实战：配置DNS客户端域名搜索后缀构造域名进行域名解析 推荐

原创

韩立刚冬青 2013-02-01 11:09:46 博主文章分类：WindowsServer

文章标签 IP地址 计算机 域名 Internet 河北企业 文章分类 服务器 阅读数 8372

2.1 实战：配置客户端使用DNS服务器解析计算机名

实战目的：

ü 在内网，用户习惯使用计算机名而不是计算机全名访问资源，你需要配置客户端，解析计算机名时在计算机名称后添加一个DNS后缀构造一个域名，然后通过DNS服务器解析该计算机的IP地址。

企业环境：

在微软河北企业护航中心网络有一个内部网络，通过ADSL连接到Internet。在内网有一个域环境。Win2k3计算机是工作组中的计算机。使用内网的计算机用户访问内网服务器时，习惯使用服务器名，而不习惯使用服务器全名，网络管理员希望配置客户机能够通过DNS服务器解析服务器名。

实验环境：

ü DCServer是ESS.COM域中的域控制器和DNS服务器。安装Windows Server 2008企业版操作系统，

ü DHCPServer是ESS.COM域中的DHCP服务器，安装Windows Server 2008企业版操作系统，负责给内网计算机分配IP地址。

ü Research服务器是ESS.COM域中的成员服务器，安装Windows Server 2008企业版操作系统，IP地址和DNS设置成自动获得。

ü Sales计算机是ESS.COM域中的成员，安装Vista企业版操作系统，IP地址和DNS设置成自动获得。

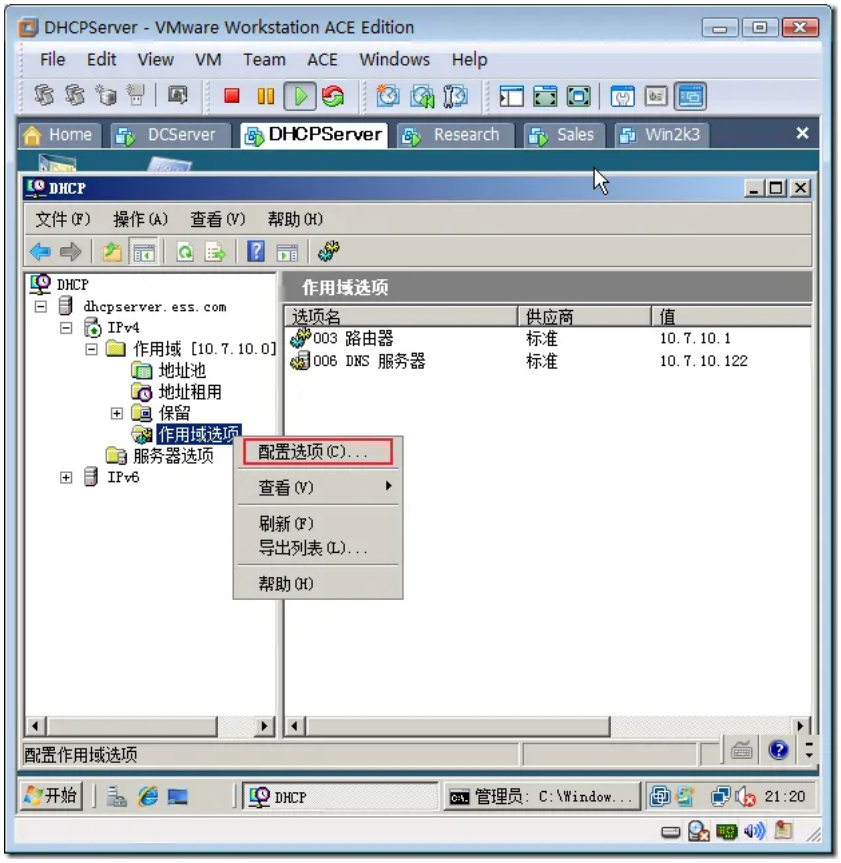
ü Win2k3是工作组中的计算机，安装有Windows Server 2003企业版操作系统。

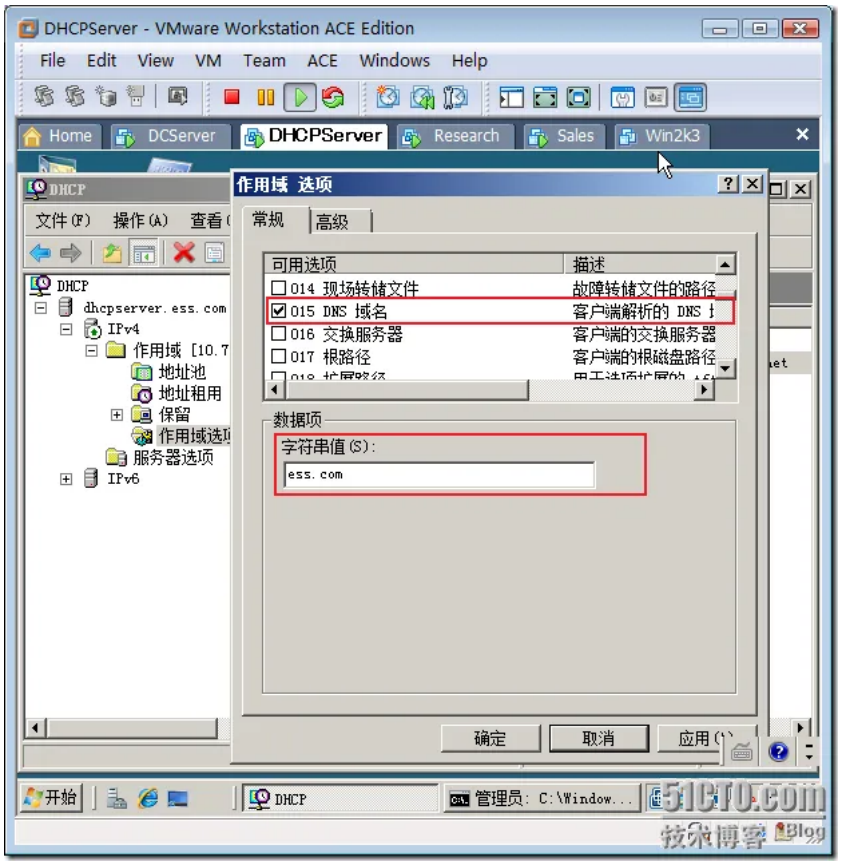
2.1.1 配置DHCP作用域选项DNS域名

在DHCServer上，点击“开始”à“程序”à“管理工具”à“DHCP”，打开DHCP管理工具。

右击“作用域选项”，点击“配置选项”。

在出现的作用域选项对话框，选中“015DNS域名”，输入ess.com，点击“确定”。

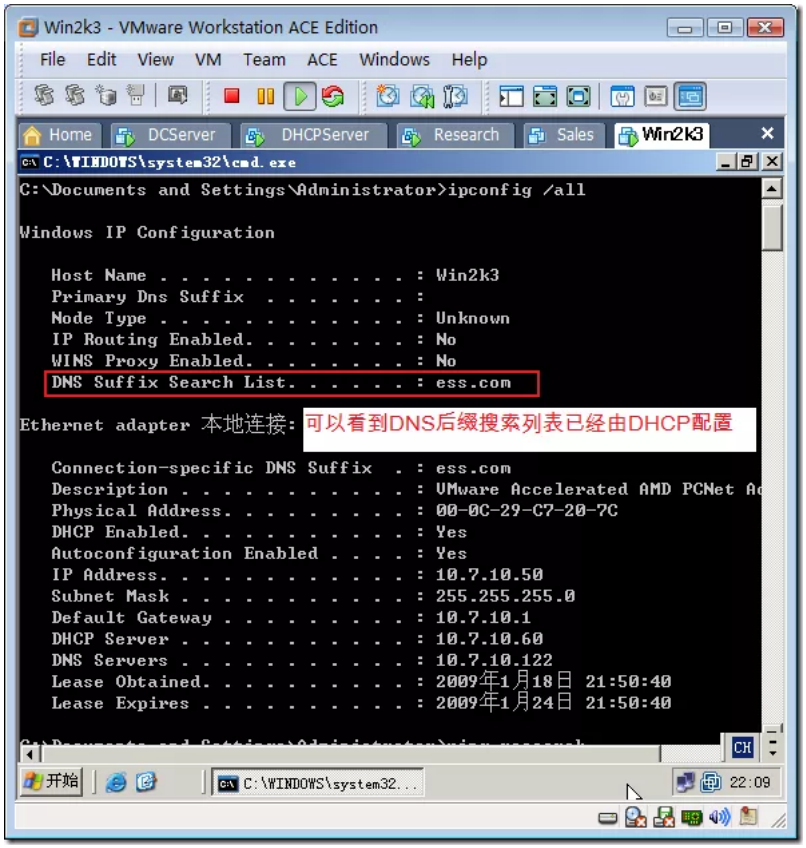


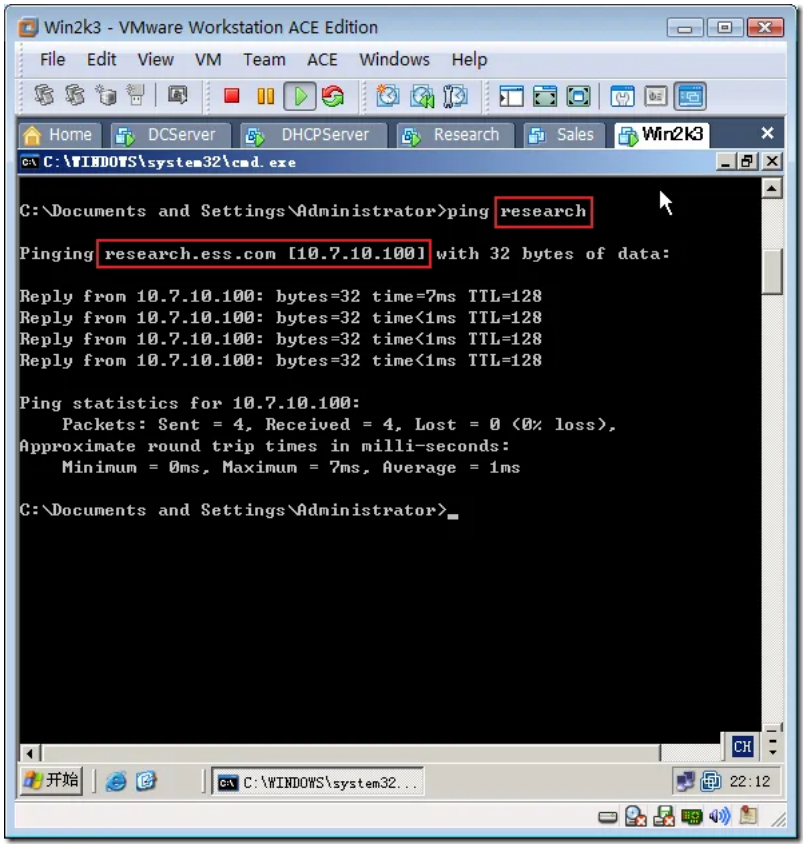


2.1.2 在客户端测试计算机名解析

在工作组中的计算机win2k3上进行名称解析测试。将IP地址设置成自动获得，在命令提示符下输入ipconfig /renew重新刷新地址租约，在输入ipconfig /all可以看到由DHCP服务器配置DNS后缀搜索列表。

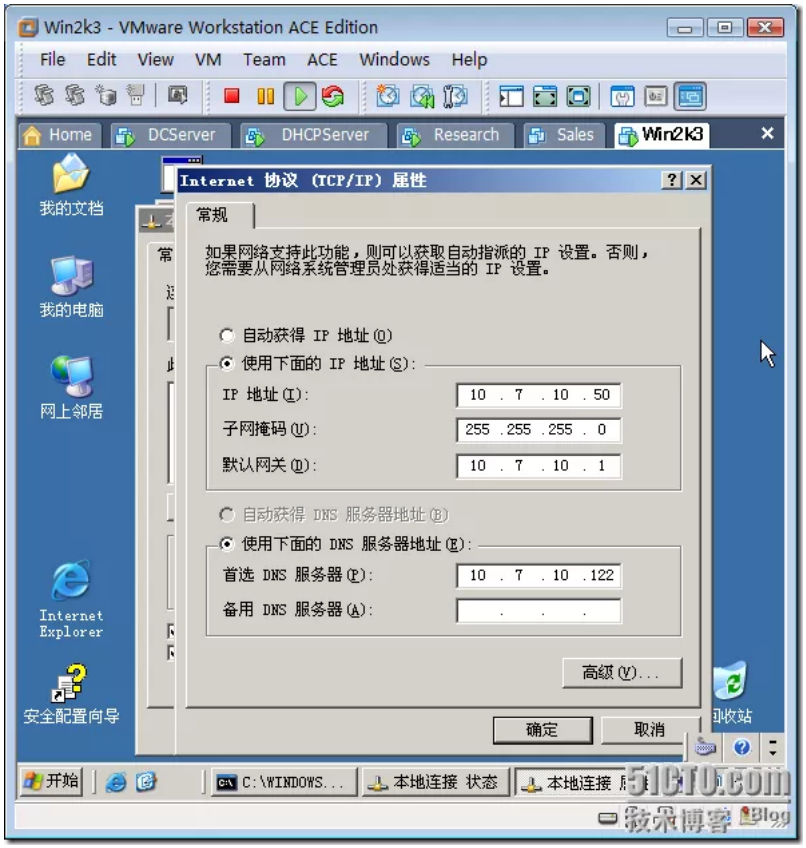
ping Research服务器的计算机名称，注意看解析到的结果是research.ess.com域名对应的IP地址。该解析结果是DNS服务器返回的。

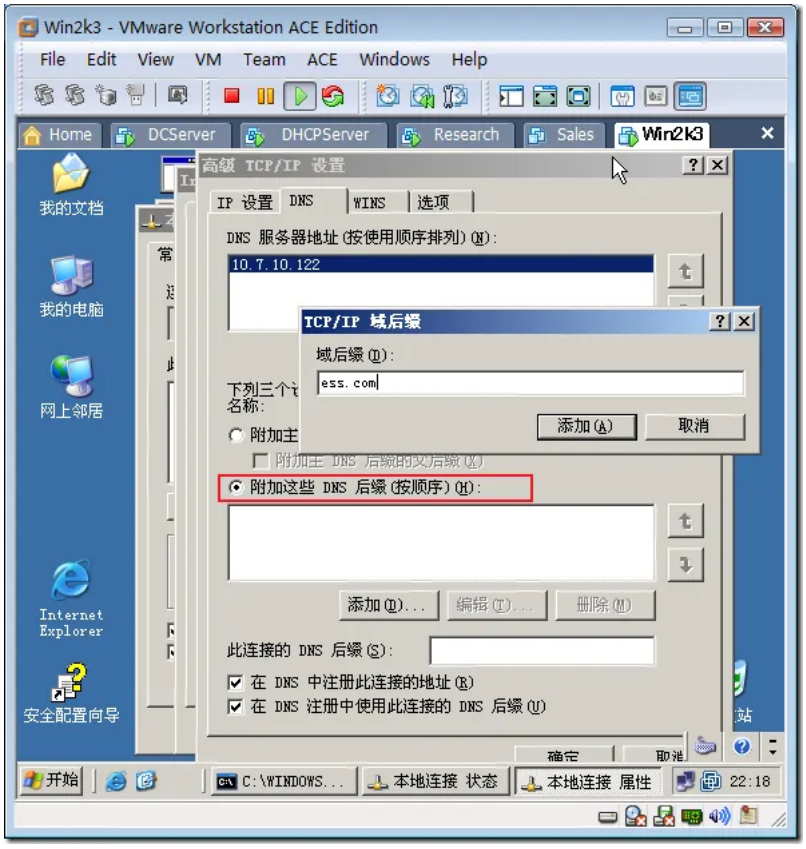




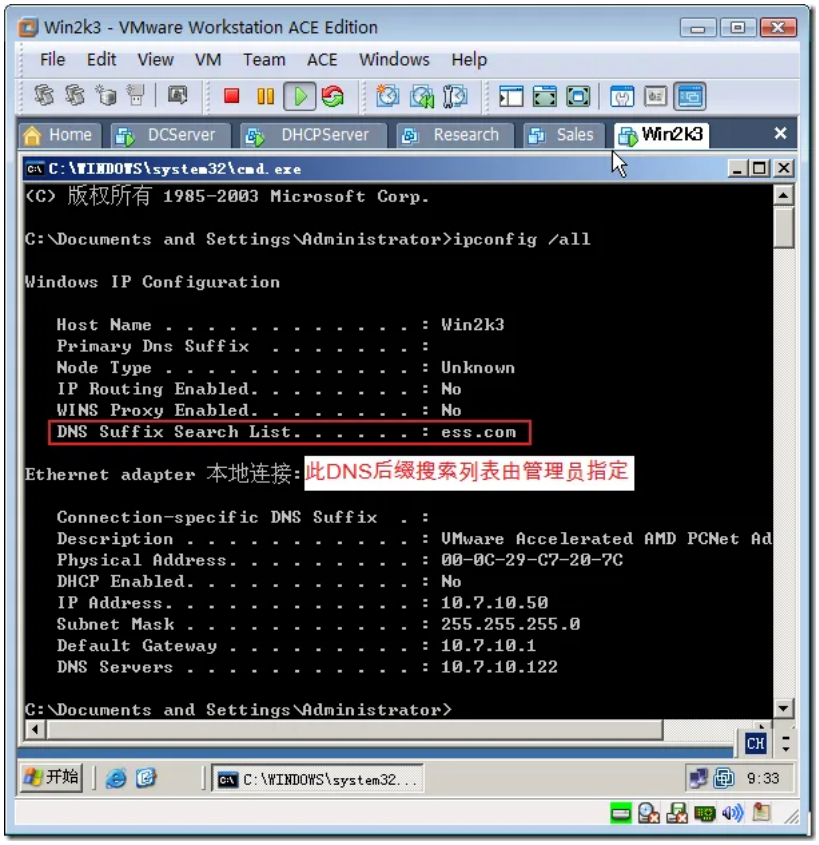
对于使用固定IP地址的计算机，需人工指定DNS后缀搜索列表。如图，打开本地连接的TCP/IP属性对话框，将IP地址和DNS服务器都设置成静态的，点击“高级”。

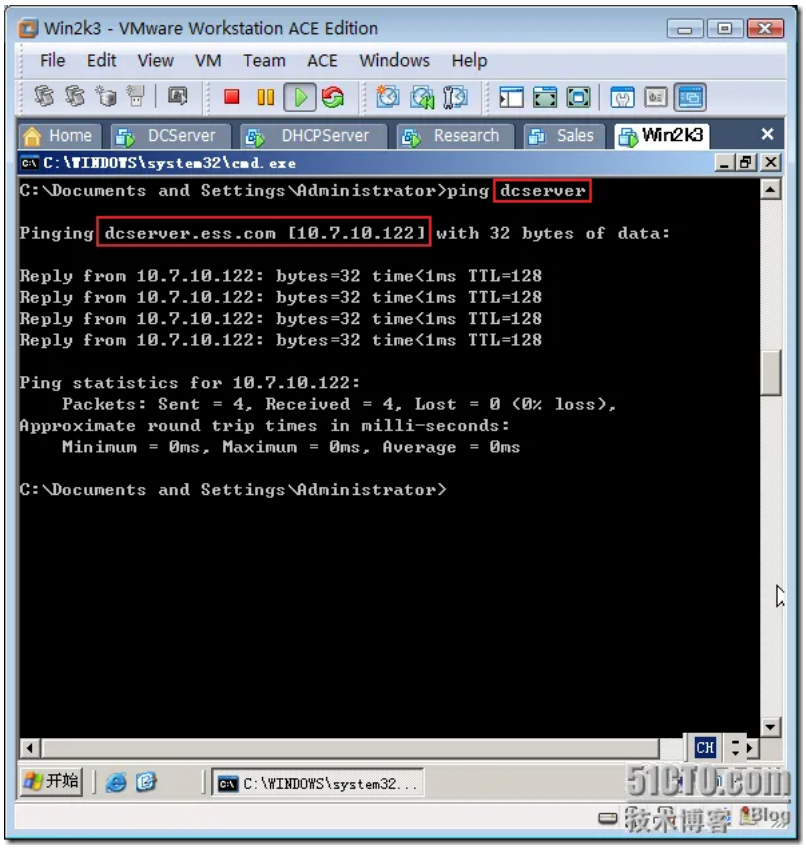
在高级TCP/IP设置对话框DNS标签下，选择“附加下列DNS后缀”，点击“添加”。在出现的对话框输入ess.com点击“添加”。当然您可以多个DNS后缀，客户端会依次添加这些DNS后缀构造域名，查找DNS服务器解析出IP地址。





在命令提示符下，输入ipconfig /all，可以看到DNS后缀搜索列表，已经添加了ess.com，ping dcserver可以看到解析到dcserver.ess.com的IP地址。


Win2k3 - VMware Workstation ACE Edition
File Edit View VM Team ACE Windows Help
Home DCServer DHCPServer Research Sales Win2k3
C:\WINDOWS\system32\cmd.exe
(C) 版权所有 1985-2003 Microsoft Corp.
C:\Documents and Settings\Administrator>ipconfig /all
Windows IP Configuration
Host Name . . . . . : Win2k3
Primary Dns Suffix . . . . . :
Node Type . . . . . : Unknown
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No
DNS Suffix Search List. . . . . : ess.com
Ethernet adapter 本地连接:
此DNS后缀搜索列表由管理员指定
Connection-specific DNS Suffix . :
Description . . . . . : VMware Accelerated AMD PCNet Ad
Physical Address. . . . . : 00-0C-29-C7-20-7C
DHCP Enabled. . . . . : No
IP Address. . . . . : 10.7.10.50
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 10.7.10.1
DNS Servers . . . . . : 10.7.10.122
C:\Documents and Settings\Administrator>
开始
C:\WINDOWS\system32...
9:33


Win2k3 - VMware Workstation ACE Edition
File Edit View VM Team ACE Windows Help
Home DCServer DHCPServer Research Sales Win2k3
C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\Administrator>ping dcserver
Pinging dcserver.ess.com [10.7.10.122] with 32 bytes of data:
Reply from 10.7.10.122: bytes=32 time<1ms TTL=128
Reply from 10.7.10.122: bytes=32 time<1ms TTL=128
Reply from 10.7.10.122: bytes=32 time<1ms TTL=128
Reply from 10.7.10.122: bytes=32 time<1ms TTL=128
Ping statistics for 10.7.10.122:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms
C:\Documents and Settings\Administrator>
开始
C:\WINDOWS\system32...
51CTO.com
技术博客 Blog

resolv.conf：search、domain、nameserver解释

resolv.conf是resolver类库使用的配置文件，每当一个程序需要通过域名来访问internet上面的其它主机时，需要利用该类库将域名转换成对应的IP，然后才可进行访问.

resolv.conf文件的配置选项不多，从man文档中看了半天，不理解domain和search使用来干嘛的。这里做个解释，防止以后忘了（环境：ubuntu12.04）：

nameserver x.x.x.x该选项用来制定DNS服务器的，可以配置多个nameserver指定多个DNS。

domain mydomain.com这个用来指定本地的域名，在没有设置search的情况下，search默认为domain的值。这个值可以随便配，目前在我看来，domain除了当search的默认值外，没有其它用途。也就说一旦配置search，那domain就没用了。

search google.com baidu.com该选项可以用来指定多个域名，中间用空格或tab键隔开。它是干嘛的呢?

如：在没有配置该选项时，执行

```
#ping  new
sping: unknown host news
```

配置search google.com baidu.com后，再执行

```
#ping news
PING news.google.com (74.125.128.101) 56(84) bytes of data.
64 bytes from hg-in-f101.1e100.net (74.125.128.101): icmp_req=1 ttl=47 time=78.9 ms
64 bytes from hg-in-f101.1e100.net (74.125.128.101): icmp_req=2 ttl=47 time=63.6 ms
```

它就去ping news.google.com了。原来当访问的域名不能被DNS解析时，resolver会将该域名加上search指定的参数，重新请求DNS，直到被正确解析或试完search指定的列表为止。

由于news不能被DNS解析，所以去尝试news.google.com，被正常解析。如果没有被解析还会去尝试news.baidu.com。

参考链接：http://dns-learning.twnic.net.tw/bind/intro4.html

設定 /etc/resolv.conf 檔案

此檔案可用來設定 DNS 用戶端要求名稱解析時，所定義的各項內容。我們分別來看一個完整的resolv.conf的檔案:

```
domain twnic.com.tw
nameserver 192.168.10.1
nameserver 192.168.2.5
search twnic.com.tw twnic.net.tw
```

- “domain”指定本地的網域名稱，如果查詢時的名稱沒有包含小數點，則會自動補上此處的網域名稱為字尾再送給DNS伺服器。
- “nameserver”指定用戶端要求進行名稱解析的 nameserver IP位址，在此可指定多部DNS伺服器，則用戶端將會依序提出查詢要求。
- “search”這個選項為非必要選項，而功能在於若使用者指定主機名稱查詢時，所需要搜尋的網域名稱。例如，當我們設 “search twnic.com.tw”時，當DNS伺服器在做名稱解析過程中，無法對輸入的名稱，例如pc1，找出相對應的IP時，則DNS會利用search的設定值加上需查詢的名稱，即pc1.twnic.com.tw來進行解析，解析失敗時則會嘗試pc1.twnic.net.tw。
- 需要注意的是當我們想嘗試多種在沒有包含小數點，於字尾補上所需要搜尋的網域名稱時，我們會在"search"中指定幾種組合給DNS伺服器，而不能在"domain"中指定。因為“domain”是指定本地的網域名稱，而搜尋時也以“domain”為優先嘗試，如果失敗之後才會嘗試"search"中的組合。

resolv.conf(5) — Linux manual page

NAME top

resolv.conf - resolver configuration file

SYNOPSIS top

/etc/resolv.conf

DESCRIPTION top

The resolver is a set of routines in the C library that provide access to the Internet Domain Name System (DNS). The resolver configuration file contains information that is read by the resolver routines the first time they are invoked by a process. The file is designed to be human readable and contains a list of keywords with values that provide various types of resolver information. The configuration file is considered a trusted source of DNS information; see the trust-ad option below for details.

If this file does not exist, only the name server on the local machine will be queried, and the search list contains the local domain name determined from the hostname.

The different configuration options are:

nameserver Name server IP address
Internet address of a name server that the resolver should query, either an IPv4 address (in dot notation), or an IPv6 address in colon (and possibly dot) notation as per RFC 2373. Up to MAXNS (currently 3, see <resolv.h>) name servers may be listed, one per keyword. If there are multiple servers, the resolver library queries them in the order listed. If no nameserver entries are present, the default is to use the name server on the local machine. (The algorithm used is to try a name server, and if the query times out, try the next, until out of name servers, then repeat trying all the name servers until a maximum number of retries are made.)

search Search list for host-name lookup.

By default, the search list contains one entry, the local domain name. It is determined from the local hostname returned by gethostname(2); the local domain name is taken to be everything after the first '.'. Finally, if the hostname does not contain a '.', the root domain is assumed as the local domain name.

This may be changed by listing the desired domain search path following the search keyword with spaces or tabs separating the names. Resolver queries having fewer than ndots dots (default is 1) in them will be attempted using each component of the search path in turn until a match is found. For environments with multiple subdomains please read options ndots:n below to avoid man-in-the-middle attacks and unnecessary traffic for the root-dns-servers. Note that this process may be slow and will generate a lot of network traffic if the servers for the listed domains are not local, and that queries will time out if no server is available for one of the domains.

If there are multiple search directives, only the search list from the last instance is used.

In glibc 2.25 and earlier, the search list is limited to six domains with a total of 256 characters. Since glibc 2.26, the search list is unlimited.

The domain directive is an obsolete name for the search directive that handles one search list entry only.

sortlist
This option allows addresses returned by gethostbyname(3) to be sorted. A sortlist is specified by IP-address-netmask pairs. The netmask is optional and defaults to the natural netmask of the net. The IP address and optional network pairs are separated by slashes. Up to 10 pairs may be specified. Here is an example:

```
sortlist 130.155.160.0/255.255.240.0 130.155.0.0
```

options
Options allows certain internal resolver variables to be modified. The syntax is

```
options option ...
```

where option is one of the following:

debug Sets RES_DEBUG in _res.options (effective only if glibc was built with debug support; see resolver(3)).

ndots:n
Sets a threshold for the number of dots which must appear in a name given to res_query(3) (see resolver(3)) before an initial absolute query will be made. The default for n is 1, meaning that if there are any dots in a name, the name will be

tried first as an absolute name before any search list elements are appended to it. The value for this option is silently capped to 15.

timeout:n
Sets the amount of time the resolver will wait for a response from a remote name server before retrying the query via a different name server. This may not be the total time taken by any resolver API call and there is no guarantee that a single resolver API call maps to a single timeout. Measured in seconds, the default is RES_TIMEOUT (currently 5, see <resolv.h>). The value for this option is silently capped to 30.

attempts:n
Sets the number of times the resolver will send a query to its name servers before giving up and returning an error to the calling application. The default is RES_DFLRETRY (currently 2, see <resolv.h>). The value for this option is silently capped to 5.

rotate Sets RES_ROTATE in _res.options, which causes round-robin selection of name servers from among those listed. This has the effect of spreading the query load among all listed servers, rather than having all clients try the first listed server first every time.

no-check-names
Sets RES_NOCHECKNAME in _res.options, which disables the modern BIND checking of incoming hostnames and mail names for invalid characters such as underscore (_), non-ASCII, or control characters.

inet6 Sets RES_USE_INET6 in _res.options. This has the effect of trying an AAAA query before an A query inside the gethostbyname(3) function, and of mapping IPv4 responses in IPv6 "tunneled form" if no AAAA records are found but an A record set exists. Since glibc 2.25, this option is deprecated; applications should use getaddrinfo(3), rather than gethostbyname(3).

ip6-bytestring (since glibc 2.3.4 to 2.24)
Sets RES_USEBSTRING in _res.options. This causes reverse IPv6 lookups to be made using the bit-label format described in RFC 2673; if this option is not set (which is the default), then nibble format is used. This option was removed in glibc 2.25, since it relied on a backward-incompatible DNS extension that was never deployed on the Internet.

ip6-dotint/no-ip6-dotint (glibc 2.3.4 to 2.24)
Clear/set RES_NOIP6DOTINT in _res.options. When this option is clear (ip6-dotint), reverse IPv6

lookups are made in the (deprecated) ip6.int zone; when this option is set (no-ip6-dotint), reverse IPv6 lookups are made in the ip6.arpa zone by default. These options are available in glibc versions up to 2.24, where no-ip6-dotint is the default. Since ip6-dotint support long ago ceased to be available on the Internet, these options were removed in glibc 2.25.

edns0 (since glibc 2.6)
Sets RES_USE_EDNS0 in _res.options. This enables support for the DNS extensions described in RFC 2671.

single-request (since glibc 2.10)
Sets RES_SNGLKUP in _res.options. By default, glibc performs IPv4 and IPv6 lookups in parallel since version 2.9. Some appliance DNS servers cannot handle these queries properly and make the requests time out. This option disables the behavior and makes glibc perform the IPv6 and IPv4 requests sequentially (at the cost of some slowdown of the resolving process).

single-request-reopen (since glibc 2.9)
Sets RES_SNGLKUPREOP in _res.options. The resolver uses the same socket for the A and AAAA requests. Some hardware mistakenly sends back only one reply. When that happens the client system will sit and wait for the second reply. Turning this option on changes this behavior so that if two requests from the same port are not handled correctly it will close the socket and open a new one before sending the second request.

no-tld-query (since glibc 2.14)
Sets RES_NOTLDQUERY in _res.options. This option causes res_nsearch() to not attempt to resolve an unqualified name as if it were a top level domain (TLD). This option can cause problems if the site has ``localhost'' as a TLD rather than having localhost on one or more elements of the search list. This option has no effect if neither RES_DEFNAMES or RES_DNSRCH is set.

use-vc (since glibc 2.14)
Sets RES_USEVC in _res.options. This option forces the use of TCP for DNS resolutions.

no-reload (since glibc 2.26)
Sets RES_NORELOAD in _res.options. This option disables automatic reloading of a changed configuration file.

trust-ad (since glibc 2.31)
Sets RES_TRUSTAD in _res.options. This option controls the AD bit behavior of the stub resolver. If a validating resolver sets the AD bit in a

response, it indicates that the data in the response was verified according to the DNSSEC protocol. In order to rely on the AD bit, the local system has to trust both the DNSSEC-validating resolver and the network path to it, which is why an explicit opt-in is required. If the trust-ad option is active, the stub resolver sets the AD bit in outgoing DNS queries (to enable AD bit support), and preserves the AD bit in responses. Without this option, the AD bit is not set in queries, and it is always removed from responses before they are returned to the application. This means that applications can trust the AD bit in responses if the trust-ad option has been set correctly.

In glibc version 2.30 and earlier, the AD is not set automatically in queries, and is passed through unchanged to applications in responses.

The search keyword of a system's resolv.conf file can be overridden on a per-process basis by setting the environment variable LOCALDOMAIN to a space-separated list of search domains.

The options keyword of a system's resolv.conf file can be amended on a per-process basis by setting the environment variable RES_OPTIONS to a space-separated list of resolver options as explained above under options.

The keyword and value must appear on a single line, and the keyword (e.g., nameserver) must start the line. The value follows the keyword, separated by white space.

Lines that contain a semicolon (;) or hash character (#) in the first column are treated as comments.

FILES top

/etc/resolv.conf, <resolv.h>

SEE ALSO top

gethostbyname(3), resolver(3), host.conf(5), hosts(5), nsswitch.conf(5), hostname(7), named(8)

Name Server Operations Guide for BIND

COLOPHON top

This page is part of release 5.13 of the Linux man-pages project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at https://www.kernel.org/doc/man-pages/.