2-2.10-py2.py3-none-any.whl (126kB)
  100% |#####| 133kB 3.5MB/s
Requirement already satisfied: Werkzeug>=0.7 in /usr/local/lib/python3.5/dist-packages (from
Requirement already satisfied: MarkupSafe>=0.23 in /usr/lib/python3/dist-packages (from Jinj
ngo)
Building wheels for collected packages: itsdangerous
  Running setup.py bdist_wheel for itsdangerous ... done
  Stored in directory: /root/.cache/pip/wheels/fc/a8/66/24d655233c757e178d45dea2de22a04c6d92
Successfully built itsdangerous
Installing collected packages: PyMongo, itsdangerous, click, Jinja2, Flask, flask-pymongo
Successfully installed Flask-0.12.2 Jinja2-2.10 PyMongo-3.6.0 click-6.7 flask-pymongo-0.5.1
```

Installing Infection Monkey

- Installing the Monkey on Linux
- Installing the Monkey Island

```
    :~# apt-get install -f ./monkey_island.deb
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'gc-monkey-island' instead of './monkey_island.deb'
The following additional packages will be installed:
  libpython-all-dev python-all python-all-dev python-pip python-wheel
The following NEW packages will be installed:
  gc-monkey-island libpython-all-dev python-all python-all-dev python-pip python-wheel
0 upgraded, 6 newly installed, 0 to remove and 0 not upgraded.
Need to get 195 kB/58.8 MB of archives.
After this operation, 864 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 python-pip all 8.1.1-2ubuntu0.4 [144 kB]
Get:2 /root/monkey_island.deb gc-monkey-island amd64 1.0 [58.6 MB]
Get:3 http://us.archive.ubuntu.com/ubuntu xenial/main amd64 libpython-all-dev amd64 2.7.11-1 [992 B]
Get:4 http://us.archive.ubuntu.com/ubuntu xenial/main amd64 python-all amd64 2.7.11-1 [978 B]
Get:5 http://us.archive.ubuntu.com/ubuntu xenial/main amd64 python-all-dev amd64 2.7.11-1 [1,000 B]
Get:6 http://us.archive.ubuntu.com/ubuntu xenial/universe amd64 python-wheel all 0.29.0-1 [40.0 kB]
Fetched 195 kB in 1s (144 kB/s)
Selecting previously unselected package python-pip.
(Reading database ... 233802 files and directories currently installed.)
Preparing to unpack .../python-pip-8.1.1-2ubuntu0.4_all.deb ...
Unpacking python-pip (8.1.1-2ubuntu0.4) ...
Selecting previously unselected package gc-monkey-island.
Preparing to unpack ./root/monkey_island.deb ...
Unpacking gc-monkey-island (1.0) ...
Selecting previously unselected package libpython-all-dev:amd64.
Preparing to unpack .../libpython-all-dev-2.7.11-1_amd64.deb ...
Unpacking libpython-all-dev:amd64 (2.7.11-1) ...
Selecting previously unselected package python-all.
Preparing to unpack .../python-all-2.7.11-1_amd64.deb ...
Unpacking python-all (2.7.11-1) ...
Selecting previously unselected package python-all-dev.
Preparing to unpack .../python-all-dev-2.7.11-1_amd64.deb ...
Unpacking python-all-dev (2.7.11-1) ...
Selecting previously unselected package python-wheel.
Preparing to unpack .../python-wheel-0.29.0-1_all.deb ...
Unpacking python-wheel (0.29.0-1) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up python-pip (8.1.1-2ubuntu0.4) ...
```

Installing Infection Monkey

- Installing the Monkey on Linux
- Verify the Monkey service is functional

```
root@kali:~# systemctl status monkey-  
● 0:1:32m 0m monkey-mongo.service - Monkey Island Mongo Service  
Loaded: loaded (/lib/systemd/system/monkey-mongo.service; disabled; vendor preset: enabled)  
Active: 0:1:32m active (running) 0m since Mon 2019-02-12 21:02:54 MST; 16min ago  
Main PID: 3489 (mongod)  
Tasks: 13  
Memory: 33.5M  
CPU: 4.761s  
CGroup: /system.slice/monkey-mongo.service  
└─3489 /var/monkey_island/bin/mongod --quiet --dbpath /var/monkey_island/db  
Feb 12 21:02:54 XeonPowerful mongod[3489]: 2018-02-12T21:02:54.999-0700 I CONTROL [initandlisten]  
Feb 12 21:02:54 XeonPowerful mongod[3489]: 2018-02-12T21:02:54.999-0700 I CONTROL [initandlisten] ** WARNING: /sys/kernel/mm/transparent_hugepage/defrag is 'always'.  
Feb 12 21:02:54 XeonPowerful mongod[3489]: 2018-02-12T21:02:54.999-0700 I CONTROL [initandlisten] ** We suggest setting it to 'never'.  
Feb 12 21:02:54 XeonPowerful mongod[3489]: 2018-02-12T21:02:54.999-0700 I CONTROL [initandlisten]  
Feb 12 21:02:54 XeonPowerful mongod[3489]: 2018-02-12T21:02:54.999-0700 I CONTROL [initandlisten] db version v3.0.7  
Feb 12 21:02:54 XeonPowerful mongod[3489]: 2018-02-12T21:02:54.999-0700 I CONTROL [initandlisten] git version: 6ce7cbe8c6b899552dad8907604559806aa2c9bd  
Feb 12 21:02:54 XeonPowerful mongod[3489]: 2018-02-12T21:02:54.999-0700 I CONTROL [initandlisten] build info: linux ip-10-183-78-195 3.2.0-4-amd64 #1 SMP Debian 3.2.46-1 x86_64 BOOST_LIB_VERSION 1_49  
Feb 12 21:02:54 XeonPowerful mongod[3489]: 2018-02-12T21:02:54.999-0700 I CONTROL [initandlisten] allocator: tcmalloc  
Feb 12 21:02:54 XeonPowerful mongod[3489]: 2018-02-12T21:02:54.999-0700 I CONTROL [initandlisten] options: { storage: { dbPath: "/var/monkey_island/db"}, systemLog: { quiet: true } }  
Feb 12 21:02:54 XeonPowerful mongod[3489]: 2018-02-12T21:02:54.993-0700 I NETWORK [initandlisten] waiting for connections on port 27017  
  
● 0:1:32m 0m monkey-island.service - Monkey Island Service  
Loaded: loaded (/lib/systemd/system/monkey-island.service; disabled; vendor preset: enabled)  
Active: 0:1:32m active (running) 0m since Mon 2019-02-12 21:02:54 MST; 16min ago  
Main PID: 3445 (start_server.sh)  
Tasks: 4  
Memory: 21.1M  
CPU: 614ms  
CGroup: /system.slice/monkey-island.service  
└─3445 /bin/bash /var/monkey_island/ubuntu/systemd/start_server.sh  
└─3448 python main.py  
Feb 12 21:02:54 XeonPowerful systemd[1]: Started Monkey Island Service.
```

Installing Infection Monkey

- Installing the Monkey on Linux
- Verify the Monkey service is functional

```
root@XeonPowerful:~# netstat -anepl | less
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       User          Inode         PID/Program name
tcp      0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      0             16165         3040/sshd
tcp      0      0 127.0.0.1:631           0.0.0.0:*               LISTEN      0             29603         4975/cupsd
tcp      0      0 0.0.0.0:443            0.0.0.0:*               LISTEN      0             28619         4482/vmware-hostd
tcp      0      0 0.0.0.0:445            0.0.0.0:*               LISTEN      0             28541         4447/smbd
tcp      0      0 0.0.0.0:902            0.0.0.0:*               LISTEN      0             25969         3488/vmware-authdla
tcp      0      0 0.0.0.0:5000           0.0.0.0:*               LISTEN      0             264961        3448/python
tcp      0      0 0.0.0.0:27017          0.0.0.0:*               LISTEN      0             265579        3489/monqod
```

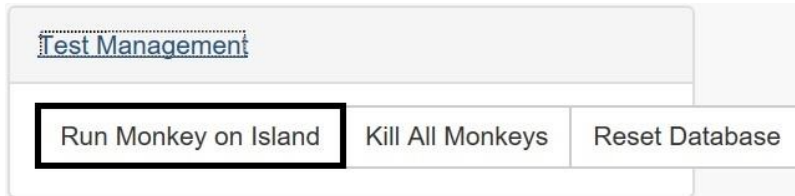
Installing Infection Monkey

- Access the Monkey's Island GUI by browsing to <https://192.168.87.8:5000/admin/index.html>



Running Infection Monkey

- The Monkey can be run in two ways =>
 - 1) With the Monkey Island as the initial attacker. This will use the Monkey Island server as a starting position for the Monkey, from which it will expand based on the configuration



Running Infection Monkey

1) With the Monkey Island as the initial attacker. This will use the Monkey Island server as a starting position for the Monkey, from which it will expand based on the configuration

Run Monkey ✕

This action will run infection monkey on the Island server.
Please choose the IP address to be used as the server for the monkeys:

▼

Run Monkey Cancel

Running Infection Monkey

1) With the Monkey Island as the initial attacker. This will use the Monkey Island server as a starting position for the Monkey, from which it will expand based on the configuration



Run Monkey

Monkey Started!

Running Infection Monkey

1) With the Monkey Island as the initial attacker. This will use the Monkey Island server as a starting position for the Monkey, from which it will expand based on the configuration



Num of Monkeys: 1 (0 exploiting were done)
Monkeys Alive: 1
Num of Hosts Not Exploited: 0
Num of Tunnels Used: 0
Display Scanned Hosts:

Map Legend

Monkey Details

XeonPowerful

Name: XeonPowerful
Description: Linux XeonPowerful 4.4.0-112-generic
#135-Ubuntu SMP Fri Jan 19 11:48:36 UTC 2018
x86_64 x86_64
Internet Access: true
Last Seen: 2018-02-12 22:03:08.040000+00:00
IP Address:

- 192.168.87.8
- 192.168.86.12
- 192.168.65.3
- 172.16.37.1

Exploited by:

- Manual Run

Running Infection Monkey

2) Running the Monkey from a machine elsewhere network

Download the appropriate Monkey executable (Linux/Windows and matching 32/64 bitness) using the following path

[https://192.168.87.8:5000/api/monkey/download/\[binaryToDownload\]](https://192.168.87.8:5000/api/monkey/download/[binaryToDownload])

(monkey-windows-32.exe, monkey-windows-64.exe, monkey-linux-32, monkey-linux-64)

```
└─$ wget --no-check-certificate https://192.168.87.8:5000/api/monkey/download/monkey-linux-64
--2018-02-13 14:55:36-- https://192.168.87.8:5000/api/monkey/download/monkey-linux-64
Connecting to 192.168.87.8:5000... connected.
WARNING: The certificate of '192.168.87.8' is not trusted.
WARNING: The certificate of '192.168.87.8' hasn't got a known issuer.
The certificate's owner does not match hostname '192.168.87.8'
HTTP request sent, awaiting response... 200 OK
Length: 8527160 (8.1M) [application/octet-stream]
Saving to: 'monkey-linux-64'

monkey-linux-64      100%[=====>] 8.13M  11.1MB/s   in 0.7s

2018-02-13 14:55:37 (11.1 MB/s) - 'monkey-linux-64' saved [8527160/8527160]

└─$ file monkey-linux-64
monkey-linux-64: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 2.6.32, BuildID[sha1]=dcb928ff56fd3f1de0f2f74950cd22212132c1, stripped
```

Running Infection Monkey

- Execute the Monkey from the console, passing the server address as a parameter and the magic keyword for execution
- Example “./monkey m0nk3y -s 192.168.87.8:5000”

```
      : ./monkey-linux-64 monkey -s 192.168.87.8:5000
Config file wasn't supplied and default path: /root/.monkey/bin wasn't found, using internal default
Loaded Configuration: {'ms08_067_remote_user_add': 'Monkey USER SUPPORT', 'pocexec_passwords': ['Password!', '1234', 'password', '12345678'], 'pocexec_user': 'Administrator', 'monkey_log_path_linux': '/tmp/user-1563', 'alive': True, 'ssh_passwords': ['Password!', '1234', 'password', '12345678'], 'scanner_classes': ['FpScanner', 'Victims_max_exploit': 7, 'tcp_target_ports': [22, 2222, 445, 135, 3389, 80, 8080, 443, 8080], 'singleton_mutex_name': '{2384ec59-0d1f-4ab9-918c-843740924281}', 'ms08_067_exploit_attempt': 5, 'smb_download_timeout': 300, 'range_size': 1, 'dropper_set_date': True, 'max_iterations': 1, 'kill_file_path_linux': '/var/run/monkey.net', 'serialize_config': False, 'tcp_scan_interval': 200, 'timeout_between_iterations': 100, 'HTTP_PORTS': [80, 8080, 443, 8080], 'depth': 2, 'ms08_067_remote_user_pass': 'Password!', 'skip_exploit_if_file_exists': True, 'self_delete_in_cleanup': False, 'retry_failed_exploitation': True, 'smb_service_name': 'InfectionMonkey', 'range_fixed': ['*'], 'local_network_scan': True, 'dropper_log_path_windows': 'stepp8\\-df1562.tmp', 'dropper_log_path_linux': '/tmp/user-1562', 'ssh_users': ['root', 'user'], 'dropper_target_path': 'C:\\Windows\\Monkey.exe', 'collect_system_info': True, 'finger_classes': ['SSHFinder', 'SSHScanner', 'HTTPFinder'], 'ping_scan_timeout': 1000, 'victim_max_find': 14, 'dropper_try_move_first': False, 'tcp_scan_timeout': 3000, 'pocexec_class': 'FpScanner', 'tcp_scan_get_banner': True, 'dropper_log_path_windows': 'stepp8\\-df1562.tmp', 'monkey_log_path_windows': 'stepp8\\-df1562.tmp', 'command_servers': ['41.50.73.31:5000'], 'monkey_log_path_windows': 'stepp8\\-df1562.tmp', 'tcp_scan_get_banner': True, 'internet_services': ['MonkeyGuardicore.com', 'www.google.com'], 'dropper_data_reference_path': '/bin/sh', 'use_file_logging': True, 'current_server': ''}
2018-02-13 14:58:39,209 [2272:INFO] main.main.IDE: >>>>>>> Initializing monkey (ChaosMonkey): PID 2272 <<<<<<<<<
2018-02-13 14:58:39,210 [2272:INFO] monkey.initialize.39: Monkey is initializing...
2018-02-13 14:58:39,210 [2272:DEBUG] system.singleton.try_lock.54: Global singleton mutex '{2384ec59-0d1f-4ab9-918c-843740924281}' acquired
2018-02-13 14:58:39,211 [2272:DEBUG] monkey.initialize.62: Added default server: 192.168.87.8:5000
2018-02-13 14:58:39,212 [2272:INFO] monkey.start.60: Monkey is running...
2018-02-13 14:58:39,213 [2272:DEBUG] control.wakeup.26: Trying to wake up with C&C servers list: ['192.168.87.8:5000', '41.50.73.31:5000']
PING monkey.guardicore.com (52.49.165.155) 56(84) bytes of data.
64 bytes from ec2-52-49-165-155.eu-west-1.compute.amazonaws.com (52.49.165.155): icmp_seq=1 ttl=36 time=242 ms

--- monkey.guardicore.com ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 242.545/242.545/242.545/0.000 ms
2018-02-13 14:58:39,856 [2272:DEBUG] control.wakeup.54: Trying to connect to server: 192.168.87.8:5000
2018-02-13 14:58:39,864 [2272:INFO] connectionpool.new_conn.135: Starting new HTTPS connection (1): 192.168.87.8
2018-02-13 14:58:39,914 [2272:DEBUG] connectionpool.make_request.383: 'POST /api/monkey HTTP/1.1' 200 99
2018-02-13 14:58:39,917 [2272:INFO] connectionpool.new_conn.135: Starting new HTTPS connection (1): 192.168.87.8
2018-02-13 14:58:39,953 [2272:DEBUG] connectionpool.make_request.383: 'GET /api/monkey/52235053614 HTTP/1.1' 200 2388
2018-02-13 14:58:39,955 [2272:INFO] control.load_control.config.125: New configuration was loaded from server: {'ms08_067_remote_user_add': 'Monkey USER SUPPORT', 'pocexec_passwords': ['Password!', '1234', 'password', '12345678'], 'pocexec_user': 'Administrator', 'monkey_log_path_linux': '/tmp/user-1562', 'alive': True, 'ssh_passwords': ['Password!', '1234', 'password', '12345678'], 'scanner_classes': ['FpScanner', 'Victims_max_exploit': 7, 'tcp_target_ports': [22, 2222, 445, 135, 3389, 80, 8080, 443, 8080], 'singleton_mutex_name': '{2384ec59-0d1f-4ab9-918c-843740924281}', 'ms08_067_exploit_attempt': 5, 'smb_download_timeout': 300, 'range_size': 1, 'dropper_set_date': True, 'max_iterations': 1, 'kill_file_path_linux': '/var/run/monkey.net', 'serialize_config': False}
```

Running Infection Monkey

- Execute the Monkey from the console, passing the server address as a parameter and the magic keyword for execution
- Example “./monkey-linux-64 m0nk3y -s 192.168.87.8:5000”



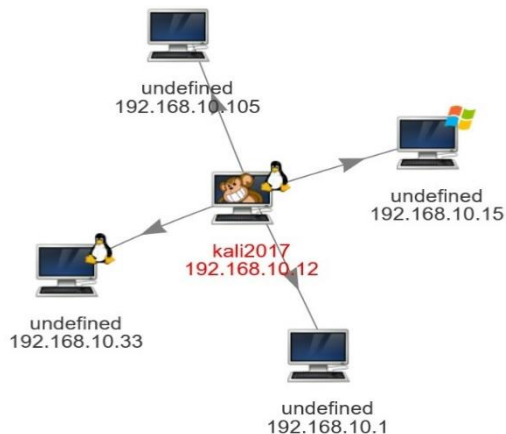
Num of Monkeys: **1** (0 exploiting were done)
Monkeys Alive: **1**
Num of Hosts Not Exploited: **0**
Num of Tunnels Used: **0**
Display Scanned Hosts: ON

Map Legend

Monkey Details

Running Infection Monkey

- 2) Execute the Monkey from the console, passing the server address as a parameter and the magic keyword for execution
- Example “./monkey-linux-64 m0nk3y -s 192.168.87.8:5000”



References

- Official website

<https://www.guardicore.com/infectionmonkey/>

- Github

<https://github.com/guardicore/monkey>