Python ile Nmap taramalarının gerçekleştirilmesi için çeşitli yöntemler bulunmaktadır. Bu iş için farklı kütüphanelerin kullanımı mümkün olduğu gibi işletim sistemi komutlarının çalıştırılarak aynı işlemin gerçekleştirilmesi mümkün olmaktadır. Burada python-nmap kütüphanesi ile Nmap taramalarını nasıl otomatize edilebileceğinden bahsedilecektir. Öncelikle sisteme gerekli kütüphanenin kurulması gerekmektedir. Kullanılan python sürümüne göre ilgili versiyonu temin edilerek kurulmalıdır.

Kurulu python sürüm bilgisi python -V komutu ile öğrenilebilir.

```
# python -V
Python 2.7.3
```

Python 2.x sürümü için http://xael.org/norman/python/python-nmap/python-nmap-0.1.4.tar.gz
3.x sürümü için ise http://xael.org/norman/python/python-nmap/python-nmap-0.3.2.tar.gz
adresindeki sürüm temin edilmelidir.

```
# wget http://xael.org/norman/python/python-nmap/python-nmap-0.1.4.tar.gz
# tar -zxvf python-nmap-0.1.4.tar.gz
# cd python-nmap-0.1.4/
# python setup.py install
```

Kütüphanenin ilişkin bazı fonksiyonların kullanımı interaktif python kabuğu aracılığı ile gösterilmiştir.

```
>>> import nmap
```
Kütüphanenenin kullanımı aktive edilir.

```
>>> nm = nmap.PortScanner()
```
Tarama için kullanılacak olan ilgili referans bilgisi oluşturulur.

```
>>> nm.scan('192.168.1.0/24', '21,22,80,443,445')
```
Hedef ip adres ve port bilgis tanımlanır.

```
>>> nm.scan('192.168.1.0/24', arguments='-n -Pn -sS -p
```

21,22,80,443,445')
Tarama için özelleştirilmiş NMAP parametreleri kullanılıyor.

```
>>> nm.command_line()
u'nmap -oX - -n -Pn -sS -p 21,22,80,443,445 192.168.1.0/24'
```
Tarama seçenekleri gösterilir.

```
>>> nm['192.168.1.1'].state()
u'up'
```
Tarama sonrası 192.168.1.1 ip adresi için durum bilgisi alınır.

```
>>> nm['192.168.1.23']['tcp'].keys()
[80, 443, 21, 22, 445]
```
Tarama sonrası 192.168.1.23 ip adresi için açık port bilgisi raporlanır.

```
>>> for host in nm.all_hosts():
... print "Host : %s"% host
... print "----------"
... for proto in nm[host].all_protocols():
... result = nm[host][proto].keys()
... result.sort()
... for port in result:
... print "\tPort : %s State: %s"% (port, nm[host][proto][port]['state'])
...
Host : 192.168.1.1
----------
Port : 21 State: open
Port : 22 State: open
Port : 80 State: open
Port : 443 State: open
Port : 445 State: filtered
Host : 192.168.1.37
----------
Port : 21 State: closed
Port : 22 State: closed
Port : 80 State: open
Port : 443 State: open
Port : 445 State: open
>>>
```

Tarama sonuçları ip adresi bazında port durumlarını belirtecek şekilde raporlanır. Kütüphanenin kullanımına ilişkin temel bir örneğe ilişkin ufak bir program aşağıda gösterilmiştir.

Aşağıdaki program normal Nmap rapor biçiminde çıktı üretmektedir. Programın kodlarına

```
#!/usr/bin/python

__VERSION__ = '0.1'
__AUTHOR__ = 'Galkan'
__DATE__ = '22.12.2013'

try:
        import nmap
        import sys
        import re
        import os
        import argparse
except ImportError,e:
        import sys
        sys.stdout.write("%s\n" %e)
        sys.exit(1)


class Tarama:
        def __init__(self):
                self.cmd_arg = "-n -Pn -sS -T4 --top-
ports 10"
                self.nmap_services_file              =
"/usr/share/nmap/nmap-services"
                self.nm = nmap.PortScanner()


        def get_service_name(self, port, proto):
                nmap_file                            =
open(self.nmap_services_file,"r")
                service = ""
                for line in nmap_file:
                        if
re.search("([^\s]+)\s%d/%s\s"% (port, proto), line):
                                service              =
re.search("([^\s]+)\s%d/%s\s"%     (port,     proto),
line).groups(1)[0]
                                break
                return service
```

```
        def run_scan(self,targets):
                self.nm.scan(hosts  =  "%s"%  targets,
arguments = "%s"% self.cmd_arg)
                for host in self.nm.all_hosts():
                        print("PORT             STATE
SERVICE")
                        for          proto          in
self.nm[host].all_protocols():
                                result              =
self.nm[host][proto].keys()
                                result.sort()
                                for port in result:
                                        res         =
str(port) + "/" + proto
                                        space  =  str("
" * (9 - len(res)))
                                        service     =
self.get_service_name(port, proto)
                                        state       =
self.nm[host][proto][port]['state']
                                        space2 = str("
" * (10 - len(state)))
                                        print
"%s/%s%s%s%s%s"                                  %
(port,proto,space,state,space2,service)


if __name__ == "__main__":
        parser                                      =
argparse.ArgumentParser(description='Nmap   Ile   Port
Tarama Programi')
        parser.add_argument('-t','--target',
help='Hedef Ip Adres Bilgisi', required=True)
        args = parser.parse_args()

        try:
                tarama = Tarama()
                tarama.run_scan(args.target)
        except Exception, e:
                print >> sys.stderr, "Hata: %s"% e
                sys.exit(2)
```

Buradaki amaç temel Nmap fonksiyonlarının python programlama dili ile

otomatize bir biçimde gerçekleştirilmesidir. Belirtilen kod çalışma dizininde ***nmap_tarama.py*** isimli dosyaya kaydedilerek çalıştırılmaktadır.

```
# ./nmap_tarama.py -t 192.168.1.1
PORT STATE SERVICE
21/tcp open ftp
22/tcp open ssh
23/tcp filtered telnet
110/tcp filtered pop3
139/tcp filtered netbios-ssn
...
```

Kütüphanenin kullanımına ilişkin daha fazla bilgi için http://xael.org/norman/python/python-nmap/ adresine göz atılabilir. Ayrıca kütüphane kaynak kodları ile birlikte gelen example.py dosyası içerisindeki örneklerde incelenebilir.

Kaynaklar:

http://xael.org/norman/python/python-nmap/
http://www.galkan.net/p/python.html
http://www.pythondersleri.com/2014/01/python-ile-nmap-programlama.html