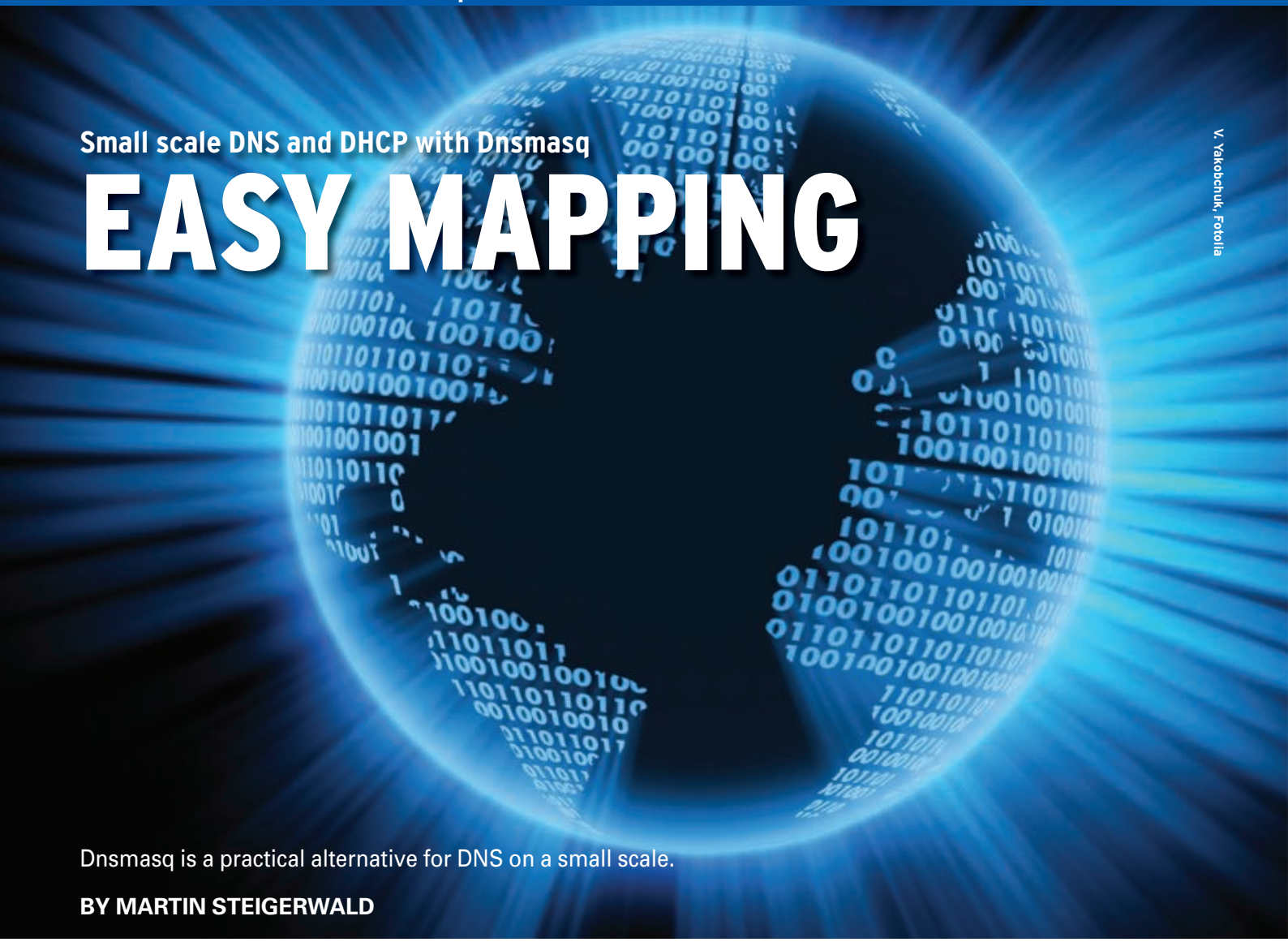

V. Yakobchuk, Fotolia

Small scale DNS and DHCP with Dnsmasq

EASY MAPPING

Dnsmasq is a practical alternative for DNS on a small scale.

BY MARTIN STEIGERWALD

DNS provides a means for associating domain names with IP addresses. A system of DNS servers operating on the Internet delivers the seamless address mapping that lets you surf the web with names instead of numbers. But what if you have a very small network that resides behind a firewall with network address translation? A simple local network doesn't need anything close to the functionality provided by a full-blown DNS implementation. Typically, it is quite enough to respond to DNS requests from hosts on the local network and forward all other requests to the provider's DNS server.

Dnsmasq is a simple, lightweight implementation of DNS, DHCP, and TFTP for small, local networks. This practical tool serves up addresses for the local network and forwards requests for external hosts to any DNS server. Because Dnsmasq integrates DNS with DHCP, it remembers the names of hosts that receive addresses through DHCP. This approach significantly reduces configuration time. All you need to do is set up each client to use DHCP, and you won't need to maintain an */etc/hosts* file for hostname-to-address mapping. The program is frugal in its use of resources, and it is therefore well suited for deployment on Linux-based router systems.

Several popular Linux distributions have Dnsmasq in their repositories. On Debian, Ubuntu, and openSUSE, you need to install the *dnsmasq* package. The popular router distributions OpenWrt, DD-WRT, and FreeWRT all include the program [1] [2]. In our lab, Dnsmasq ran on Debian from a USB stick attached to an Asus WL-500g Premium DSL router (Figure 1) [3].

Setting Up Dnsmasq

The Dnsmasq configuration file is */etc/dnsmasq.conf*. However, you have a more convenient configuration option than editing this file and redoing your changes whenever you update the program: *conf-dir=/etc/dnsmasq.d* lets you load the configuration files from the */etc/dnsmasq.d* directory. Alternatively, *conf-file* loads a single configuration file. The man page for the software documents the configuration options. After making changes, you can type */etc/init.d/dnsmasq restart* to tell the server to parse the new configuration.

The DNS server is very easy to configure. All you need to do is create a tiny configuration file named, say, */etc/dnsmasq.d/dns* (Listing 1). The *domain-needed* line tells Dnsmasq not to

Listing 1: Configuring DNS in Dnsmasq

```
# Be friendly to upstream DNS servers
domain-needed
bogus-priv
# Filter some Windows DNS requests
filterwin2k
# Only listen on LAN interfaces
interface=eth0.1
bind-interfaces
# Local domain
domain=lichtvoll.home
```

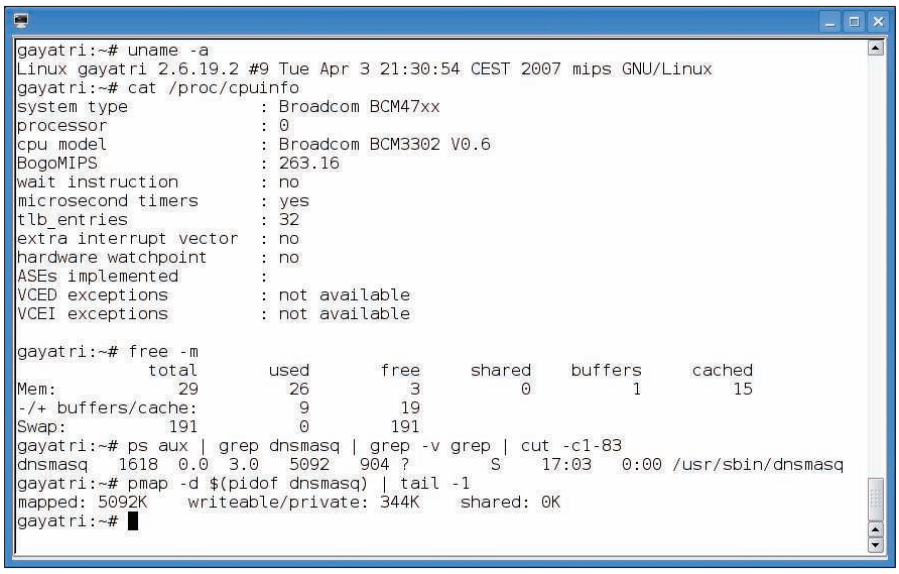

Figure 1: Dnsmasq runs without problem on a small DSL router.

ask the upstream name server unless the requested hostname includes a domain name. The *bogus-priv* line tells the tool not to pass requests for IP addresses (aka reverse lookups) to the upstream DNS server if they originate in private IP address ranges (see the "Private IP Address Ranges" box).

The *interface* and *bind-interfaces* instructions tell the DNS server only to listen for requests on the local network. Finally, *domain* specifies the local domain, which is freely configurable. It is not a good idea to use domains that already exist on the Internet. The *.home* domain is always a good choice, and you need to add it as your clients' */etc/resolv.conf* *search* option. If you specify a hostname without a domain, the DNS software will attempt to resolve the name by appending the local domain; this saves some typing.

This sample configuration leaves out one important consideration: How does Dnsmasq know which DNS server is available for host requests on the Internet? Dnsmasq takes this setting from the */etc/resolv.conf* file on the computer on which it is running. This file stores the IP addresses of up to three name servers tagged with the *nameserver* label. The Debian *resolvconf* package uses an even more elegant approach: The list of DNS servers for Dnsmasq is stored in */var/run/dnsmasq/resolv.conf*, and */etc/resolv.conf* only points to *127.0.0.1* as the name server [4].

The *resolv-file* instruction tells Dnsmasq to load its DNS servers from a different file, and *server* lets you add DNS servers directly to the configuration file. For example, the following entry supports use of the DNS servers at OpenDNS [5]:

```
# OpenDNS.com DNS-Server
server=208.67.222.222
server=208.67.220.220
```

Alternatively, you can integrate an existing DNS server for your network or other domains. In addition, Dnsmasq adds entries from */etc/hosts* to its DNS, which also is where you would add the name of the host on which Dnsmasq is running. The recommended order according to the *hosts* man page is to start with the IP address, followed by the hostname and the domain, and finally the hostname without the domain, all separated by spaces.

For a host on the local network, the domain must match the domain set in Dnsmasq; name resolution will not work if this is not the case.

DHCP in a Single Line

DHCP also is enabled quickly. The program uses the option

```
dhcp-range=↗
10.0.1.9,10.0.1.99,12h
```

in a file such as */etc/dnsmasq.d/dhcp* to assign dynamic leases with IP addresses between *10.0.1.9* and *10.0.1.99*, valid for 12 hours. Thanks to the long lease duration, the DHCP clients running on the hosts do not need to drop their IP addresses in case of a server outage.

If you so desire, you can use *dhcp-host* to assign static addresses to specific hosts. Listing 2 shows an example. Dnsmasq identifies hosts by their MAC addresses, their network interface, their hostname, or their DHCP client ID.

In the ISC DHCP client configuration file, you need an option such as *send host-name "Hostname"* (*/etc/dhcp3/dhclient.conf* for Debian and its derivates) to send the hostname. Alternatively, *send dhcp-client-identifier* will send a DHCP client ID (see *man dhcp-options*). OpenSUSE uses *dhcpcd* as its standard DHCP client, which sends hostnames by default. Alternatively, you can enable the ISC client by specifying *dhclient* for the *DHCLIENT_BIN* variable in the */etc/sysconfig/dhcp* file on your system. The configuration is in */etc/dhclient.conf*.

To discover the MAC address of a network interface on a Linux system, you can use *ip link* or *ifconfig -a*. The ARP cache contains the MAC addresses of the computers your Linux system talked

Private IP Address Ranges

In line with RFC 1918, routers do not forward some IP address ranges onto the Internet. The ranges are as follows:

- 192.168.0.0/16: 168.168.0.1 to 168.168.255.254
- 172.16.0.0/12: 172.16.0.1 to 172.31.255.254
- 10.0.0.0/8: 10.0.0.1 to 10.255.255.254

These address ranges are thus useful for setting up local networks. The IP addresses specified can be used for hosts.

Listing 2: Assigning static addresses

```
# Dynamic DHCP
dhcp-range=10.0.1.9,10.0.1.99,12h
# ThinkPad T42
dhcp-host=shambhala,10.0.0.21
# ThinkPad T23
dhcp-host=deepdance,10.0.0.99
# Sam440ep
dhcp-host=00:50:c2:5a:44:e9,gaia,10.0.0.5
# Amiga 4000 with usb ethernet adaptor
dhcp-host=00:80:c9:40:00:c0,sunshine,10.0.0.77
```

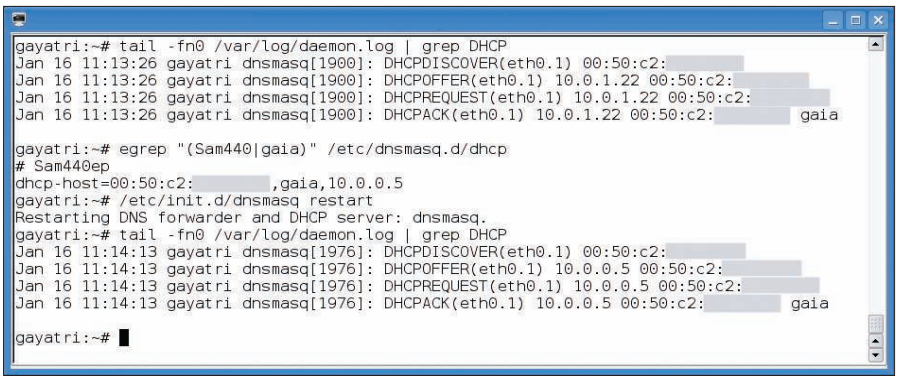

Figure 2: Setting up a DHCP host entry for a MAC address.

to last. The *ip neigh* or *arp* instruction displays the cache content. If the device you are looking for is missing from the list, pinging the device's IP address will put its MAC address in the ARP cache.

Alternatively, you can use DHCP to pick up a dynamic lease for the client and then monitor the Dnsmasq log by entering *tail -f /var/log/daemon.log | grep DHCP* on Debian, or *tail -f /var/log/messages | grep DHCP* on openSUSE (Figure 2).

Everything Working?

Now that you have set up DNS and DHCP, it is time to take Dnsmasq out for a run. To avoid difficulty, make sure you only run one DHCP server in each network segment. Stop all other DHCP services, such as the server on the access point, before this test.

Then type */etc/init.d/dnsmasq restart* to restart Dnsmasq.

The ISC DHCP client is recommended for initial tests on Linux. First deconfigure the network interface (typically *eth0*) by typing *ifdown eth0*, and then type *dhclient eth0* to launch the DHCP client. If you see output like that shown in Figure 3, DHCP-based configuration is working. The */var/lib/misc/dnsmasq.leases* file on the computer running the server has a list of the assigned leases (Figure 4).

Otherwise, type *netstat -tulpen | grep dnsmasq* to check that Dnsmasq is running. The software listens on UDP port 67 for DHCP, and on TCP and UDP ports 53 for DNS. If you type an option wrong, Dnsmasq outputs an error message on startup. The logfile contains more hints. The *resolv.conf* file typically contains the Dnsmasq server as a name server. By entering something like *host linux-user.de*, you can check name resolution.

If everything is working as expected, you can launch YaST to permanently set the interface to DHCP in *Network hardware | Network settings | Overview*. For Debian and its derivates, you need to replace the *static* option in *iface interface inet static* with *dhcp* and remove the manual configuration lines that follow.

An *ifup eth0* triggers DHCP-based configuration of the interface, if this has not already happened. Alternatively, you can use the Network Manager, which is enabled in YaST in *Network hardware | Network settings | Global Options | Network connection method*. On Debian and its derivates, the Network Manager manages any interfaces not configured in */etc/network/interfaces*.

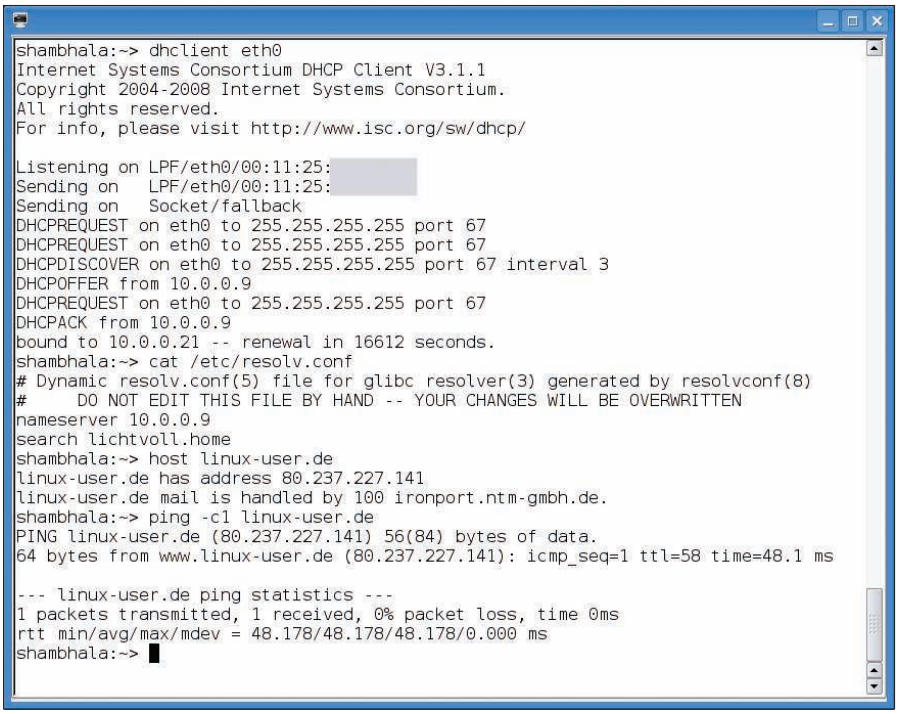

Figure 3: If DHCP and DNS are working, this is what you should see.

Conclusions

Dnsmasq keeps to functions that make sense on a small local network with Internet access. It works reliably – once configured you will quickly forget that it is running. The configuration is simple and easy to follow. ■

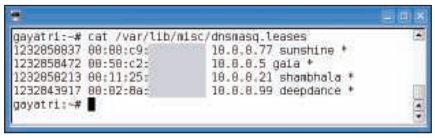

Figure 4: A LAN party with two Linux laptops and two Amiga systems.

INFO

[1] Dnsmasq in OpenWrt: *http://wiki.openwrt.org/OpenWrtDocs/dnsmasq*

[2] Dnsmasq as a DHCP server: *http://www.dd-wrt.com/wiki/index.php/DNSMasq_as_DHCP_server*

[3] Debian on the Asus WL-500g Deluxe: *http://wpkg.org/Running_Debian_on_ASUS_WL-500G_deluxe*

[4] Dnsmasq in Debian: */usr/share/doc/dnsmasq/README.Debian*

[5] OpenDNS: *http://www.opendns.com/*

THE AUTHOR

Martin Steigerwald works as a trainer, consultant, and system administrator for team(ix) GmbH in Nuremberg. His work mainly focuses on Linux training, as well as designing, installing, and maintaining robust IT infrastructures based on Debian.