
ATTACK
DEFENSE
by PentesterAcademy

Name	Memcache: Pymemcache II
URL	https://www.attackdefense.com/challengedetails?cid=222
Type	Infrastructure Attacks: Memcached

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Q1. How many memcache objects are used to store flag?

Answer: 12

Solution:

Python code:

```
from pymemcache.client.base import Client
import pprint

client = Client(('localhost', 11211))
pp = pprint.PrettyPrinter(indent=4)
pp.pprint(client.stats("items"))
pp.pprint(client.stats("cachedump","4","0"))
```

Save above python code as “code.py” file

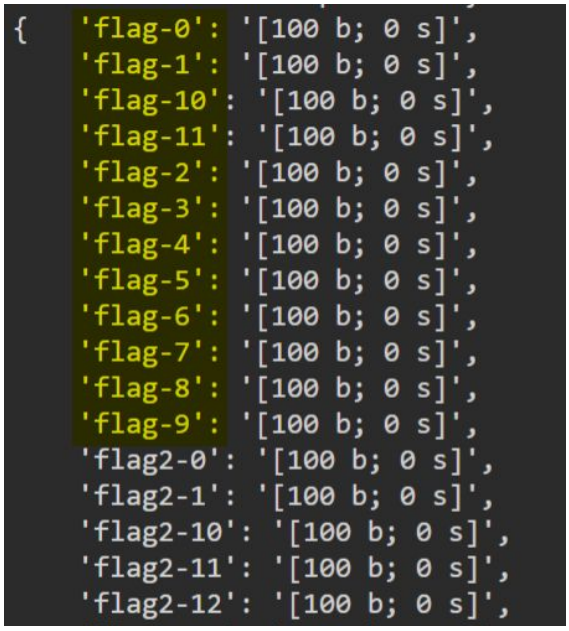
```
from pymemcache.client.base import Client
import pprint

client = Client(('localhost', 11211))
pp = pprint.PrettyPrinter(indent=4)
pp.pprint(client.stats("items"))
pp.pprint(client.stats("cachedump","4","0"))
```

Command: python code.py

```
student@attackdefense:~$ python code.py
{    'items:1:age': 261,
     'items:1:age_hot': 0,
     'items:1:age_warm': 0,
     'items:1:crawler_items_checked': 5,
     'items:1:crawler_reclaimed': 0,
     'items:1:direct_reclaims': 0,
     'items:1:evicted': 0,
     'items:1:evicted_active': 0,
     'items:1:evicted_nonzero': 0,
     'items:1:evicted_time': 0,
     'items:1:evicted_unfetched': 0,
     'items:1:expired_unfetched': 0,
     'items:1:hits_to_cold': 2,
     'items:1:hits_to_hot': 0,
     'items:1:hits_to_temp': 0,
     'items:1:hits_to_warm': 0,
     'items:1:lrutail_reflocked': 0,
     'items:1:moves_to_cold': 2,
     'items:1:moves_to_warm': 1,
     'items:1:moves_within_lru': 0,
     'items:1:number': 1,
     'items:1:number_cold': 1,
     'items:1:number_hot': 0,
     'items:1:number_warm': 0,
     'items:1:outofmemory': 0,
     'items:1:reclaimed': 0,
     'items:1:tailrepairs': 0,
```

```
'items:3:age': 1040,
'items:3:age_hot': 0,
'items:3:age_warm': 0,
'items:3:crawler_items_checked': 5,
'items:3:crawler_reclaimed': 0,
'items:3:direct_reclaims': 0,
'items:3:evicted': 0,
'items:3:evicted_active': 0,
'items:3:evicted_nonzero': 0,
'items:3:evicted_time': 0,
'items:3:evicted_unfetched': 0,
'items:3:expired_unfetched': 0,
'items:3:hits_to_cold': 0,
'items:3:hits_to_hot': 0,
'items:3:hits_to_temp': 0,
'items:3:hits_to_warm': 0,
'items:3:lrutail_reflocked': 0,
'items:3:moves_to_cold': 1,
'items:3:moves_to_warm': 0,
'items:3:moves_within_lru': 0,
'items:3:number': 1,
'items:3:number_cold': 1,
'items:3:number_hot': 0,
'items:3:number_warm': 0,
'items:3:outofmemory': 0,
'items:3:reclaimed': 0,
'items:3:tailrepairs': 0,
'items:4:age': 1040,
'items:4:age_hot': 0,
```

```
'items:4:age_warm': 272,
'items:4:crawler_items_checked': 270,
'items:4:crawler_reclaimed': 0,
'items:4:direct_reclaims': 0,
'items:4:evicted': 0,
'items:4:evicted_active': 0,
'items:4:evicted_nonzero': 0,
'items:4:evicted_time': 0,
'items:4:evicted_unfetched': 0,
'items:4:expired_unfetched': 0,
'items:4:hits_to_cold': 84,
'items:4:hits_to_hot': 0,
'items:4:hits_to_temp': 0,
'items:4:hits_to_warm': 48,
'items:4:lrutail_reflocked': 24,
'items:4:moves_to_cold': 62,
'items:4:moves_to_warm': 29,
'items:4:moves_within_lru': 25,
'items:4:number': 55,
'items:4:number_cold': 33,
'items:4:number_hot': 0,
'items:4:number_warm': 22,
'items:4:outofmemory': 0,
'items:4:reclaimed': 0,
'items:4:tailrepairs': 0}
```



Q2. What is the value of flag?

Answer: fl4g_4_m3mcach3d_g3t_mult1_t3xt

Solution:

Python code:

```
from pymemcache.client.base import Client

import pickle
def retrieve(key):
  result = client.get_multi(['%s-%s' % (key, i) for i in xrange(12)])
  serialized = ''
  for i in range(12):
        serialized=serialized+result[key+'-'+str(i)]
  return serialized

client = Client(('localhost', 11211))
print retrieve("flag")
```

Save the above python code as “code1.py”

```
from pymemcache.client.base import Client

import pickle
def retrieve(key):
  result = client.get_multi(['%s-%s' % (key, i) for i in xrange(12)])
  serialized = ''
  for i in range(12):
        serialized=serialized+result[key+'-'+str(i)]
  return serialized

client = Client(('localhost', 11211))
print retrieve("flag")
```

Command: python code1.py

```
student@attackdefense:~$ python code1.py
In computer security, Capture the Flag (CTF) is a computer security competition. CTF contests are usually designed to serve as an educational e
xercise to give participants experience in securing a machine, as well as conducting and reacting to the sort of attacks found in the real worl
d. Reverse-engineering, network sniffing, protocol analysis, system administration, programming, and cryptanalysis are all skills which have be
en required by prior CTF contests at DEF CON. There are two main styles of capture the flag competitions: attack/defense and Jeopardy!. Your fl
ag is 'fl4g_4_m3mcach3d_g3t_mult1_t3xt'
In an attack/defense style competition, each team is given a machine (or a small network) to defend on an isolated network. Teams are scored on
 both their success in defending their assigned machine and on their success in attacking the other team's machines. Depending on the nature of
 the particular CTF game, teams may either be attempting to take an opponent's flag from their machine or teams may be attempting to plant thei
r own flag on their opponent's machine. Two of the more prominent attack/defense CTF's are held every year at DEF CON, the largest hacker confe
rence, and the
student@attackdefense:~$
```

Q3. What is the value of Flag 2?

Answer: fl4g_4_m3mcach3d_g3t_mult1_b64_3nc_t3xt

Solution:

Python code:

```
from pymemcache.client.base import Client
import pickle
import base64

def retrieve2(key):
  result = client.get_multi(['%s-%s' % (key, i) for i in xrange(32)])
  serialized = ''
  for i in range(17):
        serialized=serialized+result[key+'-'+str(i)]
  return base64.b64decode(serialized)

client = Client(('localhost', 11211))
print retrieve2("flag2")
```

Save the above python code as “code2.py”

```
from pymemcache.client.base import Client
import pickle
import base64

def retrieve2(key):
  result = client.get_multi(['%s-%s' % (key, i) for i in xrange(32)])
  serialized = ''
  for i in range(17):
        serialized=serialized+result[key+'-'+str(i)]
  return base64.b64decode(serialized)

client = Client(('localhost', 11211))
print retrieve2("flag2")
```

Command: python code2.py

```
student@attackdefense:~$ python code2.py
In computer security, Capture the Flag (CTF) is a computer security competition. CTF contests are usually designed to serve as an educational e
xercise to give participants experience in securing a machine, as well as conducting and reacting to the sort of attacks found in the real worl
d. Reverse-engineering, network sniffing, protocol analysis, system administration, programming, and cryptanalysis are all skills which have be
en required by prior CTF contests at DEF CON. There are two main styles of capture the flag competitions: attack/defense and Jeopardy!. Your fl
ag is 'fl4g_4_m3mcach3d_g3t_mult1_b64_3nc_t3xt'
In an attack/defense style competition, each team is given a machine (or a small network) to defend on an isolated network. Teams are scored on
 both their success in defending their assigned machine and on their success in attacking the other team's machines. Depending on the nature of
 the particular CTF game, teams may either be attempting to take an opponent's flag from their machine or teams may be attempting to plant thei
r own flag on their opponent's machine. Two of the more prominent attack/defense CTF's are held every year at DEF CON, the largest hacker confe
rence, and the NYU-CSAW (Cyber Security Awareness Week), the largest student cyber
student@attackdefense:~$
```

Q4. What is the value of Flag 3?

Answer: fl4g_4_m3mcach3d_g3t_mult1_b64_h3x_3ncod3d_t3xt

Solution:

Python code:

```
from pymemcache.client.base import Client
import pickle
import binascii

def retrieve3(key):
  result = client.get_multi(['%s-%s' % (key, i) for i in xrange(32)])
  serialized = ''
  for i in range(25):
        serialized=serialized+result[key+'-'+str(i)]
  return binascii.unhexlify(serialized)

client = Client(('localhost', 11211))
print retrieve3("flag3")
```

Save the above python code as “code3.py”

```
from pymemcache.client.base import Client
import pickle
import binascii

def retrieve3(key):
  result = client.get_multi(['%s-%s' % (key, i) for i in xrange(32)])
  serialized = ''
  for i in range(25):
        serialized=serialized+result[key+'-'+str(i)]
  return binascii.unhexlify(serialized)

client = Client(('localhost', 11211))
print retrieve3("flag3")
```

Command: python code3.py

```
student@attackdefense:~$ python code3.py
In computer security, Capture the Flag (CTF) is a computer security competition. CTF contests are usually designed to serve as an educational e
xercise to give participants experience in securing a machine, as well as conducting and reacting to the sort of attacks found in the real worl
d. Reverse-engineering, network sniffing, protocol analysis, system administration, programming, and cryptanalysis are all skills which have be
en required by prior CTF contests at DEF CON. There are two main styles of capture the flag competitions: attack/defense and Jeopardy!. Your fl
ag is 'fl4g_4_m3mcach3d_g3t_mult1_b64_h3x_3ncod3d_t3xt'
In an attack/defense style competition, each team is given a machine (or a small network) to defend on an isolated network. Teams are scored on
 both their success in defending their assigned machine and on their success in attacking the other team's machines. Depending on the nature of
 the particular CTF game, teams may either be attempting to take an opponent's flag from their machine or teams may be attempting to plant thei
r own flag on their opponent's machine. Two of the more prominent attack/defense CTF's are held every year at DEF CON, the largest hacker confe
rence, and the NYU-CSAW (Cyber Security Awareness
student@attackdefense:~$
```

References:

1. Memcached (https://memcached.org/)
2. Pymemcache (https://pypi.org/project/pymemcache/)