Webmin & DNSMASQ

Create a snapshot! (Use the same VM we used last week that already has BIND running)

# Webmin

Webmin is a web-based interface for system administration for Unix. Using any browser that supports tables and forms (and Java for the file manager module), you can setup user accounts, Apache, DNS, file sharing and so on.

Webmin consists of a simple web server, and a number of CGI programs which directly update system files like /etc/inetd.conf and /etc/passwd. The web server and all CGI programs are written in Perl v5, and use no non-standard Perl modules

Reference: [www.webmin.com](http://www.webmin.com)

# To Install Webmin

To install webmin, we need make sure all our package repos are up to date, so we’ll run the following command:

* Run **sudo apt update**, then **sudo apt upgrade**

From here, we have add the webmin repository to our list of available package repository sources. Fortunately, the webmin project provides a simple script to do the hard work for you. You just need to download it and run it:

**sudo wget** [**https://raw.githubusercontent.com/webmin/webmin/master/setup-repos.sh**](https://raw.githubusercontent.com/webmin/webmin/master/setup-repos.sh)

(feel free to have a look at the file – see what it’s doing)

**sudo sh ./setup-repos.sh**

Text

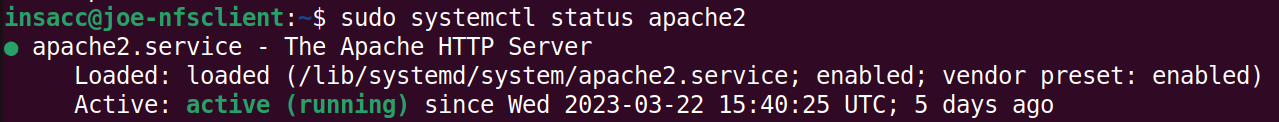
Description automatically generated

Next, we’ll re-run **sudo apt update**, and we’ll notice we’re now getting package updates from webmin as well as ubuntu.com

Text

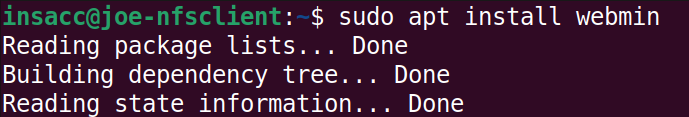
Description automatically generated

NOTE: We need to make sure apache is installed and running before, as webmin will attempt to create a new site for our administration.



Once we have the repository configured (added), we can run the following to install webmin:

**sudo apt install webmin**



This installs webmin and creates the necessary configuration files, as well as a website already running at the url mentioned in the log. In this case, it’s [**https://ubuntu-client:10000/**](https://ubuntu-client:10000/)

Depending on your /etc/hosts file or a previously set up DNS, you can use whatever local url is present.

Here’s my hosts file:

Text

Description automatically generated

So, the rest of the lecture we’ll be administering everything from coos291.com. Let’s open it in firefox: coos291.com:10000

Graphical user interface, text, application, email

Description automatically generated

We’ll get a security warning when we try to browse to the site over HTTPS without a valid security certificate. We’ll bypass for now and install a certificate within the interface. We’ll be given a prompt to log in:

Graphical user interface, application

Description automatically generated

Once we’ve signed, we’ll be presented with the **Webmin Dashboard**:

Graphical user interface, application

Description automatically generated

This will give you a high-level view of your system configuration, resources being used, as well as other things that are of interest. We’ll notice on the side to the left, we have a navigation menu, we’ll get to that in a bit.

For now, we’re going to change our System hostname as well as attempt to retrieve a certificate for that hostname. Click on the bell icon on the right then choose System hostname “joe-nfsclient.localdomain” (which will be different on your installation – ex: yourname.looneytunes)

Graphical user interface, text, application

Description automatically generated

From this screen, we’ll change our hostname as well as the IP address for our DNS Server. We’ll use whatever IP address our machine has, since we’re turning it into a DNS server as well.

Graphical user interface, text, application

Description automatically generated

Save and go back. After that, we’ll need to enable encryption and get our certificate.

From the left side navigation, click Webmin, then Webmin Configuration

Graphical user interface, application, Word

Description automatically generated

Graphical user interface, application

Description automatically generated

Then select SSL Encryption, then the Let’s Encrypt tab on the top right:

Graphical user interface, text, application, email

Description automatically generated

Click Request Certificate:

Text

Description automatically generated

It’s ok if this fails, just know that this is built into Webmin. 😊

There’s a plethora of things we can administer through Webmin. From the System menu:

For instance, we can configure any of the services, like Apache2: Graphical user interface, application, email

Description automatically generated

Text

Description automatically generated

We can change use passwords:

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

The Users and Groups let’s us configure users and groups, the listing matches that of the /etc/passwd file:

A screenshot of a computer

Description automatically generated

We can change use settings, like their default shell:

Graphical user interface, text, application, email

Description automatically generatedText

Description automatically generated

Change the shell to /bin/sh:

Graphical user interface, text, application, email

Description automatically generated A picture containing text

Description automatically generated

Make sure to put it back!

Looking at the Servers section in the left-nav:

Graphical user interface

Description automatically generated

(You should also have BIND!)

We can see we can administer Apache2 through webmin as well. We can see a list of virtual hosts, create virtual hosts, configure global settings etc:

Graphical user interface, table

Description automatically generated with medium confidence

Choosing one of the VH’s gives us numerous configuration options as well, some of which should look very familiar to you!

Graphical user interface, application, Teams

Description automatically generated

Example:

Graphical user interface, application

Description automatically generated

We know how to do this via the command line!

From the tools menu in the left-nav, we can see the different tools available to us:

For example, we have a traditional UI file manager.

Graphical user interface, application

Description automatically generated

From the Networking section in the left-nav, we have a bunch of options for general network configuration, firewalls, default dns services, etc:

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application, Word

Description automatically generated

The Host Addresses page reads and allows you to modify the /etc/hosts file:

Graphical user interface

Description automatically generated with medium confidence Text

Description automatically generated

Moving on to Hardware, we can administer RAID, LVM, partitions and disks, as well as printers:

Graphical user interface, text, application, email

Description automatically generated

We aren’t concerned with server clustering, so skip the “Cluster” module.

Most of the remaining settings that concern us will be in the Un-used Modules section. I will configure the BIND server here. LDAP is also here and we will look at this next class!

Graphical user interface, text, application, Word

Description automatically generated

Click Install Now

Graphical user interface, text, application, email

Description automatically generated

Text

Description automatically generated

Locate BIND in either Unused Modules or Servers:

Graphical user interface, application

Description automatically generated

Here, we can edit our configuration, add to Access Control Lists, add additional DNS server and check our current config for errors.

Let’s add access to our new DNS server on our network by going to Access Control Lists:

Graphical user interface, text, application, email

Description automatically generated

We can edit other global server configuration options:

Click on Edit Config File under BIND portal.

Graphical user interface, application

Description automatically generated

We will edit the first three files:

Graphical user interface, text, application

Description automatically generated

Let’s edit our /etc/bind/named.conf.options file:

Text

Description automatically generated

The forwarders sections will list other DNS servers that we’ll forward requests if our DNS server cannot fulfill the initial request.

Let’s check out configuration by click on Check BIND Config in our Portal.

Graphical user interface, text, application

Description automatically generatedYAY!!

This will run the command **named-checkzone** behind the scenes

Next, we’ll create our Master Zones for Forward and Reverse lookups. Click on **Create Master Zone** in the portal:

Graphical user interface, application, table

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Make sure you select Forward as your Zone type, enter your domain name, your master server can be prepended with **ns** and fill in an email address. Click on “Create” to create your master zone.

Next, we’ll create some A records for our master zone:

Click on Addresses within the master zone configuration page and set an A record for your DNS Server (current IP address):

Graphical user interface, text, application, email

Description automatically generated

The ns record has been created for us:

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

**We’ll now create a reverse zone.**

Go back to your DNS Portal and Click on Create Master Zone again, but this time select the Reverse option:

Graphical user interface, text, application, email

Description automatically generated

Enter the first 3 octets of your network for your domain name, add your master server, and an email.

We’ll need to create a PTR record in this zone so that when IP address 192.168.56.100 is accessed, DNS will return ns.coos291.com. Do that from Reverse Address section, add your ip address and ns.coos291.com:

Graphical user interface, text, application, email

Description automatically generated

We’ll need to check our BIND Config now after all these changes.

We can stop and restart our BIND via the web interface. In the top right of the BIND portal, we’ll see the following icons:  
A picture containing diagram

Description automatically generated

Next, we’ll make sure we’re using our current IP for our DNS server. Let’s edit our **/etc/resolv.conf** file with the command:

sudo nano /etc/resolv.conf

Text

Description automatically generated

Now, when we start doing nslookups and digs, we’ll have our DNS server running locally taking care of the requests, then forward other requests to the other IP addresses in our Forwarders section in our bind options file:

Text

Description automatically generated

Our DNS server should be working now, feel free to add other A records to test nslookup and dig.

Close your webmin.

# Installing DNSMASQ

Next we will set up dnsmasq, which is a lightweight caching forwarder for dns.

Let’s start by installing dnsmasq:

Text

Description automatically generated with medium confidence

After installation, the system will try to start dnsmasq. This will fail, but that’s okay. It’s failing because Ubuntu includes another DNS resolver by default, called system-resolved, which is already listening on port 53. We’ll take care of that in a couple minutes.

Next, we need to tell NetworkManager to use our new software (dnsmasq) as the DNS server. To do this, we’ll edit the file /etc/NetworkManager/NetworkManager.conf

Text

Description automatically generated

By default, dnsmasq only forwards requests, it doesn’t store the results (that is, by default it’s a DNS forwarder but not a DNS cache). We can change that. Change to the directory /etc/NetworkManager/dnsmasq.d and create a file called cache.conf (name doesn’t matter, but that’s what I’ll call mine) with the contents:

cache-size=1000

log-queries

Text

Description automatically generated

This sets up dnsmasq to be a cache server with a 1000-entry cache. Also, so that we can see what it’s doing (for investigating only), we’ll log any name server queries to the file **/var/log/syslog**.

Now that everything is configured correctly, we will disable systemd-resolved so that it doesn’t start up automatically anymore, and then reboot. We will also disable BIND to make sure that it isn’t getting in the way of dnsmasq using port 53.

**sudo systemctl disable systemd-resolved named**

**sudo reboot**

Text

Description automatically generated

To check our logs, we can use the command:

grep dnsmasq /var/log/syslog 🡨 allows us to see all the DNS queries

During bootup, potentially a number of lookups can be performed, most notable for NTP. You can also get a complete dump of the cache by sending a USR1 signal to the dnsmasq process:

sudo pkill -USR1 dnsmasq 🡨 dumps the contents of the cache into the log file

grep dnsmasq /var/log/syslog 🡨view the log again to see the cache

Dump the cache, then lookup a new site, and dump the cache again, you should see an entry for it. Doing subsequent lookups for the same URL will return a match much faster.

Next we can adjust some other settings for dnsmasq through the dnsmasq.conf file:

Within that file, we can change the port on which dnsmasq will listen for DNS requests. This defaults to port 53 UDP by default. You can explicitly set the port using the **port** option:



We can disable forwarding of names without a dot or domain part:

Text

Description automatically generated

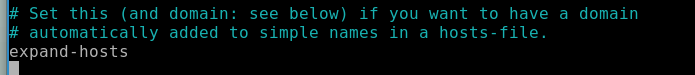
Domain needed is a performance and security improvement as internal queries are not sent to the top level DNS server.

Bogus-priv - a security feature which drops DNS responses containing private IP addresses received from the public internet.

Define an interface (eg: interface=enp0s8) or the IP address (eg: listen-address:192.168.x.x) on which dnsmasq can listen for the DNS requests. This usually defaults to the loopback address.



Enable DNSMASQ to automatically append the domain part to the simple names as well as set the default cache size



Restart dnsmasq to make these additional configuration changes take effect: **sudo systemctl restart dnsmasq**