

第3天服务器信息清洗

一、 subprocess 执行本机命令

`subprocess` 的 `getoutput()` 方法可以接收一个字符串的参数，这个参数会被认为是当前操作系统的命令去执行，并返回字符串类型的命令执行结果，假设命令出错，会返回相应的错误信息。

比如我们执行一条命令，来获取的厂商机器信息 目标是得到这些信息

- 厂商 就是 `Manufacturer` 对应的值 比如 Del
- 服务器型号(名字) 就是 `Product Name` 对应的值 比如 Del R710
- 服务器序列号 就是 `Serial Number` 对应的值

```
1 [root@qfedu.com ~]# rpm -q dmidecode
2 dmidecode-3.1-2.el7.x86_64
3 [root@qfedu.com ~]# dmidecode -q -t 1 2>/dev/null
4 System Information
5     Manufacturer: Alibaba Cloud
6     Product Name: Alibaba Cloud ECS
7     Version: pc-i440fx-2.1
8     Serial Number: 0f7e3d86-7742-4612-9f93-
e3a9e4754157
9     UUID: 0f7e3d86-7742-4612-9f93-e3a9e4754157
10    Wake-up Type: Power Switch
11    SKU Number: Not Specified
12    Family: Not Specified
13
```

在 python 中可以这样做

```
1 In [1]: import subprocess
2
3 In [2]: prod_info = "dmidecode -q -t 1 2>/dev/null"
4
5 In [3]: prod_info
6 Out[3]: 'dmidecode -q -t 1 2>/dev/null'
7
8 In [4]: subprocess.getoutput(prod_info)
9 Out[4]: 'System Information\n\tManufacturer: Alibaba
Cloud\n\tProduct Name: Alibaba Cloud ECS\n\tVersion:
pc-i440fx-2.1\n\tSerial Number: 0f7e3d86-7742-4612-
9f93-e3a9e4754157\n\tUUID: 0f7e3d86-7742-4612-9f93-
e3a9e4754157\n\tWake-up Type: Power Switch\n\tSKU
Number: Not Specified\n\tFamily: Not Specified\n'
10
11 In [5]:
```

可以看到输出结果是一个整体的字符串，和 shell 中输出的少有不同，就是这里每一行后面都有一个换行符 '\n' 那我们要想对每一行进行处理，可以使用 `split('\n')` 进行分割，当然我们这里使用另一个方法 `splitlines()`，它默认使用的分隔符就是换行符

```
1 In [5]: ret = subprocess.getoutput(prod_info)
2
3 In [6]: ret.splitlines()
4 Out[6]:
5 ['System Information',
6  '\tManufacturer: Alibaba Cloud',
7  '\tProduct Name: Alibaba Cloud ECS',
8  '\tVersion: pc-i440fx-2.1',
9  '\tSerial Number: 0f7e3d86-7742-4612-9f93-
e3a9e4754157',
```

```

10     '\tUUID: 0f7e3d86-7742-4612-9f93-e3a9e4754157',
11     '\tWake-up Type: Power Switch',
12     '\tSKU Number: Not Specified',
13     '\tFamily: Not Specified']
14
15 In [7]:

```

那接着我们即可以循环列表中的每个元素（也就是每行），在循环中处理每行内容，得到我们想要的数据

```

1 In [7]: for line in ret.splitlines():
2         ...:     print(line)
3         ...:
4 System Information
5     Manufacturer: Alibaba Cloud
6     Product Name: Alibaba Cloud ECS
7     Version: pc-i440fx-2.1
8     Serial Number: 0f7e3d86-7742-4612-9f93-
e3a9e4754157
9     UUID: 0f7e3d86-7742-4612-9f93-e3a9e4754157
10    Wake-up Type: Power Switch
11    SKU Number: Not Specified
12    Family: Not Specified
13
14 In [8]: for line in ret.splitlines():
15         ...:     if 'Manufacturer:' in line:
16         ...:         print(line)
17         ...:
18     Manufacturer: Alibaba Cloud

```

可以看到我们拿到了我们需要的第一个数据，并且可以进行进一步的处理，比如转换成一个字典，其他可以如法炮制。

```

1 In [12]: prod_dic = {}
2         ...: for line in ret.splitlines():

```

```

3      ....:      k = ''
4      ....:      line = line.strip()
5      ....:      print(line)
6      ....:      if ': ' in line:
7      ....:          k, v = line.split(': ')
8      ....:          print(k)
9      ....:          if k == 'Manufacturer':
10     ....:              prod_dic[k.lower()] = v
11     ....:          elif k == 'Product Name':
12     ....:              prod_dic[k.lower()] = v
13     ....:          elif k == 'Serial Number':
14     ....:              prod_dic[k.lower()] = v
15     ....:
16
17 In [13]: prod_dic
18 Out[13]:
19 {'serial_number': '0f7e3d86-7742-4612-9f93-
    e3a9e4754157',
20  'manufacturer': 'Alibaba Cloud',
21  'product_name': 'Alibaba Cloud ECS'}

```

和我们的目标越来越接近了，但你会发现有问题

1. 多个判断语句导致代码臃肿
2. 并且 if 语句中存在重复的代码

那继续优化，思路是可以提前定义一个映射的字典

```

1  #!/usr/bin/env python3
2  import subprocess
3
4  prod_info = 'dmidecode -q -t 1 2>/dev/null'
5
6  # 执行系统命令并返回结果
7  ret = subprocess.getoutput(prod_info)

```

```
8 prod_dic = {}
9
10 # 定义映射字典
11 map_dic = {
12     "Manufacturer": "manufacturer",
13     "Product Name": "pod_name",
14     "Serial Number": "sn"
15 }
16
17 for line in ret.splitlines():
18     line = line.strip()
19     try: # 异常处理语句
20         k, v = line.split(": ")
21         if k in map_dic:
22             # k = map_dic.get(k)
23             prod_dic[map_dic.get(k)] = v
24     except ValueError as e:
25         print(e)
26 # print('....>>>')
27 print(prod_dic)
28 # 输出信息
29 {'manufacturer': 'VMware, Inc.', 'pod_name':
   'VMware7,1', 'sn': 'VMware-56 4d 2b 4b 91 1e 48 15-
   5b d2 73 9c ec 98 da 22'}
30
```

二、获取服务器的硬件基础信息

1. 基础信息

```

1 # 主机名
2 cmd_machine = 'uname -n'
3
4 # 内核版本
5 cmd_kernel = 'uname -r'
6
7 # 操作系统
8 cmd_os_version = "cat /etc/redhat-release

```

2. 厂家和产品信息

```

1 [root@qfedu.com]# dmidecode -q -t 1 2>/dev/null
2 System Information
3     Manufacturer: Alibaba Cloud #
  厂商
4     Product Name: Alibaba Cloud ECS # 机器型
  号
5     Version: pc-i440fx-2.1
6     Serial Number: 0f7e3d86-7742-4612-9f93-
e3a9e4754157
7     UUID: 0f7e3d86-7742-4612-9f93-e3a9e4754157
8     Wake-up Type: Power Switch
9     SKU Number: Not Specified
10    Family: Not Specified

```

3. CPU 信息

3.1 查看物理CPU型号

```

1 grep 'model name' /proc/cpuinfo | uniq

```

```

1 In [1]: import subprocess
2

```

```
3 In [2]: cmd_cpu_name = "grep 'model name'
   /proc/cpuinfo | uniq"
4
5 In [3]: subprocess.getoutput(cmd_cpu_name)
6 Out[3]: 'model name\t: Intel(R) Xeon(R) Platinum
   8163 CPU @ 2.50GHz'
7
8 In [4]: cpu_name =
   subprocess.getoutput(cmd_cpu_name).split(": ")[1]
9
10 In [5]: cpu_name
11 Out[5]: 'Intel(R) Xeon(R) Platinum 8163 CPU @
   2.50GHz'
12
13 In [6]: cpu = {"cpu_name": cpu_name}
14
15 In [7]: cpu
16 Out[7]: {'cpu_name': 'Intel(R) Xeon(R) Platinum 8163
   CPU @ 2.50GHz'}
17
18 In [8]:
```

3.2 查看物理CPU颗数

```
1 | grep 'physical id' /proc/cpuinfo | sort -u | wc -l
```

```

1 In [8]: cmd_cpu_pyc = "grep 'physical id'
   /proc/cpuinfo | sort -u | wc -l"
2
3 In [9]: subprocess.getoutput(cmd_cpu_pyc)
4 Out[9]: '1'
5
6 In [10]: cpu["pyc"] =
   int(subprocess.getoutput(cmd_cpu_pyc))
7
8 In [11]: cpu
9 Out[11]: {'cpu_name': 'Intel(R) Xeon(R) Platinum
   8163 CPU @ 2.50GHz', 'cpu_pyc': 1}
10
11 In [12]:

```

3.3 查看每颗物理 CPU 的核心数

```

1 grep 'cpu cores' /proc/cpuinfo | uniq # 每颗 CPU 的
   核心数，不是总核心数

```

```

1 In [13]: subprocess.getoutput("grep 'cpu cores'
   /proc/cpuinfo | uniq")
2 Out[13]: 'cpu cores\t: 1'
3
4 In [14]: cpu_cores_each = subprocess.getoutput("grep
   'cpu cores' /proc/cpuinfo | uniq")
5
6 In [15]: cpu_cores_each =
   int(cpu_cores_each.split(": ")[1])
7
8 In [16]: cpu_cores_each
9 Out[16]: 1
10
11 In [17]: cpu["cores_each"] = cpu_cores_each

```



```

12
13 In [18]: cpu
14 Out[18]:
15 {'cpu_name': 'Intel(R) Xeon(R) Platinum 8163 CPU @
    2.50GHz',
16  'cpu_num': 1,
17  'cpu_cores_each': 1}
18
19 In [19]:

```

4. 内存信息

- 阿里云虚拟主机

```

1 [root@qfedu.com]# dmidecode -q -t 17 2>/dev/null
2 Memory Device
3     Total Width: Unknown
4     Data Width: Unknown
5     Size: 4096 MB           # 容量
6     Form Factor: DIMM
7     Set: None
8     Locator: DIMM 0        # 插槽号
9     Bank Locator: Not Specified
10    Type: RAM               # 类型   物理的有 DDR3
    DDR4
11    Type Detail: Other
12    Speed: Unknown         # 速率   物理的有 1333 等
13    Manufacturer: Alibaba Cloud
14    Serial Number: Not Specified
15    Asset Tag: Not Specified
16    Part Number: Not Specified
17    Rank: Unknown
18    Configured Clock Speed: Unknown
19    Minimum Voltage: Unknown

```

20	Maximum Voltage: Unknown
21	Configured Voltage: Unknown

- 物理 R710 服务器

1	Memory Device
2	Total Width: 72 bits
3	Data Width: 64 bits
4	Size: 8192 MB
5	Form Factor: DIMM
6	Set: 6
7	Locator: DIMM_B2
8	Bank Locator: Not Specified
9	Type: DDR3
10	Type Detail: Synchronous Registered (Buffered)
11	Speed: 1333 MT/s
12	Manufacturer: 00CE00B380CE
13	Serial Number: 82B79F71
14	Asset Tag: 02120363
15	Part Number: M393B1K70DH0-YH9
16	Rank: 2
17	Memory Device
18	Total Width: 72 bits
19	Data Width: 64 bits
20	Size: 8192 MB
21	Form Factor: DIMM
22	Set: 6
23	Locator: DIMM_B3
24	Bank Locator: Not Specified
25	Type: DDR3
26	Type Detail: Synchronous Registered (Buffered)
27	Speed: 1333 MT/s
28	Manufacturer: 00CE00B380CE
29	Serial Number: 32CDDE81
30	Asset Tag: 02120361

31 Part Number: M393B1K70CH0-YH9
32 Rank: 2
33 Memory Device
34 Total Width: 72 bits
35 Data Width: 64 bits
36 Size: No Module Installed
37 Form Factor: DIMM
38 Set: 4
39 Locator: DIMM_B4
40 Bank Locator: Not Specified
41 Type: DDR3
42 Type Detail: Synchronous
43 Speed: Unknown
44 Manufacturer:
45 Serial Number:
46 Asset Tag:
47 Part Number:
48 Rank: Unknown
49 Memory Device
50 Total Width: 72 bits
51 Data Width: 64 bits
52 Size: 8192 MB
53 Form Factor: DIMM
54 Set: 5
55 Locator: DIMM_B5
56 Bank Locator: Not Specified
57 Type: DDR3
58 Type Detail: Synchronous Registered (Buffered)
59 Speed: 1333 MT/s
60 Manufacturer: 00CE04B380CE
61 Serial Number: 85966B82
62 Asset Tag: 02113621
63 Part Number: M393B1K70DH0-YH9
64 Rank: 2
65 Memory Device

```
66      Total Width: 72 bits
67      Data Width: 64 bits
68      Size: 8192 MB
69      Form Factor: DIMM
70      Set: 6
71      Locator: DIMM_B6
72      Bank Locator: Not Specified
73      Type: DDR3
74      Type Detail: Synchronous Registered (Buffered)
75      Speed: 1333 MT/s
76      Manufacturer: 000000B380CE
77      Serial Number: 00000000
78      Asset Tag: 02121563
79      Part Number:
80      Rank: 2
```

作业

请使用以上信息，编写一个脚本，输出如下信息

```
1  {
2      "base_info": {
3          "host_name": "db_server",
4          "kernel": "3.10.0-957.21.3.el7.x86_64",
5          "os": "CentOS Linux release 7.6.1810
(Core)",
6          'manufacturer': 'Alibaba Cloud',
7          'pod_name': 'Alibaba Cloud ECS',
8          'sn': '0f7e3d86-7742-4612-9f93-e3a9e4754157'
9      },
10     "cpu": {
11         'name': 'Intel(R) Xeon(R) Platinum 8163 CPU
@ 2.50GHz',
12         'num': 1,
```

```

13         'cores_each': 1
14     },
15     "mem": [
16         {
17             'capacity': '8192 MB',
18             'slot': 'DIMM_A3',
19             'model': 'DDR3',
20             'speed': '1333 MT/s',
21             'manufacturer': '00CE04B380CE',
22             'sn': '8362A2F8'
23         },
24         {
25             'capacity': 'No Module Installed',
26             'slot': 'DIMM_A4',
27             'model': 'DDR3',
28             'speed': 'Unknown'
29         }
30         .....略.....
31     ]
32 }

```

内存源数据使用上面 R710 的，映射字典使用下面这个

```

1 key_map = {
2     'Size': 'capacity',
3     'Locator': 'slot',
4     'Type': 'model',
5     'Speed': 'speed',
6     'Manufacturer': 'manufacturer',
7     'Serial Number': 'sn',
8 }
9

```

内存处理参考代码

```

1  def parse(data):
2      key_map = {
3          'Size': 'capacity',
4          'Locator': 'slot',
5          'Type': 'model',
6          'Speed': 'speed',
7          'Manufacturer': 'manufacturer',
8          'Serial Number': 'sn'
9      }
10
11
12     info_mem = []
13     # 首先把服务器上的所有插槽分开，并发到一个列表中
14     # 这个语法叫列表生成式，表示循环中的元素为真时候，将
mem  添加到列表中
15     memory_list = [ mem for mem in
data.split('Memory Device') if mem]
16
17
18     for item in memory_list:
19         # 把每个插槽的信息放到一个字典中
20         single_slot = {}
21
22         for line in item.splitlines():
23             line = line.strip()
24             if len(line.split(': ')) == 2:
25                 key, val = line.split(': ')
26                 if key in key_map:
27                     # 获取到映射字典的 value 作为新字典
的 key
28                     single_slot[key_map[key]] = val
29                 # 含有插槽信息的字典:

```

```
30         # {'capacity': '8192 MB', 'slot': 'DIMM_A3',  
    'model': 'DDR3', 'speed': '1333 MT/s',  
    'manufacturer': '00CE04B380CE', 'sn': '8362A2F8'}  
31  
32         # 由于存在多个内存插槽，每个插槽的号码是不一样的  
33         # 所以可以把当前内存的插槽号作为总体内存字典中的一个  
    key, 值就是当前含有插槽信息的字典  
34         info_mem.append = single_slot  
35     return info_mem
```