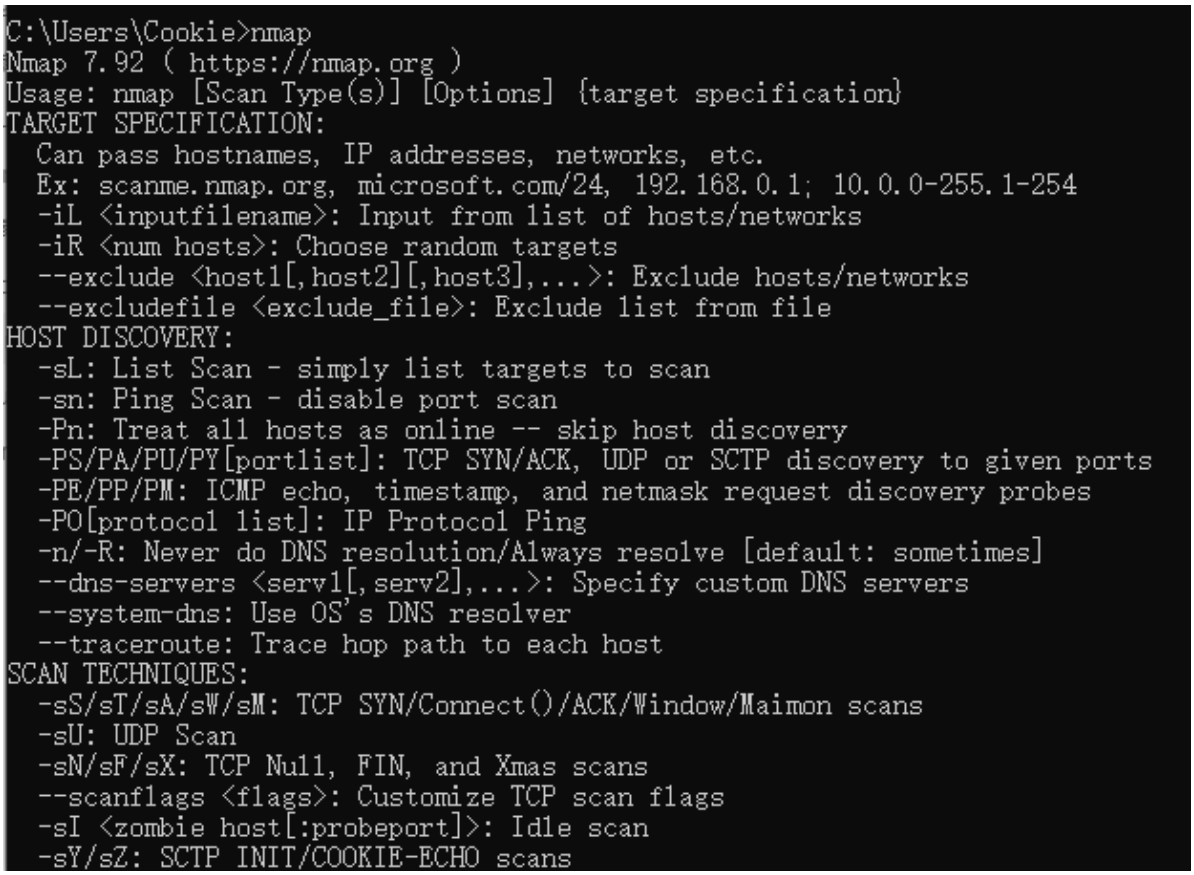
Nmap

选项概要

当Nmap不带选项运行时，选项概要会被输出：



它可以帮助我们记住常用的选项，一些不常用的选项不在这里列出。

```
Usage: nmap [Scan Type(s)] [Options] {target specification}
TARGET SPECIFICATION:
Can pass hostnames, IP addresses, networks, etc.
Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0-255.0-255.1-254
-iL <inputfilename>: Input from list of hosts/networks
-iR <num hosts>: Choose random targets
--exclude <host1[,host2][,host3],...>: Exclude hosts/networks
--excludefile <exclude_file>: Exclude list from file
HOST DISCOVERY:
-sL: List Scan - simply list targets to scan
-sP: Ping Scan - go no further than determining if host is online
-P0: Treat all hosts as online -- skip host discovery
-PS/PA/PU [portlist]: TCP SYN/ACK or UDP discovery probes to given ports
-PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes
-n/-R: Never do DNS resolution/Always resolve [default: sometimes resolve]
SCAN TECHNIQUES:
-sS/sT/sA/sW/sM: TCP SYN/Connect()/ACK/Window/Maimon scans
-sN/sF/sX: TCP Null, FIN, and Xmas scans
--scanflags <flags>: Customize TCP scan flags
-sI <zombie host[:probeport]>: Idlescan
```

```
-sO: IP protocol scan
-b <ftp relay host>: FTP bounce scan
PORT SPECIFICATION AND SCAN ORDER:
-p <port ranges>: Only scan specified ports
Ex: -p22; -p1-65535; -p U:53,111,137,T:21-25,80,139,8080
-F: Fast - Scan only the ports listed in the nmap-services file)
-r: Scan ports consecutively - don't randomize
SERVICE/VERSION DETECTION:
-sV: Probe open ports to determine service/version info
--version-light: Limit to most likely probes for faster identification
--version-all: Try every single probe for version detection
--version-trace: Show detailed version scan activity (for debugging)
OS DETECTION:
-O: Enable OS detection
--osscan-limit: Limit OS detection to promising targets
--osscan-guess: Guess OS more aggressively
TIMING AND PERFORMANCE:
-T[0-6]: Set timing template (higher is faster)
--min-hostgroup/max-hostgroup <msec>: Parallel host scan group sizes
--min-parallelism/max-parallelism <msec>: Probe parallelization
--min-rtt-timeout/max-rtt-timeout/initial-rtt-timeout <msec>: Specifies
probe round trip time.
--host-timeout <msec>: Give up on target after this long
--scan-delay/--max-scan-delay <msec>: Adjust delay between probes
FIREWALL/IDS EVASION AND SPOOFING:
-f; --mtu <val>: fragment packets (optionally w/given MTU)
-D <decoy1,decoy2[,ME],...>: Cloak a scan with decoys
-S <IP_Address>: Spoof source address
-e <iface>: Use specified interface
-g/--source-port <portnum>: Use given port number
--data-length <num>: Append random data to sent packets
--ttl <val>: Set IP time-to-live field
--spoof-mac <mac address, prefix, or vendor name>: Spoof your MAC address
OUTPUT:
-oN/-oX/-oS/-oG <file>: Output scan results in normal, XML, s|<rIpt kIddi3,
and Grepable format, respectively, to the given filename.
-oA <basename>: Output in the three major formats at once
-v: Increase verbosity level (use twice for more effect)
-d[level]: Set or increase debugging level (Up to 9 is meaningful)
--packet-trace: Show all packets sent and received
--iflist: Print host interfaces and routes (for debugging)
--append-output: Append to rather than clobber specified output files
--resume <filename>: Resume an aborted scan
--stylesheet <path/URL>: XSL stylesheet to transform XML output to HTML
--no-stylesheet: Prevent Nmap from associating XSL stylesheet w/XML output
MISC:
-6: Enable IPv6 scanning
-A: Enables OS detection and Version detection
--datadir <dirname>: Specify custom Nmap data file location
--send-eth/--send-ip: Send packets using raw ethernet frames or IP packets
--privileged: Assume that the user is fully privileged
-V: Print version number
-h: Print this help summary page.
EXAMPLES:
nmap -v -A scanme.nmap.org
```

```
nmap -v -sP 192.168.0.0/16 10.0.0.0/8
nmap -v -iR 10000 -P0 -p 80
```

Nmap (“Network Mapper<网络映射器>”) 是一款开放源代码的网络探测和安全审核的工具。它的设计目标是快速扫描大型网络，当然用它扫描单个主机也没有问题。Nmap以新颖的方式使用原始IP报文来发现网络上有哪些主机，这些主机提供什么服务(应用程序名和版本)，这些服务运行在什么操作系统(包括版本信息)， 它们使用什么类型的报文过滤器/防火墙，以及一堆其它功能。虽然Nmap通常用于安全审核，许多系统管理员和网络管理员也用它来做一些日常的工作，比如查看整个网络的信息， 管理服务升级计划，以及监视主机和服务的运行。

Nmap输出的是扫描目标的列表，以及每个目标的补充信息，至于是哪些信息则依赖于所使用的选项，通常会列出端口号，协议，服务名称和状态。

端口状态：
open（开放的）：意味着目标机器上的应用程序正在该端口监听连接/报文。
filtered（被过滤的） ：意味着防火墙、过滤器或者其它网络障碍阻止了该端口被访问，Nmap无法得知它是open还是closed。
closed（关闭的）：端口没有应用程序在它上面监听，但是它们随时可能开放。
unfiltered（未被过滤的）：当端口对Nmap的探测做出响应，但是Nmap无法确定它们是关闭还是开放时，这些端口就被认为是unfiltered。
open|filtered 和 closed|filtered ：状态组合，即Nmap无法确定该端口处于两个状态中的哪一个状态。

除了端口表以外，Nmap还能提供关于目标机器的进一步信息，包括反向域名、操作系统猜测、设备类型和MAC地址。

一个典型的Nmap扫描如下方所示。在这个例子中，选项`-A`， 用来进行操作系统及其版本的探测，`-T4`可以加快执行速度，接着是目标主机名。

例1. 一个典型的Nmap扫描

`nmap -A -T4 scanme.nmap.org`

```
# nmap -A -T4 scanme.nmap.org

Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.16s latency).
Other addresses for scanme.nmap.org (not scanned): 45.33.32.156
Not shown: 985 closed tcp ports (reset)
PORT      STATE    SERVICE        VERSION
9/tcp     filtered discard
22/tcp    open     ssh            OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu
Linux; protocol 2.0)
| ssh-hostkey:
|   256 96:02:bb:5e:57:54:1c:4e:45:2f:56:4c:4a:24:b2:57 (ECDSA)
|_  256 33:fa:91:0f:e0:e1:7b:1f:6d:05:a2:b0:f1:54:41:56 (ED25519)
42/tcp    filtered nameserver
80/tcp    open     http           Apache httpd 2.4.7 ((Ubuntu))
|_http-title: Go ahead and ScanMe!
|_http-favicon: Nmap Project
|_http-server-header: Apache/2.4.7 (Ubuntu)
135/tcp   filtered msrpc
139/tcp   filtered netbios-ssn
```

```
445/tcp   filtered microsoft-ds
593/tcp   filtered http-rpc-epmap
1068/tcp  filtered instl_bootc
1434/tcp  filtered ms-sql-m
3128/tcp  filtered squid-http
4444/tcp  filtered krb524
6669/tcp  filtered irc
9929/tcp  open     nping-echo     Nping echo
31337/tcp open     tcpwrapped
Aggressive OS guesses: Linux 3.18 (89%), IPCop 2.0 (Linux 2.6.32) (87%), Linux
2.6.32 (87%), Linux 2.6.32 or 3.10 (87%), Linux 3.11 - 4.1 (87%), Linux 3.16
(87%), Linux 3.2 - 3.8 (87%), Linux 3.3 (87%), Linux 3.5 (87%), Linux 3.8 (87%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 16 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 1720/tcp)
HOP RTT       ADDRESS
1   1.00 ms   192.168.0.1
2   1.00 ms   192.168.1.1
3   3.00 ms   100.101.0.1
4   ...
5   5.00 ms   202.102.217.81
6   17.00 ms  202.97.110.65
7   ... 8
9   147.00 ms 202.97.50.114
10  136.00 ms 218.30.54.245
11  142.00 ms 64.86.160.4
12  145.00 ms 207.45.208.13
13  144.00 ms a23-203-158-53.deploy.static.akamaitechnologies.com
(23.203.158.53)
14  153.00 ms if-0-0-2-997.gw2.fnc1.us.linode.com (213.52.131.188)
15  176.00 ms ae1.r11.sjc01.ien.netarch.akamai.com (23.207.232.35)
16  156.00 ms scanme.nmap.org (45.33.32.156)

OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 70.62 seconds
```

目标说明

除了选项，所有出现在Nmap命令行上的都被视为对目标主机的说明。 最简单的情况是指定一个目标IP地址或主机名。

有时候我们希望扫描整个网络的相邻主机。为此，Nmap支持**CIDR风格**的地址。可以附加一个/在一个IP地址或主机名后面， Nmap将会扫描所有和该参考IP地址具有 相同比特的所有IP地址或主机。 **例如，192.168.10.0/24将会扫描192.168.10.0 (二进制格式: 11000000 10101000 00001010 00000000)和192.168.10.255 (二进制格式: 11000000 10101000 00001010 11111111)之间的256台主机。**
192.168.10.40/24 代表同样的意义。
假设主机scanme.nmap.org的IP地址是205.217.153.62，scanme.nmap.org/16 将扫描205.217.0.0和205.217.255.255之间的65,536 个IP地址。 所允许的最小值是/1， 这将会扫描半个互联网。最大值是/32，这将会扫描该主机或IP地址， 因为所有的比特都固定了。

CIDR标志位很简洁但有时候不够灵活。例如，想要扫描 192.168.0.0/16，但希望略过任何以.0或者.255结束的IP地址，因为它们通常是广播地址。Nmap通过八位字节地址范围支持这样的扫描，可以用逗号分开的数字或范围列表为IP地址的每个八位字节指定它的范围。例如，192.168.0-255.1-254 将略过在该范围内以.0和.255结束的地址。范围不必限于最后的8位：0-255.0-255.13.37 将在整个互联网范围内扫描所有以13.37结束的地址。这种大范围的扫描对互联网调查研究或许有用。

IPv6地址只能用规范的IPv6地址或主机名指定。 CIDR 和八位字节范围不支持IPv6，因为它们对于IPv6几乎没什么用。

Nmap命令行接受多个主机说明，它们不必是相同类型，如：nmap scanme.nmap.org 192.168.0.0/8 10.0.0.1。

虽然目标通常在命令行指定，下列选项也可用来控制目标的选择：

- **-iL（从列表中输入）**

  从 中读取目标说明。在命令行输入一堆主机名显得很笨拙，然而经常需要这样。 例如，您的DHCP服务器可能导出10,000个当前租约的列表，而您希望对它们进行扫描。如果您不是使用未授权的静态IP来定位主机，或许您想要扫描所有IP地址。 只要生成要扫描的主机的列表，用-iL 把文件名作为选项传给Nmap。列表中的项可以是Nmap在命令行上接受的任何格式(IP地址，主机名，CIDR，IPv6，或者八位字节范围)。 每一项必须以一个或多个空格，制表符或换行符分开。 如果您希望Nmap从标准输入而不是实际文件读取列表， 您可以用一个连字符(-)作为文件名。

- **-iR（随机选择目标）**

  对于互联网范围内的调查和研究， 您也许想随机地选择目标。 选项告诉 Nmap生成多少个IP。不合需要的IP如特定的私有，组播或者未分配的地址自动略过。选项 0 意味着永无休止的扫描。记住，一些网管对于未授权的扫描可能会很感冒并加以抱怨。 使用该选项的后果自负! 如果在某个雨天的下午，您觉得实在无聊， 试试这个命令**nmap -sS -PS80 -iR 0 -p 80**随机地找一些网站浏览。

- **--exclude <host1,host2,host3，...> (排除主机/网络)**

  如果在指定的扫描范围有一些主机或网络不是想要扫描的目标， 那就用该选项加上以逗号分隔的列表排除它们。该列表用正常的Nmap语法， 因此它可以包括主机名，CIDR，八位字节范围等等。当希望扫描的网络包含执行关键任务的服务器，已知的对端口扫描反应强烈的系统或者被其他人看管的子网时，这也许有用。

- **--excludefile（排除文件中的列表）**

  这和--exclude 选项的功能一样，只是所排除的目标是用以换行符，空格，或者制表符分隔的 提供的，而不是在命令行上输入的。

主机发现（思考一下可以用来做什么？）

任何网络探测任务的最初几个步骤之一就是把一组IP范围（有时该范围是巨大的）缩小为一列活动的或者感兴趣的主机。扫描每个IP的每个端口很慢，通常也没必要。 当然，什么样的主机令您感兴趣主要依赖于扫描的目的。网管也许只对运行特定服务的主机感兴趣，而从事安全的人员则可能对一个主机都感兴趣，只要它有IP地址:-)。一个系统管理员也许仅仅使用Ping来定位内网上的主机，而一个外部入侵测试人员则可能绞尽脑汁用各种方法试图突破防火墙的封锁。

由于主机发现的需求五花八门，Nmap提供了一箩筐的选项来定制需求。主机发现有时候也叫做ping扫描，但它远远超越世人皆知的ping工具发送简单的ICMP回声请求报文。用户完全可以通过使用列表扫描(-sL)或者通过关闭ping (-P0)跳过ping的步骤，也可以使用多个端口把TCP SYN/ACK，UDP和ICMP 任意组合起来玩一玩。这些探测的目的是获得响应以显示某个IP地址是否是活动的(正在被某主机或者网络设备使用)。**在许多网络上，在给定的时间，往往只有小部分的IP地址是活动的。这种情况在私有地址空间**

如10.0.0.0/8尤其普遍。私有网络有16,000,000个IP，但一些使用它的公司连1000台机器都没有。主机发现能够找到零星分布于IP地址海洋上的机器。

如果没有给出主机发现的选项，Nmap 就发送一个TCP ACK报文到80端口和一个ICMP回声请求到每台目标机器。 一个例外是ARP扫描用于局域网上的任何目标机器。对于非特权UNIX shell用户，使用connect()系统调用会发送一个SYN报文而不是ACK。这些默认行为和使用-PA -PE选项的效果相同。 扫描局域网时，这种主机发现一般够用了，但是对于安全审核，建议进行更加全面的探测。

-P选项（用于选择ping的类型）可以被结合使用，可以通过使用不同的TCP端口/标志位和ICMP码发送许多探测报文来增加穿透防守严密的防火墙的机会。另外要注意的是即使指定了其它 -P*选项，ARP发现(-PR)对于局域网上的目标而言是默认行为，因为它总是更快更有效。

下列选项控制主机发现。

- **-sL (列表扫描)**

  列表扫描是主机发现的退化形式，**它仅仅列出指定网络上的每台主机，不发送任何报文到目标主机，也并不能确定主机是否存活，所以扫描速度是很快的。默认情况下，Nmap会对主机进行反向域名解析以获取它们的名字。**例如，fw.chi.playboy.com是花花公子芝加哥办公室的防火墙。Nmap最后还会报告IP地址的总数。列表扫描可以很好的确保您拥有正确的目标IP。如果主机的域名出乎您的意料，那么就值得进一步检查以防错误地扫描其它组织的网络。既然只是打印目标主机的列表，像其它一些高级功能如端口扫描，操作系统探测或者Ping扫描的选项就没有了。

- **-sP (Ping扫描)**

  该选项告诉Nmap仅进行ping扫描(主机发现)，然后输出对扫描做出响应的那些主机，没有进一步的测试 (如端口扫描或者操作系统探测)。 这比列表扫描更积极，常常用于和列表扫描相同的目的。它可以得到些许目标网络的信息而不被特别注意到。 **对于攻击者来说，了解多少主机正在运行比列表扫描提供的一列IP和主机名往往更有价值。**系统管理员往往也很喜欢这个选项。 它可以很方便地得出网络上有多少机器正在运行或者监视服务器是否正常运行。常常有人称它为地毯式ping，它比ping广播地址更可靠，因为许多主机对广播请求不响应。-sP选项在默认情况下， 发送一个ICMP回声请求和一个TCP报文到80端口。如果非特权用户执行，就发送一个SYN报文 (用connect()系统调用)到目标机的80端口。 当特权用户扫描局域网上的目标机时，会发送ARP请求(-PR)，除非使用了--send-ip选项。 -sP选项可以和除-P0之外的任何发现探测类型-P* 选项结合使用以达到更大的灵活性。 一旦使用了任何探测类型和端口选项，默认的探测(ACK和回应请求)就被覆盖了。 **当防守严密的防火墙位于运行Nmap的源主机和目标网络之间时， 推荐使用那些高级选项。否则，当防火墙捕获并丢弃探测包或者响应包时，一些主机就不能被探测到。**

- **-P0 (无ping)**

  该选项完全跳过Nmap发现阶段。 通常Nmap在进行高强度的扫描时用它确定正在运行的机器。 **默认情况下，Nmap只对正在运行的主机进行高强度的探测如端口扫描，版本探测，或者操作系统探测。用-P0禁止主机发现会使Nmap对每一个指定的目标IP地址进行所要求的扫描。**所以如果在命令行指定一个B类目标地址空间(/16)， 所有 65,536 个IP地址都会被扫描。 -P0的第二个字符是数字0而不是字母O。 **和列表扫描一样，跳过正常的主机发现，但不是打印一个目标列表， 而是继续执行所要求的功能，就好像每个IP都是活动的。**

- -PS [portlist] (TCP SYN Ping)

  该选项发送一个设置了SYN标志位的空TCP报文。 默认目的端口为80 (可以通过改变nmap.h) 文件中的DEFAULT-TCP-PROBE-PORT值进行配置，但不同的端口也可以作为选项指定。 甚至可以指定一个以逗号分隔的端口列表(如 -PS22，23，25，80，113，1050，35000)， 在这种情况下，每个端口会被并发地扫描。SYN标志位告诉对方您正试图建立一个连接。 通常目标端口是关闭的，一个RST (复位) 包会发回来。 如果碰巧端口是开放的，目标会进行TCP三步握手的第二步，回应一个SYN/ACK TCP报文。然后运行Nmap的机器则会扼杀这个正在建立的连接， 发送一个RST而非ACK报文，否则，一个完全的连接将会建立。 RST报文是运行Nmap的机器而不是Nmap本身响应的，因为它对收到的SYN/ACK感到很意外。Nmap并不关心端口开放还是关闭。 无论RST还是SYN/ACK

响应都告诉Nmap该主机正在运行。在UNIX机器上，通常只有特权用户 root 能否发送和接收原始的TCP报文。因此作为一个变通的方法，对于非特权用户， Nmap会为每个目标主机进行系统调用connect()，它也会发送一个SYN 报文来尝试建立连接。如果connect()迅速返回成功或者一个ECONNREFUSED 失败，下面的TCP堆栈一定已经收到了一个SYN/ACK或者RST，该主机将被标志位为在运行。 如果连接超时了，该主机就标志位为down掉了。这种方法也用于IPv6 连接，因为Nmap目前还不支持原始的IPv6报文。

- -PA [portlist] (TCP ACK Ping)

  TCP ACK ping和刚才讨论的SYN ping相当类似，区别就是设置TCP的ACK标志位而不是SYN标志位。ACK报文表示确认一个建立连接的尝试，但该连接尚未完全建立。所以远程主机应该总是回应一个RST报文， 因为它们并没有发出过连接请求到运行Nmap的机器，如果它们正在运行的话。-PA选项使用和SYN探测相同的默认端口(80)，也可以用相同的格式指定目标端口列表。如果非特权用户尝试该功能，或者指定的是IPv6目标，前面说过的connect()方法将被使用。这个方法并不完美，因为它实际上发送的是SYN报文，而不是ACK报文。提供SYN和ACK两种ping探测的原因是使通过防火墙的机会尽可能大。许多管理员会配置他们的路由器或者其它简单的防火墙来封锁SYN报文，除非连接目标是那些公开的服务器像公司网站或者邮件服务器。这可以阻止其它进入组织的连接，同时也允许用户访问互联网。这种无状态的方法几乎不占用防火墙/路由器的资源，因而被硬件和软件过滤器广泛支持。Linux Netfilter/iptables 防火墙软件提供方便的 --syn选项来实现这种无状态的方法。当这样的无状态防火墙规则存在时，发送到关闭目标端口的SYN ping探测 (-PS) 很可能被封锁。这种情况下，ACK探测格外有闪光点，因为它正好利用了这样的规则。另外一种常用的防火墙用有状态的规则来封锁非预期的报文。这一特性已开始只存在于高端防火墙，但是这些年类它越来越普遍了。Linux Netfilter/iptables 通过 --state选项支持这一特性，它根据连接状态把报文进行分类。SYN探测更有可能用于这样的系统，由于没头没脑的ACK报文通常会被识别成伪造的而丢弃。解决这个两难的方法是通过即指定 -PS又指定-PA来即发送SYN又发送ACK。

- -PU [portlist] (UDP Ping)

  还有一个主机发现的选项是UDP ping，它发送一个空的(除非指定了--data-length UDP报文到给定的端口。端口列表的格式和前面讨论过的-PS和-PA选项还是一样。如果不指定端口，默认是31338。该默认值可以通过在编译时改变nmap.h文件中的 DEFAULT-UDP-PROBE-PORT值进行配置。默认使用这样一个奇怪的端口是因为对开放端口 进行这种扫描一般都不受欢迎。如果目标机器的端口是关闭的，UDP探测应该马上得到一个ICMP端口无法到达的回应报文。 这对于Nmap意味着该机器正在运行。 许多其它类型的ICMP错误，像主机/网络无法到达或者TTL超时则表示down掉的或者不可到达的主机。 没有回应也被这样解释。如果到达一个开放的端口，大部分服务仅仅忽略这个空报文而不做任何回应。这就是为什么默认探测端口是31338这样一个极不可能被使用的端口。少数服务如chargen会响应一个空的UDP报文，从而向Nmap表明该机器正在运行。该扫描类型的主要优势是它可以穿越只过滤TCP的防火墙和过滤器。 例如。我曾经有过一个Linksys BEFW11S4无线宽带路由器。默认情况下， 该设备对外的网卡过滤所有TCP端口，但UDP探测仍然会引发一个端口不可到达 的消息，从而暴露了它自己。

- -PE; -PP; -PM (ICMP Ping Types)

  除了前面讨论的这些不常见的TCP和UDP主机发现类型，Nmap也能发送世人皆知的ping程序所发送的报文。Nmap发送一个ICMP type 8 (回声请求)报文到目标IP地址，期待从运行的主机得到一个type 0 (回声响应)报文。对于网络探索者而言，不幸的是，许多主机和防火墙现在封锁这些报文，而不是按期望的那样响应。因此，仅仅ICMP扫描对于互联网上的目标通常是不够的。 但对于系统管理员监视一个内部网络，它们可能是实际有效的途径。 使用-PE选项打开该回声请求功能。虽然回声请求是标准的ICMP ping查询， Nmap并不止于此。ICMP标准 (RFC 792)还规范了时间戳请求，信息请求 request，和地址掩码请求，它们的代码分别是13，15和17。 虽然这些查询的表面目的是获取信息如地址掩码和当前时间，它们也可以很容易地用于主机发现。很简单，回应的系统就是在运行的系统。Nmap目前没有实现信息请求报文， 因为它们还没有被广泛支持。RFC 1122坚持 “主机不应该实现这些消息”。 时间戳和地址掩码查询可以分别用-PP和-PM选项发送。 时间戳

响应(ICMP代码14)或者地址掩码响应(代码18)表示主机在运行。 当管理员特别封锁了回声请求报文而忘了其它ICMP查询可能用于相同目的时，这两个查询可能很有价值。

- -PR (ARP Ping)

  最常见的Nmap使用场景之一是扫描一个以太局域网。 在大部分局域网上，特别是那些使用基于RFC1918私有地址范围的网络，在一个给定的时间绝大部分 IP地址都是不使用的。 当Nmap试图发送一个原始IP报文如ICMP回声请求时，操作系统必须确定对应于目标IP的硬件 地址(ARP)，这样它才能把以太帧送往正确的地址。这一般比较慢而且会有些问题，因为操作系统设计者认为一般不会在短时间内对没有运行的机器作几百万次的ARP请求。当进行ARP扫描时，Nmap用它优化的算法管理ARP请求。当它收到响应时，Nmap甚至不需要担心基于IP的ping报文，既然它已经知道该主机正在运行了。 这使得ARP扫描比基于IP的扫描更快更可靠。所以默认情况下，如果Nmap发现目标主机就在它所在的局域网上，它会进行ARP扫描。即使指定了不同的ping类型(如 -PI或者 -PS)，Nmap也会对任何相同局域网上的目标机使用ARP。 如果您真的不想要ARP扫描，指定 --send-ip。

- -n (不用域名解析)

  告诉Nmap永不对它发现的活动IP地址进行反向域名解析。既然DNS一般比较慢，这可以让事情更快些。

- -R (为所有目标解析域名)

  告诉Nmap永远对目标IP地址作反向域名解析。一般只有当发现机器正在运行时才进行这项操作。

- --system-dns (使用系统域名解析器)

  默认情况下，Nmap通过直接发送查询到您的主机上配置的域名服务器来解析域名。为了提高性能，许多请求 (一般几十个) 并发执行。如果希望使用系统自带的解析器，就指定该选项 (通过getnameinfo()调用一次解析一个IP)。

端口扫描基础

虽然Nmap这些年来功能越来越多，但它最初也是从一个端口扫描器开始的，并且这仍然是它的核心功能。 nmap **这个简单的命令会扫描主机上的超过1660个TCP端口。许多传统的端口扫描器只列出所有端口是开放还是关闭的，Nmap的信息粒度比它们要细得多。它把端口分成六个状态: open(开放的)，closed(关闭的)，filtered(被过滤的)， unfiltered(未被过滤的)， open|filtered(开放或者被过滤的)，或者 closed|filtered(关闭或者被过滤的)。

这些状态并非端口本身的性质，而是描述Nmap怎样看待它们。例如，对于同样的目标机器的135/tcp端口，从同网络扫描显示它是开放的，而跨网络进行完全相同的扫描则可能显示它是filtered(被过滤的)。

Nmap所识别的6个端口状态。

- open(开放的)

  应用程序正在该端口接收TCP连接或者UDP报文。发现这一点常常是端口扫描的主要目标。安全意识强的人们知道每个开放的端口都是攻击的入口。攻击者或者入侵测试者想要发现开放的端口。 而管理员则试图关闭它们或者用防火墙保护它们以免妨碍了合法用户。 非安全扫描可能对开放的端口也感兴趣，因为它们显示了网络上那些服务可供使用。

- closed(关闭的)

  关闭的端口对于Nmap也是可访问的(它接受Nmap的探测报文并作出响应)，但没有应用程序在其上监听。它们可以显示该IP地址上(主机发现，或者ping扫描)的主机正在运行，也对部分操作系统探测有所帮助。因为关闭的关口是可访问的，也许过会儿值得再扫描一下，可能一些又开放了。系统管理员可能会考虑用防火墙封锁这样的端口，那样的话就会被显示为被过滤的状态，下面讨论。

- filtered(被过滤的)

由于包过滤阻止探测报文到达端口，Nmap无法确定该端口是否开放。过滤可能来自专业的防火墙设备，路由器规则或者主机上的软件防火墙。这样的端口让攻击者感觉很挫折，因为它们几乎不提供任何信息。有时候它们响应ICMP错误消息如类型3代码13 (无法到达目标：通信被管理员禁止)，但更普遍的是过滤器只是丢弃探测帧，不做任何响应。这迫使Nmap重试若干次以防探测包是由于网络阻塞丢弃的，会使得扫描速度明显变慢。

- unfiltered(未被过滤的)

  未被过滤状态意味着端口可访问，但Nmap不能确定它是开放还是关闭。 只有用于映射防火墙规则集的ACK扫描才会把端口分类到这种状态。用其它类型的扫描如窗口扫描，SYN扫描，或者FIN扫描来扫描未被过滤的端口可以帮助确定端口是否开放。

- open|filtered(开放或者被过滤的)

  当无法确定端口是开放还是被过滤的，Nmap就把该端口划分成这种状态。开放的端口不响应就是一个例子。没有响应也可能意味着报文过滤器丢弃了探测报文或者它引发的任何响应。因此Nmap无法确定该端口是开放的还是被过滤的。UDP，IP协议， FIN，Null，和Xmas扫描可能把端口归入此类。

- closed|filtered(关闭或者被过滤的)

  该状态用于Nmap不能确定端口是关闭的还是被过滤的，它只可能出现在IPID Idle扫描中。

端口扫描技术

大部分扫描类型只对特权用户可用，这是因为他们发送接收原始报文，在Unix系统需要root权限。在Windows上推荐使用administrator账户，但是当WinPcap已经被加载到操作系统时，非特权用户也可以正常使用Nmap。当Nmap在1997年发布时，需要root权限是一个严重的局限，因为很多用户只有共享的shell账户。现在，世界变了，计算机便宜了，更多人拥有互联网连接 ，桌面UNIX系统 (包括Linux和MAC OS X)很普遍了。Windows版本的Nmap现在也有了，这使它可以运行在更多的桌面上。 由于所有这些原因，用户不再需要用有限的共享shell账户运行Nmap。 这是很幸运的，因为特权选项让Nmap强大得多也灵活得多。

虽然Nmap努力产生正确的结果，但请记住所有结果都是基于目标机器(或者它们前面的防火墙)返回的报文的。这些主机也许是不值得信任的，它们可能响应以迷惑或误导Nmap的报文。 更普遍的是非RFC兼容的主机以不正确的方式响应Nmap探测，FIN，Null和Xmas扫描特别容易遇到这个问题，这些是特定扫描类型的问题。

这一节讨论Nmap支持的大约十几种扫描技术。一般一次只用一种方法，除了UDP扫描(-sU)可能和任何一种TCP扫描类型结合使用。友情提示一下，端口扫描类型的选项格式是-s， 其中是个显眼的字符，通常是第一个字符，一个例外是deprecated FTP bounce扫描(-b)。默认情况下，Nmap执行一个SYN扫描，但是如果用户没有权限发送原始报文(在UNIX上需要root权限)或者如果指定的是IPv6目标，Nmap调用connect()。本节列出的扫描中，非特权用户只能执行connect()和ftp bounce扫描。

- **-sS (TCP SYN扫描)**

  SYN扫描作为默认的也是最受欢迎的扫描选项，是有充分理由的。**它执行得很快，在一个没有入侵防火墙的快速网络上，每秒钟可以扫描数千个端口。SYN扫描相对来说不张扬，不易被注意到，因为它从来不完成TCP连接。**它也不像Fin/Null/Xmas，Maimon和Idle扫描依赖于特定平台，而可以应对任何兼容的 TCP协议栈。它还可以明确可靠的区分open(开放的)， closed(关闭的)，和filtered(被过滤的) 状态，常常被称为半连接扫描， 因为它不打开一个完全的TCP连接。它发送一个SYN报文，就像您真的要打开一个连接，然后等待响应。 SYN/ACK表示端口在监听 (开放)，而RST (复位)表示没有监听者。如果数次重发后仍没响应，该端口就被标记为被过滤。如果收到ICMP不可到达错误 (类型3，代码1，2，3，9，10，或者13)，该端口也被标记为被过滤。

- **-sT (TCP connect()扫描)**

当SYN扫描不能用时，TCP Connect()扫描就是**默认的TCP扫描**。当用户没有权限发送原始报文或者扫描IPv6网络时，就是这种情况。 Instead of writing raw packets as most other scan types do，Nmap通过创建connect() 系统调用要求操作系统和目标机以及端口建立连接，而不像其它扫描类型直接发送原始报文。 这是和Web浏览器，P2P客户端以及大多数其它网络应用程序用以建立连接一样的高层系统调用。它是叫做Berkeley Sockets API编程接口的一部分。Nmap用该API获得每个连接尝试的状态信息，而不是读取响应的原始报文。当SYN扫描可用时，它通常是更好的选择。因为Nmap对高层的connect()调用比对原始报文控制更少， 所以前者效率较低。 该系统调用完全连接到开放的目标端口而不是像SYN扫描进行半开放的复位。这不仅花更长时间，需要更多报文得到同样信息，目标机也更可能记录下连接。IDS(入侵检测系统)可以捕获两者，但大部分机器没有这样的警报系统。当Nmap连接，然后不发送数据又关闭连接，许多普通UNIX系统上的服务会在syslog留下记录，有时候是一条加密的错误消息。 此时，有些真正脆弱的服务会崩溃，虽然这不常发生。如果管理员在日志里看到来自同一系统的一堆连接尝试，他应该知道系统被扫描了。

- **-sU (UDP扫描)**

  虽然互联网上很多流行的服务运行在TCP协议上，UDP服务也不少。 DNS，SNMP，和DHCP (注册的端口是53，161/162，和67/68)是最常见的三个。 因为UDP扫描一般较慢，比TCP更困难，一些安全审核人员忽略这些端口。 这是一个错误，因为可探测的UDP服务相当普遍，攻击者当然不会忽略整个协议。 所幸，Nmap可以帮助记录并报告UDP端口。UDP扫描用-sU选项激活。它可以和TCP扫描如 SYN扫描 (-sS)结合使用来同时检查两种协议。UDP扫描发送空的(没有数据)UDP报头到每个目标端口。 如果返回ICMP端口不可到达错误(类型3，代码3)， 该端口是closed(关闭的)。 其它ICMP不可到达错误(类型3， 代码1，2，9，10，或者13)表明该端口是filtered(被过滤的)。 偶尔地，某服务会响应一个UDP报文，证明该端口是open(开放的)。 如果几次重试后还没有响应，该端口就被认为是 open|filtered(开放|被过滤的)。 这意味着该端口可能是开放的，也可能包过滤器正在封锁通信。 可以用版本扫描(-sV)帮助区分真正的开放端口和被过滤的端口。UDP扫描的巨大挑战是怎样使它更快速。 开放的和被过滤的端口很少响应，让Nmap超时然后再探测，以防探测帧或者响应丢失。关闭的端口常常是更大的问题。 它们一般发回一个ICMP端口无法到达错误。但是不像关闭的TCP端口响应SYN或者Connect 扫描所发送的RST报文，许多主机在默认情况下限制ICMP端口不可到达消息。 Linux和Solaris对此特别严格。例如， Linux 2.4.20内核限制一秒钟只发送一条目标不可到达消息 (见net/ipv4/icmp。c)。Nmap探测速率限制并相应地减慢来避免用那些目标机会丢弃的无用报文来阻塞 网络。不幸的是，Linux式的一秒钟一个报文的限制使65,536个端口的扫描要花 18小时以上。加速UDP扫描的方法包括并发扫描更多的主机，先只对主要端口进行快速 扫描，从防火墙后面扫描，使用--host-timeout跳过慢速的 主机。

- -sN; -sF; -sX (TCP Null，FIN，and Xmas扫描)

  这三种扫描类型 (甚至用下一节描述的 --scanflags 选项的更多类型) 在TCP RFC 中发掘了一个微妙的方法来区分open(开放的)和 closed(关闭的)端口。第65页说"如果 [目标]端口状态是关闭的.... 进入的不含RST的报文导致一个RST响应。" 接下来的一页 讨论不设置SYN，RST，或者ACK位的报文发送到开放端口: "理论上，这不应该发生，如果您确实收到了，丢弃该报文，返回。 "如果扫描系统遵循该RFC，当端口关闭时，任何不包含SYN，RST，或者ACK位的报文会导致 一个RST返回，而当端口开放时，应该没有任何响应。只要不包含SYN，RST，或者ACK， 任何其它三种(FIN，PSH，and URG)的组合都行。Nmap有三种扫描类型利用这一点：Null扫描 (-sN)不设置任何标志位(tcp标志头是0)FIN扫描 (-sF)只设置TCP FIN标志位。Xmas扫描 (-sX)设置FIN，PSH，和URG标志位，就像点亮圣诞树上所有的灯一样。除了探测报文的标志位不同，这三种扫描在行为上完全一致。 如果收到一个RST报文，该端口被认为是 closed(关闭的)，而没有响应则意味着 端口是open|filtered(开放或者被过滤的)。 如果收到ICMP不可到达错误(类型 3，代号 1，2，3，9，10，或者13)，该端口就被标记为 被过滤的。这些扫描的关键优势是它们能躲过一些无状态防火墙和报文过滤路由器。 另一个优势是这些扫描类型甚至比SYN扫描还要隐秘一些。但是别依赖它 -- 多数 现代的IDS产品可以发现它们。一个很大的不足是并非所有系统都严格遵循RFC 793。 许多系统不管端口开放还是关闭，都响应RST。 这导致所有端口都标记为closed(关闭的)。 这样的操作系统主要有Microsoft Windows，许多Cisco设备，BSDI，以及IBM OS/400。 但是这种扫描对多数

UNIX系统都能工作。这些扫描的另一个不足是 它们不能辨别open(开放的)端口和一些特定的filtered(被过滤的)端口，从而返回 open|filtered(开放或者被过滤的)。

- -sA (TCP ACK扫描)

  这种扫描与目前为止讨论的其它扫描的不同之处在于它不能确定open(开放的)或者open|filtered(开放或者过滤的))端口。 它用于发现防火墙规则，确定它们是有状态的还是无状态的，哪些端口是被过滤的。ACK扫描探测报文只设置ACK标志位(除非您使用 --scanflags)。当扫描未被过滤的系统时， open(开放的)和closed(关闭的) 端口都会返回RST报文。Nmap把它们标记为unfiltered(未被过滤的)，意思是 ACK报文不能到达，但至于它们是open(开放的)或者 closed(关闭的) 无法确定。不响应的端口 或者发送特定的ICMP错误消息(类型3，代号1，2，3，9，10， 或者13)的端口，标记为 filtered(被过滤的)。

- -sW (TCP窗口扫描)

  除了利用特定系统的实现细节来区分开放端口和关闭端口，当收到RST时不总是打印unfiltered，窗口扫描和ACK扫描完全一样。 它通过检查返回的RST报文的TCP窗口域做到这一点。 在某些系统上，开放端口用正数表示窗口大小(甚至对于RST报文) 而关闭端口的窗口大小为0。因此，当收到RST时，窗口扫描不总是把端口标记为 unfiltered， 而是根据TCP窗口值是正数还是0，分别把端口标记为open或者 closed该扫描依赖于互联网上少数系统的实现细节， 因此您不能永远相信它。不支持它的系统会通常返回所有端口closed。 当然，一台机器没有开放端口也是有可能的。 如果大部分被扫描的端口是 closed，而一些常见的端口 (如 22， 25，53) 是 filtered，该系统就非常可疑了。 偶尔地，系统甚至会显示恰恰相反的行为。 如果您的扫描显示1000个开放的端口和3个关闭的或者被过滤的端口， 那么那3个很可能也是开放的端口。

- -sM (TCP Maimon扫描)

  Maimon扫描是用它的发现者Uriel Maimon命名的。他在 Phrack Magazine issue #49 (November 1996)中描述了这一技术。 Nmap在两期后加入了这一技术。 这项技术和Null，FIN，以及Xmas扫描完全一样，除了探测报文是FIN/ACK。 根据RFC 793 (TCP)，无论端口开放或者关闭，都应该对这样的探测响应RST报文。 然而，Uriel注意到如果端口开放，许多基于BSD的系统只是丢弃该探测报文。

- --scanflags (定制的TCP扫描)

  真正的Nmap高级用户不需要被这些现成的扫描类型束缚。 --scanflags选项允许您通过指定任意TCP标志位来设计您自己的扫描。 让您的创造力流动，躲开那些仅靠本手册添加规则的入侵检测系统！--scanflags选项可以是一个数字标记值如9 (PSH和FIN)， 但使用字符名更容易些。 只要是URG， ACK，PSH， RST，SYN，and FIN的任何组合就行。例如，--scanflags URGACKPSHRSTSYNFIN设置了所有标志位，但是这对扫描没有太大用处。 标志位的顺序不重要。除了设置需要的标志位，您也可以设置 TCP扫描类型(如-sA或者-sF)。 那个基本类型告诉Nmap怎样解释响应。例如， SYN扫描认为没有响应意味着 filtered端口，而FIN扫描则认为是open|filtered。 除了使用您指定的TCP标记位，Nmap会和基本扫描类型一样工作。 如果您不指定基本类型，就使用SYN扫描。

- -sI <zombie host[:probeport]> (Idlescan)

  这种高级的扫描方法允许对目标进行真正的TCP端口盲扫描 (意味着没有报文从您的真实IP地址发送到目标)。相反，side-channel攻击 利用zombie主机上已知的IP分段ID序列生成算法来窥探目标上开放端口的信息。 IDS系统将显示扫描来自您指定的zombie机(必须运行并且符合一定的标准)。 这种奇妙的扫描类型太复杂了，不能在此完全描述，所以我写一篇非正式的论文， 发布在http://nmap.org/book/idlescan.html。除了极端隐蔽(由于它不从真实IP地址发送任何报文)， 该扫描类型可以建立机器间的基于IP的信任关系。 端口列表*从zombie 主机的角度*。显示开放的端口。 因此您可以尝试用您认为(通过路由器/包过滤规则)可能被信任的 zombies扫描目标。如果您由于IPID改变希望探测zombie上的特定端口， 您可以在zombie 主机后加上一个冒号和端口号。 否则Nmap会使用默认端口(80)。

- -sO (IP协议扫描)

  IP 协议扫描可以让您确定目标机支持哪些IP协议 (TCP，ICMP，IGMP，等等)。从技术上说，这不是端口扫描 ，既然它遍历的是IP协议号而不是TCP或者UDP端口号。 但是它仍使用 -p选项选择要扫描的协议号， 用正常的端口表格式报告结果，甚至用和真正的端口扫描一样 的扫描引擎。因此它和端口扫描非常接近，也被放在这里讨论。除了本身很有用，协议扫描还显示了开源软件的力量。 尽管基本想法非常简单，我过去从没想过增加这一功能也没收到任何对它的请求。 在2000年夏天，Gerhard Rieger孕育了这个想法，写了一个很棒的补丁程序，发送到nmap-hackers邮件列表。 我把那个补丁加入了Nmap，第二天发布了新版本。 几乎没有商业软件会有用户有足够的热情设计并贡献他们的改进。协议扫描以和UDP扫描类似的方式工作。它不是在UDP报文的端口域上循环， 而是在IP协议域的8位上循环，发送IP报文头。 报文头通常是空的，不包含数据，甚至不包含所申明的协议的正确报文头 TCP，UDP，和ICMP是三个例外。它们三个会使用正常的协议头，因为否则某些系 统拒绝发送，而且Nmap有函数创建它们。协议扫描不是注意ICMP端口不可到达消息， 而是ICMP *协议*不可到达消息。如果Nmap从目标主机收到 任何协议的任何响应，Nmap就把那个协议标记为open。 ICMP协议不可到达 错误(类型 3，代号 2) 导致协议被标记为 closed。其它ICMP不可到达协议(类型 3，代号 1，3，9，10，或者13) 导致协议被标记为 filtered (虽然同时他们证明ICMP是 open)。如果重试之后仍没有收到响应， 该协议就被标记为open|filtered

- -b (FTP弹跳扫描)

  FTP协议的一个有趣特征(RFC 959) 是支持所谓代理ftp连接。它允许用户连接到一台FTP服务器，然后要求文件送到一台第三方服务器。 这个特性在很多层次上被滥用，所以许多服务器已经停止支持它了。其中一种就是导致FTP服务器对其它主机端口扫描。 只要请求FTP服务器轮流发送一个文件到目标主机上的所感兴趣的端口。 错误消息会描述端口是开放还是关闭的。 这是绕过防火墙的好方法，因为FTP服务器常常被置于可以访问比Web主机更多其它内部主机的位置。 Nmap用-b选项支持ftp弹跳扫描。参数格式是 :@:。 是某个脆弱的FTP服务器的名字或者IP地址。 您也许可以省略:， 如果服务器上开放了匿名用户(user:anonymous password:-wwwuser@)。 端口号(以及前面的冒号) 也可以省略，如果使用默认的FTP端口(21)。当Nmap1997年发布时，这个弱点被广泛利用，但现在大部分已经被fix了。 脆弱的服务器仍然存在，所以如果其它都失败了，这也值得一试。 如果您的目标是绕过防火墙，扫描目标网络上的开放的21端口(或者甚至任何ftp服务，如果您用版本探测扫描所有端口)， 然后对每个尝试弹跳扫描。Nmap会告诉您该主机脆弱与否。 如果您只是试着玩Nmap，您不必(事实上，不应该)限制您自己。 在您随机地在互联网上寻找脆弱的FTP服务器时，考虑一下系统管理员不太喜欢您这样滥用他们的服务器。

端口说明和扫描顺序

除了所有前面讨论的扫描方法， Nmap提供选项说明哪些端口被扫描以及扫描是随机还是顺序进行。 默认情况下，Nmap用指定的协议对端口 1~1024 以及 nmap-services 文件中列出的更高的端口进行扫描。

- **-p（只扫描指定的端口）**

  该选项指明想扫描的端口，覆盖默认值，单个端口和用连字符表示的端口范围(如1-1023)都可以。 范围的开始以及/或者结束值可以被省略， 分别导致Nmap使用1和65535，所以可以指定 -p- 从端口1扫描到65535。 如果特别指定，也可以扫描端口0。对于IP协议扫描(-sO)，该选项指定您希望扫描的协议号 (0-255)。当既扫描TCP端口又扫描UDP端口时，您可以通过在端口号前加上T: 或者U:指定协议。 协议限定符一直有效您直到指定另一个。 例如，参数 -p U:53，111，137，T:21-25，80，139，8080 将扫描UDP 端口53，111，和137，同时扫描列出的TCP端口。注意，要既扫描 UDP又扫描TCP，您必须指定 -sU ，以及至少一个TCP扫描类型(如 -sS，-sF，或者 -sT)。如果没有给定协议限定符， 端口号会被加到所有协议列表。

- -F (快速 (有限的端口) 扫描)

在nmap的nmap-services文件中(对于-sO，是协议文件)指定您想要扫描的端口。这比扫描所有65535个端口快得多，因为该列表包含如此多的TCP端口(1200多)，这和默认的TCP扫描 (大约1600个端口)速度差别不是很大。如果您用--datadir选项指定您自己的小小的nmap-services文件 ，差别会很惊人。

- **-r (不要按随机顺序扫描端口)**

  默认情况下，Nmap按随机顺序扫描端口 (除了出于效率的考虑，常用的端口前移)。这种随机化通常都是受欢迎的， 但您也可以指定-r来按顺序进行端口扫描。

服务和版本探测

把Nmap指向一个远程机器，它可能告诉您端口25/tcp，80/tcp，和53/udp是开放的，使用包含大约2,200个著名的服务的nmap-services数据库。Nmap可以报告那些端口可能分别对应于一个邮件服务器(SMTP)，web服务器(HTTP)，和域名服务器(DNS)。 这种查询通常是正确的。事实上，绝大多数在TCP端口25监听的守护进程是邮件 服务器。然而，您不应该把赌注押在这上面！管理员完全可以在一些奇怪的端口上运行服务。

即使Nmap是对的，假设运行服务的确实是 SMTP，HTTP和DNS，那也不是特别多的信息。 当为您的公司或者客户作安全评估(或者甚至简单的网络明细清单)时，您确实想知道正在运行什么邮件和域名服务器以及它们的版本，有一个精确的版本号对了解服务器有什么漏洞有巨大帮助，版本探测可以帮您获得该信息。

在用某种其它类型的扫描方法发现TCP 和/或者UDP端口后，版本探测会询问这些端口，确定到底什么服务正在运行。nmap-service-probes 数据库包含查询不同服务的探测报文和解析识别响应的匹配表达式。Nmap试图确定服务协议 (如 ftp，ssh，telnet，http)，应用程序名(如ISC Bind，Apache httpd，Solaris telnetd)，版本号，主机名，设备类型(如打印机，路由器)，操作系统 (如Windows，Linux)以及其它的细节，如是否可以连接X server，SSH协议版本，或者KaZaA用户名。当然，并非所有服务都提供所有这些信息。 如果Nmap被编译成支持OpenSSL，它将连接到SSL服务器，推测什么服务在加密层后面监听。当发现RPC服务时，Nmap RPC grinder (-sR)会自动被用于确定RPC程序和它的版本号。如果在扫描某个UDP端口后仍然无法确定该端口是开放的还是被过滤的，那么该端口状态就被标记为open|filtered。 版本探测将试图从这些端口引发一个响应(就像它对开放端口做的一样)， 如果成功，就把状态改为开放。open|filtered TCP端口用同样的方法对待。注意Nmap -A选项在其它情况下打开版本探测。有一篇关于版本探测的原理，使用文章在http://www.insecure.org/nmap/vscan/。

当Nmap从某个服务收到响应，但不能在数据库中找到匹配时，它就打印一个特殊的fingerprint和一个URL给您提交。如果您知道什么服务运行在端口，请花两分钟提交您的发现，让每个人受益。由于这些提交，Nmap有350种以上协议，如smtp，ftp，http等大约3,000条模式匹配。

用下列的选项打开和控制版本探测。

- **-sV (版本探测)**

  打开版本探测，也可以用 -A 同时打开操作系统探测和版本探测。

- **--allports (不为版本探测排除任何端口)**

  默认情况下，Nmap版本探测会跳过 9100 TCP端口，因为一些打印机会打印发送到该端口的任何数据，会导致数十页HTTP get请求、二进制 SSL会话请求等等被打印出来。这一行为可以通过修改或删除nmap-service-probes 中的Exclude指示符改变，也可以不理会任何Exclude指示符，指定--allports扫描所有端口。

- --version-intensity (设置版本扫描强度)

当进行版本扫描(-sV)时，nmap发送一系列探测报文，每个报文都被赋予一个1到9之间的值。被赋予较低值的探测报文对大范围的常见服务有效，而被赋予较高值的报文一般没什么用。强度水平说明了应该使用哪些探测报文。数值越高，服务越有可能被正确识别。然而，高强度扫描花更多时间。强度值必须在0和9之间。默认是7。当探测报文通过nmap-service-probes ports指示符注册到目标端口时，无论什么强度水平，探测报文都会被尝试。这保证了DNS探测将永远在任何开放的53端口尝试，SSL探测将在443端口尝试，等等。

- --version-light (打开轻量级模式)

  这是 --version-intensity 2的方便的别名。轻量级模式使版本扫描快许多，但它识别服务的可能性也略微小一点。

- --version-all (尝试每个探测)

  --version-intensity 9的别名，保证对每个端口尝试每个探测报文。

- --version-trace (跟踪版本扫描活动)

  这导致Nmap打印出详细的关于正在进行的扫描的调试信息。它是您用--packet-trace所得到的信息的子集。

- -sR (RPC扫描)

  这种方法和许多端口扫描方法联合使用。它对所有被发现开放的TCP/UDP端口执行SunRPC程序NULL命令，来试图确定它们是否RPC端口，如果是，是什么程序和版本号。因此您可以有效地获得和**rpcinfo -p**一样的信息，即使目标的端口映射在防火墙后面(或者被TCP包装器保护)。Decoys目前不能和RPC scan一起工作。这作为版本扫描(-sV)的一部分自动打开。由于版本探测包括它并且全面得多，-sR很少被需要。

操作系统探测

Nmap最著名的功能之一是用TCP/IP协议栈fingerprinting进行远程操作系统探测。Nmap发送一系列TCP和UDP报文到远程主机，检查响应中的每一个比特。在进行一打测试如TCP ISN采样、TCP选项支持和排序、IPID采样和初始窗口大小检查之后，Nmap把结果和数据库nmap-os-fingerprints中超过1500个已知的操作系统的fingerprints进行比较，如果有匹配，就打印出操作系统的详细信息。每个fingerprint包括一个自由格式的关于OS的描述文本和一个分类信息，它提供供应商名称(如Sun)，下面的操作系统(如Solaris)，OS版本(如10)，设备类型(通用设备，路由器，switch，游戏控制台等)。

如果Nmap不能猜出操作系统，并且有些好的已知条件(如至少发现了一个开放端口和一个关闭端口)，Nmap会提供一个URL，如果您确知运行的操作系统，您可以把fingerprint提交到那个URL。这样您就扩大了Nmap的操作系统知识库，从而让每个Nmap用户都受益。

操作系统检测可以进行其它一些测试，这些测试可以利用处理过程中收集到的信息。例如运行时间检测，使用TCP时间戳选项(RFC 1323)来估计主机上次重启的时间，这仅适用于提供这类信息的主机。另一种是TCP序列号预测分类，用于测试针对远程主机建立一个伪造的TCP连接的可能难度。这对于利用基于源IP地址的可信关系(rlogin，防火墙过滤等)或者隐含源地址的攻击非常重要。这一类欺骗攻击现在很少见，但一些主机仍然存在这方面的漏洞。实际的难度值基于统计采样，因此可能会有一些波动。通常采用英国的分类较好，如“worthy challenge”或者 “trivial joke”。在详细模式(-v)下只以普通的方式输出，如果同时使用-O，还报告IPID序列产生号。很多主机的序列号是“增加”类别，即在每个发送包的IP头中增加ID域值，这对一些先进的信息收集和欺骗攻击来说是个漏洞。

http://nmap.org/book/osdetect.html 文档使用多种语言描述了版本检测的方式、使用和定制。

采用下列选项启用和控制操作系统检测：

- **-O (启用操作系统检测)**

也可以使用 -A 来同时启用操作系统检测和版本检测。

- --osscan-limit (针对指定的目标进行操作系统检测)

  如果发现一个打开和关闭的TCP端口时，操作系统检测会更有效。 采用这个选项，Nmap只对满足这个条件的主机进行操作系统检测，这样可以节约时间，特别在使用-P0扫描多个主机时。这个选项仅在使用 -O或-A 进行操作系统检测时起作用。

- --osscan-guess; --fuzzy (推测操作系统检测结果)

  当Nmap无法确定所检测的操作系统时，会尽可能地提供最相近的匹配，Nmap默认进行这种匹配，使用上述任一个选项使得Nmap的推测更加有效。

时间和性能

Nmap开发的最高优先级是性能。在本地网络对一个主机的默认扫描(**nmap ****)需要0.2秒，而仅仅眨眼的时间，就可以扫描上万甚至几十万的主机。此外，一些特定的扫描选项会明显增加扫描时间，如UDP扫描和版本检测。同样，防火墙配置以及特殊的响应速度限制也会增加时间。Nmap使用了并行算法和许多先进的算法来加速扫描，用户对Nmap如何工作有最终的控制权。高级用户可以仔细地调整Nmap命令，在满足时间要求的同时获得他们所关心的信息。

改善扫描时间的技术有：忽略非关键的检测、升级最新版本的Nmap(性能增强不断改善)。 优化时间参数也会带来实质性的变化，这些参数如下。

- --min-hostgroup ; --max-hostgroup（调整并行扫描组的大小)

  Nmap具有并行扫描多主机端口或版本的能力，Nmap将多个目标IP地址 空间分成组，然后在同一时间对一个组进行扫描。通常，大的组更有效。缺 点是只有当整个组扫描结束后才会提供主机的扫描结果。如果组的大小定义 为50，则只有当前50个主机扫描结束后才能得到报告(详细模式中的补充信息 除外)。默认方式下，Nmap采取折衷的方法。开始扫描时的组较小， 最小为5，这样便于尽快产生结果；随后增长组的大小，最大为1024。确切的 大小依赖于所给定的选项。为保证效率，针对UDP或少量端口的TCP扫描，Nmap 使用大的组。--max-hostgroup选项用于说明使用最大的组，Nmap不 会超出这个大小。--min-hostgroup选项说明最小的组，Nmap 会保持组大于这个值。如果在指定的接口上没有足够的目标主机来满足所 指定的最小值，Nmap可能会采用比所指定的值小的组。这两个参数虽然很少使用， 但都用于保持组的大小在一个指定的范围之内。这些选项的主要用途是说明一个最小组的大小，使得整个扫描更加快速。通常 选择256来扫描C类网段。对于端口数较多的扫描，超出该值没有意义。对于 端口数较少的扫描，2048或更大的组大小是有帮助的。

- --min-parallelism ; --max-parallelism（调整探测报文的并行度)

  这些选项控制用于主机组的探测报文数量，可用于端口扫描和主机发现。默认状态下， Nmap基于网络性能计算一个理想的并行度，这个值经常改变。如果报文被丢弃， Nmap降低速度，探测报文数量减少。随着网络性能的改善，理想的探测报文数量会缓慢增加。 这些选项确定这个变量的大小范围。默认状态下，当网络不可靠时，理想的并行度值 可能为1，在好的条件下，可能会增长至几百。最常见的应用是--min-parallelism值大于1，以加快 性能不佳的主机或网络的扫描。这个选项具有风险，如果过高则影响准确度，同时 也会降低Nmap基于网络条件动态控制并行度的能力。这个值设为10较为合适， 这个值的调整往往作为最后的手段。--max-parallelism选项通常设为1，以防止Nmap在同一时间 向主机发送多个探测报文，和选择--scan-delay同时使用非常有用，虽然 这个选项本身的用途已经很好。

- --min-rtt-timeout， --max-rtt-timeout， --initial-rtt-timeout（调整探测报文超时)

Nmap使用一个运行超时值来确定等待探测报文响应的时间，随后会放弃或重新 发送探测报文。Nmap基于上一个探测报文的响应时间来计算超时值，如果网络延迟比较显著 和不定，这个超时值会增加几秒。初始值的比较保守(高)，而当Nmap扫描无响应 的主机时，这个保守值会保持一段时间。这些选项以毫秒为单位，采用小的--max-rtt-timeout值，使 --initial-rtt-timeout值大于默认值可以明显减少扫描时间，特别 是对不能ping通的扫描(-P0)以及具有严格过滤的网络。如果使用太小的值，使得很多探测报文超时从而重新发送，而此时可能响应消息正在发送，这使得整个扫描的时 间会增加。如果所有的主机都在本地网络，对于--max-rtt-timeout值来 说，100毫秒比较合适。如果存在路由，首先使用ICMP ping工具ping主机，或使用其 它报文工具如hpings，可以更好地穿透防火墙。查看大约10个包的最大往返时间，然后将 --initial-rtt-timeout设成这个时间的2倍，--max-rtt-timeout 可设成这个时间值的3倍或4倍。通常，不管ping的时间是多少，最大的rtt值不得小于100ms， 不能超过1000ms。--min-rtt-timeout这个选项很少使用，当网络不可靠时，Nmap的默认值也显得过于强烈，这时这个选项可起作用。当网络看起来不可靠时，Nmap仅将 超时时间降至最小值，这个情况是不正常的，需要向nmap-dev邮件列表报告bug。

- --host-timeout（放弃低速目标主机）

  由于性能较差或不可靠的网络硬件或软件、带宽限制、严格的防火墙等原因， 一些主机需要*很长*的时间扫描。这些极少数的主机扫描往往占 据了大部分的扫描时间。因此，最好的办法是减少时间消耗并且忽略这些主机，使用 --host-timeout选项来说明等待的时间(毫秒)。通常使用1800000 来保证Nmap不会在单个主机上使用超过半小时的时间。需要注意的是，Nmap在这半小时中可以 同时扫描其它主机，因此并不是完全放弃扫描。超时的主机被忽略，因此也没有针对该主机的 端口表、操作系统检测或版本检测结果的输出。

- --scan-delay ; --max-scan-delay（调整探测报文的时间间隔）

  这个选项用于Nmap控制针对一个主机发送探测报文的等待时间(毫秒)，在带宽 控制的情况下这个选项非常有效。Solaris主机在响应UDP扫描探测报文报文时，每秒 只发送一个ICMP消息，因此Nmap发送的很多数探测报文是浪费的。--scan-delay 设为1000，使Nmap低速运行。Nmap尝试检测带宽控制并相应地调整扫描的延迟，但 并不影响明确说明何种速度工作最佳。--scan-delay的另一个用途是躲闭基于阈值的入侵检测和预防 系统(IDS/IPS)。

- **-T <Paranoid|Sneaky|Polite|Normal|Aggressive|Insane> (设置时间模板)**

  上述优化时间控制选项的功能很强大也很有效，但有些用户会被迷惑，往往造成选择合适参数的时间超过了所需优化的扫描时间。因此，**Nmap提供了一些简单的方法，使用6个时间模板，使用时采用-T选项及数字(0 - 5) 或名称。模板名称有paranoid (0)、sneaky (1)、polite (2)、normal(3)、 aggressive (4)和insane (5)。**前两种模式（T0、T1）用于IDS躲避，Polite模式（T2）降低了扫描速度以使用更少的带宽和目标主机资源。**默认模式为Normal（T3），因此 -T3 实际上是未做任何优化。**Aggressive模式（T4）假设用户具有合适及可靠的网络从而加速扫描。Insane模式（T5）假设用户具有特别快的网络或者愿意为获得速度而牺牲准确性。用户可以根据自己的需要选择不同的模板，由Nmap负责选择实际的时间值。 模板也会针对其它的优化控制选项进行速度微调。**例如，-T4 针对TCP端口禁止动态扫描延迟超过10ms，-T5对应的值为5ms。** 模板可以和优化调整控制选项组合使用，但模板必须首先指定，否则其标准值会覆盖用户指定的值。建议在扫描可靠的网络时使用 -T4，即使在自己要增加优化控制选项时也使用(在命令行的开始)，从而从这些额外的较小的优化中获益。如果用于有足够的带宽或以太网连接，仍然建议使用-T4选项。 有些用户喜欢-T5选项，但这个过于强烈。有时用户考虑到避免使主机崩溃或者希望更礼貌一些会采用-T2选项。他们并没意识到-T Polite选项是如何慢，这种模式的扫描比默认方式实际上要多花10倍的时间。默认时间选项(-T3)很少有主机崩溃和带宽问题，比较适合于谨慎的用户。不进行版本检测比进行时间调整能更有效地解决这些问题。虽然-T0和-T1选项可能有助于避免IDS告警，但在进行上千个主机或端口扫描时，会显著增加时间。对于这种长时间的扫描，宁可设定确切的时间值，而不要去依赖封装的-T0和-T1选项。T0选项的主要影响是对于连续扫描，在一个时间只能扫描一个端口， 每个探测报文的发送间隔为5分钟。T1和T2选项比较类似，探测报文间隔分别为15秒和0.4秒。T3是Nmap的默认选项，包含了并行扫描。T4选项与 --max-rtt-timeout 1250 --initial-rtt-

timeout 500 等价，最大TCP扫描延迟为10ms。T5等价于 --max-rtt-timeout 300 --min-rtt-timeout 50 --initial-rtt-timeout 250 --host-timeout 900000，最大TCP扫描延迟为5ms。

防火墙/IDS躲避和欺骗

很多Internet先驱们设想了一个全球开放的网络，使用全局的IP地址空间，使得任何两个节点之间都有虚拟连接，这使得主机间可以作为真正的对等体，相互间提供服务和获取信息。

然而，这些全球连接的设想受到了地址空间短缺和安全考虑的限制。在90年代早期，各种机构开始部署防火墙来实现减少连接的目的，大型网络通过代理、NAT和包过滤器与未过滤的Internet隔离。不受限的信息流被严格控制的可信通信通道信息流所替代。

类似防火墙的网络隔离使得对网络的搜索更加困难，随意的搜索变得不再简单。然而，Nmap提供了很多特性用于理解这些复杂的网络，并且检验这些过滤器是否正常工作。此外，Nmap提供了绕过某些较弱的防范机制的手段。检验网络安全状态最有效的方法之一是尝试欺骗网络，将自己想象成一个攻击者，使用本节提供的技术来攻击自己的网络。如使用FTP bounce扫描、Idle扫描、分片攻击或尝试穿透自己的代理。

有时也会出现不同的意见，比如建议Nmap不应该提供躲闭防火墙规则或欺骗IDS的功能， 这些功能可能会被攻击者滥用，然而管理员却可以利用这些功能来增强安全性。实际上，即便是关闭Nmap绕过或者欺骗的功能，其他的攻击方法仍可被攻击者利用，他们可以发现其它工具或Nmap的补丁程序。同时，管理员发现攻击者的工作更加困难，相较于采取措施来预防执行FTP Bounce攻击的工具而言，部署先进的、打过补丁的FTP服务器更加有效。

- **-f (报文分段); --mtu (使用指定的MTU)**

  扫描时(包挺ping扫描)使用小的IP包分段，其思路是将TCP头分段在几个包中，使得包过滤器、 IDS以及其它工具的检测更加困难。
  必须小心使用这个选项，有些系统在处理这些小包时存在问题，例如旧的网络嗅探器Sniffit在接收到第一个分段时会立刻出现分段错误。该选项使用一次，Nmap在IP头后将包分成8个字节或更小。因此，一个20字节的TCP头会被分成3个包，其中2个包分别有TCP头的8个字节，另1个包有TCP头的剩下4个字节。当然，每个包都有一个IP头。再次使用-f可使用16字节的分段(减少分段数量)。使用--mtu选项可以自定义偏移的大小，使用时不需要-f，偏移量必须是8的倍数。包过滤器和防火墙对所有的IP分段排队，如Linux核心中的 CONFIG-IP-ALWAYS-DEFRAG 配置项，分段包不会直接使用。一些网络无法承受这样所带来的性能冲击，会将这个配置禁止。其它禁止的原因有分段包会通过不同的路由进入网络。一些源系统在内核中对发送的报文进行分段，使用iptables连接跟踪模块的Linux就是一个例子。当使用类似Ethereal的嗅探器时，扫描必须保证发送的报文要分段。如果主机操作系统会产生问题，尝试使用--send-eth选项以避开IP层而直接发送原始的以太网帧。

- -D <decoy1 [，decoy2][，ME]，...> (使用诱饵隐蔽扫描)

  为使诱饵扫描起作用，需要使远程主机认为是诱饵在扫描目标网络。

  IDS可能会报个某个IP的5-10个端口扫描，但并不知道哪个IP在扫描以及 哪些不是诱饵。但这种方式可以通过路由跟踪、响应丢弃以及其它主动 机制在解决。这是一种常用的隐藏自身IP地址的有效技术。使用逗号分隔每个诱饵主机，也可用自己的真实IP作为诱饵，这时可使用 ME选项说明。如果在第6个位置或 更后的位置使用ME选项，一些常用 端口扫描检测器(如Solar Designer's excellent scanlogd)就不会报告 这个真实IP。如果不使用ME选项，Nmap 将真实IP放在一个随机的位置注意，作为诱饵的主机须在工作状态，否则会导致目标主机的SYN洪水攻击。 如果在网络中只有一个主机在工作，那就很容易确定哪个主机在扫描。也可 使用IP地址代替主机名(被诱骗的网络就不可能在名字服务器日志中发现)。诱饵可用在初始的ping扫描(ICMP、SYN、ACK等)阶段或真正的端口扫描 阶段。诱饵也可以用于远程操作系统检测(-O)。在进行版 本检测或TCP连接扫描时，

诱饵无效。使用过多的诱饵没有任何价值，反而导致扫描变慢并且结果不准确。 此外，一些ISP会过滤欺骗的报文，但很多对欺骗IP包没有任何限制。

- **-S <IP_Address> (源地址欺骗)**

  在某些情况下，Nmap可能无法确定扫描源地址(如果这样，Nmap会给出提示)，此时，可以使用-S选项并说明所需发送包的接口IP地址。这个标志的另一个用处是欺骗性的扫描，使得目标认为是另一个地址在进行扫描。

- **--source-port ; -g（源端口欺骗)**

  仅依赖于源端口号就信任数据流是一种常见的错误配置，这个问题非常好理解。例如一个管理员部署了一个新的防火墙，但招来了很多用户的不满，因为他们的应用停止工作了。可能是由于外部的UDP DNS服务器响应无法进入网络，而导致DNS的崩溃。FTP是另一个常见的例子，在FTP传输时，远程服务器尝试和内部用户建立连接以传输数据。对这些问题有安全解决方案，通常是应用级代理或协议分析防火墙模块，但也存在一些不安全的方案。注意到DNS响应来自于53端口，FTP连接来自于20端口，很多管理员会掉入一个陷阱，即允许来自于这些端口的数据进入网络。他们认为这些端口里不会有值得注意的攻击和漏洞利用。此外，管理员或许认为这是一个短期的措施，直至他们采取更安全的方案。但他们忽视了安全的升级。不仅仅是工作量过多的网络管理员掉入这种陷阱，很多产品本身也会有这类 不安全的隐患，甚至是微软的产品，Windows 2000和Windows XP中包含的IPsec过滤器也包含了一些隐含规则，允许所有来自88端口(Kerberos)的TCP和UDP数据流。另一个常见的例子是Zone Alarm个人防火墙到2.1.25版本仍然允许源端口53(DNS)或67(DHCP)的UDP包进入。Nmap提供了-g和--source-port选项(它们是等价的)，用于利用上述弱点，只需要提供一个端口号，Nmap就可以从这些端口发送数据。为使特定的操作系统正常工作，Nmap必须使用不同的端口号。DNS请求会忽略--source-port选项，这是因为Nmap依靠系统库来处理。大部分TCP扫描，包括SYN扫描，可以完全支持这些选项，UDP扫描同样如此。

- **--data-length（发送报文时附加随机数据)**

  正常情况下，Nmap发送最少的报文，只含一个包头，TCP包通常是40字节，ICMP ECHO请求只有28字节。这个选项告诉Nmap在发送的报文上附加指定数量的随机字节，操作系统检测(-O)包不受影响，但大部分ping和端口扫描包受影响，这会使处理变慢，但对扫描的影响较小。

- --ttl (设置IP time-to-live域)

  设置IPv4报文的time-to-live域为指定的值。

- --randomize-hosts (对目标主机的顺序随机排列)

  告诉Nmap在扫描主机前对每个组中的主机随机排列，最多可达 8096个主机。这会使得扫描针对不同的网络监控系统来说变得不是很 明显，特别是配合值较小的时间选项时更有效。如果需要对一个较大的组进行随机排列，需要增大nmap.h文件中 PING-GROUP-SZ的值，并重新编译。另一种方法是使用列表扫描 (-sL -n -oN)，产生目标IP的列表， 使用Perl脚本进行随机化，然后使用-iL提供给Nmap。

- **--spoof-mac <mac address，prefix，or vendor name> (MAC地址欺骗)**

  要求Nmap在发送原以太网帧时使用指定的MAC地址，这个选项隐含了 --send-eth选项，以保证Nmap真正发送以太网包。MAC地址有几种格式。如果简单地使用字符串“0”，Nmap选择一个完全随机的MAC 地址。如果给定的字符品是一个16进制偶数(使用:分隔)，Nmap将使用这个MAC地址。 如果是小于12的16进制数字，Nmap会随机填充剩下的6个字节。如果参数不是0或16进制字符串，Nmap将通过nmap-mac-prefixes查找 厂商的名称(大小写区分)，如果找到匹配，Nmap将使用厂商的OUI(3字节前缀)，然后随机填充剩余的3个节字。正确的--spoof-mac参数有：Apple，0，01:02:03:04:05:06，deadbeefcafe，0020F2和Cisco。

输出

任何安全工具只有在输出结果时才是有价值的，如果没有通过组织和易于理解的方式来表达，复杂的测试和算法几乎没有意义。

Nmap提供了一些方式供用户和其它软件使用，实际上，没有一种方式可以使所有人满意。 因此Nmap提供了一些格式，包含了方便直接查看的交互方式和方便软件处理的XML格式。除了提供输出格式外，Nmap还提供了选项来控制输出的细节以及调试信息。输出内容可发送给标准输出或命名文件，可以追加或覆盖。输出文件还可被用于继续中断的扫描。

Nmap提供5种不同的输出格式。默认的方式是interactive output， 发送给标准输出(stdout)。normal output方式类似于interactive，但显示较少的运行时间信息和告警信息，这是由于这些信息是在扫描完全结束后用于分析，而不是交互式的。XML输出是最重要的输出类型，可被转换成HTML，对于程序处理非常方便， 如用于Nmap图形用户接口或导入数据库。另外两种输出类型比较简单，grepable output格式，在一行中包含目标主机最多的信息；sCRiPt KiDDi3 0utPUt 格式，用于考虑自己的用户 | <-r4d。

交互式输出是默认方式，没有相应的命令行选项，其它四种格式选项使用相同的语法，采用一个参数，即存放结果的文件名。多种格式可同时使用，但一种格式只能使用一次。例如，在标准输出用于查看的同时，可将结果保存到XML文件用于程序分析，这时可以使用选项-oX myscan.xml -oN myscan.nmap。 为便于描述的简化，本章使用类似于myscan.xml的简单文件名， 建议采用更具有描述性的文件名。文件名的选择与个人喜好有关，建议增加 扫描日期以及一到两个单词来描述，并放置于一个目录中。

在将结果输出到文件的同时，Nmap仍将结果发送给标准输出。例如， 命令nmap -oX myscan.xml target将输出XML至myscan.xml，并在stdout上打印相同的交互式结果，而此时-oX选项没有采用。可以使用连字符作为选项来改变，这使得Nmap禁止交互式输出，而是将结果打印到所指定的标准输出流中。因此，命令nmap -oX - target只 输出XML至标准输出stdout，严重错误仍然是输出到标准错误流stderr中。

与其它Nmap参数不同，日志文件选项的空格(如-oX)和文件名或连字符是必需的。如果省略了标记，例如-oG-或 -oXscan.xml，Nmap的向后兼容特点将建立标准格式的输出文件，相应的文件名为G-和Xscan.xml。

Nmap还提供了控制扫描细节以及输出文件的添加或覆盖的选项，这些选项如下所述。

Nmap输出格式

- -oN（标准输出）

  要求将标准输出直接写入指定的文件。

- -oX（XML输出）

  要求XML输出直接写入指定的文件。Nmap包含了一个文档类型定义(DTD)，使XML解析器有效地进行XML输出。这主要是为了程序应用，同时也可以协助人工解释 Nmap的XML输出。DTD定义了合法的格式元素，列举可使用的属性和 值。最新的版本可在 http://www.insecure.org/nmap/data/nmap.dtd获取。XML提供了可供软件解析的稳定格式输出，主要的计算机 语言都提供了免费的XML解析器，如C/C++，Perl，Python和Java。 针对这些语言有一些捆绑代码用于处理Nmap的输出和特定的执行程序。 例如perl CPAN中的Nmap::Scanner 和Nmap::Parser。 对几乎所有与Nmap有接口的主要应用来说，XML是首选的格式。XML输出引用了一个XSL样式表，用于格式化输出结果，类似于 HTML。最方便的方法是将XML输出加载到一个Web浏览器，如Firefox 或IE。由于nmap.xsl文件的绝对 路径，因此通常只能在运行了Nmap的机器上工作(或类似配置的机器)。类似于任何支持Web机器的HTML文件，--stylesheet 选项可用于建立可移植的XML文件。

细节和调试选项

- **-v (提高输出信息的详细度)**

通过提高详细度，Nmap可以输出扫描过程的更多信息。 输出发现的打开端口，若Nmap认为扫描需要更多时间会显示估计的结束时间。这个选项使用两次，会提供更详细的信息。这个选项使用两次以上不起作用。大部分的变化仅影响交互式输出，也有一些影响标准和脚本小子输出。其它输出类型由机器处理，此时Nmap默认提供详细的息，不需要人工干预。然而，其它模式也会有一些变化，省略一些细节可以减小输出大小。例如，Grep输出中的注释行提供所有扫描端口列表，但由于这些信息过长，因此只能在细节模式中输出。

- -d [level] (提高或设置调试级别)

  当详细模式也不能为用户提供足够的数据时，使用调试可以得到更多的信息。使用细节选项(-v)时，可启用命令行参数 (-d)，多次使用可提高调试级别。也可在-d 后面使用参数设置调试级别。例如，-d9设定级别9。这是 最高的级别，将会产生上千行的输出，除非只对很少的端口和目标进行简单扫描。如果Nmap因为Bug而挂起或者对Nmap的工作及原理有疑问，调试输出非常有效。主要是开发人员用这个选项，调试行不具备自我解释的特点。 例如，Timeoutvals: srtt: -1 rttvar: -1 to: 1000000 delta 14987 ==> srtt: 14987 rttvar: 14987 to: 100000。如果对某行输出不明白，可以忽略、查看源代码或向开发列表(nmap-dev)求助。有些输出行会有自 我解释的特点，但随着调试级别的升高，会越来越含糊。

其它选项

本节描述一些重要的(和并不重要)的选项，这些选项不适合其它任何地方。

- -6 (启用IPv6扫描)

  从2002年起，Nmap提供对IPv6的一些主要特征的支持。ping扫描(TCP-only)、 连接扫描以及版本检测都支持IPv6。除增加-6选项外， 其它命令语法相同。当然，必须使用IPv6地址来替换主机名，如 3ffe:7501:4819:2000:210:f3ff:fe03:14d0。 除"所关注的端口"行的地址部分为IPv6地址。IPv6目前未在全球广泛采用，目前在一些国家(亚洲)应用较多，一些高级操作系统支持IPv6。使用Nmap的IPv6功能，扫描的源和目的都需要配置IPv6。如果ISP(大部分)不分配IPv6地址，Nmap可以采用免费的隧道代理。一种较好的选择是BT Exact，位于https://tb.ipv6.btexact.com/。 此外，还有Hurricane Electric，位于http://ipv6tb.he.net/。6to4隧道是另一种常用的免费方法。

实例

下面给出一些实例，简单的、复杂的到深奥的。为更具体，一些例子使用了实际的IP地址和域名。在这些位置，可以使用你自己网络的地址/域名替换。注意，扫描其它网络不一定合法，一些网络管理员不愿看到未申请过的扫描，会产生报怨。因此，先获得允许是最好的办法。

如果是为了测试，scanme.nmap.org 允许被扫描，但仅允许使用Nmap扫描并禁止测试漏洞或进行DoS攻击。为 保证带宽，对该主机的扫描每天不要超过12次。如果这个免费扫描服务被滥用，系统将崩溃而且Nmap将报告解析 指定的主机名/IP地址失败：scanme.nmap.org。这些免费扫描要求也适用于scanme2/3.nmap.org等等，虽然这些主机目前还不存在。

nmap -v scanme.nmap.org

这个选项扫描主机scanme.nmap.org中所有的保留TCP端口，选项-v启用细节模式。

nmap -sS -O scanme.nmap.org/24

进行秘密SYN扫描，对象为主机scanme所在的"C类"网段的255台主机。同时尝试确定每台工作主机的操作系统类型。因为进行SYN扫描和操作系统检测，这个扫描需要有根权限。

nmap -sV -p 22,53,110,143,4564 198.116.0-255.1-127

进行主机列举和TCP扫描，对象为B类198.116网段中255个8位子网。这个测试用于确定系统是否运行了sshd、DNS、imapd或4564端口。如果这些端口打开，将使用版本检测来确定哪种应用在运行。