Beware Here Be Musings

I enjoy using Linux, I have to be paid to use Microsoft Windows.

DNS with bind9 and DHCP on Ubuntu 14.04

Updated for Ubuntu 14.04-3 LTS. If you are a home user and your network has grown such that you are tired of using all static IP addresses and having to configure the /etc/hosts files by hand, then use the great tool dnsmasq. See my How to set up dnsmasq. Dnsmasq is so much simpler to setup and maintain than bind9.

If you are determined to use bind9 and isc-dhcp server then carry on. This is what I do.

I start off with a minimal Ubuntu desktop install with MATE. This is all tried and tested out in a test Networking Lab with VirtualBox.

```
Backup your system before you go any further!
```

This is what we will be installing with this howto.

- DNS Server:
  - Name: dns-server.dragon.org.uk
  - Server Type: Authoritative.
  - Forward Lookup Zone: dragon.lab.
  - Reverse Lookup Zone: 200.1.10.in-addr.arpa.
- Network:
  - Subnet: 10.1.200.0/24
  - DNS/DHCP Server Address: 10.1.200.3
  - DNS/DHCP Server Name: lab-dns-dhcp
  - Gateway Address: 10.1.200.1
  - Broadcast Address: 10.1.200.255
  - Netmask: 255.255.255.0
- DHCP Server:
  - Dynamic Pool: 10.1.200.100 to 10.1.200.119
  - With Updates to DNS (bind9)

The interfaces file should have something like this:

```
auto eth0
iface eth0 inet static
   address 10.1.200.3
   gateway 10.1.200.1
   netmask 255.255.255.0
   dns-nameservers  8.8.8.8
```

Update your system with the latest patches and security fixes.

```
sudo apt-get update
sudo apt-get dist-upgrade
```

Let's pretend it's Windows and reboot as we installed some updates. That gives you a minute or so to grab a very swift coffee 🙂

```
sudo reboot
```

Installing the software is painless both modules should be available from the usual Ubuntu repos.

```
sudo apt-get install isc-dhcp-server bind9
```

DNS Configuration

The configuration files for bind9 are cryptic and not particularly intuitive. It is very easy to break a working setup, let alone fail to get it working, by missing off a single semi-colon or full stop. Talking about full stops, if your system fails to work it is most likely a missing full stop that is stopping it from working. Look at the log output in /var/log/syslog.

The official documentation for bind9 which is rather extensive and very well written, so go there first, there is no need to go anywhere else for help 🙂 Get it from http://www.isc.org it is also installed with the bind9-doc package.

So you insist on following my howto, let us begin. The first configuration file to look at, is one you should not change. Add your changes to the files which are included or to those included further down the branches. Let's have a look at it anyway.

```
cd /etc/bind
less /etc/bind/named.conf
```

```
// This is the primary configuration file for the BIND DNS server named.
//
// Please read /usr/share/doc/bind9/README.Debian.gz for information on the
// structure of BIND configuration files in Debian, *BEFORE* you customize
// this configuration file.
//
// If you are just adding zones, please do that in /etc/bind/named.conf.local

include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
```

To the first config file we will edit.

```
sudo nano named.conf.options
```

```
acl internals {
    localhost;
    localnets;
};

options {
   directory "/var/cache/bind";

   // If there is a firewall between you and nameservers you want
   // to talk to, you may need to fix the firewall to allow multiple
   // ports to talk.  See http://www.kb.cert.org/vuls/id/800113

   // If your ISP provided one or more IP addresses for stable
   // nameservers, you probably want to use them as forwarders.
   // Uncomment the following block, and insert the addresses replacing
   // the all-0's placeholder.

   forwarders {
       // DNS on the internet you could also add
```

```
        // the DNS servers from your ISP
        10.1.200.1;
        // Google's DNS
//      8.8.8.8
    };
    allow-query {
       internals;
    };
    // restrict recursion
    allow-recursion {
       internals;
    };
    allow-transfer {
       internals;
    };
    //========================================================================
    // If BIND logs error messages about the root key being expired,
    // you will need to update your keys.  See https://www.isc.org/bind-keys
    //========================================================================
    // turn off zone encryption. The auto flag still generates
    // warnings in the log file
    dnssec-enable no;
    // dnssec-validation auto;

    listen-on-v6 { any; };
    auth-nxdomain no;    # conform to RFC1035
};
```

Add the section for "acl internals {}", and update the commented out section for forwarders. This limits access to your server and LAN only. It will stop the scum bags in China using your DNS. Not sure why the guys in China are so interested in hacking my servers. You can now use the name **internals** in the three sections for **allow-query, allow-recursion** and **allow-transfer**. These are now limited to our own internal network, LAN.

The forwarders section, where your DNS server asks to lookup IP addrerss for hosts it does not know. This is where you place the DNS from your ISP. I'm using 10.1.200.1 which in the one on my lab-router. You can also use Google's 8.8.8.8, 8.8.4.4 or OpenDNS 208.67.220.220, 208.67.222.222.

The installation process creates the crypto file needed when our new DHCP server talks back to this DNS server. The command below creates a file **/etc/bind/rndc.key** replacing the one generated by the install process, so it's not strictly necessary to run this at all but it will not do any harm to run it again.

```
sudo /usr/sbin/rndc-confgen -a
```

It should look something like this, do not include the secret string in any log posts. 🙂 The one here has been mangled from the original. The text in quotes "**rncd.key**" can be changed to anything you like, within reason.
Just remember that if you do change the text, also update the text where it is accessed in both **/etc/bind/named.conf*** and **/etc/dhcp/dhcpd.conf**

```
key "rndc-key" {
  algorithm hmac-md5;
  secret "Ay6c74c/HELP/MEr26ZPlw==";
};
```

The file contents should be kept secret. We will change the permissions so the casual reader cannot see it.

```
sudo chown root:bind /etc/bind/rndc.key
sudo chmod 640 /etc/bind/rndc.key
```

Adding DNS Zones

Let's get on with defining our first zones. Before that include the **rndc.key** file created above into the named.conf.local file. We will use it again later. The first zone is the forward looking zone for dragon.lab the second one is the reverse lookup zone definition.

The files which the zones point to can be called anything, make them meaningful. They must be placed in /var/lib/bind and should have permissions **644** and be owned:group by **bind:bind**. More of that a little later.

That is all we need here.

```
sudo nano /etc/bind/named.conf.local
```

```
//
// Do any local configuration here
//
include "/etc/bind/rndc.key";

zone "dragon.lab" {
     type master;
     file "/var/lib/bind/dragon.lab.zone";
     allow-update { key rndc-key; };
};

# This is the zone definition for reverse DNS. replace 200.1.10 with your network
# address in reverse notation - e.g my network address is 10.1.200
zone "200.1.10.in-addr.arpa" {
     type master;
     file "/var/lib/bind/dragon.lab.rev.zone";
     allow-update { key rndc-key; };
};

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";
```

I have seen some people add dummy zones like this:

```
zone "facebook.com" {
  type master;
  file "/var/lib/bind/block-domains";
};
```

These zones are designed to stop unwanted advertising, doubleclick.net, or other commonly used junk on web pages like icons to Facebook and Twitter. I believe this type of zone is unwarranted, the user can always install and configure an ad-blocker. It is not up to the **network guy** to block this type of content.

Configuring the Zones

Zones files go in the directory **/var/lib/bind** and should be owned:group by root:bind and be Read/Write by both.

```
sudo nano /var/lib/bind/dragon.lab.zone
```

```
$ORIGIN .
$TTL 907200     ; 1 week 3 days 12 hours
dragon.lab              IN SOA  ns1.dragon.lab. admin.dragon.lab. (
                                2014071403 ; serial
                                28800      ; refresh (8 hours)
```

```
                                3600       ; retry (1 hour)
                                604800     ; expire (1 week)
                                38400      ; minimum (10 hours 40 minutes)
                                )
                        NS      ns1.dragon.lab.
$ORIGIN dragon.lab.
lab-router1             A       10.1.200.1
ns1                     A       10.1.200.3
lab-dns-dhcp            CNAME   ns1   ; the name of the server we are building
```

Make sure you get the syntax correct. **Note** the trailing full stop '.' on the hostname **ns1.dragon.lab.**, (two places) and the email **address admin.dragon.lab.**. Get these wrong and your DNS will not work.

To fully explain what each bit of this file is doing, read the official documentation it is rather good. A quick run through may help out a little here though.

- **$ORIGIN** Sets the domain name that will be appended to any unqualified records.
- **$TTL** The time-to-live of the RR field is a 32-bit integer represented in units of seconds. The TTL describes how long a RR can be cached before it should be discarded.
  Rather than using seconds you can add a suffix for
    - m minutes
    - h hours
    - d days
    - w weeks
- **SOA** Start of Authority record, contains general administrative and control information about the domain. It has the following format
    - Name dragon.lab
    - Class IN
    - Type SOA
    - Name-Server ns1.dragon.lab. (Note the trailing full stop)
    - Email-Address admin.dragon.lab (Note: the @ is replaced with a full stop and there is a trailing one)
    - Serial-No Usually the date as YYYYMMDDSS where SS is a number counting up, used by slave DNS servers.
    - Refresh How often slave DNS server should check back
    - Retry If the above fails retry after this long
    - Expiry Remove data this old when retries fail.
    - Minimum-TTL Time to remember that NXDOMAIN, not found, was returned by the master DNS
- All the **NS, MX, A** and **CNAME** Records for the zone.

The reverse lookup zone file looks like this.

```
sudo nano /var/lib/bind/dragon.lab.rev.zone
```

Here is the complete reverse lookup. The **in-addr.arpa** domain is a legacy domain from the early days of the Internet. We also use PRT records not MX, A or CNAME records here. Watch out for trailing full stops!

```
$ORIGIN .
$TTL 907200     ; 1 week 3 days 12 hours
200.1.10.in-addr.arpa IN SOA    ns1.dragon.lab. admin.dragon.lab. (
                                2014071402 ; serial
                                28800      ; refresh (8 hours)
                                604800     ; retry (1 week)
                                604800     ; expire (1 week)
                                86400      ; minimum (1 day)
                                )
                        NS      ns1.dragon.lab.
$ORIGIN 200.1.10.in-addr.arpa.
```

```
1                       PTR     lab-router1.dragon.lab.
3                       PTR     dns-server.dragon.lab.
                        PTR     dragon.lab.
```

Change the permissions on the two new zone files we created, so they are in the bind group and read and writable by the owner and the group.

```
sudo chown bind:bind /var/lib/bind/*zone
sudo chmod 664 /var/lib/bind/*zone
```

If you will be using this DNS/DHCP server along with Samba4 to create an Active directory server you should also add another CNAME record to /var/lib/bind/dragon.lab.zone at the end./var/lib/bind/dragon.lab.rev.zone.

```
lab-addc1               CNAME   ns1
```

Also add the reverse lookup record to /var/lib/bind/dragon.lab.rev.zone, but this time before the "PTR dragon.lab. line. . Adding these records just means you can access the machine with two different names, in this case dns-dhcp and lab-addc1. I use the hostname lab-addc1 as the SAMBA AD DC server name.

```
3                       PTR     dns-server.dragon.lab.
3                       PTR     lab-addc1.dragon.lab.
                        PTR     dragon.lab.
```

If you want to add some more host names with static IP addresses make the entries in /var/lib/bind/dragon.lab.zone, for lookups and /var/lib/bind/dragon.lab.rev.zone for the reverse lookups. Say we have a machine called **desktop1** which has an IP address of 10.1.200.20 we can use the following:

/var/lib/bind/dragon.lab.zone

```
desktop1              A   10.1.200.20
```

/var/lib/bind/dragon.lab.zone

```
20                      PTR     desktop1.dragon.lab.
```

We should be able to start or restart our DNS server now. Remember to check the error/warning output in your /var/log/syslog. It will point out where you missed off those pesky full stops. 🙂 Use the tail command below in a new terminal window and the restart in your existing terminal. You can see the log lines as they are written.

```
sudo tail -fn10 /var/log/syslog
sudo service bind9 restart
```

If you see any messages like the one below they usually mean you forget to change the permissions above.

```
managed-keys-zone: journal file is out of date: removing journal file
```

Now we have a very basic zone file setup. The NIC on the DNS server should be setup to look only at localhost, **127.0.0.1**. change the the line for **dns-nameservers**, the interfaces file should now have something like this in it. Remember we used the Google DNS or OpenDNS in the **named.conf.local** file as our forwarder so your lab-dns-dhcp server will look there if it cannot find an answer from bind.

```
auto eth0
iface eth0 inet static
   address 10.1.200.3
   gateway 10.1.200.1
```

```
    netmask 255.255.255.0
    dns-nameservers  127.0.0.1
```

Reload the new settings in the interfaces file by bouncing eth0 or rebooting the server.

```
sudo ifdown eth0 ; sudo ifup eth0
```

Now for a quick test to see something is working, lookup the lab-dns-dhcp server.

```
dig lab-dns-dhcp@dragon.lab
```

```
; <<>> DiG 9.9.5-3ubuntu0.5-Ubuntu <<>> lab-dns-dhcp dragon.lab
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 49518
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;lab-dns-dhcp.                  IN      A

;; Query time: 1 msec
;; SERVER: 10.1.200.1#53(10.1.200.1)
;; WHEN: Sun Sep 27 11:36:33 BST 2015
;; MSG SIZE  rcvd: 30

;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 5992
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;dragon.lab.                    IN      A

;; AUTHORITY SECTION:
.                       1799    IN      SOA     a.root-servers.net. nstld.verisign-grs.com. 2015092700 1800 900 604800 86

;; Query time: 40 msec
;; SERVER: 10.1.200.1#53(10.1.200.1)
;; WHEN: Sun Sep 27 11:36:33 BST 2015
;; MSG SIZE  rcvd: 114
```

You can use the following tools to debug your DNS server. The messages in syslog are usually pretty informative.

- tail -f /var/log/syslog
- dig [hostname]
- host [hostname]
- ping [hostname]

The man pages for all of them are a good source of information and their command line switches.

Let us do a quick check that the reverse lookup is also working. This time lookup the IP address 10.1.200.1 that is my lab-router, you may need to use a different IP depending on what you placed in your **dragon.lab.rev.zone** file.

```
dig -x lab-router1
```

```
; <<>> DiG 9.9.5-3ubuntu0.5-Ubuntu <<>> -x 10.1.200.1
;; global options: +cmd
```

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1124
;; flags: qr aa rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;1.200.1.10.in-addr.arpa.       IN      PTR

;; ANSWER SECTION:
1.200.1.10.in-addr.arpa. 0      IN      PTR     lab-router1.dragon.lab.

;; Query time: 2 msec
;; SERVER: 10.1.200.1#53(10.1.200.1)
;; WHEN: Sun Sep 27 11:39:41 BST 2015
;; MSG SIZE  rcvd: 77
```

Do not continue until your DNS is working.

DHCP Configuration

The config for DHCP is all in one file and a little less verbose. Let's dive right in and edit the file.
You should be able to work out what the options do, if not look in **man dhcp.conf**

```
sudo nano /etc/dhcp/dhcpd.conf
```

```
ddns-updates on;
ddns-update-style interim;
update-static-leases on;
authoritative;

# This option points to the same rndc.key we create for bind9.
include "/etc/bind/rndc.key";
allow unknown-clients;
use-host-decl-names on;
default-lease-time 86400; #24  hours
max-lease-time 86400;
log-facility local7;

# dragon.lab DNS zones
zone dragon.lab. {
  primary 10.1.200.3; # This server is the primary DNS server for the zone
  key rndc-key;       # Use the key we defined earlier for dynamic updates
}
zone 200.1.10.in-addr.arpa. {
  primary 10.1.200.3; # This server is the primary reverse DNS for the zone
  key rndc-key;       # Use the key we defined earlier for dynamic updates
}

# dragon.lab LAN range
subnet 10.1.200.0 netmask 255.255.255.0 {
  range 10.1.200.100 10.1.200.119;
  option subnet-mask 255.255.255.0;
  option routers 10.1.200.1;
  option domain-name-servers 10.1.200.3;
  option domain-name "dragon.lab";
  ddns-domainname "dragon.lab.";
  ddns-rev-domainname "200.1.10.in-addr.arpa.";
}

# Static hosts give an IP addr by MAC address
#group {
```

```
#  # Fist static host
#  host static1.dragon.lab {
#      hardware ethernet 01:23:45:67:89:ab;
#      fixed-address 10.1.200.120;
#      ddns-hostname "static1";
#  }
#}
```

If try and start up the dhcp daemon it will throw an error and fail to start. Access to the rndc.key file is being blocked by apparmor. Edit the apparmor config file for dhcpd. If you seach towars the end there is a line for /etc/dhcp/ddns-keys/** r, comment this out and add the one below to replace it.

```
sudo nano /etc/apparmor.d/usr.sbin.dhcpd
```

```
#  /etc/dhcp/ddns-keys/** r,
  /etc/bind/rndc.key  r,
```

Testing That Lot

Reboot the server to get our services, DNS, DHCP & apparmor, restarted and along with all their dependencies. Take a look at the sys-log file for any errors or warnings.

Now boot up another machine which will use the DHCP service we have created. As it boots the output in syslog should look similar to the following:

```
Jul 20 14:54:18 lab-dns-dhcp dhcpd: DHCPDISCOVER from 08:00:27:d3:5e:8a via eth0
Jul 20 14:54:19 lab-dns-dhcp dhcpd: DHCPOFFER on 10.1.200.105 to 08:00:27:d3:5e:8a (lab-desktop) via eth0
Jul 20 14:54:19 lab-dns-dhcp named[2190]: client 10.1.200.2#15077/key rndc-key: signer "rndc-key" approved
Jul 20 14:54:19 lab-dns-dhcp named[2190]: client 10.1.200.2#15077/key rndc-key: updating zone 'dragon.lab/IN': adding an
Jul 20 14:54:19 lab-dns-dhcp named[2190]: client 10.1.200.2#15077/key rndc-key: updating zone 'dragon.lab/IN': adding an
Jul 20 14:54:19 lab-dns-dhcp dhcpd: DHCPREQUEST for 10.1.200.105 (10.1.200.2) from 08:00:27:d3:5e:8a (lab-desktop) via et
Jul 20 14:54:19 lab-dns-dhcp dhcpd: DHCPACK on 10.1.200.105 to 08:00:27:d3:5e:8a (lab-desktop) via eth0
Jul 20 14:54:19 lab-dns-dhcp dhcpd: Added new forward map from lab-desktop.dragon.lab. to 10.1.200.105
Jul 20 14:54:19 lab-dns-dhcp named[2190]: client 10.1.200.2#15077/key rndc-key: signer "rndc-key" approved
Jul 20 14:54:19 lab-dns-dhcp named[2190]: client 10.1.200.2#15077/key rndc-key: updating zone '200.1.10.in-addr.arpa/IN':
Jul 20 14:54:19 lab-dns-dhcp named[2190]: client 10.1.200.2#15077/key rndc-key: updating zone '200.1.10.in-addr.arpa/IN':
Jul 20 14:54:19 lab-dns-dhcp dhcpd: Added reverse map from 105.200.1.10.200.1.10.in-addr.arpa. to lab-desktop.dragon.lab.
```

You should see, in order, DHCPDISCOVER, DHCPOFFER followed by a line saying it is “updating zone ‘dragon…” and then some lines with DHCPREQUEST, DHCPACK and “Added new forward map” If you do, you are all done!

Try using dig to find the IP of your dhcp enabled machine and then do a reverse lookup on that IP. You should see the answer lines similar to these.

```
lab-desktop.dragon.lab. 3600    IN      A       10.1.200.100

100.200.1.10.in-addr.arpa. 3600 IN      PTR     lab-desktop.dragon.lab.
```

If not you missed something out, I found it was usually missing semicolons or permissions not being set correctly or a pesky missing full stop. The syslog is a great help finding the problem.

That was rather long. I Bet you are glad you grabbed that swift coffee right at the beginning. 🙂

NTP Server

If you are setting up a DNS and DHCP server you will also need a NTP server to synchronise the clocks on the PC’s connected to your

LAN. An NTP server uses very little in resources, it will not need a dedicated machine to run on. We can add one to the DNS/DHCP server. It only takes a few edits from a default install and you will be done.

To install and configure an NTP server follow my HOWTO

Updates To The DNS root Zone

The root name server list is updated occasionally. To keep yours up to date, download the latest with:

```
sudo mv /etc/bind/db.root /etc/bind/db.root_old
sudo wget -q -O /etc/bind/db.root http://www.internic.net/zones/named.root
sudo service bind9 reload
```

You might want to set up a crontab job to do this for you, frequently and on a regular basis.

Trouble shooting

If anything is not working check the /var/log/syslog output first. You can also look in the DHCP leases file /var/lib/dhcp/dhcpd.leases.

Unable to add forward map from host.dragon.lab

You may see this when testing a DHCP client. The client will get an IP address but the DNS will not get updated. Check that the zones in dhcp.conf and named.conf.local are the same. In dhcp.cong the name ends in a full stop.

Also check the primary lines in dhcp.conf actually point to your DNS server.

If you see something like “insecurity proof failed”

If you see something like the following in syslog

```
error (chase DS servers) resolving ‘dragon.lab/DS/IN’: 10.1.200.101#23
named[4321]: error (insecurity proof failed) resolving ‘com/NS/IN’: 10.1.200.101#23
```

You need to comment out the following line in /etc/bind/named.conf.options, it appears to be set by default.

```
dnssec-validation auto;
```

This entry was posted in 14.04 LTS, Installing and Configuration, Networking, System Administration, Ubuntu and tagged bind9, dhcp, dns, NTP on July 20, 2014 [http://blogging.dragon.org.uk/dns-with-bind9-and-dhcp-on-ubuntu-14-04/] .

---

7 thoughts on “DNS with bind9 and DHCP on Ubuntu 14.04”

Can Bican
December 22, 2015 at 12:32 am

One small typo:

Not

dig lab-dns-dhcp dragon.lab

But

dig lab-dns-dhcp @dragon.lab

Michael
September 24, 2015 at 8:30 pm

Hello!

Thanks for this very useful guide! It has saved a lot of time!

But I think I have found one little bug...
ddns-rev-domainname "200.1.10.in-addr.arpa."; –> in my case this will cause an error when I try a reverse lookup on a client who has got the IP from the dhcp
(Because the dhcp has written this wrong statement into the zone-file: 200.1.10.200.1.10.in-addr.arpa.)

I have fixed this Problem with this correction:
ddns-rev-domainname "in-addr.arpa.";

And also I have changed
"zone 200.1.10.in-addr.arpa. ..."
into "zone in-addr.arpa. ..."

I hope this contribution is helpful
MIchael

Richard Post author
September 27, 2015 at 11:18 am

Thanks for pointing out the mistake in the zones. I only needed to to change the DHCP setting for ddns-rev-domainname "in-addr.arpa."; I have also rechecked it on 14.04-3.

Ryan
October 7, 2014 at 1:10 pm

Great How-to! Just the right amount of hand holding for someone like me with an understanding of the concepts but not the implementation. I've learned a bunch from this.

One thing that did throw me for a bit was the zone definitions. The code does not format correctly in Chrome which caused some confusion as I was entering it myself instead of copying/pasting.

Thanks for posting this! Now on to the AD DC..

Jeff Brown
September 24, 2014 at 4:27 am

Many thanks for the tutorial. After stumbling down a few other paths, I found this, and now have BIND, NTP, DNS, and DHCP services running on Ubuntu 14.04.
The next step for me is the Samba AD-DC config, but sleep followed by coffee is needed first!

Lars Urban
August 20, 2014 at 7:34 pm

Hello again,

also this is a nice work, setup your config (in my case without DHCP) and it is running without problems.
Thank you for your Work !

ps.:
A little typo :
sudo ntp restart –> sudo service ntp restart

😉

Richard Post author
August 21, 2014 at 5:15 pm

I am glad you found the tutorial useful and can amend it to suit your needs.
Also thank you for letting me know about the typo. My only comment about that is DOH!! Do you know how many times I read and re-read that?