Remove "unsafe" ciphers from existing SSL profiles

remove_unsafe_ciphers.py

```
#!/usr/bin/env python


import argparse
import getpass
from requests.packages import urllib3
from avi.sdk.avi_api import ApiSession


UNSAFE_CIPHERS = {
    # Below is not in Controller's unsafe list but probably should be!
    #'TLS_RSA_WITH_3DES_EDE_CBC_SHA',
    'TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA',
    'TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256',
    'TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA',
    'TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384',
    'TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA',
    'TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256',
    'TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA',
    'TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384',
    'TLS_RSA_WITH_AES_128_GCM_SHA256',
    'TLS_RSA_WITH_AES_256_GCM_SHA384',
    'TLS_RSA_WITH_AES_256_CBC_SHA256',
    'TLS_RSA_WITH_AES_128_CBC_SHA',
    'TLS_RSA_WITH_AES_128_CBC_SHA256',
    'TLS_RSA_WITH_AES_256_CBC_SHA'
}

urllib3.disable_warnings()

parser = argparse.ArgumentParser(
                    formatter_class=argparse.RawDescriptionHelpFormatter)
parser.add_argument('-c', '--controller',
                    help='FQDN or IP address of Avi Vantage controller')
parser.add_argument('-u', '--user', help='Avi Vantage username',
                    default='admin')
parser.add_argument('-p', '--password', help='Avi Vantage password')
parser.add_argument('-t', '--tenant', help='Tenant containing SSL Profiles',
                    default='admin')
parser.add_argument('-n', '--name',
                    help='SSL Profile(s) to update ("begins with" criteria)',
                    default='')

args = parser.parse_args()

if args:

    # If not specified on the command-line, prompt the user for the
    # controller IP address and/or password

    controller = args.controller
    user = args.user
    password = args.password
    tenant = args.tenant
    name = args.name

    while not controller:
        controller = input('Controller:')

    while not password:
        password = getpass.getpass('Password for %s@%s:' %
                                   (user, controller))

    api = ApiSession.get_session(controller, user, password)
```

```
    sslprofiles = api.get('sslprofile', tenant=tenant,
                          params='search=(name,%s)' % name).json()

    for sslprofile in sslprofiles['results']:
        print('Processing SSL Profile %s' % sslprofile['name'])
        ciphers = set(sslprofile['cipher_enums'])
        ciphers_removed = ciphers & UNSAFE_CIPHERS
        if ciphers_removed:
            print('Removing the following ciphers:')
            print(', '.join(ciphers_removed))
            sslprofile['cipher_enums'] = list(ciphers - UNSAFE_CIPHERS)
            resp = api.put('sslprofile/%s' % sslprofile['uuid'], sslprofile,
                           tenant=tenant)
            if resp.status_code == 200:
                print('OK!')
            else:
                print('Got error %d' % (resp.status_code))
        else:
            print('No ciphers need to be removed')

        print()
        print()
```

e.g.

```
python remove_unsafe_ciphers.py -c <controller ip> -n System-
```

```
Will prompt for admin password and update ciphers in SSL profiles whose names begin with "System-" in the admin tenant
```

```
python remove_unsafe_ciphers.py -c <controller ip> -t *
```

Will prompt for admin password and update ciphers in all SSL profiles in all tenants.