X-Force Red

Our Mission: Hacking Anything to Secure Everything

Automating Hashtopolis

Presenters:
EvilMog, Senior Managing Consultant, X-Force Red

19 April 2019

Disclaimer

This talk has not been vetted by IBM, IBM Security or X-Force Red

Opinions expressed in this talk are mine and mine alone

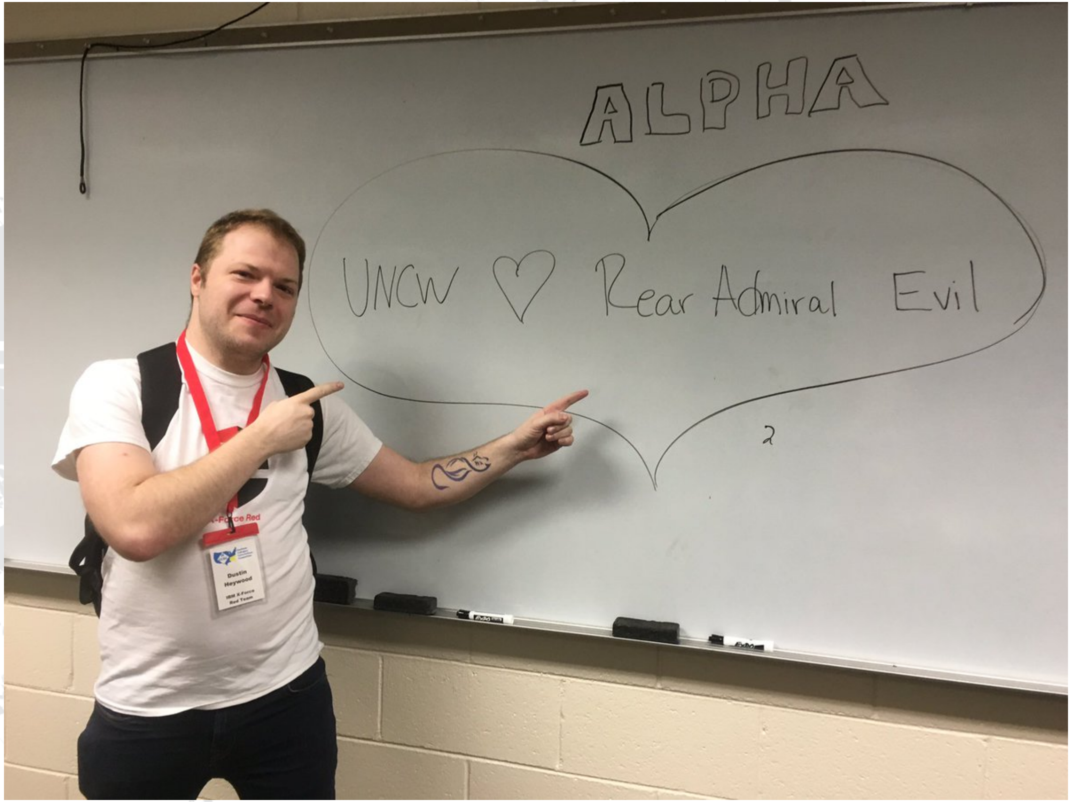
I am also not speaking on behalf of Hashcat, Hashtopolis, Team Hashcat, The Church of Wifi, CsP, the illuminati, or bigfoot.

No warrantees express or implied, I am not responsible if you set your house on fire, break your hashtopolis instance, blow up your database or otherwise severely break something and/or get yourself fired.

Please seek the assistance of a professional before doing this in production.

Who I am

- EvilMog
- Bishop of the Church of Wifi
- Hacker for X-Force Red
- Member of Team Hashcat



Why are you here

- **Hashtopolis is awesome**
- **Hashtopolis manages Hashcat**
- **Hashtopolis can be Automated with JSON**
- **That’s right you can be lazy**

Hashtopolis - Terminology

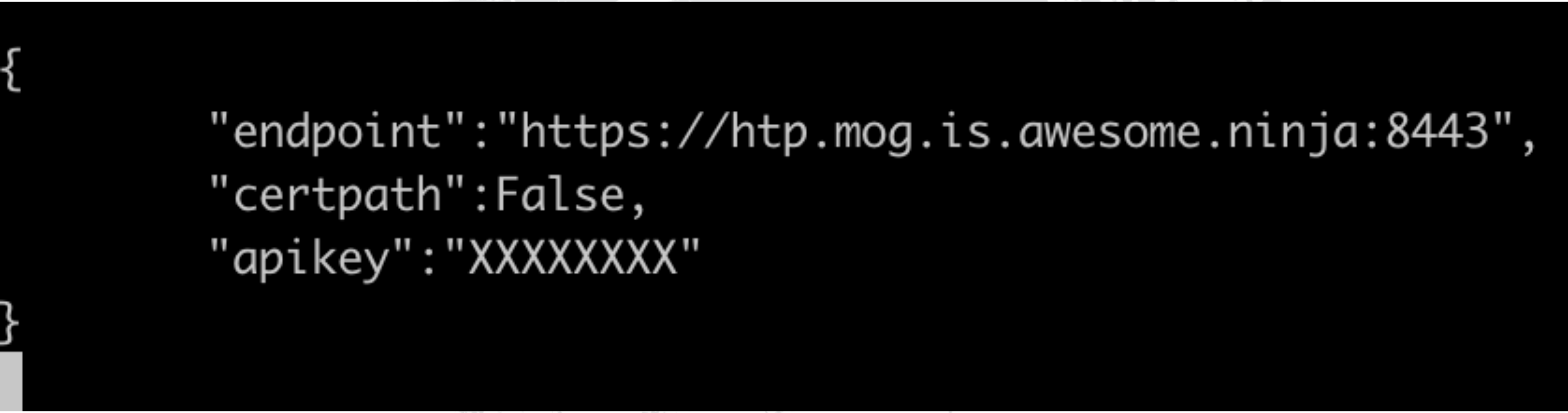
- Agent – A hashcat worker node
- Trust – An agent is trusted to run secret hashlists
- Secret – A hashlist that cannot be sent to untrusted agents
- Task – A hashcat job
- Hashlist – A list of hashes
- Hashcat – An awesome password cracker
  - Thanks Atom
- Hashtopolis – Manages hashcat to distribute work
  - Thanks s3!nlc, hops, winxp et al

Getting started

- Step 1) Install Hashtopolis
  - https://github.com/s3inlc/hashtopolis
- Step 2) Generate an API Key
  - Users -> API Management
- Step 3) Read https://github.com/s3inlc/hashtopolis/blob/master/doc/user-api/user-api.pdf
- Step 4) Make a new directory for your project
- Step 5) In that directory clone this repo as your submodule:
  - https://github.com/evilmog/htpclientapi
- Step 6) ??????
- Step 7) Profit (maybe)

config.json

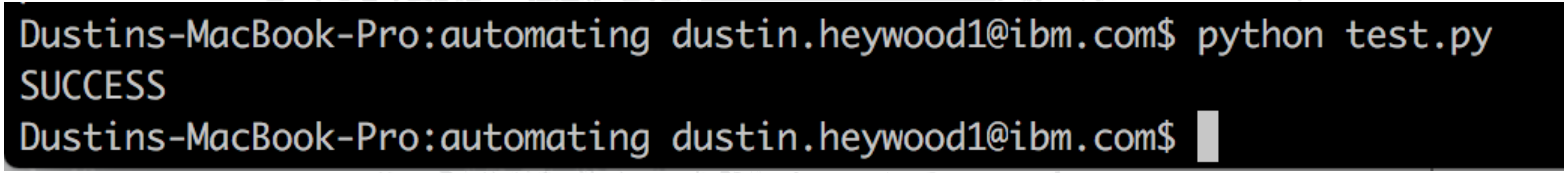
- 1) **Create a config.json**



```
{
        "endpoint":"https://htp.mog.is.awesome.ninja:8443",
        "certpath":False,
        "apikey":"XXXXXXXX"
}
```

Test connection script

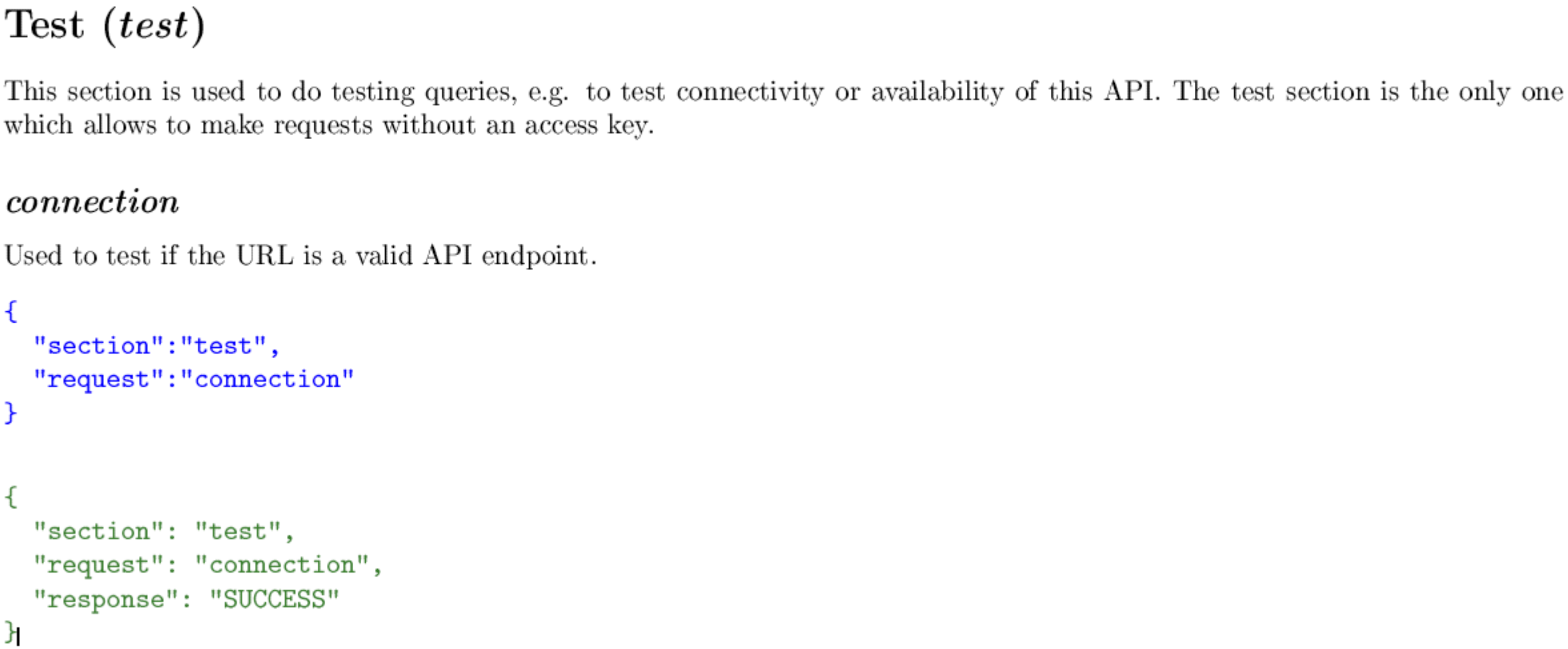
```
from htpclientapi.functions import *
x = test_connection()
print x
```



```
Dustins-MacBook-Pro:automating dustin.heywood1@ibm.com$ python test.py
SUCCESS
Dustins-MacBook-Pro:automating dustin.heywood1@ibm.com$ 
```

Test Connection under the hood

- Everything is JSON, nice and simple



Test (*test*)

This section is used to do testing queries, e.g. to test connectivity or availability of this API. The test section is the only one which allows to make requests without an access key.

connection

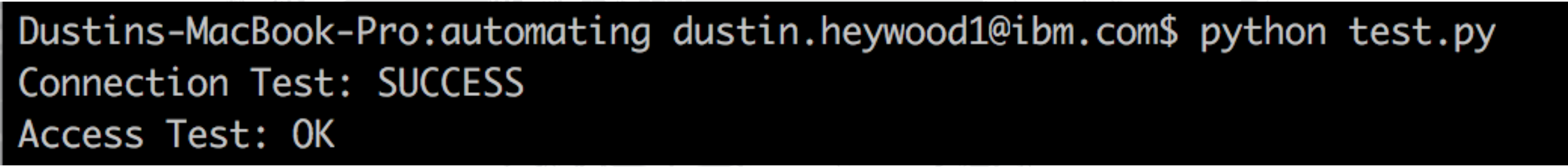
Used to test if the URL is a valid API endpoint.

```
{
  "section":"test",
  "request":"connection"
}
```

```
{
  "section": "test",
  "request": "connection",
  "response": "SUCCESS"
}
```

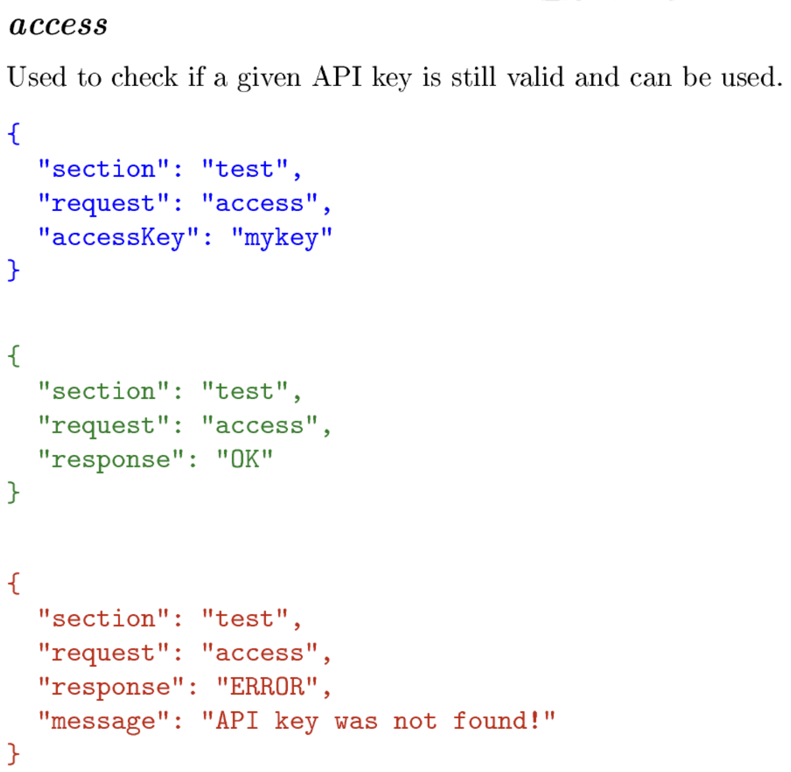
Make sure your API Key works

```
from htpclientapi.functions import *

x = test_connection()
y = test_access()
print "Connection Test: " + x
print "Access Test: " + y
```



```
Dustins-MacBook-Pro:automating dustin.heywood1@ibm.com$ python test.py
Connection Test: SUCCESS
Access Test: OK
```

Access check under the hood



access

Used to check if a given API key is still valid and can be used.

```
{
  "section": "test",
  "request": "access",
  "accessKey": "mykey"
}
```

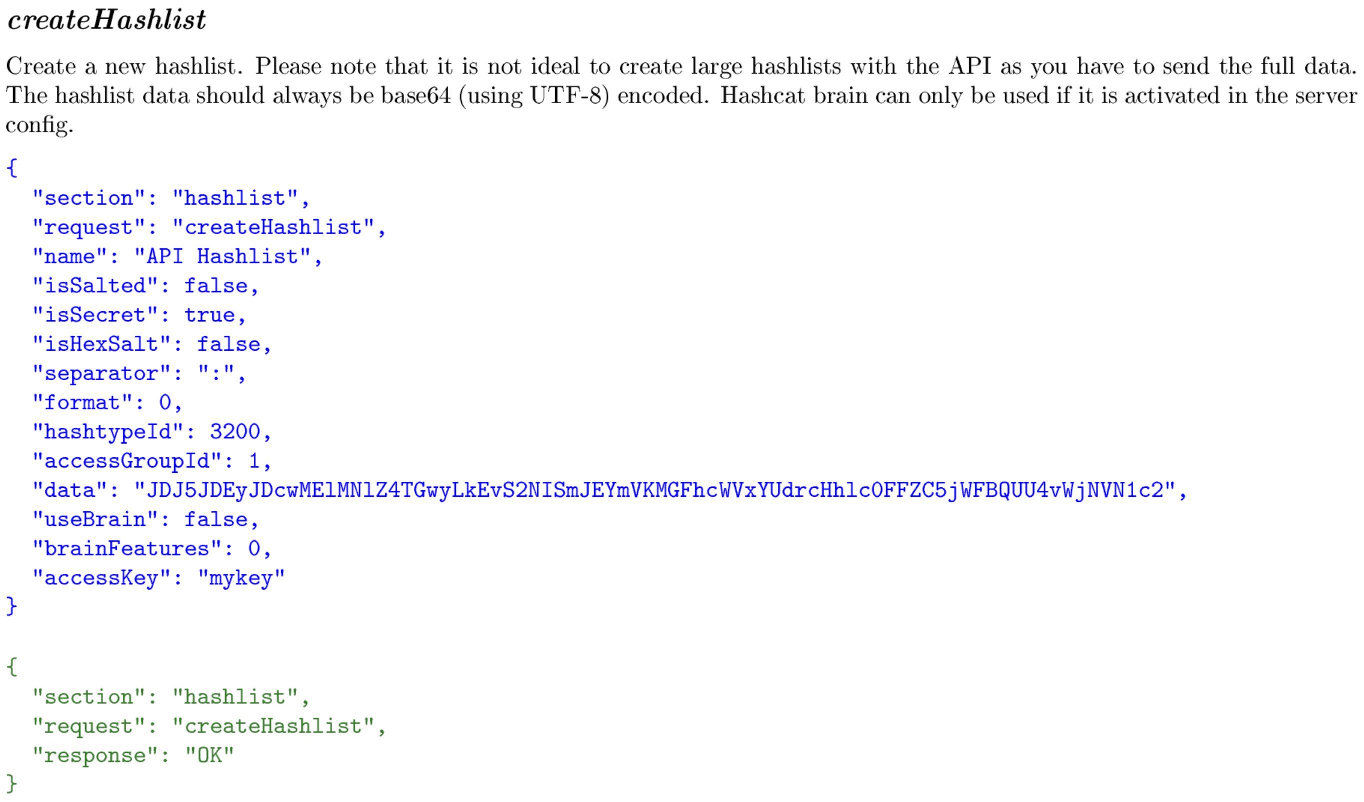
```
{
  "section": "test",
  "request": "access",
  "response": "OK"
}
```

```
{
  "section": "test",
  "request": "access",
  "response": "ERROR",
  "message": "API key was not found!"
}
```

Create Hashlist

```
from htpclientapi.functions import *
import base64

data = open("example0.hash", "r").read()
encoded = base64.b64encode(data)

# options order
# name, issalted, issecret, ishexsalt, separator, hashformat,
# hashtypeid, accessgroupid, data, usebrain, brainfeatures

result = createhashlist("Example0 Hash", False, True, False, ":", 0, 0, 1, encoded,
        False, 0)

print result
```

Create Hashlist Under the Hood



createHashlist

Create a new hashlist. Please note that it is not ideal to create large hashlists with the API as you have to send the full data. The hashlist data should always be base64 (using UTF-8) encoded. Hashcat brain can only be used if it is activated in the server config.

```
{
  "section": "hashlist",
  "request": "createHashlist",
  "name": "API Hashlist",
  "isSalted": false,
  "isSecret": true,
  "isHexSalt": false,
  "separator": ":",
  "format": 0,
  "hashtypeId": 3200,
  "accessGroupId": 1,
  "data": "JDJ5JDEyJDcwMElMNlZ4TGwyLkEvS2NISmJEYmVKMGFhcWVxYUdrcHhlc0FFZC5jWFBQUU4vWjNVN1c2",
  "useBrain": false,
  "brainFeatures": 0,
  "accessKey": "mykey"
}
```

```
{
  "section": "hashlist",
  "request": "createHashlist",
  "response": "OK"
}
```

Create pathwell masks as tasks

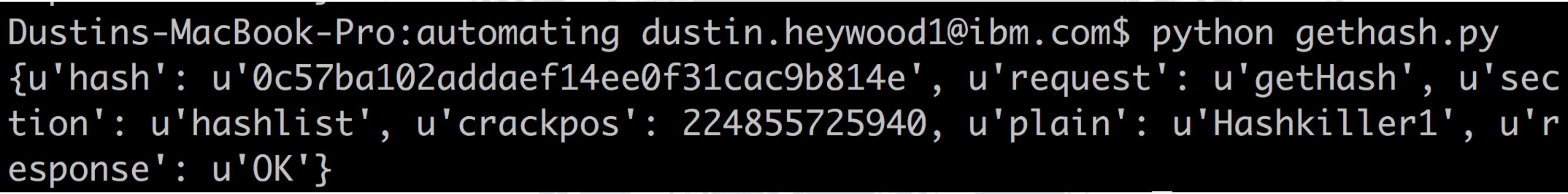
```
from htpclientapi.functions import *

hashlistid = str(1)
priority = 102
benchmark = "speed”
pathwellfile = open("pathwell.txt", 'r')

for line in pathwellfile:
    priority = priority – 1
    pmask = line.rstrip()
    newtask = createtask(
                    ("PATHWELL -a3 - " + pmask), hashlistid,
                    ("#HL# -a 3 " + pmask), 1200, 5, benchmark,
                    "#FFFFFF", False, False, 0, 1, str(priority), [], False)
    print newtask
```

Delete all tasks

```
from htpclientapi.functions import *

data = listtasks()

for line in data['tasks’]:
    killtask = deletetask(line['taskId’])
    print killtask
```

lookup cracked hash

```
from htpclientapi.functions import *

x = gethash("0c57ba102addaef14ee0f31cac9b814e")
print x
```

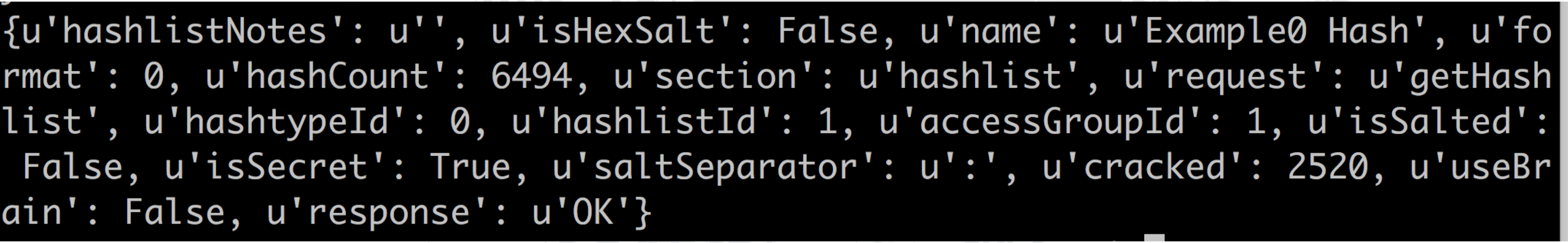


```
Dustins-MacBook-Pro:automating dustin.heywood1@ibm.com$ python gethash.py
{u'hash': u'0c57ba102addaef14ee0f31cac9b814e', u'request': u'getHash', u'sec
tion': u'hashlist', u'crackpos': 224855725940, u'plain': u'Hashkiller1', u'r
esponse': u'OK'}
```

get hashlist details

from htpclientapi.functions import *

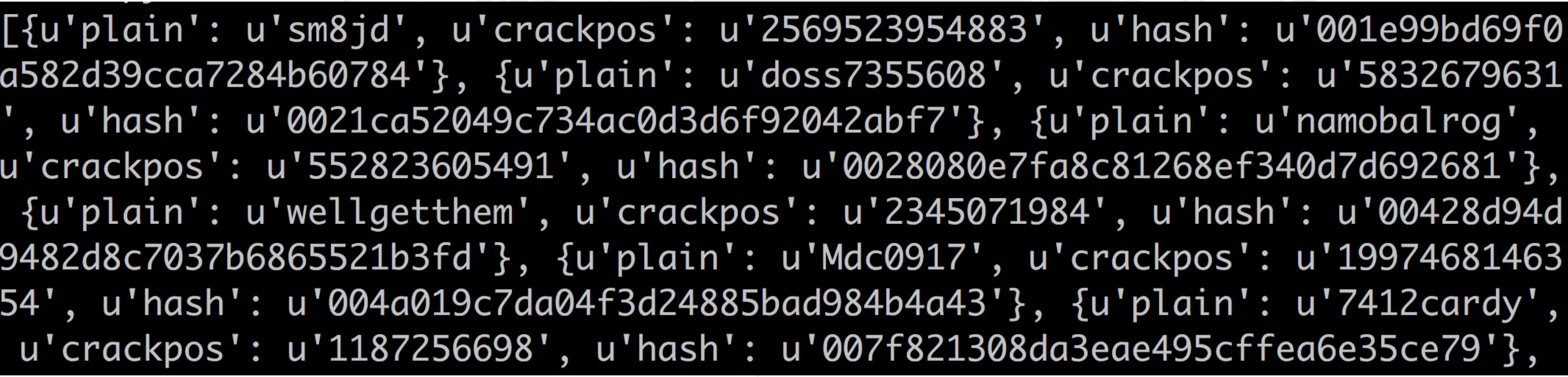
x = gethashlist(1)
print x



```
{u'hashlistNotes': u'', u'isHexSalt': False, u'name': u'Example0 Hash', u'fo
rmat': 0, u'hashCount': 6494, u'section': u'hashlist', u'request': u'getHash
list', u'hashtypeId': 0, u'hashlistId': 1, u'accessGroupId': 1, u'isSalted':
 False, u'isSecret': True, u'saltSeparator': u':', u'cracked': 2520, u'useBr
ain': False, u'response': u'OK'}
```

get all cracked hashes for a hashlist

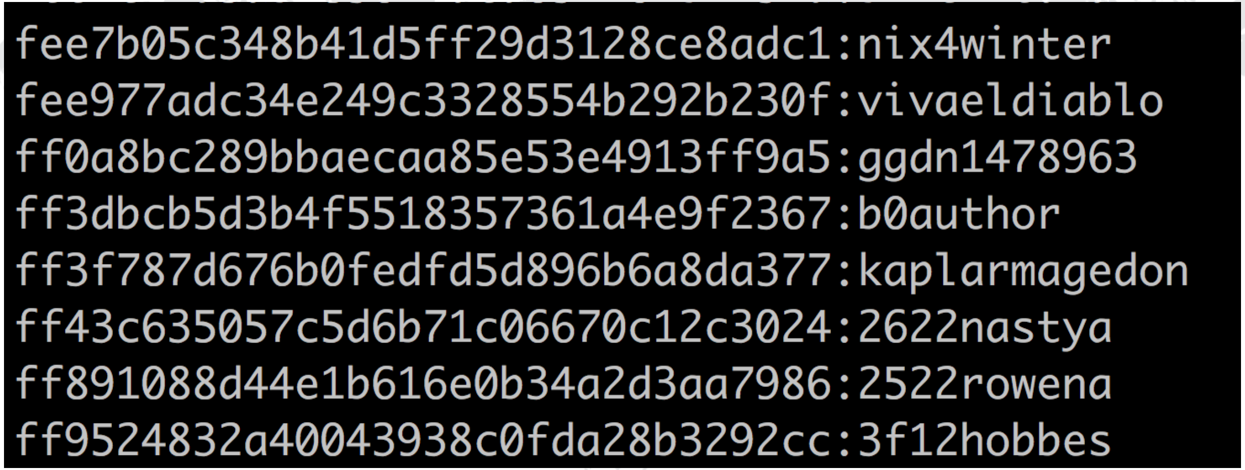
```
from htpclientapi.functions import *

hashes = gethashlistcracked(1)['cracked’]
print hashes
```



```
[{u'plain': u'sm8jd', u'crackpos': u'2569523954883', u'hash': u'001e99bd69f0
a582d39cca7284b60784'}, {u'plain': u'doss7355608', u'crackpos': u'5832679631
', u'hash': u'0021ca52049c734ac0d3d6f92042abf7'}, {u'plain': u'namobalrog',
u'crackpos': u'552823605491', u'hash': u'0028080e7fa8c81268ef340d7d692681'},
 {u'plain': u'wellgetthem', u'crackpos': u'2345071984', u'hash': u'00428d94d
9482d8c7037b6865521b3fd'}, {u'plain': u'Mdc0917', u'crackpos': u'19974681463
54', u'hash': u'004a019c7da04f3d24885bad984b4a43'}, {u'plain': u'7412cardy',
 u'crackpos': u'1187256698', u'hash': u'007f821308da3eae495cffea6e35ce79'},
```

get cracked hashes an alternative way

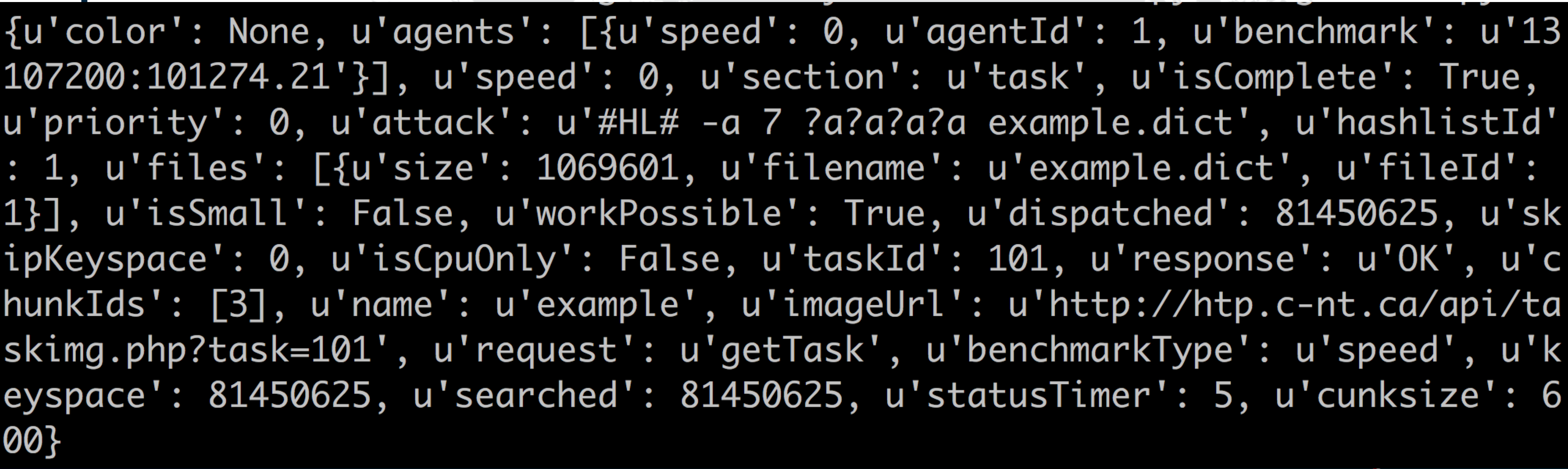
```
from htpclientapi.functions import *

hashes = gethashlistcracked(1)['cracked']

for hash in hashes:
  print hash['hash']+ ":" + hash['plain']
```



```
fee7b05c348b41d5ff29d3128ce8adc1:nix4winter
fee977adc34e249c3328554b292b230f:vivaeldiablo
ff0a8bc289bbaecaa85e53e4913ff9a5:ggdn1478963
ff3dbcb5d3b4f5518357361a4e9f2367:b0author
ff3f787d676b0fedfd5d896b6a8da377:kaplarmagedon
ff43c635057c5d6b71c06670c12c3024:2622nastya
ff891088d44e1b616e0b34a2d3aa7986:2522rowena
ff9524832a40043938c0fda28b3292cc:3f12hobbes
```

Get Task Details

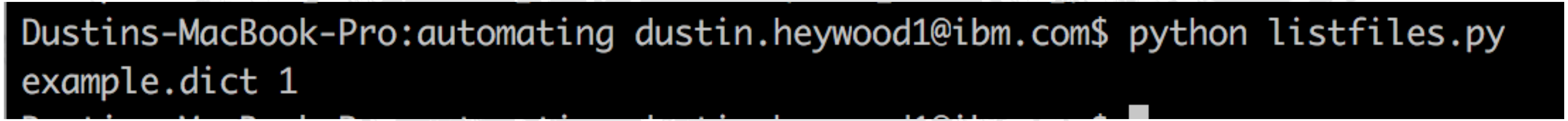
```
from htpclientapi.functions import *

taskId = 101
data = gettask(taskId)
print data
```



```
{u'color': None, u'agents': [{u'speed': 0, u'agentId': 1, u'benchmark': u'13
107200:101274.21'}], u'speed': 0, u'section': u'task', u'isComplete': True,
u'priority': 0, u'attack': u'#HL# -a 7 ?a?a?a?a example.dict', u'hashlistId'
: 1, u'files': [{u'size': 1069601, u'filename': u'example.dict', u'fileId':
1}], u'isSmall': False, u'workPossible': True, u'dispatched': 81450625, u'sk
ipKeyspace': 0, u'isCpuOnly': False, u'taskId': 101, u'response': u'OK', u'c
hunkIds': [3], u'name': u'example', u'imageUrl': u'http://htp.c-nt.ca/api/ta
skimg.php?task=101', u'request': u'getTask', u'benchmarkType': u'speed', u'k
eyspace': 81450625, u'searched': 81450625, u'statusTimer': 5, u'cunksize': 6
00}
```

List Files

```
from htpclientapi.functions import *

for line in files['files’]:
    print line['filename'] + " " + str(line['fileId'])
```

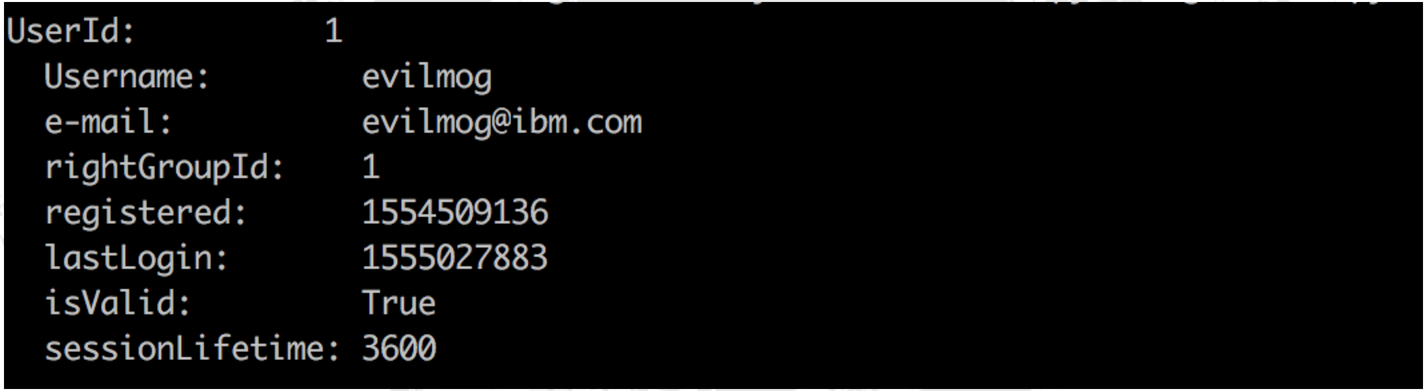


```
Dustins-MacBook-Pro:automating dustin.heywood1@ibm.com$ python listfiles.py
example.dict 1
```

Get User List

```
from htpclientapi.functions import *

data = listusers()

for user in data["users"]:
    userdata = getuser(str(user["userId"]))
    print "UserId:           " + str(userdata["userId"])
    print "  Username:        " + str(userdata["username"])
    print "  e-mail:          " + str(userdata["email"])
    print "  rightGroupId:    " + str(userdata["rightGroupId"])
    print "  registered:      " + str(userdata["registered"])
    print "  lastLogin:       " + str(userdata["lastLogin"])
    print "  isValid:         " + str(userdata["isValid"])
    print "  sessionLifetime: " + str(userdata["sessionLifetime"])
    print " "
```

User List



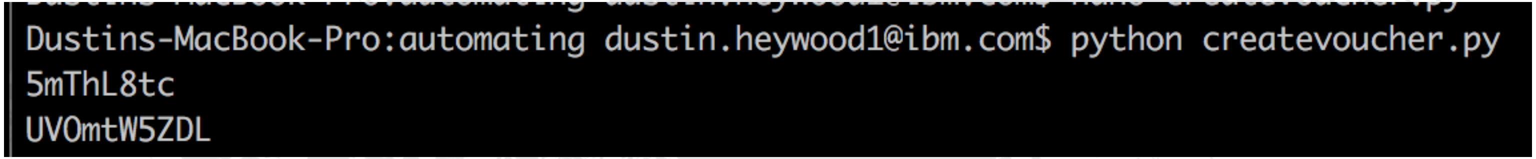
```
UserId:         1
  Username:         evilmog
  e-mail:           evilmog@ibm.com
  rightGroupId:     1
  registered:       1554509136
  lastLogin:        1555027883
  isValid:          True
  sessionLifetime: 3600
```

Create a new User

```
from htpclientapi.functions import *

username = "test”
email = "test@test.com"
rightgroupid = 1

createuser(username, email, rightgroupid)

userlist = listusers()

for line in userlist['users’]:
  if line['username'] == username:
    setuserpassword(line['userId'], "P@$$w0rd")
```

Create Vouchers / List Vouchers

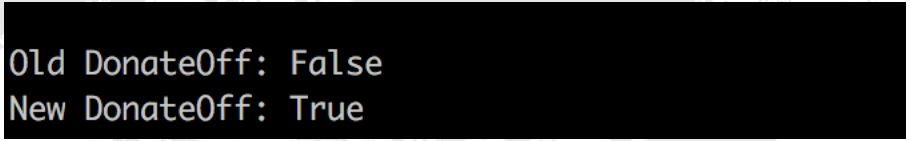
```
from htpclientapi.functions import *

createagentvoucherrandom()
vouchers = listagentvouchers()

for voucher in vouchers['vouchers’]:
  print voucher
```



```
Dustins-MacBook-Pro:automating dustin.heywood1@ibm.com$ nano createvoucher.py
Dustins-MacBook-Pro:automating dustin.heywood1@ibm.com$ python createvoucher.py
5mThL8tc
UVOmtW5ZDL
```

Configuration – Changing Values

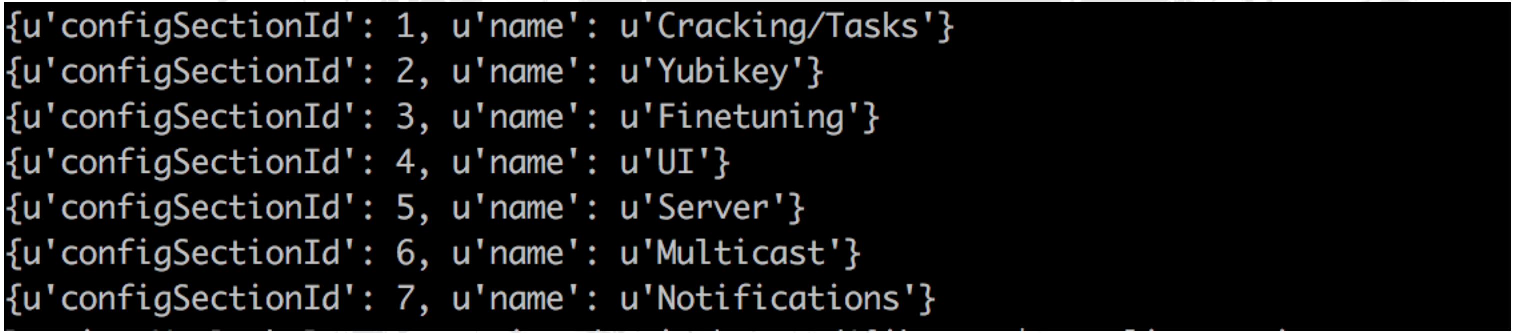
```
from htpclientapi.functions import *

force = False
donate = getconfig('donateOff’)
print "Old DonateOff: " + str(donate['value’])
setconfig('donateOff', True, force)
donate = getconfig('donateOff’)
print "New DonateOff: " + str(donate['value'])
```



```
Old DonateOff: False
New DonateOff: True
```

List Configuration Sections

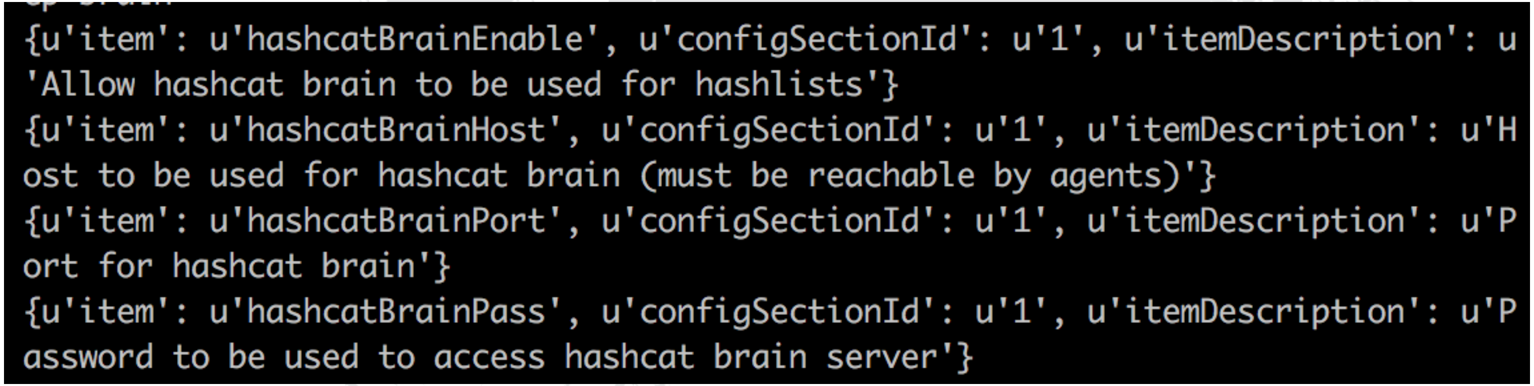
```
from htpclientapi.functions import *

sections = listsections()

for line in sections['configSections’]:
  print line
```



```
{u'configSectionId': 1, u'name': u'Cracking/Tasks'}
{u'configSectionId': 2, u'name': u'Yubikey'}
{u'configSectionId': 3, u'name': u'Finetuning'}
{u'configSectionId': 4, u'name': u'UI'}
{u'configSectionId': 5, u'name': u'Server'}
{u'configSectionId': 6, u'name': u'Multicast'}
{u'configSectionId': 7, u'name': u'Notifications'}
```

List Configuration Items

```
from htpclientapi.functions import *

configitems = listconfig()

for line in configitems['items’]:
  print line
```



```
{u'item': u'hashcatBrainEnable', u'configSectionId': u'1', u'itemDescription': u
'Allow hashcat brain to be used for hashlists'}
{u'item': u'hashcatBrainHost', u'configSectionId': u'1', u'itemDescription': u'H
ost to be used for hashcat brain (must be reachable by agents)'}
{u'item': u'hashcatBrainPort', u'configSectionId': u'1', u'itemDescription': u'P
ort for hashcat brain'}
{u'item': u'hashcatBrainPass', u'configSectionId': u'1', u'itemDescription': u'P
assword to be used to access hashcat brain server'}
```

Reactivate all deactivated agents

```
from htpclientapi.functions import *

agents = listagents()



for line in agents['agents’]:
  agentid = line['agentId’]
  taskunassignagent(agentid)
  setagentcpuonly(agentid, False)
  setagenttrusted(agentid, True)
  setagentextraparams(agentid, "-O -w3")
  setagentactive(agentid, True)
```

The end

- That’s right that’s all folks
- Hit me up on twitter @Evil_Mog
- Check out github:
  - https://github.com/s3inlc/hashtopolis
  - https://github.com/s3inlc/hashtopolis/blob/master/doc/user-api/user-api.pdf
  - https://github.com/evilmog/htpclientapi

Donate to the hashtopolis project, these folks need beer, or buy them a beer at cyphercon

- **BTC 15gi3X5L4VPa5S2yygztYaN7MF7VA26Zaf - ETH 0x06B3Ae7561AD763eF58Df9C37deB6757bDA2BC0c**
- **Special thanks to s3in!c, he has saved my bacon as has the rest of the CsP crew**
- **Special thanks to Atom for making hashcat**

X-Force Red

THANK YOU

FOLLOW US ON:

- ibm.com/security
- securityintelligence.com
- xforce.ibmcloud.com
- @ibmsecurity
- youtube/user/ibmsecuritysolutions

Statement of Good Security Practices: IT system security involves protecting systems and information through prevention, detection and response to improper access from within and outside your enterprise. Improper access can result in information being altered, destroyed, misappropriated or misused or can result in damage to or misuse of your systems, including for use in attacks on others. No IT system or product should be considered completely secure and no single product, service or security measure can be completely effective in preventing improper use or access. IBM systems, products and services are designed to be part of a lawful, comprehensive security approach, which will necessarily involve additional operational procedures, and may require other systems, products or services to be most effective. IBM does not warrant that any systems, products or services are immune from, or will make your enterprise immune from, the malicious or illegal conduct of any party.

IBM