Splunk

Tuesday, March 17, 2020 1:11 PM

Splunk comprises 3 main components

1. Splunk Forwarder: It is used for collecting the logs.
2. Splunk Indexer: It is used for Parsing and Indexing the data.
3. Splunk Search Head: Provides a web interface for searching, analyzing and reporting.

How to install Splunk

First, we are going to download the Splunk binary using the wget command as shown

```
wget -O splunk-8.0.1-6db836e2fb9e-Linux-x86_64.tgz 'https://www.splunk.com/bin/splunk/DownloadActivityServlet?architecture=x86_64&platform=linux&version=8.0.1&product=splunk&filename=splunk-8.0.1-6db836e2fb9e-Linux-x86_64.tgz&wget=true'
```

Download md5 sum for verify:

```
wget https://download.splunk.com/products/splunk/releases/8.0.1/linux/splunk-8.0.1-6db836e2fb9e-Linux-x86_64.tgz.md5
```

Verify:

```
user@splunk:~$ md5sum -c splunk-8.0.1-6db836e2fb9e-Linux-x86_64.tgz.md5
splunk-8.0.1-6db836e2fb9e-Linux-x86_64.tgz: OK
```

Create user for run splunk as non-root user:

```
user@splunk:~$ sudo useradd splunkstart --system --shell=/usr/sbin/nologin
```

Untar splunk-8.0.1–6db836e2fb9e-Linux-x86_64.tgz:

```
user@splunk:~$ sudo tar zxvf splunk-8.0.1-6db836e2fb9e-Linux-x86_64.tgz -C /opt/
```

Run the chown command to change the ownership of the splunk directory and everything under it to the user that you want to run the software:

```
user@splunk:~$ sudo chown -R splunkstart:splunkstart /opt/splunk
```

Enable boot-start as non-root user (accept the license agreement and enter username and password for local admin, when prompted):

```
user@splunk:~$ sudo /opt/splunk/bin/splunk enable boot-start -user splunkstart -systemd-managed 1
```

Output

Next, We shall enable Splunk to always start when the server starts
This command will generate the Output as seen below

```
SPLUNK SOFTWARE LICENSE AGREEMENT

THIS SPLUNK SOFTWARE LICENSE AGREEMENT ("AGREEMENT") GOVERNS THE LICENSING,
INSTALLATION AND USE OF SPLUNK SOFTWARE. BY DOWNLOADING AND/OR INSTALLING SPLUNK
SOFTWARE: (A) YOU ARE INDICATING THAT YOU HAVE READ AND UNDERSTAND THIS
AGREEMENT, AND AGREE TO BE LEGALLY BOUND BY IT ON BEHALF OF THE COMPANY,
GOVERNMENT, OR OTHER ENTITY FOR WHICH YOU ARE ACTING (FOR EXAMPLE, AS AN
EMPLOYEE OR GOVERNMENT OFFICIAL) OR, IF THERE IS NO COMPANY, GOVERNMENT OR OTHER
ENTITY FOR WHICH YOU ARE ACTING, ON BEHALF OF YOURSELF AS AN INDIVIDUAL; AND (B)
YOU REPRESENT AND WARRANT THAT YOU HAVE THE AUTHORITY TO ACT ON BEHALF OF AND
BIND SUCH COMPANY, GOVERNMENT OR OTHER ENTITY (IF ANY). WITHOUT LIMITING THE
FOREGOING, YOU (AND YOUR ENTITY, IF ANY) ACKNOWLEDGE THAT BY SUBMITTING AN ORDER
FOR THE SPLUNK SOFTWARE, YOU (AND YOUR ENTITY (IF ANY)) HAVE AGREED TO BE BOUND
BY THIS AGREEMENT. As used in this Agreement, "Splunk," refers to Splunk Inc., a
Delaware corporation, with its principal place of business at 270 Brannan
Street, San Francisco, California 94107, U.S.A.; and "Customer" refers to the
company, government, or other entity on whose behalf you have entered into this
Agreement or, if there is no such entity, you as an individual.
4. FORCE MAJEURE. Splunk will not be responsible for any failure or delay i
n
its performance under these Terms and Conditions due to causes beyond its
reasonable control, including, but not limited to, labor disputes, strikes,
lockouts, shortages of or inability to obtain labor, energy, raw materials or
supplies, war, acts of terror, riot, acts of God or governmental action.

Splunk Software License Agreement 04.24.2018
```

Accept the Software license by typing Y

```
Do you agree with this license? [y/n]: Y
```

Output

```
This appears to be your first time running this version of Splunk.
```

```
An Admin password must be set before installation proceeds.
Password must contain at least:
* 8 total printable ASCII character(s).
Please enter a new password:
Please confirm new password:
```

Now you can start Splunkd service:

```
user@splunk:~$ sudo systemctl start Splunkd
```

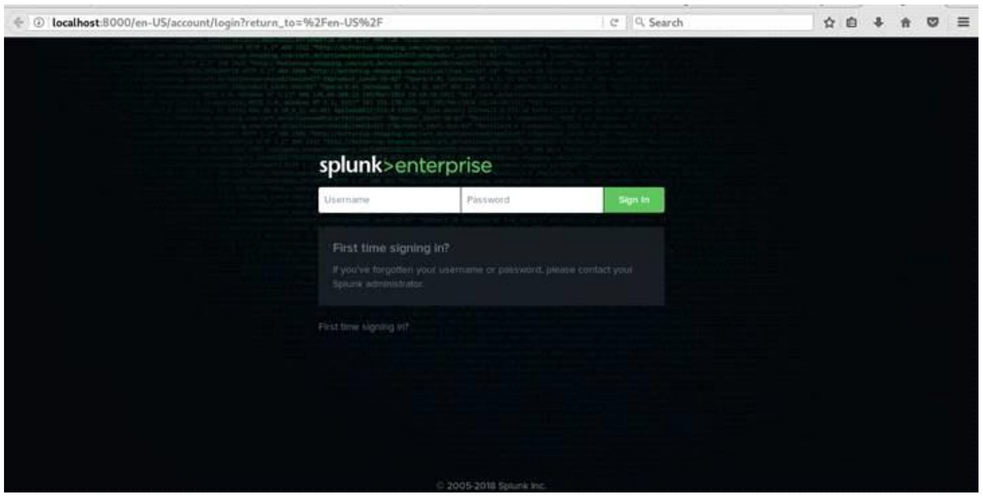
Check service status and port listening:

```
user@splunk:~$ systemctl status Splunkd
* Splunkd.service - Systemd service file for Splunk, generated by 'splunk enable boot-start'
   Loaded: loaded (/etc/systemd/system/Splunkd.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2019-12-26 15:49:51 EET; 1h 21min ago
 Main PID: 727 (splunkd)
```

```
user@splunk:~$ ss -tln
State   Recv-Q  Send-Q Local Address:Port  Peer Address:Port
LISTEN  0      128    127.0.0.53%lo:53    0.0.0.0:*
LISTEN  0      128    0.0.0.0:22          0.0.0.0:*
LISTEN  0      128    0.0.0.0:8089        0.0.0.0:*
LISTEN  0      128    0.0.0.0:8191        0.0.0.0:*
LISTEN  0      128    0.0.0.0:8000        0.0.0.0:*
LISTEN  0      5      127.0.0.1:8065      0.0.0.0:*
LISTEN  0      128    [::]:22             [::]:*
```

Now you can access web interface with URL http://192.168.1.154:8000

Enter the login credentials you created when you agreed to the user agreement and hit "Sign In"

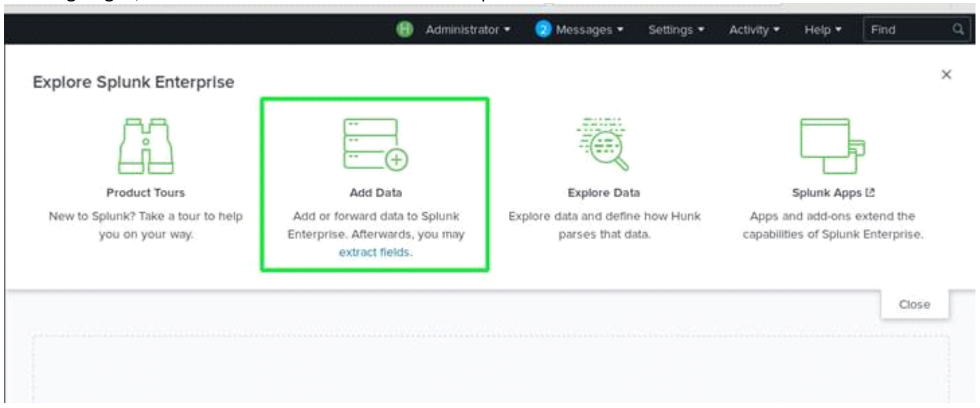

localhost:8000/en-US/account/login?return_to=%2Fen-US%2F
splunk>enterprise
Username
Password
Sign In
First time signing in?
If you've forgotten your username or password, please contact your Splunk administrator.
First time signing in?
© 2005-2018 Splunk Inc.

Hurray! We have successfully installed Splunk on our Ubuntu 18.04 System.

Let's now monitor log file /var/log/messages. To achieve that, follow the steps below

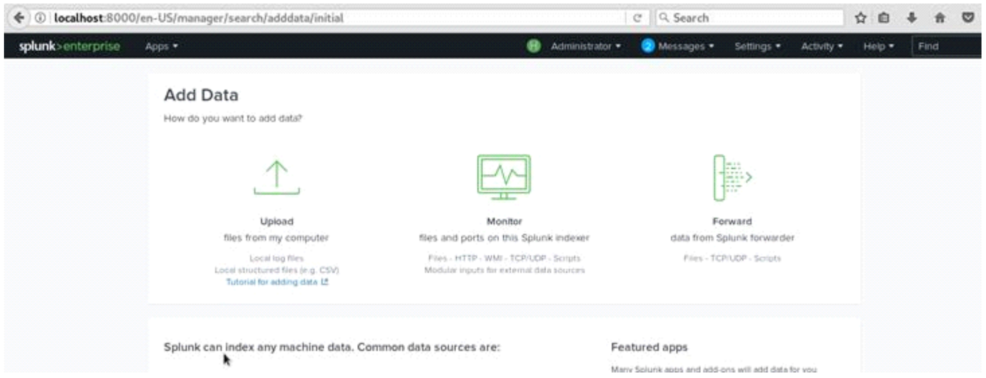
Step1:

After signing in, head out and click on the '**Add Data** 'option.


Administrator
Messages
Settings
Activity
Help
Find
Explore Splunk Enterprise
Product Tours
New to Splunk? Take a tour to help you on your way.
Add Data
Add or forward data to Splunk Enterprise. Afterwards, you may extract fields.
Explore Data
Explore data and define how Hunk parses that data.
Splunk Apps
Apps and add-ons extend the capabilities of Splunk Enterprise.
Close

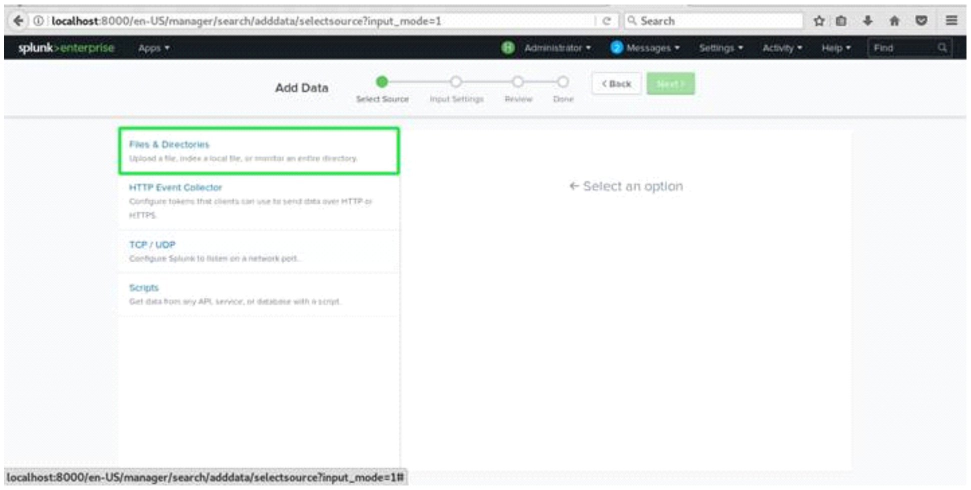
Step 2:

3 options will be presented to you: Upload, Monitor, and Forward.
Each option is self-explanatory with a short description of the purpose. in our case,
our task is to monitor logs from /var/log/messages folder so we go ahead and select the '**Monitor'** option.


localhost:8000/en-US/manager/search/adddata/initial
splunk>enterprise
Apps
Administrator
Messages
Settings
Activity
Help
Find
Add Data
How do you want to add data?
Upload
files from my computer
Local log files
Local structured files (e.g. CSV)
Tutorial for adding data
Monitor
files and ports on this Splunk indexer
Files - HTTP - WMI - TCP/UDP - Scripts
Modular inputs for external data sources
Forward
data from Splunk forwarder
Files - TCP/UDP - Scripts
Splunk can index any machine data. Common data sources are:
Featured apps
Many Splunk apps and add-ons will add data for you

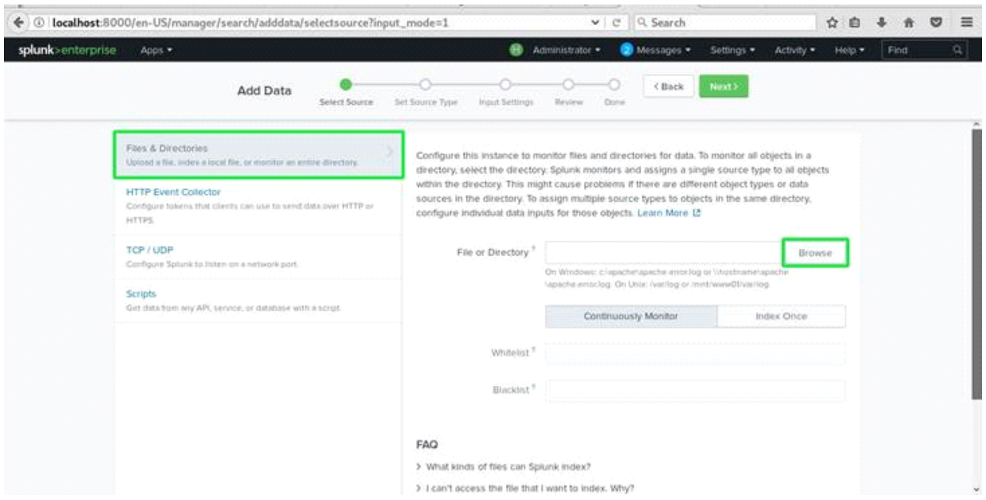
Step 3:

Here, you'll be presented with more options. in this case, click on '**Files and Directories** 'since our log file is contained in a directory structure


localhost:8000/en-US/manager/search/adddata/selectsource?input_mode=1
splunk>enterprise
Apps
Administrator
Messages
Settings
Activity
Help
Find
Add Data
Select Source
Input Settings
Review
Done
< Back
Next >
Files & Directories
Upload a file, index a local file, or monitor an entire directory.
HTTP Event Collector
Configure tokens that clients can use to send data over HTTP or HTTPS.
TCP / UDP
Configure Splunk to listen on a network port.
Scripts
Get data from any API, service, or database with a script.
← Select an option
localhost:8000/en-US/manager/search/adddata/selectsource?input_mode=1#

Step 4:

Browse to the target folder you with to gather statistics from



localhost:8000/en-US/manager/search/adddata/selectsource?input_mode=1

Add Data — Select Source

Files & Directories
Upload a file, index a local file, or monitor an entire directory.

HTTP Event Collector
Configure tokens that clients can use to send data over HTTP or HTTPS.

TCP / UDP
Configure Splunk to listen on a network port.

Scripts
Get data from any API, service, or database with a script.

Configure this instance to monitor files and directories for data. To monitor all objects in a directory, select the directory. Splunk monitors and assigns a single source type to all objects within the directory. This might cause problems if there are different object types or data sources in the directory. To assign multiple source types to objects in the same directory, configure individual data inputs for those objects. Learn More

File or Directory: Browse
On Windows: c:\apache\apache.error.log or \\hostname\apache\apache.error.log. On Unix: /var/log or /mnt/www01/var/log.

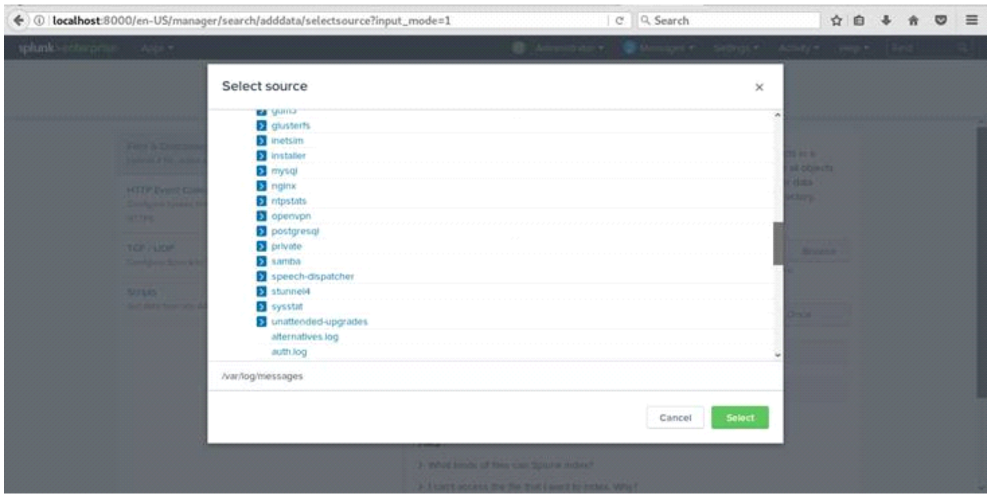
Continuously Monitor | Index Once

Whitelist

Blacklist

FAQ
> What kinds of files can Splunk index?
> I can't access the file that I want to index. Why?

In our case, browse till you get to the /var/log/messages file path



Select source

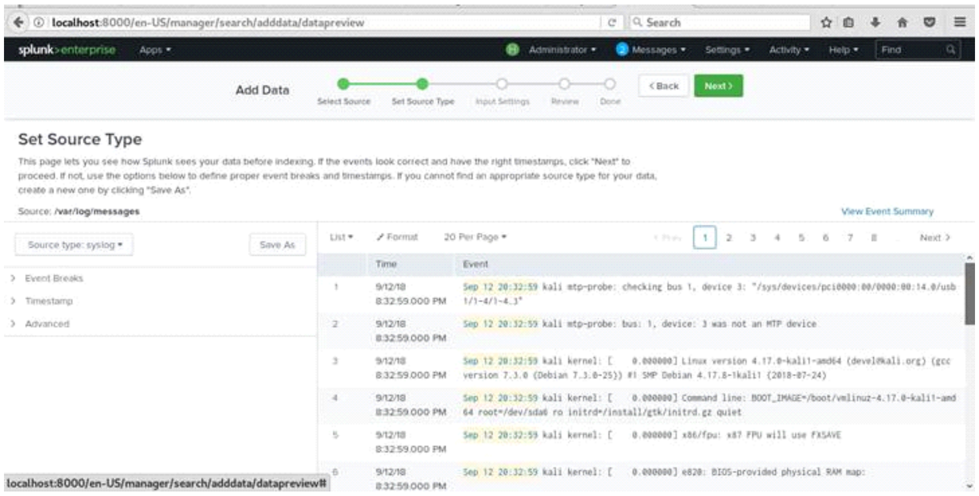
```
glusterfs
inetsim
installer
mysql
nginx
ntpstats
openvpn
postgresql
private
samba
speech-dispatcher
stunnel4
sysstat
unattended-upgrades
alternatives.log
auth.log
```

/var/log/messages

Cancel | Select

Step 5:

In the next page, you'll see how Splunk will display your data before indexing it. If you are satisfied with all the settings, click **'Next'**



localhost:8000/en-US/manager/search/adddata/datapreview

Set Source Type

This page lets you see how Splunk sees your data before indexing. If the events look correct and have the right timestamps, click "Next" to proceed. If not, use the options below to define proper event breaks and timestamps. If you cannot find an appropriate source type for your data, create a new one by clicking "Save As".

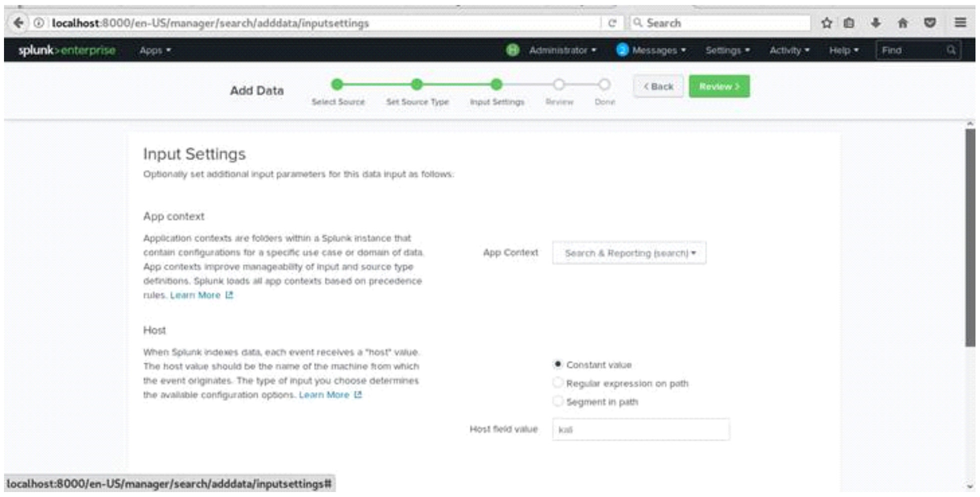
Source: /var/log/messages

Source type: syslog

	Time	Event
1	9/12/18 8:32:59.000 PM	Sep 12 20:32:59 kali mtp-probe: checking bus 1, device 3: "/sys/devices/pci0000:00/0000:00:14.0/usb1/1-4/1-4.3"
2	9/12/18 8:32:59.000 PM	Sep 12 20:32:59 kali mtp-probe: bus: 1, device: 3 was not an MTP device
3	9/12/18 8:32:59.000 PM	Sep 12 20:32:59 kali kernel: [0.000000] Linux version 4.17.0-kali1-amd64 (devel@kali.org) (gcc version 7.3.0 (Debian 7.3.0-25)) #1 SMP Debian 4.17.8-1kali1 (2018-07-24)
4	9/12/18 8:32:59.000 PM	Sep 12 20:32:59 kali kernel: [0.000000] Command line: BOOT_IMAGE=/boot/vmlinuz-4.17.0-kali1-amd64 root=/dev/sda6 ro initrd=/install/gtk/initrd.gz quiet
5	9/12/18 8:32:59.000 PM	Sep 12 20:32:59 kali kernel: [0.000000] x86/fpu: x87 FPU will use FXSAVE
6	9/12/18 8:32:59.000 PM	Sep 12 20:32:59 kali kernel: [0.000000] e820: BIOS-provided physical RAM map:

Step 6:

The next page is the '**Input settings** 'Click '**Review** ' to have a glimpse of your settings before you start indexing


localhost:8000/en-US/manager/search/adddata/inputsettings

Input Settings

Optionally set additional input parameters for this data input as follows:

App context

Application contexts are folders within a Splunk instance that contain configurations for a specific use case or domain of data. App contexts improve manageability of input and source type definitions. Splunk loads all app contexts based on precedence rules. Learn More

App Context: Search & Reporting (search)

Host

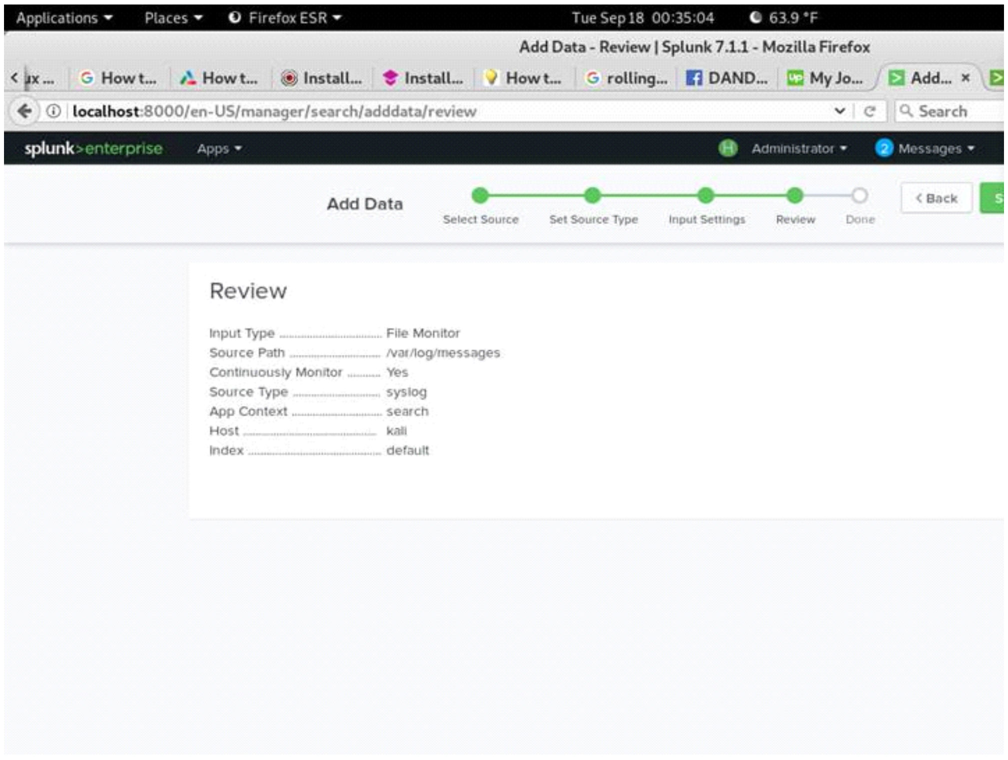
When Splunk indexes data, each event receives a "host" value. The host value should be the name of the machine from which the event originates. The type of input you choose determines the available configuration options. Learn More

● Constant value
○ Regular expression on path
○ Segment in path

Host field value: kali

Step 7:

The ‘review’ page will give you a brief summary of the main settings that you have selected. Click ‘Submit’

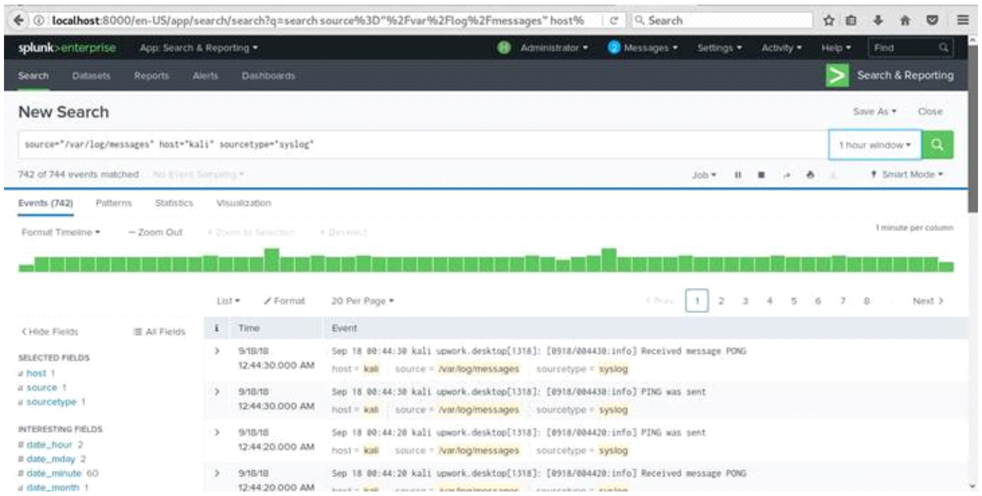

Add Data - Review | Splunk 7.1.1 - Mozilla Firefox

localhost:8000/en-US/manager/search/adddata/review

Review

```
Input Type .................. File Monitor
Source Path ................. /var/log/messages
Continuously Monitor ........ Yes
Source Type ................. syslog
App Context ................. search
Host ........................ kali
Index ....................... default
```

Statistics based on the parameters you chose will start being displayed



```
source="/var/log/messages" host="kali" sourcetype="syslog"
742 of 744 events matched

9/18/18 12:44:30.000 AM  Sep 18 00:44:30 kali upwork.desktop[1318]: [0918/004430:info] Received message PONG
                         host = kali   source = /var/log/messages   sourcetype = syslog
9/18/18 12:44:30.000 AM  Sep 18 00:44:30 kali upwork.desktop[1318]: [0918/004430:info] PING was sent
                         host = kali   source = /var/log/messages   sourcetype = syslog
9/18/18 12:44:20.000 AM  Sep 18 00:44:20 kali upwork.desktop[1318]: [0918/004420:info] PING was sent
                         host = kali   source = /var/log/messages   sourcetype = syslog
9/18/18 12:44:20.000 AM  Sep 18 00:44:20 kali upwork.desktop[1318]: [0918/004420:info] Received message PONG
```

Navigate to the folder where you have downloaded the Debian file and install Splunk using the dpkg command

Copying '/opt/splunk/etc/openldap/ldap.conf.default' to '/opt/splunk/etc/openldap/ldap.conf'.

Generating RSA private key, 2048 bit long modulus

................+++

.....+++

e is 65537 (0x10001)

writing RSA key

Generating RSA private key, 2048 bit long modulus

....+++

....+++

e is 65537 (0x10001)

writing RSA key

Moving '/opt/splunk/share/splunk/search_mrsparkle/modules.new' to '/opt/splunk/share/splunk/search_mrsparkle/modules'.

Init script installed at /etc/init.d/splunk.

Init script is configured to run at boot.

Now we are going to Start Splunk

systemctl start spunk

To verify that Splunk is indeed rrunning,run

systemctl status splunk

```
● splunk.service - LSB: Start splunk
Loaded: loaded (/etc/init.d/splunk; generated)
Active: active (running) since Mon 2018-09-17 23:39:47 EAT; 33s ago
Docs: man:systemd-sysv-generator(8)
Process: 2514 ExecStart=/etc/init.d/splunk start (code=exited, status=0/SUCCESS)
Tasks: 164 (limit: 4574)
Memory: 750.5M
CGroup: /system.slice/splunk.service
```

```
├─2577 splunkd -p 8089 start
├─2578 [splunkd pid=2577] splunkd -p 8089 start [process-runner]
├─2589 mongod --dbpath=/opt/splunk/var/lib/splunk/kvstore/mongo --storageEngine=mmapv1 --port=8191 --timeStampFormat=iso8601-utc --smallfil
├─2663 /opt/splunk/bin/splunkd instrument-resource-usage -p 8089 --with-kvstore
├─2673 /opt/splunk/bin/python -O /opt/splunk/lib/python2.7/site-packages/splunk/appserver/mrsparkle/root.py --proxied=127.0.0.1,8065,8000
├─3038 [splunkd pid=2577] [search-launcher]
└─3039 [splunkd pid=2577] [search-launcher] [process-runner]

Sep 17 23:39:14 kali splunk[2514]: All installed files intact.
Sep 17 23:39:14 kali splunk[2514]: Done
Sep 17 23:39:14 kali splunk[2514]: All preliminary checks passed.
Sep 17 23:39:14 kali splunk[2514]: Starting splunk server daemon (splunkd)...
Sep 17 23:39:14 kali splunk[2514]: Done
Sep 17 23:39:47 kali splunk[2514]: Waiting for web server at http://127.0.0.1:8000 to be available............................. Done
Sep 17 23:39:47 kali splunk[2514]: If you get stuck, we're here to help.
Sep 17 23:39:47 kali splunk[2514]: Look for answers here: http://docs.splunk.com
Sep 17 23:39:47 kali splunk[2514]: The Splunk web interface is at http://kali:8000
Sep 17 23:39:47 kali systemd[1]: Started LSB: Start splunk.
```

Graylog

Tuesday, March 17, 2020 7:47 PM

Prerequisites
Make sure to install and configure the following software before installing and starting any Graylog services:

Java (>= 8)
Elasticsearch (5.x or 6.x)
MongoDB (3.6 or 4.0)

Warning

Graylog 3 does not work with Elasticsearch 7.x!
Graylog 3 does not work with MongoDB 4.2!

wget https://downloads.graylog.org/releases/graylog/graylog-3.2.3.tgz

This guide describes the fastest way to install Graylog on Ubuntu 18.04 LTS. All links and packages are present at the time of writing but might need to be updated later on.

Warning

This guide **does not cover** security settings! The server administrator must make sure the graylog server is not publicly exposed, and is following security best practices.

Prerequisites

Taking a minimal server setup as base will need this additional packages:

```
$ sudo apt-get update && sudo apt-get upgrade
$ sudo apt-get install apt-transport-https openjdk-8-jre-headless uuid-runtime pwgen
```

If you get an error stating *Unable to locate package*, you likely need to enable the universe repository which can be done typing the below command, and subsequent commands as follows:

```
$ sudo add-apt-repository universe
$ sudo apt-get update && sudo apt-get upgrade
$ sudo apt-get install apt-transport-https openjdk-8-jre-headless uuid-runtime pwgen
```

MongoDB

The official MongoDB repository provides the most up-to-date version and is the recommended way of installing MongoDB [1]:

```
$ sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv 9DA31620334BD75D9DCB49F368818C72E52529D4
$ echo "deb [ arch=amd64 ] https://repo.mongodb.org/apt/ubuntu bionic/mongodb-org/4.0 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.0.list
$ sudo apt-get update
$ sudo apt-get install -y mongodb-org
```

1

For e.g. corporate proxies and other non-free environments you can use a keyserver approach via wget. **wget -qO- 'http://keyserver.ubuntu.com/pks/lookup?op=get&search=0x9DA31620334BD75D9DCB49F368818C72E52529D4' | sudo apt-key add -**

The last step is to enable MongoDB during the operating system's startup and verify it is running.

```
$ sudo systemctl daemon-reload
$ sudo systemctl enable mongod.service
$ sudo systemctl restart mongod.service
$ sudo systemctl --type=service --state=active | grep mongod
```

Elasticsearch

Graylog can be used with Elasticsearch 6.x, please follow the below instructions to install the open source version of Elasticsearch.

```
$ wget -q https://artifacts.elastic.co/GPG-KEY-elasticsearch -O myKey
$ sudo apt-key add myKey
$ echo "deb https://artifacts.elastic.co/packages/oss-6.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-6.x.list
$ sudo apt-get update && sudo apt-get install elasticsearch-oss
```

The above instructions are a derivative from the Elasticsearch install page

Modify the Elasticsearch configuration file (/etc/elasticsearch/elasticsearch.yml) and set the cluster name to graylog and uncomment action.auto_create_index: false to enable the action:

```
$ sudo vim /etc/elasticsearch/elasticsearch.yml
cluster.name: graylog
action.auto_create_index: false
```

After you have modified the configuration, you can start Elasticsearch and verify it is running.

```
$ sudo systemctl daemon-reload
$ sudo systemctl enable elasticsearch.service
$ sudo systemctl restart elasticsearch.service
$ sudo systemctl --type=service --state=active | grep elasticsearch
```

Graylog

Now install the Graylog repository configuration and Graylog itself with the following commands:

```
$ wget https://packages.graylog2.org/repo/packages/graylog-3.2-repository_latest.deb
$ sudo dpkg -i graylog-3.2-repository_latest.deb
$ sudo apt-get update && sudo apt-get install graylog-server graylog-enterprise-plugins graylog-integrations-plugins graylog-enterprise-integrations-plugins
```

Hint

If you do not want the Integrations Plugins or the Enterprise Plugins installed, then simply run sudo apt-get install graylog-server

Edit the Configuration File

Read the instructions *within* the configurations file and edit as needed, located at /etc/graylog/server/server.conf. Additionally add password_secret and root_password_sha2 as these are *mandatory* and **Graylog will not start without them**.

To create your root_password_sha2 run the following command:

```
$ echo -n "Enter Password: " && head -1 </dev/stdin | tr -d '\n' | sha256sum | cut -d" " -f1
```

To be able to connect to Graylog you should set http_bind_address to the public host name or a public IP address of the machine you can connect to. More information about these settings can be found in Configuring the web interface.

Note

If you’re operating a single-node setup and would like to use HTTPS for the Graylog web interface and the Graylog REST API, it’s possible to use NGINX or Apache as a reverse proxy.

The last step is to enable Graylog during the operating system’s startup and verify it is running.

```
$ sudo systemctl daemon-reload
$ sudo systemctl enable graylog-server.service
$ sudo systemctl start graylog-server.service
$ sudo systemctl --type=service --state=active | grep graylog
```

The next step is to ingest messages into your Graylog and extract the messages with extractors or use the Pipelines to work with the messages.

Multiple Server Setup

If you plan to have multiple server taking care of different roles in your cluster like we have in this big production setup you need to modify only a few settings. This is covered in our Multi-node Setup guide. The default file location guide will give you the file you need to modify in your setup.

OpenLDAP

Tuesday, March 17, 2020 1:14 PM

Introduction

Lightweight Directory Access Protocol (LDAP) is a standard protocol designed to manage and access hierarchical directory info rmation over a network. It can be used to store any kind of information, though it is most often used as a centralized authentication system or for corporate email and phone directories.

Before starting this tutorial, you should have an Ubuntu 18.04 server set up with Apache, PHP (LAMP) on Ubuntu 18.04, skipping MySQL as we will not need the MySQL database server.

Additionally, since we will be entering passwords into the web interface, we should secure Apache with SSL encryption. Read How To Secure Apache with Let's Encrypt on Ubuntu 16.04 to download and configure free SSL certificates. You will need a domain name to complete this step. We will use these same ce rtificates to provide secure LDAP connections as well.

Step 1 — Installing and Configuring the LDAP Server

Our first step is to install the LDAP server and some associated utilities. Luckily, the packages we need are all available i n Ubuntu's default repositories.

Log into your server. Since this is our first time using apt-get in this session, we'll refresh our local package index, then install the packages we want:

```
sudo apt-get update
sudo apt-get install slapd ldap-utils
```

During the installation, you will be asked to select and confirm an administrator password for LDAP. You can enter anything h ere, because you'll have the opportunity to update it in just a moment.

Even though we just installed the package, we're going to go right ahead and reconfigure it. The slapd package has the abilit y to ask a lot of important configuration questions, but by default they are skipped over in the installation process. We gain access to all of the prompts by telling our system to reconfigure the packa ge:

```
sudo dpkg-reconfigure slapd
```

There are quite a few new questions to answer in this process. We will be accepting most of the defaults. Let's go through th e questions:

- Omit OpenLDAP server configuration? No
- DNS domain name?
  This option will determine the base structure of your directory path. Read the message to understand exactly how this will be implemented. You can actually select whatever value you'd like, even if you don't own the actual domain. However, this tutorial assumes you have a proper domain name for the server, so you should u se that. We'll use example.com throughout the tutorial.
- Organization name?
  For this guide, we will be using example as the name of our organization. You may choose anything you feel is appropriate.
- Administrator password? enter a secure password twice
- Database backend? MDB
- Remove the database when slapd is purged? No
- Move old database? Yes
- Allow LDAPv2 protocol? No

At this point, your LDAP server is configured and running. Open up the LDAP port on your firewall so external clients can con nect:

```
sudo ufw allow ldap
```

Let's test our LDAP connection with ldapwhoami, which should return the username we're connected as:

```
ldapwhoami -H ldap:// -x
```

```
Output
anonymous
```

anonymous is the result we're expecting, since we ran ldapwhoami without logging in to the LDAP server. This means the server is running and answering queries. Next we'll set up a web interface to manage LDAP data.

Step 2 — Installing and Configuring the phpLDAPadmin Web Interface

Although it is very possible to administer LDAP through the command line, most users will find it easier to use a web interfa ce. We're going to install phpLDAPadmin, a PHP application which provides this functionality.

The Ubuntu repositories contain a phpLDAPadmin package. You can install it with apt-get:

```
sudo apt-get install phpldapadmin
```

This will install the application, enable the necessary Apache configurations, and reload Apache.

The web server is now configured to serve the application, but we need to make some additional changes. We need to configure phpLDAPadmin to use our domain, and to not autofill the LDAP login information.

Begin by opening the main configuration file with root privileges in your text editor:

```
sudo nano /etc/phpldapadmin/config.php
```

Look for the line that starts with $servers->setValue('server','name'. In nano you can search for a string by typing CTRL-W, then the string, then ENTER. Your cursor will be placed on the correct line.

This line is a display name for your LDAP server, which the web interface uses for headers and messages about the server. Cho ose anything appropriate here:

/etc/phpldapadmin/config.php
```
$servers->setValue('server','name','Example LDAP');
```

Next, move down to the $servers->setValue('server','base' line. This config tells phpLDAPadmin what the root of the LDAP hierarchy is. This is based on the value we typed in when reconfiguring the slapd package. In our example we selected example.com and we need to translate this into LDAP syntax by putting each domain component (everything not a dot) into a dc= notation:

/etc/phpldapadmin/config.php
```
$servers->setValue('server','base', array('dc=example,dc=com'));
```

Now find the login bind_id configuration line and comment it out with a # at the beginning of the line:

/etc/phpldapadmin/config.php
```
#$servers->setValue('login','bind_id','cn=admin,dc=example,dc=com');
```

This option pre-populates the admin login details in the web interface. This is information we shouldn't share if our phpLDAPadmin page is pu blicly accessible.

The last thing that we need to adjust is a setting that controls the visibility of some phpLDAPadmin warning messages. By def ault the application will show quite a few warning messages about template files. These have no impact on our current use of the software. We can hide them by searching for the hide_template_warning parameter, uncommenting the line that contains it, and setting it to **true**:

/etc/phpldapadmin/config.php
```
$config->custom->appearance['hide_template_warning'] = true;
```

This is the last thing that we need to adjust. Save and close the file to finish. We don't need to restart anything for the c hanges to take effect.

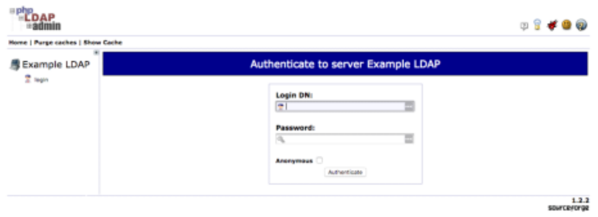
Next we'll log into phpLDAPadmin.

Step 3 — Logging into the phpLDAPadmin Web Interface

Having made the necessary configuration changes to phpLDAPadmin, we can now begin to use it. Navigate to the application in y our web browser. Be sure to substitute your domain for the highlighted area below:

https://example.com/phpldapadmin

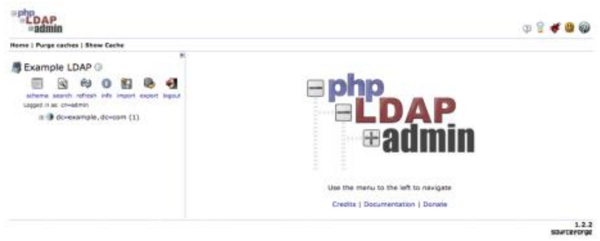
The phpLDAPadmin landing page will load. Click on the **login** link in the left-hand menu on the page. A login form will be presented:



The **Login DN** is the username that you will be using. It contains the account name as a cn= section, and the domain name you selected for the server broken into dc= sections as described in previous steps. The default admin account that we set up during install is called **admin**, so for our example we would type in the following:

```
cn=admin,dc=example,dc=com
```
After entering the appropriate string for your domain, type in the admin password you created during configuration, then clic k the **Authenticate** button.

You will be taken to the main interface:



At this point, you are logged into the phpLDAPadmin interface. You have the ability to add users, organizational units, group s, and relationships.

LDAP is flexible in how you can structure your data and directory hierarchies. You can create whatever kind of structure you' d like and also create rules for how they interact.

Since this process is the same on Ubuntu 16.04 as it was on previous versions, you can follow the steps laid out in the *Add Organizational Units, Groups, and Users* section of the LDAP installation article for Ubuntu 12.04.

Those steps will work well on this installation of phpLDAPadmin, so follow along to get some practice working with the interf ace and learning how to structure your data.

Now that we've logged in and familiarized ourselves with the web interface, let's take a moment to provide more security to o ur LDAP server.

Step 4 – Configuring StartTLS LDAP Encryption

Although we've encrypted our web interface, external LDAP clients are still connecting to the server and passing information around in plain text. Let's use our Let's Encrypt SSL certificates to add encryption to our LDAP server.

Copying the Let's Encrypt Certificates

Because the slapd daemon runs as the user **openldap**, and Let's Encrypt certificates can only be read by the **root** user, we'll need make a few adjustments to allow slapd access to the certificates. We'll create a short script that will copy the certificates to /etc/ssl/, the standard system directory for SSL certificates and keys. The reason we're making a script to do this, instead of just entering the commands manually, is that we'll need to repeat this pr ocess automatically whenever the Let's Encrypt certificates are renewed. We'll update the certbot cron job later to enable this.

First, open a new text file for the shell script:

```
sudo nano /usr/local/bin/renew.sh
```

This will open a blank text file. Paste in the following script. Be sure to update the SITE=example.com portion to reflect where your Let's Encrypt certificates are stored. You can find the correct value by listing out the certificate directory with sudo ls /etc/letsencrypt/live.

/usr/local/bin/renew.sh

```
#!/bin/sh
SITE=example.com

# move to the correct let's encrypt directory
cd /etc/letsencrypt/live/$SITE

# copy the files
cp cert.pem /etc/ssl/certs/$SITE.cert.pem
cp fullchain.pem /etc/ssl/certs/$SITE.fullchain.pem
cp privkey.pem /etc/ssl/private/$SITE.privkey.pem

# adjust permissions of the private key
chown :ssl-cert /etc/ssl/private/$SITE.privkey.pem
chmod 640 /etc/ssl/private/$SITE.privkey.pem

# restart slapd to load new certificates
systemctl restart slapd
```

This script moves into the Let's Encrypt certificate directory, copies files over to /etc/ssl, then updates the private key's permissions to make it readable by the system's **ssl-cert** group. It also restarts slapd, which will ensure that new certificates are loaded when this script is run from our certbot renewal cron job.

Save and close the file, then make it executable:

```
sudo chmod u+x /usr/local/bin/renew.sh
```

Then run the script with sudo:

```
sudo /usr/local/bin/renew.sh
```

Verify that the script worked by listing out the new files in /etc/ssl:

```
sudo su -c 'ls -al /etc/ssl/{certs,private}/example.com*'
```

The sudo command above is a little different than normal. The su -c '. . .' portion wraps the whole ls command in a **root** shell before executing it. If we didn't do this, the * wildcard filename expansion would run with your non-sudo user's permissions, and it would fail because /etc/ssl/private is not readable by your user.

ls will print details about the three files. Verify that the ownership and permissions look correct:

Output

```
-rw-r--r-- 1 root root     1793 May 31 13:58 /etc/ssl/certs/example.com.cert.pem
-rw-r--r-- 1 root root     3440 May 31 13:58 /etc/ssl/certs/example.com.fullchain.pem
-rw-r----- 1 root ssl-cert 1704 May 31 13:58 /etc/ssl/private/example.com.privkey.pem
```

Next we'll automate this with certbot.

Updating the Certbot Renewal Cron Job

We need to update our certbot cron job to run this script whenever the certificates are updated:

```
sudo crontab -e
```

You should already have a certbot renew line. Add the highlighted portion below:

crontab

```
15 3 * * * /usr/bin/certbot renew --quiet --renew-hook /usr/local/bin/renew.sh
```

Save and close the crontab. Now, whenever certbot renews the certificates, our script will be run to copy the files, adjust permissions, and restart the slapd server.

Configuring slapd to Offer Secure Connections

We need to add the **openldap** user to the **ssl-cert** group so slapd can read the private key:

```
sudo usermod -aG ssl-cert openldap
```

Restart slapd so it picks up the new group:

```
sudo systemctl restart slapd
```

Finally, we need to configure slapd to actually use these certificates and keys. To do this we put all of our config changes in an *LDIF* file — which stands for LDAP data interchange format — and then load the changes into our LDAP server with the ldapmodify command.

Open up a new LDIF file:

```
cd ~
nano ssl.ldif
```

This will open a blank file. Paste the following into the file, updating the filenames to reflect your domain:

ssl.ldif

```
dn: cn=config
changetype: modify
add: olcTLSCACertificateFile
olcTLSCACertificateFile: /etc/ssl/certs/example.com.fullchain.pem
-
add: olcTLSCertificateFile
olcTLSCertificateFile: /etc/ssl/certs/example.com.cert.pem
-
add: olcTLSCertificateKeyFile
olcTLSCertificateKeyFile: /etc/ssl/private/example.com.privkey.pem
```

Save and close the file, then apply the changes with ldapmodify:

```
sudo ldapmodify -H ldapi:// -Y EXTERNAL -f ssl.ldif
```

Output

```
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
modifying entry "cn=config"
```

We don't need to reload slapd to load the new certificates, this happened automatically when we updated the config with ldapmodify. Run the ldapwhoami command one more time, to verify. This time we need to use the proper hostname and add the -ZZ option to force a secure connection:

```
ldapwhoami -H ldap://example.com -x -ZZ
```

We need the full hostname when using a secure connection because the client will check to make sure that the hostname matches the hostname on the certificate. This prevents man-in-the-middle attacks where an attacker could intercept your connection and impersonate your server.

The ldapwhoami command should return anonymous, with no errors. We've successfully encrypted our LDAP connection.

Configure LDAP Server

Tuesday, March 17, 2020 1:36 PM

Configure LDAP Server in order to share users' accounts in your local networks.

[1] Install OpenLDAP.

```
root@dlp:~# apt -y install slapd ldap-utils
# set LDAP admin password during installation like follows
 +--------------------------| Configuring slapd |-------------------------+
 | Please enter the password for the admin entry in your LDAP directory.  |
 |                                                                        |
 | Administrator password:                                                |
 |                                                                        |
 | ********_______________________________________________________________ |
 |                                                                        |
 |                          <Ok>                                          |
 |                                                                        |
 +------------------------------------------------------------------------+

# confirm settings
root@dlp:~# slapcat
dn: dc=srv,dc=world
objectClass: top
objectClass: dcObject
objectClass: organization
o: srv.world
dc: srv
structuralObjectClass: organization
entryUUID: ea77e6e6-fc14-1037-85b5-a3ba02104140
creatorsName: cn=admin,dc=srv,dc=world
createTimestamp: 20180604073035Z
entryCSN: 20180604073035.735717Z#000000#000#000000
modifiersName: cn=admin,dc=srv,dc=world
modifyTimestamp: 20180604073035Z

dn: cn=admin,dc=srv,dc=world
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: LDAP administrator
userPassword:: e1NTSEF9QlNFODVkY1htb0ZuQzBlODIxNURYVzFUZEk1eTRWem0=
structuralObjectClass: organizationalRole
entryUUID: ea7c7918-fc14-1037-85b6-a3ba02104140
creatorsName: cn=admin,dc=srv,dc=world
createTimestamp: 20180604073035Z
entryCSN: 20180604073035.765746Z#000000#000#000000
modifiersName: cn=admin,dc=srv,dc=world
modifyTimestamp: 20180604073035Z
```

[2] Add base dn for Users and Groups.

```
root@dlp:~# vi base.ldif
# create new
# change to your own suffix for the field [dc=srv,dc=world]
dn: ou=people,dc=srv,dc=world
objectClass: organizationalUnit
ou: people

dn: ou=groups,dc=srv,dc=world
objectClass: organizationalUnit
ou: groups

root@dlp:~# ldapadd -x -D cn=admin,dc=srv,dc=world -W -f base.ldif
Enter LDAP Password:     # LDAP admin password (set in installation of openldap)
adding new entry "ou=people,dc=srv,dc=world"

adding new entry "ou=groups,dc=srv,dc=world"
```

Add User Accounts

Tuesday, March 17, 2020 1:36 PM

Add LDAP User Accounts in the OpenLDAP Server.

[1] Add a user.

```
# generate encrypted password
root@dlp:~# slappasswd
New password:
Re-enter new password:
{SSHA}xxxxxxxxxxxxxxxxx
root@dlp:~# vi ldapuser.ldif
# create new
# replace to your own domain name for "dc=***,dc=***" section
dn: uid=bionic,ou=people,dc=srv,dc=world
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
cn: bionic
sn: ubuntu
userPassword: {SSHA}xxxxxxxxxxxxxxxxx
loginShell: /bin/bash
uidNumber: 2000
gidNumber: 2000
homeDirectory: /home/bionic

dn: cn=bionic,ou=groups,dc=srv,dc=world
objectClass: posixGroup
cn: bionic
gidNumber: 2000
memberUid: bionic

root@dlp:~# ldapadd -x -D cn=admin,dc=srv,dc=world -W -f ldapuser.ldif
Enter LDAP Password:
adding new entry "uid=bionic,ou=people,dc=srv,dc=world"

adding new entry "cn=bionic,ou=groups,dc=srv,dc=world"
```

[2] Add users and groups in local passwd/group to LDAP directory.

```
root@dlp:~# vi ldapuser.sh
# extract local users and groups who have 1000-9999 digit UID
# replace "SUFFIX=***" to your own domain name
# this is an example, free to modify
#!/bin/bash

SUFFIX='dc=srv,dc=world'
LDIF='ldapuser.ldif'

echo -n > $LDIF
GROUP_IDS=()
grep "x:[1-9][0-9][0-9][0-9]:" /etc/passwd | (while read TARGET_USER
do
  USER_ID="$(echo "$TARGET_USER" | cut -d':' -f1)"
```

```
    USER_NAME="$(echo "$TARGET_USER" | cut -d':' -f5 | cut -d',' -f1 )"
    [ ! "$USER_NAME" ] && USER_NAME="$USER_ID"

    LDAP_SN="$(echo "$USER_NAME" | awk '{print $2}')"
    [ ! "$LDAP_SN" ] && LDAP_SN="$USER_ID"

    LASTCHANGE_FLAG="$(grep "${USER_ID}:" /etc/shadow | cut -d':' -f3)"
    [ ! "$LASTCHANGE_FLAG" ] && LASTCHANGE_FLAG="0"

    SHADOW_FLAG="$(grep "${USER_ID}:" /etc/shadow | cut -d':' -f9)"
    [ ! "$SHADOW_FLAG" ] && SHADOW_FLAG="0"

    GROUP_ID="$(echo "$TARGET_USER" | cut -d':' -f4)"
    [ ! "$(echo "${GROUP_IDS[@]}" | grep "$GROUP_ID")" ] && GROUP_IDS=("${GROUP_IDS[@]}" "$GROUP_ID")

    echo "dn: uid=$USER_ID,ou=people,$SUFFIX" >> $LDIF
    echo "objectClass: inetOrgPerson" >> $LDIF
    echo "objectClass: posixAccount" >> $LDIF
    echo "objectClass: shadowAccount" >> $LDIF
    echo "sn: $LDAP_SN" >> $LDIF
    echo "givenName: $(echo "$USER_NAME" | awk '{print $1}')" >> $LDIF
    echo "cn: $(echo "$USER_NAME" | awk '{print $1}')" >> $LDIF
    echo "displayName: $USER_NAME" >> $LDIF
    echo "uidNumber: $(echo "$TARGET_USER" | cut -d':' -f3)" >> $LDIF
    echo "gidNumber: $(echo "$TARGET_USER" | cut -d':' -f4)" >> $LDIF
    echo "userPassword: {crypt}$(grep "${USER_ID}:" /etc/shadow | cut -d':' -f2)" >> $LDIF
    echo "gecos: $USER_NAME" >> $LDIF
    echo "loginShell: $(echo "$TARGET_USER" | cut -d':' -f7)" >> $LDIF
    echo "homeDirectory: $(echo "$TARGET_USER" | cut -d':' -f6)" >> $LDIF
    echo "shadowExpire: $(passwd -S "$USER_ID" | awk '{print $7}')" >> $LDIF
    echo "shadowFlag: $SHADOW_FLAG" >> $LDIF
    echo "shadowWarning: $(passwd -S "$USER_ID" | awk '{print $6}')" >> $LDIF
    echo "shadowMin: $(passwd -S "$USER_ID" | awk '{print $4}')" >> $LDIF
    echo "shadowMax: $(passwd -S "$USER_ID" | awk '{print $5}')" >> $LDIF
    echo "shadowLastChange: $LASTCHANGE_FLAG" >> $LDIF
    echo >> $LDIF
done

for TARGET_GROUP_ID in "${GROUP_IDS[@]}"
do
    LDAP_CN="$(grep ":${TARGET_GROUP_ID}:" /etc/group | cut -d':' -f1)"

    echo "dn: cn=$LDAP_CN,ou=groups,$SUFFIX" >> $LDIF
    echo "objectClass: posixGroup" >> $LDIF
    echo "cn: $LDAP_CN" >> $LDIF
    echo "gidNumber: $TARGET_GROUP_ID" >> $LDIF

    for MEMBER_UID in $(grep ":${TARGET_GROUP_ID}:" /etc/passwd | cut -d':' -f1,3)
    do
        UID_NUM=$(echo "$MEMBER_UID" | cut -d':' -f2)
        [ $UID_NUM -ge 1000 -a $UID_NUM -le 9999 ] && echo "memberUid: $(echo "$MEMBER_UID" | cut -d':' -f1)" >> $LDIF
    done
    echo >> $LDIF
done
)
```

```
root@dlp:~# bash ldapuser.sh
root@dlp:~# ldapadd -x -D cn=admin,dc=srv,dc=world -W -f ldapuser.ldif
Enter LDAP Password:
adding new entry "uid=ubuntu,ou=people,dc=srv,dc=world"

adding new entry "uid=redhat,ou=people,dc=srv,dc=world"

adding new entry "uid=debian,ou=people,dc=srv,dc=world"

adding new entry "cn=ubuntu,ou=groups,dc=srv,dc=world"

adding new entry "cn=redhat,ou=groups,dc=srv,dc=world"

adding new entry "cn=debian,ou=groups,dc=srv,dc=world"
```

[3] If you'd like to delete LDAP User or Group, Do as below.

```
root@dlp:~# ldapdelete -x -W -D 'cn=admin,dc=srv,dc=world' "uid=ubuntu,ou=people,dc=srv,dc=world"
Enter LDAP Password:
root@dlp:~# ldapdelete -x -W -D 'cn=admin,dc=srv,dc=world' "cn=ubuntu,ou=groups,dc=srv,dc=world"
Enter LDAP Password:
```

LDAP over SSL/TLS

Tuesday, March 17, 2020 1:36 PM

Configure LDAP over SSL/TLS to make connection be secure.
[1]
On this exmaple, create and use self-signed certificates like here.
[2] Configure LDAP Server.

```
root@dlp:~# cp /etc/ssl/private/server.key \
/etc/ssl/private/server.crt \
/etc/ssl/certs/ca-certificates.crt \
/etc/ldap/sasl2/
root@dlp:~# chown openldap. /etc/ldap/sasl2/server.key \
/etc/ldap/sasl2/server.crt \
/etc/ldap/sasl2/ca-certificates.crt
root@dlp:~# vi mod_ssl.ldif
# create new
dn: cn=config
changetype: modify
add: olcTLSCACertificateFile
olcTLSCACertificateFile: /etc/ldap/sasl2/ca-certificates.crt
-
replace: olcTLSCertificateFile
olcTLSCertificateFile: /etc/ldap/sasl2/server.crt
-
replace: olcTLSCertificateKeyFile
olcTLSCertificateKeyFile: /etc/ldap/sasl2/server.key

root@dlp:~# ldapmodify -Y EXTERNAL -H ldapi:/// -f mod_ssl.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
modifying entry "cn=config"
```

[3] Configure LDAP Client
If you'd like to make sure the connection between LDAP server and client is encrypted, use tcpdump and other network capture software on LDAP server.

```
root@www:~# echo "TLS_REQCERT allow" >> /etc/ldap/ldap.conf
root@www:~# vi /etc/ldap.conf
# line 261: uncomment
ssl start_tls
root@www:~# exit
logout
www login: ubuntu     # LDAP user
Password:
Last login: Tue Jun  5 11:22:06 JST 2018 on ttyS0


  System information as of Tue Jun  5 15:05:32 JST 2018

  System load:  0.0              Processes:           93
  Usage of /:   6.4% of 28.45GB   Users logged in:     0
  Memory usage: 3%               IP address for ens3: 10.0.0.31
  Swap usage:   0%

 * Meltdown, Spectre and Ubuntu: What are the attack vectors,
   how the fixes work, and everything else you need to know
   - https://ubu.one/u2Know

16 packages can be updated.
8 updates are security updates.

ubuntu@www:~$     # logined
```

LDAP Replication

Tuesday, March 17, 2020 1:34 PM

Configure OpenLDAP Replication to continue Directory service if OpenLDAP master server would be down.
OpenLDAP master server is called [Provider] and OpenLDAP Slave server is called [Consumer] on OpenLDAP.
[1]
Configure Basic LDAP Server settings on both Provider and Consumer, refer to here.
[2] Configure LDAP Provider. Add syncprov module.

```
root@dlp:~# vi mod_syncprov.ldif
# create new
dn: cn=module,cn=config
objectClass: olcModuleList
cn: module
olcModulePath: /usr/lib/ldap
olcModuleLoad: syncprov.la

root@dlp:~# ldapadd -Y EXTERNAL -H ldapi:/// -f mod_syncprov.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
adding new entry "cn=module,cn=config"

root@dlp:~# vi syncprov.ldif
# create new
dn: olcOverlay=syncprov,olcDatabase={1}mdb,cn=config
objectClass: olcOverlayConfig
objectClass: olcSyncProvConfig
olcOverlay: syncprov
olcSpSessionLog: 100

root@dlp:~# ldapadd -Y EXTERNAL -H ldapi:/// -f syncprov.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
adding new entry "olcOverlay=syncprov,olcDatabase={1}mdb,cn=config"
```

[3] Configure LDAP Consumer.

```
root@node01:~# vi syncrepl.ldif
# create new
dn: olcDatabase={1}mdb,cn=config
changetype: modify
add: olcSyncRepl
olcSyncRepl: rid=001
  # LDAP server's URI
  provider=ldap://10.0.0.30:389/
  bindmethod=simple
  # own domain name
  binddn="cn=admin,dc=srv,dc=world"
  # directory manager's password
  credentials=password
  searchbase="dc=srv,dc=world"
  # includes subtree
```

```
 scope=sub
 schemachecking=on
 type=refreshAndPersist
 # [retry interval] [retry times] [interval of re-retry] [re-retry times]
 retry="30 5 300 3"
 # replication interval
 interval=00:00:05:00

root@node01:~# ldapadd -Y EXTERNAL -H ldapi:/// -f syncrepl.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
modifying entry "olcDatabase={1}mdb,cn=config"

# confirm settings to search datas
root@node01:~# ldapsearch -x -b 'ou=people,dc=srv,dc=world'
dn: ou=people,dc=srv,dc=world
objectClass: organizationalUnit
ou: people

# bionic, people, srv.world
dn: uid=bionic,ou=people,dc=srv,dc=world
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
cn: bionic
sn: ubuntu
loginShell: /bin/bash
uidNumber: 2000
gidNumber: 2000
homeDirectory: /home/bionic
uid: bionic
.....
.....
```

[4] Configure LDAP Client to bind LDAP Consumer, too.

```
root@www:~# vi /etc/ldap.conf
# line 30: add LDAP Consumer
uri ldap://dlp.srv.world ldap://node01.srv.world
```

Multi-Master Replication

Tuesday, March 17, 2020 1:34 PM

Configure OpenLDAP Multi-Master Replication.
For the Settings of Provider/Consumer, it's impossible to add datas on Consumer server, but if configure this Multi -Master Settings, it's possbile to add on any Master server.
[1]
Configure Basic LDAP Server settings on all server, refer to here.
[2] Configure like follows on all servers. Add syncprov module.

```
root@dlp:~# vi mod_syncprov.ldif
# create new
dn: cn=module,cn=config
objectClass: olcModuleList
cn: module
olcModulePath: /usr/lib/ldap
olcModuleLoad: syncprov.la

root@dlp:~# ldapadd -Y EXTERNAL -H ldapi:/// -f mod_syncprov.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
adding new entry "cn=module,cn=config"

root@dlp:~# vi syncprov.ldif
# create new
dn: olcOverlay=syncprov,olcDatabase={1}mdb,cn=config
objectClass: olcOverlayConfig
objectClass: olcSyncProvConfig
olcOverlay: syncprov
olcSpSessionLog: 100

root@dlp:~# ldapadd -Y EXTERNAL -H ldapi:/// -f syncprov.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
adding new entry "olcOverlay=syncprov,olcDatabase={1}mdb,cn=config"
```

[3] Configure like follows on all servers. But only the parameters [olcServerID] and [provider=***], set different value on each server.

```
root@dlp:~# vi master01.ldif
# create new
dn: cn=config
changetype: modify
replace: olcServerID
# specify uniq ID number on each server
olcServerID: 101

dn: olcDatabase={1}mdb,cn=config
changetype: modify
add: olcSyncRepl
olcSyncRepl: rid=001
  # specify another LDAP server's URI
  provider=ldap://10.0.0.51:389/
  bindmethod=simple
  # own domain name
  binddn="cn=admin,dc=srv,dc=world"
  # directory manager's password
  credentials=password
  searchbase="dc=srv,dc=world"
  # includes subtree
  scope=sub
  schemachecking=on
  type=refreshAndPersist
  # [retry interval] [retry times] [interval of re-retry] [re-retry times]
  retry="30 5 300 3"
  # replication interval
  interval=00:00:05:00
-
add: olcMirrorMode
olcMirrorMode: TRUE

dn: olcOverlay=syncprov,olcDatabase={1}mdb,cn=config
changetype: add
objectClass: olcOverlayConfig
objectClass: olcSyncProvConfig
```

```
olcOverlay: syncprov

root@dlp:~# ldapmodify -Y EXTERNAL -H ldapi:/// -f master01.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
modifying entry "cn=config"

modifying entry "olcDatabase={1}mdb,cn=config"

adding new entry "olcOverlay=syncprov,olcDatabase={1}mdb,cn=config"
```

[4] Configure LDAP Client to bind all LDAP server.

```
root@www:~# vi /etc/ldap.conf
# line 30: add LDAP Consumer
uri ldap://dlp.srv.world ldap://node01.srv.world
```

LDAP Account Manager

Tuesday, March 17, 2020 1:34 PM

Install LDAP Account Manager to manage LDAP user accounts on Web GUI.
[1]
Install and start Apache2, refer to here.
[2]
Install PHP, refer to here.
[3] Install LDAP Account Manager.

```
root@dlp:~# apt -y install ldap-account-manager
root@dlp:~# vi /etc/php/7.2/apache2/php.ini
# line 404: set memory_limit more than 64M
memory_limit = 128M
root@dlp:~# vi /etc/apache2/conf-enabled/ldap-account-manager.conf
# line 12: change access permition if you need
#Require all granted
Require ip 127.0.0.1 10.0.0.0/24
root@dlp:~# systemctl restart apache2
```

[4] Access to [http://(your hostname or IP address)/lam/] with web browser from any Clients which are in the Network you set to allow. LDAP Account Manag er Login form is shown, then click [LAM configuration] which is on upper-right to set your server's profile.

[5] Click [Edit server prpfiles].

[6] Login with a LAM Admin user [lam]. Default password is the same with username.

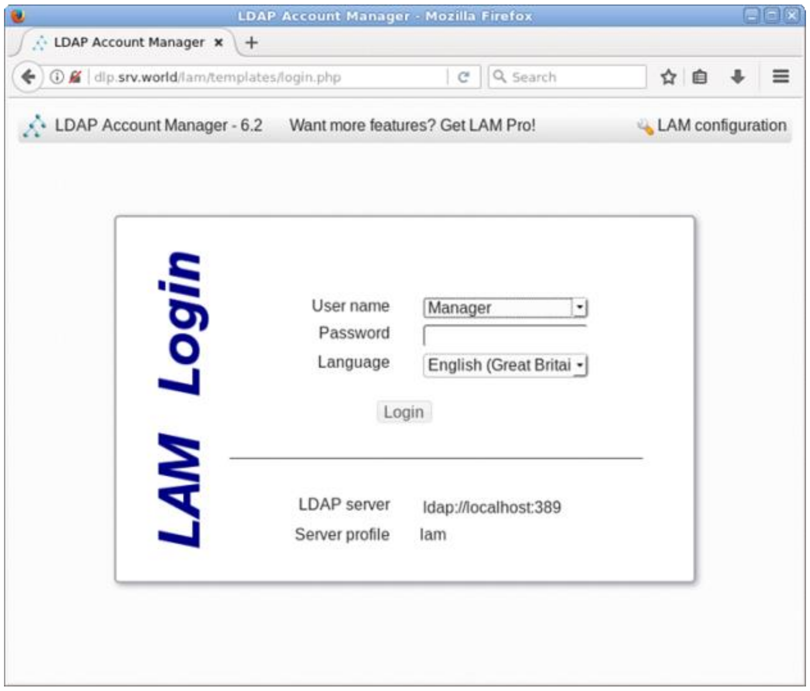
[7] Configure your LDAP server URL and Suffix.

[8] Scroll down and Consigure LDAP admin user's Suffix and also change default password for [lam] user. Next, Scroll up and move to [Account types] tab.

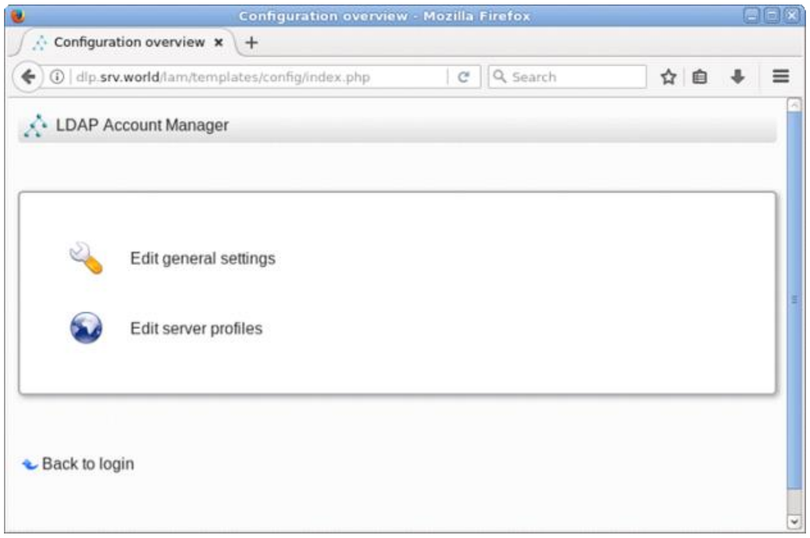
[9] Scroll down and Configure Users and Group Suffix. After setting, Click [Save] button.

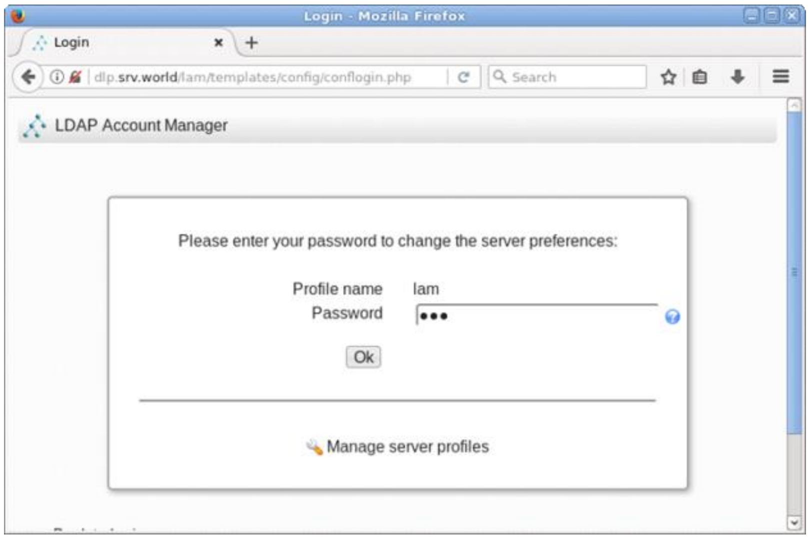
[10] After saving settings, Login form is shown, login with LDAP admin user.

[11] Just logined. It's possible to manage LDAP user accounts on here.


LDAP Account Manager - Mozilla Firefox
LDAP Account Manager
dlp.srv.world/lam/templates/login.php
Search
LDAP Account Manager - 6.2
Want more features? Get LAM Pro!
LAM configuration
LAM Login
Login

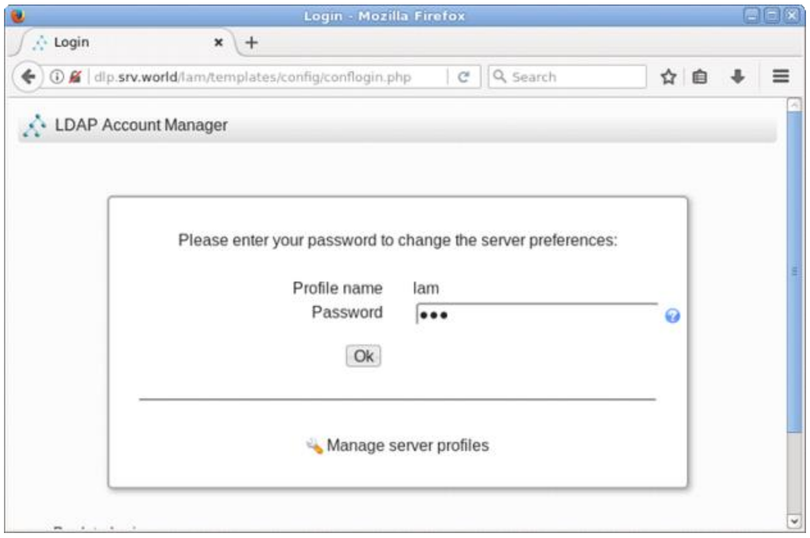
User name: Manager
Password:
Language: English (Great Britai
LDAP server: ldap://localhost:389
Server profile: lam





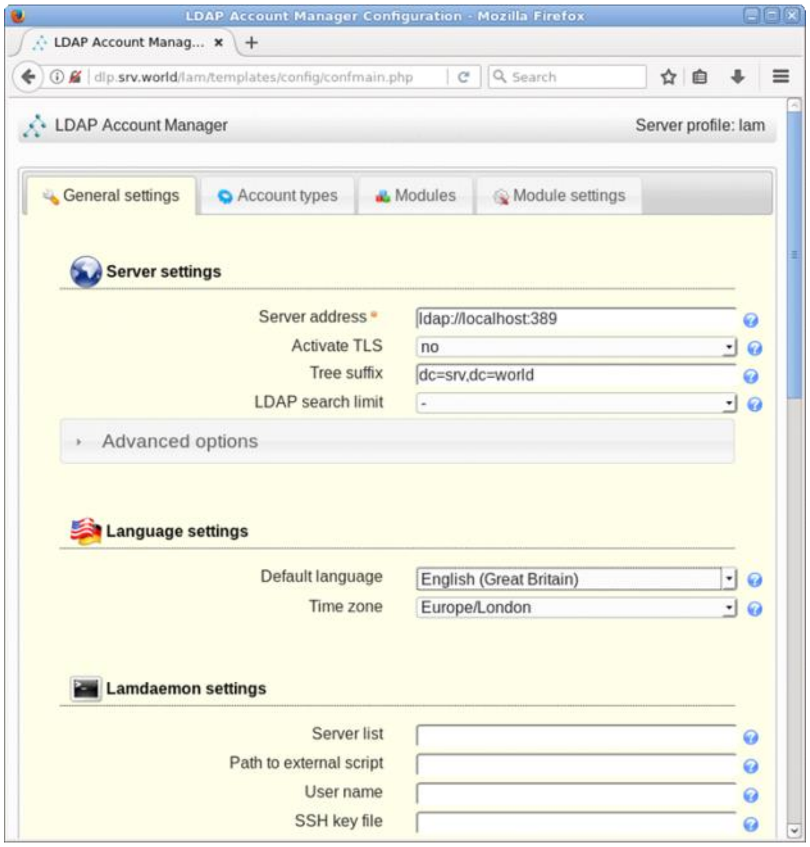
Please enter your password to change the server preferences:

Profile name	lam
Password	•••



Please enter your password to change the server preferences:

Profile name	lam
Password	•••



Server settings

Server address *	ldap://localhost:389
Activate TLS	no
Tree suffix	dc=srv,dc=world
LDAP search limit	-

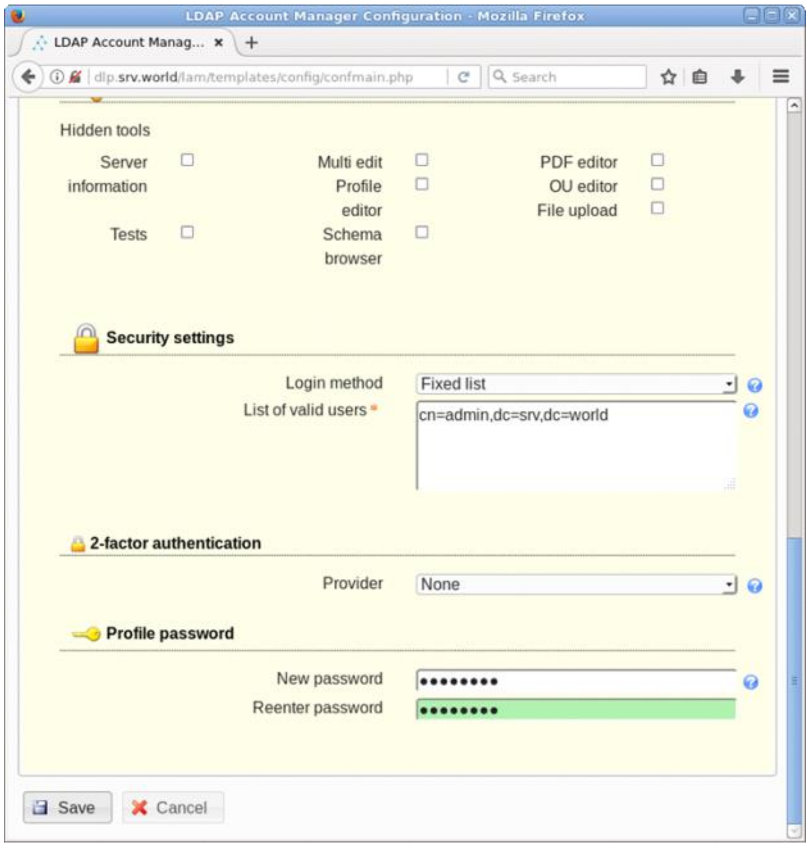
Advanced options

Language settings

Default language	English (Great Britain)
Time zone	Europe/London

Lamdaemon settings

Server list	
Path to external script	
User name	
SSH key file	


LDAP Account Manager Configuration - Mozilla Firefox
LDAP Account Manag...
dlp.srv.world/lam/templates/config/confmain.php
Search

Hidden tools

Server information	☐	Multi edit	☐	PDF editor	☐
		Profile editor	☐	OU editor	☐
Tests	☐	Schema browser	☐	File upload	☐

Security settings

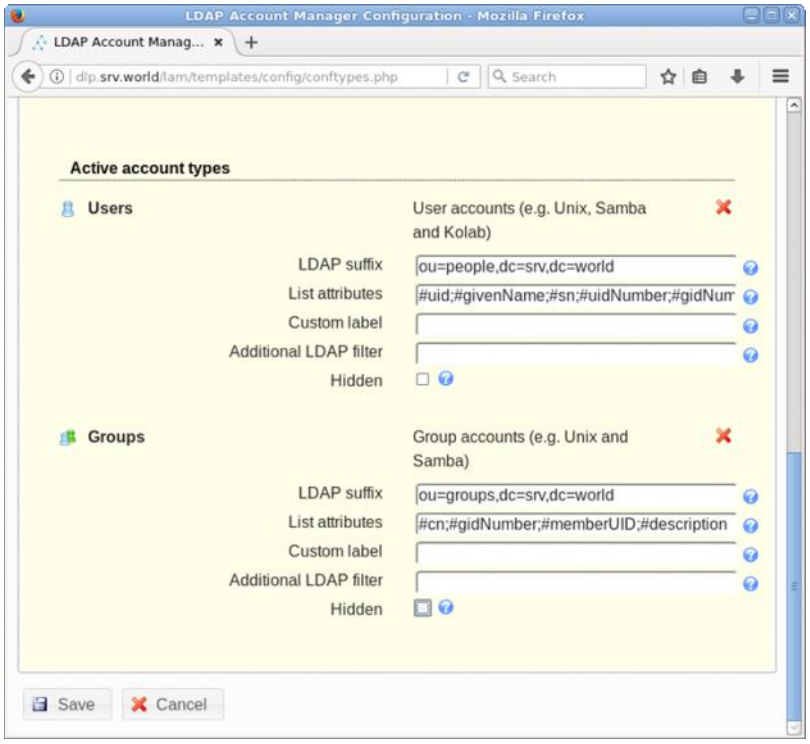
Login method	Fixed list
List of valid users *	cn=admin,dc=srv,dc=world

2-factor authentication

Provider	None

Profile password

New password	••••••••
Reenter password	••••••••


LDAP Account Manager Configuration - Mozilla Firefox
LDAP Account Manag...
dlp.srv.world/lam/templates/config/conftypes.php
Search

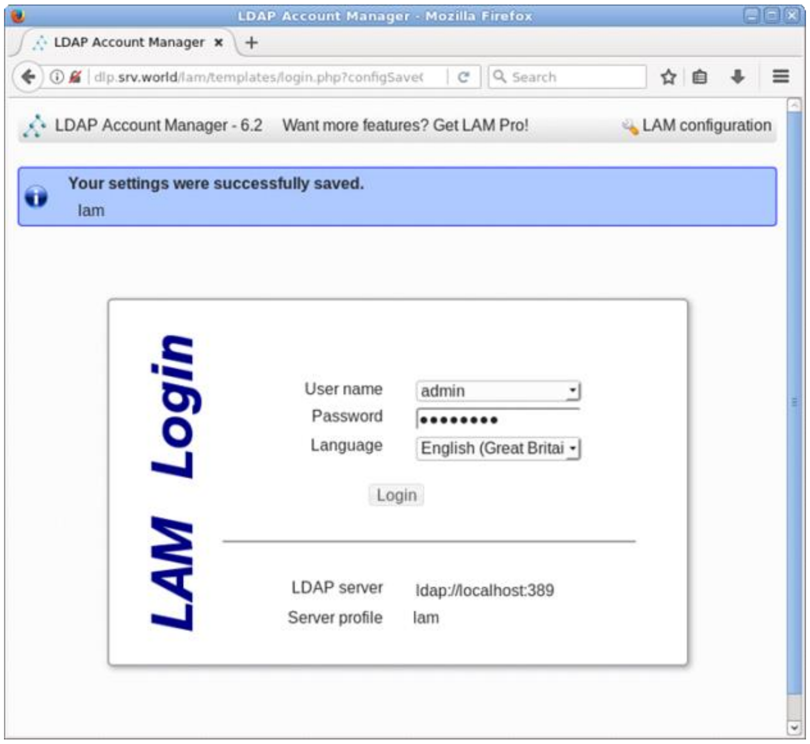
Active account types

Users — User accounts (e.g. Unix, Samba and Kolab)

LDAP suffix	ou=people,dc=srv,dc=world
List attributes	#uid;#givenName;#sn;#uidNumber;#gidNum
Custom label	
Additional LDAP filter	
Hidden	☐

Groups — Group accounts (e.g. Unix and Samba)

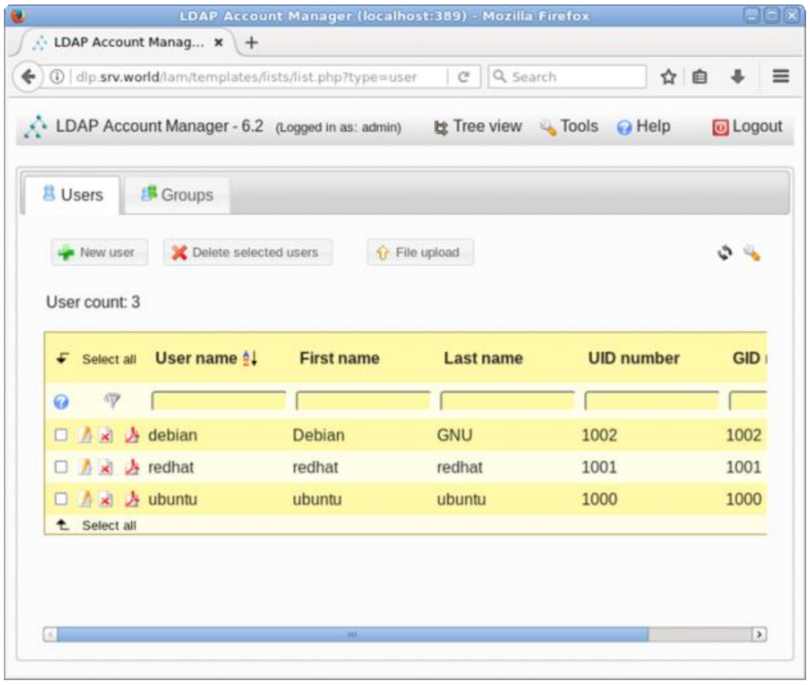
LDAP suffix	ou=groups,dc=srv,dc=world
List attributes	#cn;#gidNumber;#memberUID;#description
Custom label	
Additional LDAP filter	
Hidden	☐


LDAP Account Manager - Mozilla Firefox
LDAP Account Manager
dlp.srv.world/lam/templates/login.php?configSave(
Search
LDAP Account Manager - 6.2
Want more features? Get LAM Pro!
LAM configuration

Your settings were successfully saved.
lam

User name	admin
Password	●●●●●●●●
Language	English (Great Britai

LDAP server	ldap://localhost:389
Server profile	lam


LDAP Account Manager (localhost:389) - Mozilla Firefox
LDAP Account Manag...
dlp.srv.world/lam/templates/lists/list.php?type=user
Search
LDAP Account Manager - 6.2 (Logged in as: admin)
Tree view
Tools
Help
Logout
Users
Groups
New user
Delete selected users
File upload
Select all

User count: 3

User name	First name	Last name	UID number	GID
debian	Debian	GNU	1002	1002
redhat	redhat	redhat	1001	1001
ubuntu	ubuntu	ubuntu	1000	1000

BIND / DNS

Tuesday, March 17, 2020 1:20 PM

Prerequisites
To complete this tutorial, you will need the following infrastructure. Create each server in the same datacenter with private networking enabled:

A fresh Ubuntu 18.04 server to serve as the Primary DNS server, ns1
(Recommended) A second Ubuntu 18.04 server to serve as a Secondary DNS server, ns2
Additional servers in the same datacenter that will be using your DNS servers

On each of these servers, configure administrative access via a sudo user and a firewall by following our Ubuntu 18.04 initial server setup guide.

If you are unfamiliar with DNS concepts, it is recommended that you read at least the first three parts of our Introduction to Managing DNS.

Example Infrastructure and Goals
For the purposes of this article, we will assume the following:

We have two servers which will be designated as our DNS name servers. We will refer to these as ns1 and ns2 in this guide.
We have two additional client servers that will be using the DNS infrastructure we create. We will call these host1 and host2 in this guide. You can add as many as you'd like for your infrastructure.
All of these servers exist in the same datacenter. We will assume that this is the nyc3 datacenter.
All of these servers have private networking enabled (and are on the 10.128.0.0/16 subnet. You will likely have to adjust this for your servers).
All servers are connected to a project that runs on “example.com”. Since our DNS system will be entirely internal and private, you do not have to purchase a domain name. However, using a domain you own may help avoid conflicts with publicly routable domains.

With these assumptions, we decide that it makes sense to use a naming scheme that uses “nyc3.example.com” to refer to our private subnet or zone. Therefore, host1's private Fully-Qualified Domain Name (FQDN) will be host1.nyc3.example.com. Refer to the following table the relevant details:

Host	Role	Private FQDN	Private IP Address
ns1	Primary DNS Server	ns1.nyc3.example.com	10.128.10.11
ns2	Secondary DNS Server	ns2.nyc3.example.com	10.128.20.12
host1	Generic Host 1	host1.nyc3.example.com	10.128.100.101
host2	Generic Host 2	host2.nyc3.example.com	10.128.200.102

Note
Your existing setup will be different, but the example names and IP addresses will be used to demonstrate how to configure a DNS server to provide a
functioning internal DNS. You should be able to easily adapt this setup to your own environment by replacing the host names and private IP addresses with your own.
It is not necessary to use the region name of the datacenter in your naming scheme, but we use it here to denote that these hosts belong to a particular datacenter's private network.
If you utilize multiple datacenters, you can set up an internal DNS within each respective datacenter.

By the end of this tutorial, we will have a primary DNS server, ns1, and optionally a secondary DNS server, ns2, which will serve as a backup.

Let's get started by installing our Primary DNS server, ns1.

Installing BIND on DNS Servers

Note
Text that is highlighted in red is important! It will often be used to denote something that needs to be replaced with your own settings or
that it should be modified or added to a configuration file. For example, if you see something like host1.nyc3.example.com, replace it with
the FQDN of your own server. Likewise, if you see host1_private_IP, replace it with the private IP address of your own server.

On both DNS servers, ns1 and ns2, update the apt package cache by typing:

```
ns$ sudo apt-get update
```

Now install BIND:

```
ns$ sudo apt-get install bind9 bind9utils bind9-doc
```

Setting Bind to IPv4 Mode
Before continuing, let's set BIND to IPv4 mode since our private networking uses IPv4 exclusively. On both servers, edit the bind9 default settings file by typing:

```
ns$ sudo nano /etc/default/bind9
```

Add “-4” to the end of the OPTIONS parameter. It should look like the following:

```
/etc/default/bind9
. . .
OPTIONS="-u bind -4"
```

Save and close the file when you are finished.

Restart BIND to implement the changes:

```
ns$ sudo systemctl restart bind9
```

Now that BIND is installed, let's configure the primary DNS server.

Configuring the Primary DNS Server
BIND's configuration consists of multiple files, which are included from the main configuration file, named.conf. These filenames begin with named because that is the name of the process that BIND runs (short for “domain name daemon”). We will start with configuring the options file.

Configuring the Options File
On ns1, open the named.conf.options file for editing:

```
ns1$ sudo nano /etc/bind/named.conf.options
```

Above the existing options block, create a new ACL (access control list) block called “trusted”. This is where we will define a list of clients that we will allow recursive DNS queries from (i.e. your servers that are in the same datacenter as ns1). Using our example private IP addresses, we will add ns1, ns2, host1, and host2 to our list of trusted clients:

```
/etc/bind/named.conf.options — 1 of 3
acl "trusted" {
        10.128.10.11;    # ns1 - can be set to localhost
        10.128.20.12;    # ns2
        10.128.100.101;  # host1
        10.128.200.102;  # host2
};

options {

        . . .
```

Now that we have our list of trusted DNS clients, we will want to edit the options block.
Currently, the start of the block looks like the following:

```
/etc/bind/named.conf.options — 2 of 3
        . . .
};

options {
        directory "/var/cache/bind";
        . . .
}
```

Below the directory directive, add the highlighted configuration lines
(and substitute in the proper ns1 IP address) so it looks something like this:

```
/etc/bind/named.conf.options — 3 of 3
        . . .

};

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

        recursion yes;                 # enables resursive queries
        allow-recursion { trusted; };  # allows recursive queries from "trusted" clients
        listen-on { 10.128.10.11; };   # ns1 private IP address - listen on private network only
        allow-transfer { none; };      # disable zone transfers by default

        forwarders {
                8.8.8.8;
                8.8.4.4;
        };

        . . .
};
```

When you are finished, save and close the named.conf.options file. The above configuration specifies that only your own servers (the “trusted” ones) will be able to query your DNS server for outside domains.

Next, we will configure the local file, to specify our DNS zones.

Configuring the Local File
On ns1, open the named.conf.local file for editing:

```
ns1$ sudo nano /etc/bind/named.conf.local
```

Aside from a few comments, the file should be empty. Here, we will specify our forward and reverse zones. DNS zones designate a specific scope for managing and defining DNS records. Since our domains will all be within the “nyc3.example.com” subdomain, we will use that as our forward zone. Because our servers’ private IP addresses are each in the 10.128.0.0/16 IP space, we will set up a reverse zone so that we can define reverse lookups within that range.

Add the forward zone with the following lines, substituting the zone name with your own and the secondary DNS server’s private IP address in the allow-transfer directive:

```
/etc/bind/named.conf.local — 1 of 2
zone "nyc3.example.com" {
    type master;
    file "/etc/bind/zones/db.nyc3.example.com"; # zone file path
    allow-transfer { 10.128.20.12; };           # ns2 private IP address - secondary
};
```

Assuming that our private subnet is 10.128.0.0/16, add the reverse zone by with the following lines
(note that our reverse zone name starts with “128.10” which is the octet reversal of “10.128”):

```
/etc/bind/named.conf.local — 2 of 2
    . . .
};

zone "128.10.in-addr.arpa" {
    type master;
    file "/etc/bind/zones/db.10.128";  # 10.128.0.0/16 subnet
    allow-transfer { 10.128.20.12; };  # ns2 private IP address - secondary
};
```

If your servers span multiple private subnets but are in the same datacenter, be sure to specify an additional zone and zone file for each distinct subnet. When you are finished adding all of your desired zones, save and exit the named.conf.local file.

Now that our zones are specified in BIND, we need to create the corresponding forward and reverse zone files.

Creating the Forward Zone File
The forward zone file is where we define DNS records for forward DNS lookups. That is, when the DNS receives a name query, “host1.nyc3.example.com” for example, it will look in the forward zone file to resolve host1’s corresponding private IP address.

Let’s create the directory where our zone files will reside. According to our named.conf.local configuration, that location should be /etc/bind/zones:

```
ns1$ sudo mkdir /etc/bind/zones
```

We will base our forward zone file on the sample db.local zone file. Copy it to the proper location with the following commands:

```
ns1$ sudo cp /etc/bind/db.local /etc/bind/zones/db.nyc3.example.com
```

Now let’s edit our forward zone file:

```
ns1$ sudo nano /etc/bind/zones/db.nyc3.example.com
```

Initially, it will look something like the following:

/etc/bind/zones/db.nyc3.example.com — original

```
$TTL    604800
@       IN      SOA     localhost. root.localhost. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      localhost.      ; delete this line
@       IN      A       127.0.0.1       ; delete this line
@       IN      AAAA    ::1             ; delete this line
```

First, you will want to edit the SOA record. Replace the first “localhost” with ns1’s FQDN, then replace “root.localhost” with “admin.nyc3.example.com”. Every time you edit a zone file, you need to increment the serial value before you restart the named process. We will increment it to “3”. It should now look something like this:

/etc/bind/zones/db.nyc3.example.com — updated 1 of 3

```
@       IN      SOA     ns1.nyc3.example.com. admin.nyc3.example.com. (
                              3         ; Serial

                              . . .
```

Next, delete the three records at the end of the file (after the SOA record). If you’re not sure which lines to delete, they are marked with a “delete this line” comment above.

At the end of the file, add your name server records with the following lines (replace the names with your own). Note that the second column specifies that these are “NS” records:

/etc/bind/zones/db.nyc3.example.com — updated 2 of 3

```
. . .

; name servers - NS records
    IN      NS      ns1.nyc3.example.com.
    IN      NS      ns2.nyc3.example.com.
```

Now, add the A records for your hosts that belong in this zone. This includes any server whose name we want to end with “.nyc3.example.com”
(substitute the names and private IP addresses). Using our example names and private IP addresses, we will add A records for ns1, ns2, host1, and host2 like so:

/etc/bind/zones/db.nyc3.example.com — updated 3 of 3

```
. . .

; name servers - A records
ns1.nyc3.example.com.          IN      A       10.128.10.11
ns2.nyc3.example.com.          IN      A       10.128.20.12

; 10.128.0.0/16 - A records
host1.nyc3.example.com.        IN      A      10.128.100.101
host2.nyc3.example.com.        IN      A      10.128.200.102
```

Save and close the db.nyc3.example.com file.

Our final example forward zone file looks like the following:

/etc/bind/zones/db.nyc3.example.com — updated

```
$TTL    604800
@       IN      SOA     ns1.nyc3.example.com. admin.nyc3.example.com. (
                  3     ; Serial
             604800     ; Refresh
              86400     ; Retry
            2419200     ; Expire
             604800 )   ; Negative Cache TTL
;
; name servers - NS records
     IN      NS      ns1.nyc3.example.com.
     IN      NS      ns2.nyc3.example.com.

; name servers - A records
ns1.nyc3.example.com.          IN      A       10.128.10.11
ns2.nyc3.example.com.          IN      A       10.128.20.12
```

```
; 10.128.0.0/16 - A records
host1.nyc3.example.com.       IN      A      10.128.100.101
host2.nyc3.example.com.       IN      A      10.128.200.102
```

Now let's move onto the reverse zone file(s).

Creating the Reverse Zone File(s)
Reverse zone files are where we define DNS PTR records for reverse DNS lookups. That is, when the DNS receives a query by IP address, "10.128.100.101" for example, it will look in the reverse zone file(s) to resolve the corresponding FQDN, "host1.nyc3.example.com" in this case.

On ns1, for each reverse zone specified in the named.conf.local file, create a reverse zone file. We will base our reverse zone file(s) on the sample db.127 zone file. Copy it to the proper location with the following commands (substituting the destination filename so it matches your reverse zone definition):

```
ns1$ sudo cp /etc/bind/db.127 /etc/bind/zones/db.10.128
```

Edit the reverse zone file that corresponds to the reverse zone(s) defined in named.conf.local:

```
ns1$ sudo nano /etc/bind/zones/db.10.128
```

Initially, it will look something like the following:

/etc/bind/zones/db.10.128 — original
```
$TTL    604800
@       IN      SOA     localhost. root.localhost. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      localhost.      ; delete this line
1.0.0   IN      PTR     localhost.      ; delete this line
```
In the same manner as the forward zone file, you will want to edit the SOA record and increment the serial value. It should look something like this:

/etc/bind/zones/db.10.128 — updated 1 of 3
```
@       IN      SOA     ns1.nyc3.example.com. admin.nyc3.example.com. (
                              3         ; Serial

                              . . .
```
Now delete the two records at the end of the file (after the SOA record). If you're not sure which lines to delete, they are marked with a "delete this line" comment above.

At the end of the file, add your name server records with the following lines (replace the names with your own). Note that the second column specifies that these are "NS" records:

/etc/bind/zones/db.10.128 — updated 2 of 3
```
. . .

; name servers - NS records
      IN      NS      ns1.nyc3.example.com.
      IN      NS      ns2.nyc3.example.com.
```

Then add PTR records for all of your servers whose IP addresses are on the subnet of the zone file that you are editing. In our example, this includes all of our hosts because they are all on the 10.128.0.0/16 subnet. Note that the first column consists of the last two octets of your servers' private IP addresses in reversed order. Be sure to substitute names and private IP addresses to match your servers:

/etc/bind/zones/db.10.128 — updated 3 of 3
```
. . .

; PTR Records
11.10   IN      PTR     ns1.nyc3.example.com.    ; 10.128.10.11
12.20   IN      PTR     ns2.nyc3.example.com.    ; 10.128.20.12
101.100 IN      PTR     host1.nyc3.example.com.  ; 10.128.100.101
102.200 IN      PTR     host2.nyc3.example.com.  ; 10.128.200.102
```

Save and close the reverse zone file (repeat this section if you need to add more reverse zone files).

Our final example reverse zone file looks like the following:

/etc/bind/zones/db.10.128 — updated
```
$TTL    604800
@       IN      SOA     nyc3.example.com. admin.nyc3.example.com. (
                              3         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
; name servers
      IN      NS      ns1.nyc3.example.com.
      IN      NS      ns2.nyc3.example.com.

; PTR Records
11.10   IN      PTR     ns1.nyc3.example.com.    ; 10.128.10.11
12.20   IN      PTR     ns2.nyc3.example.com.    ; 10.128.20.12
101.100 IN      PTR     host1.nyc3.example.com.  ; 10.128.100.101
102.200 IN      PTR     host2.nyc3.example.com.  ; 10.128.200.102
```

We're done editing our files, so next we can check our files for errors.

Checking the BIND Configuration Syntax
Run the following command to check the syntax of the named.conf* files:

```
ns1$ sudo named-checkconf
```

If your named configuration files have no syntax errors, you will return to your shell prompt and see no error messages. If there are problems with your configuration files, review the error message and the “Configure Primary DNS Server” section, then try named-checkconf again.

The named-checkzone command can be used to check the correctness of your zone files. Its first argument specifies a zone name, and the second argument specifies the corresponding zone file, which are both defined in named.conf.local.

For example, to check the “nyc3.example.com” forward zone configuration, run the following command (change the names to match your forward zone and file):

```
$ sudo named-checkzone nyc3.example.com db.nyc3.example.com
```

And to check the “128.10.in-addr.arpa” reverse zone configuration, run the following command (change the numbers to match your reverse zone and file):

```
$ sudo named-checkzone 128.10.in-addr.arpa /etc/bind/zones/db.10.128
```

When all of your configuration and zone files have no errors in them, you should be ready to restart the BIND service.

Restarting BIND
Restart BIND:

```
ns1$ sudo systemctl restart bind9
```

If you have the UFW firewall configured, open up access to BIND by typing:

```
ns1$ sudo ufw allow Bind9
```

Your primary DNS server is now setup and ready to respond to DNS queries. Let’s move on to creating the secondary DNS server.

Configuring the Secondary DNS Server
In most environments, it is a good idea to set up a secondary DNS server that will respond to requests if the primary becomes unavailable. Luckily, the secondary DNS server is much easier to configure.

On ns2, edit the named.conf.options file:

```
ns2$ sudo nano /etc/bind/named.conf.options
```

At the top of the file, add the ACL with the private IP addresses of all of your trusted servers:

/etc/bind/named.conf.options — updated 1 of 2 (secondary)
```
acl "trusted" {
        10.128.10.11;   # ns1
        10.128.20.12;   # ns2 - can be set to localhost
        10.128.100.101;  # host1
        10.128.200.102;  # host2
};

options {

        . . .
```

Below the directory directive, add the following lines:

/etc/bind/named.conf.options — updated 2 of 2 (secondary)
```
        recursion yes;
        allow-recursion { trusted; };
        listen-on { 10.128.20.12; };      # ns2 private IP address
        allow-transfer { none; };          # disable zone transfers by default

        forwarders {
                8.8.8.8;
                8.8.4.4;
        };
```

Save and close the named.conf.options file. This file should look exactly like ns1’s named.conf.options file except it should be configured to listen on ns2’s private IP address.

Now edit the named.conf.local file:

```
ns2$ sudo nano /etc/bind/named.conf.local
```

Define slave zones that correspond to the master zones on the primary DNS server. Note that the type is “slave”, the file does not contain a path, and there is a masters directive which should be set to the primary DNS server’s private IP address. If you defined multiple reverse zones in the primary DNS server, make sure to add them all here:

/etc/bind/named.conf.local — updated (secondary)
```
zone "nyc3.example.com" {
    type slave;
    file "db.nyc3.example.com";
    masters { 10.128.10.11; };  # ns1 private IP
};

zone "128.10.in-addr.arpa" {
    type slave;
    file "db.10.128";
    masters { 10.128.10.11; };  # ns1 private IP
};
```

Now save and close the named.conf.local file.

Run the following command to check the validity of your configuration files:

```
ns2$ sudo named-checkconf
```

Once that checks out, restart BIND:

```
ns2$ sudo systemctl restart bind9
```

Allow DNS connections to the server by altering the UFW firewall rules:

```
ns2$ sudo ufw allow Bind9
```

Now you have primary and secondary DNS servers for private network name and IP address resolution.
Now you must configure your client servers to use your private DNS servers.

Configuring DNS Clients
Before all of your servers in the "trusted" ACL can query your DNS servers, you must configure each of them to use ns1 and ns2 as name servers. This process varies depending on OS, but for most Linux distributions it involves adding your name servers to the /etc/resolv.conf file.

Ubuntu 18.04 Clients
On Ubuntu 18.04, networking is configured with Netplan, an abstraction that allows you to write standardized network configuration and apply it to incompatible backend networking software. To configure DNS, we need to write a Netplan configuration file.

First, find the device associated with your private network by querying the private subnet with the ip address command:

```
$ ip address show to 10.128.0.0/16
```

Output
```
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    inet 10.128.100.101/16 brd 10.128.255.255 scope global eth1
          valid_lft forever preferred_lft forever
```

In this example, the private interface is eth1.

Next, create a new file in /etc/netplan called 00-private-nameservers.yaml:

```
$ sudo nano /etc/netplan/00-private-nameservers.yaml
```

Inside, paste the following contents. You will need to modify the interface of the private network, the addresses of your ns1 and ns2 DNS servers, and the DNS zone:

Note: Netplan uses the YAML data serialization format for its configuration files. Because YAML uses indentation and whitespace to define its data structure, make sure that
your definition uses consistent indentation to avoid errors.

/etc/netplan 00-private-nameservers.yaml
```
network:
    version: 2
    ethernets:
        eth1:                                 # Private network interface
            nameservers:
                addresses:
                - 10.128.10.11              # Private IP for ns1
                - 10.132.20.12              # Private IP for ns2
                search: [ nyc3.example.com ]  # DNS zone
```

Save and close the file when you are finished.

Next, tell Netplan to attempt to use the new configuration file by using netplan try. If there are problems that cause a loss of networking, Netplan will automatically roll back the changes after a timeout:

```
$ sudo netplan try
```

Output
```
Warning: Stopping systemd-networkd.service, but it can still be activated by:
  systemd-networkd.socket
Do you want to keep these settings?

Press ENTER before the timeout to accept the new configuration

Changes will revert in 120 seconds
```

If the countdown is updating correctly at the bottom, the new configuration is at least functional enough to not break your SSH connection. Press ENTER to accept the new configuration.

Now, check that the system's DNS resolver to determine if your DNS configuration has been applied:

```
$ sudo systemd-resolve --status
```

Scroll down until you see the section for your private network interface. You should see the private IP addresses for your DNS servers listed first,
followed by some fallback values. Your domain should should be in the "DNS Domain":

Output
```
. . .
Link 3 (eth1)
      Current Scopes: DNS
       LLMNR setting: yes
MulticastDNS setting: no
      DNSSEC setting: no
```

```
   DNSSEC supported: no
       DNS Servers: 10.128.10.11
                  10.128.20.12
                  67.207.67.2
                  67.207.67.3
        DNS Domain: nyc3.example.com
. . .
```

Your client should now be configured to use your internal DNS servers.

Ubuntu 16.04 and Debian Clients
On Ubuntu 16.04 and Debian Linux servers, you can edit the /etc/network/interfaces file:

```
$ sudo nano /etc/network/interfaces
```

Inside, find the dns-nameservers line, and prepend your own name servers in front of the list that is currently there.
Below that line, add a dns-search option pointed to the base domain of your infrastructure. In our case, this would be "nyc3.example.com":

```
/etc/network/interfaces
   . . .

   dns-nameservers 10.128.10.11 10.128.20.12 8.8.8.8
   dns-search nyc3.example.com

   . . .
```
Save and close the file when you are finished.

Now, restart your networking services, applying the new changes with the following commands. Make sure you replace eth0 with the name of your networking interface:

```
$ sudo ifdown --force eth0 && sudo ip addr flush dev eth0 && sudo ifup --force eth0
```

This should restart your network without dropping your current connection. If it worked correctly, you should see something like this:

```
Output
RTNETLINK answers: No such process
Waiting for DAD... Done
```

Double check that your settings were applied by typing:

```
$ cat /etc/resolv.conf
```

You should see your name servers in the /etc/resolv.conf file, as well as your search domain:

```
Output
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by resolvconf(8)
#     DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVERWRITTEN
nameserver 10.128.10.11
nameserver 10.128.20.12
nameserver 8.8.8.8
search nyc3.example.com
```

Your client is now configured to use your DNS servers.

CentOS Clients
On CentOS, RedHat, and Fedora Linux, edit the /etc/sysconfig/network-scripts/ifcfg-eth0 file. You may have to substitute eth0 with the name of your primary network interface:

```
$ sudo nano /etc/sysconfig/network-scripts/ifcfg-eth0
```
Search for the DNS1 and DNS2 options and set them to the private IP addresses of your primary and secondary name servers.
Add a DOMAIN parameter that with your infrastructure's base domain. In this guide, that would be "nyc3.example.com":

```
/etc/sysconfig/network-scripts/ifcfg-eth0
. . .
DNS1=10.128.10.11
DNS2=10.128.20.12
DOMAIN='nyc3.example.com'
. . .
```

Save and close the file when you are finished.

Now, restart the networking service by typing:

```
$ sudo systemctl restart network
```

The command may hang for a few seconds, but should return you to the prompt shortly.

Check that your changes were applied by typing:

```
$ cat /etc/resolv.conf
```

You should see your name servers and search domain in the list:

```
/etc/resolv.conf
nameserver 10.128.10.11
nameserver 10.128.20.12
search nyc3.example.com
```

Your client should now be able to connect to and use your DNS servers.

Testing Clients
Use nslookup to test if your clients can query your name servers. You should be able to do this on all of the clients that you have configured and are in the “trusted” ACL.

For CentOS clients, you may need to install the utility with:

```
$ sudo yum install bind-utils
```

We can start by performing a forward lookup.

Forward Lookup

For example, we can perform a forward lookup to retrieve the IP address of host1.nyc3.example.com by running the following command:

```
$ nslookup host1
```

Querying “host1” expands to “host1.nyc3.example.com because of the search option is set to your private subdomain, and DNS queries will attempt to look on that subdomain before looking for the host elsewhere. The output of the command above would look like the following:

```
Output
Server:     127.0.0.53
Address:    127.0.0.53#53

Non-authoritative answer:
Name:   host1.nyc3.example.com
Address: 10.128.100.101
```

Next, we can check reverse lookups.

Reverse Lookup
To test the reverse lookup, query the DNS server with host1’s private IP address:

```
$ nslookup 10.128.100.101
```

You should see output that looks like the following:

```
Output
11.10.128.10.in-addr.arpa   name = host1.nyc3.example.com.

Authoritative answers can be found from:
```

If all of the names and IP addresses resolve to the correct values, that means that your zone files are configured properly. If you receive unexpected values, be sure to review the zone files on your primary DNS server (e.g. db.nyc3.example.com and db.10.128).

Congratulations! Your internal DNS servers are now set up properly! Now we will cover maintaining your zone records.

Maintaining DNS Records
Now that you have a working internal DNS, you need to maintain your DNS records so they accurately reflect your server environment.

Adding a Host to DNS
Whenever you add a host to your environment (in the same datacenter), you will want to add it to DNS. Here is a list of steps that you need to take:

Primary Name Server
- Forward zone file: Add an "A” record for the new host, increment the value of “Serial”
- Reverse zone file: Add a “PTR” record for the new host, increment the value of “Serial”
- Add your new host’s private IP address to the “trusted” ACL (named.conf.options)

Test your configuration files:

```
$ sudo named-checkconf
$ sudo named-checkzone nyc3.example.com db.nyc3.example.com
$ sudo named-checkzone 128.10.in-addr.arpa /etc/bind/zones/db.10.128
```

Then reload BIND:

```
$ sudo systemctl reload bind9
```

Your primary server should be configured for the new host now.

Secondary Name Server
- Add your new host’s private IP address to the “trusted” ACL (named.conf.options)

Check the configuration syntax:

```
$ sudo named-checkconf
```
Then reload BIND:

```
$ sudo systemctl reload bind9
```
Your secondary server will now accept connections from the new host.

Configure New Host to Use Your DNS

Configure /etc/resolv.conf to use your DNS servers

Test using nslookup

Removing Host from DNS

If you remove a host from your environment or want to just take it out of DNS, just remove all the things that were added when you added the server to DNS (i.e. the reverse of the steps above).

Conclusion
Now you may refer to your servers' private network interfaces by name, rather than by IP address. This makes configuration of services and applications easier because you no longer have to remember the private IP addresses, and the files will be easier to read and understand. Also, now you can change your configurations to point to a new servers in a single place, your primary DNS server, instead of having to edit a variety of distributed configuration files, which eases maintenance.

Once you have your internal DNS set up, and your configuration files are using private FQDNs to specify network connections, it is critical that your DNS servers are properly maintained.
If they both become unavailable, your services and applications that rely on them will cease to function properly. This is why it is recommended to set up your DNS with at least one secondary server, and to maintain working backups of all of them.

BIND : Install / Configure

Tuesday, March 17, 2020 1:22 PM

Install BIND to configure DNS server which resolves domain name or IP address.
DNS uses 53/TCP,UDP.

[1] Install BIND 9.

```
root@dlp:~# apt -y install bind9 bind9utils
```

[2] Configure BIND 9.

This example is set with grobal IP address [172.16.0.80/29], Private IP address [10.0.0.0/24], Domain name [srv.world]. However, Please use your own IPs and domain name when you set config on your server. (Actually, [172.16.0.80/29] is for private IP address, though.)

```
root@dlp:~# vi /etc/bind/named.conf
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
# comment out
# include "/etc/bind/named.conf.default-zones";
# add
include "/etc/bind/named.conf.internal-zones";
include "/etc/bind/named.conf.external-zones";
root@dlp:~# vi /etc/bind/named.conf.internal-zones
# create new
# define for internal section
view "internal" {
        match-clients {
                localhost;
                10.0.0.0/24;
        };
        # set zone for internal
        zone "srv.world" {
                type master;
                file "/etc/bind/srv.world.lan";
                allow-update { none; };
        };
        # set zone for internal *note
        zone "0.0.10.in-addr.arpa" {
                type master;
                file "/etc/bind/0.0.10.db";
                allow-update { none; };
        };
        include "/etc/bind/named.conf.default-zones";
};

root@dlp:~# vi /etc/bind/named.conf.external-zones
# create new
# define for external section
view "external" {
        match-clients { any; };
        # allow any query
        allow-query { any; };
        # prohibit recursions
        recursion no;
        # set zone for external
        zone "srv.world" {
                type master;
                file "/etc/bind/srv.world.wan";
                allow-update { none; };
        };
        # set zone for external *note
        zone "80.0.16.172.in-addr.arpa" {
                type master;
                file "/etc/bind/80.0.16.172.db";
                allow-update { none; };
        };
};

# *note : For How to write for reverse resolving, Write network address reversely like below
# Case of 10.0.0.0/24
# network address       ⇒ 10.0.0.0
# range of network      ⇒ 10.0.0.0 - 10.0.0.255
# how to write        ⇒ 0.0.10.in-addr.arpa

# Case of 172.16.0.80/29
# network address       ⇒ 172.16.0.80
# range of network      ⇒ 172.16.0.80 - 172.16.0.87
# how to write        ⇒ 80.0.16.172.in-addr.arpa
```

[3]   Limit ranges you allow to access if needed.

```
root@dlp:~# vi /etc/bind/named.conf.options
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 {
        //      0.0.0.0;
        // };

         # query range you allow
         allow-query { localhost; 10.0.0.0/24; };
         # the range to transfer zone files
         allow-transfer { localhost; 10.0.0.0/24; };
         # recursion range you allow
         allow-recursion { localhost; 10.0.0.0/24; };

        //========================================================================
        // 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
        //============================================================================
        dnssec-validation auto;

        auth-nxdomain no;    # conform to RFC1035
        # change if not use IPV6
        listen-on-v6 { none; };
};
```

BIND : Start and Verify Resolution

Tuesday, March 17, 2020 1:23 PM

Restart BIND to take effect changes and make sure it's no ploblem.

[1] Change DNS setting to refer to local DNS.

```
root@dlp:~# vi /etc/netplan/01-netcfg.yaml
# change to own
   nameservers:
      addresses: [10.0.0.30]

root@dlp:~# netplan apply
root@dlp:~# systemctl restart bind9
```

[2] Try to resolv Name or Address normally.

```
root@dlp:~# dig dlp.srv.world.

; <<>> DiG 9.11.3-1ubuntu1-Ubuntu <<>> dlp.srv.world.
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 35748
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;dlp.srv.world.                 IN      A

;; ANSWER SECTION:
dlp.srv.world.          0       IN      A       10.0.0.30

;; Query time: 1 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: Tue May 08 17:15:07 JST 2018
;; MSG SIZE  rcvd: 58

root@dlp:~# dig -x 10.0.0.30

; <<>> DiG 9.11.3-1ubuntu1-Ubuntu <<>> -x 10.0.0.30
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 52054
;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;30.0.0.10.in-addr.arpa.                IN      PTR

;; ANSWER SECTION:
30.0.0.10.in-addr.arpa. 86400   IN      PTR     dlp.srv.world.
```

```
;; Query time: 2 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: Tue May 08 17:16:52 JST 2018
;; MSG SIZE  rcvd: 118
```

BIND : Set CNAME

Tuesday, March 17, 2020 1:23 PM

If you'd like to set another name to your Host, define CNAME record in zone file.

[1] Set CNAME record in zone file.

```
root@dlp:~# vi /etc/bind/srv.world.lan
$TTL 86400
@   IN  SOA     dlp.srv.world. root.srv.world. (
        # update serial
        2018050802  ;Serial
        3600        ;Refresh
        1800        ;Retry
        604800      ;Expire
        86400       ;Minimum TTL
)
        IN  NS      dlp.srv.world.
        IN  A       10.0.0.30
        IN  MX 10   dlp.srv.world.

dlp     IN  A       10.0.0.30

# aliase IN CNAME server's name
ftp     IN  CNAME   dlp.srv.world.

root@dlp:~# rndc reload
server reload successful
root@dlp:~# dig ftp.srv.world.

; <<>> DiG 9.11.3-1ubuntu1-Ubuntu <<>> ftp.srv.world.
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15143
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;ftp.srv.world.                 IN      A

;; ANSWER SECTION:
ftp.srv.world.          86400   IN      CNAME   dlp.srv.world.
dlp.srv.world.          7199    IN      A       10.0.0.30

;; Query time: 0 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: Tue May 08 17:25:38 JST 2018
;; MSG SIZE  rcvd: 76
```

BIND : Configure Slave DNS Server

Tuesday, March 17, 2020 1:24 PM

Configure BIND as a Slave DNS Server.
The following example shows an environment that master DNS is [172.16.0.82], Slave DNS is [slave.example.host].
[1] Configure DNS master server.

```
root@dlp:~# vi /etc/bind/named.conf.options
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 {
        //      0.0.0.0;
        // };

         allow-query { localhost; 10.0.0.0/24; };
         # add a range you allow to transfer zones
         allow-transfer { localhost; 10.0.0.0/24; 172.16.0.80/29; };
         allow-recursion { localhost; 10.0.0.0/24; };

        //========================================================================
        // 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
        //========================================================================
        dnssec-validation auto;

        auth-nxdomain no;    # conform to RFC1035
        listen-on-v6 { none; };
};

root@dlp:~# rndc reload
server reload successful
```

[2] Configure DNS slave server.

```
root@slave:~# vi /etc/bind/named.conf.external-zones
        # add settings like follows
        zone "srv.world" {
                type slave;
                masters { 172.16.0.82; };
                file "/etc/bind/slaves/srv.world.wan";
        };

root@slave:~# mkdir /etc/bind/slaves
root@slave:~# chown bind. /etc/bind/slaves
```

```
root@slave:~# rndc reload
server reload successful
root@slave:~# ls /etc/bind/slaves
srv.world.wan     # zone file in master DNS has been just transfered
```

Dovecot

Tuesday, March 17, 2020 1:24 PM

Dovecot Server

Tuesday, March 17, 2020 1:49 PM

Dovecot is a Mail Delivery Agent, written with security primarily in mind. It supports the major mailbox formats: mbox or Maildir. This section explain how to set it up as an imap or pop3 server.

1 - Installation
To install a basic Dovecot server with common pop3 and imap functions, run the following command in the command prompt:

```
sudo apt install dovecot-imapd dovecot-pop3d
```

There are various other Dovecot modules like dovecot-sieve (mail filtering), dovecot-solr (full text search), ...

2 - Configuration
To configure Dovecot, you can edit the file **/etc/dovecot/dovecot.conf** and its included configfiles in **/etc/dovecot/conf.d/**. By default all installed protocols will be enabled via an include directive in **/etc/dovecot/dovecot.conf**.

```
!include_try /usr/share/dovecot/protocols.d/*.protocol
```

IMAPS and POP3S are more secure because they use SSL encryption to connect. A basic self-signed ssl certificate is automatically set up by package ssl-cert and used by dovecot in **/etc/dovecot/conf.d/10-ssl.conf**.

By default mbox format is configured, if required you can also use maildir. More about that can be found in the comments at **/etc/dovecot/conf.d//10-mail.conf** Further benefits and details are discussed on the Dovecot web site.

You should configure your Mail Transport Agent (MTA) to transfer the incoming mail to the selected type of mailbox if it is different from the one you have configured.

Once you have configured dovecot, restart the Dovecot daemon in order to test your setup:

```
sudo systemctl restart dovecot.service
```

If you have enabled imap, or pop3, you can also try to log in with the commands telnet localhost pop3 or telnet localhost imap2. If you see something like the following, the installation has been successful:

```
Sample@rainbow:~$ telnet localhost pop3
Trying 127.0.0.1...
Connected to localhost.localdomain.
Escape character is '^]'.
+OK Dovecot ready.
```

3 - Dovecot SSL Configuration
Dovecot is now automatically configured to use SSL. It uses the package ssl-cert which provides a self signed certificate. You can edit the file /etc/dovecot/conf.d/10-ssl.conf and amend following lines if you want to set up a custom certificate (See Certificates for more details.):

```
ssl_cert = </etc/dovecot/private/dovecot.pem
```

ssl_key = </etc/dovecot/private/dovecot.key
You can get the SSL certificate from a Certificate Issuing Authority or you can create self signed SSL certificate. Please refer to Certificates for details about how to create self signed SSL certificate. Once you create the certificate, you will have a key file and a certificate file that you want to make known in the config shown above.

4 - Firewall Configuration for an Email Server
To access your mail server from another computer, you must configure your firewall to allow connections to the server on the necessary ports.

IMAP - 143

IMAPS - 993

POP3 - 110

POP3S - 995

How To Install And Configure Dovecot On Ubuntu

Tuesday, March 17, 2020 1:25 PM

How To Install And Configure Dovecot On Ubuntu

Step 1: Install Dovecot on Ubuntu
Dovecot packages are available via Ubuntu default repositories… Using a simple command, you'll be able to install it in no time.

To get Dovecot installed, run the commands below:

```
sudo apt update
sudo apt install dovecot-core dovecot-pop3d dovecot-imapd
```

When prompted whether to except the packages that will be downloaded and installed, type y for Yes.

Once the installation is complete, you can stop, start and enable the service using the commands below:

```
sudo systemctl stop dovecot
sudo systemctl start dovecot
sudo systemctl enable dovecot
```

Then enable command enables the service to automatically start up everytime the system boots up..

Step 2: Configure Dovecot
Now that Dovecot is installed, move below to learn how to configure it..

Just like many other mail servers, Dovecot comes with lots of different configuration options. Some are basic while others are more advanced..

Dovecot main configuration file is located at /etc/dovecot/dovecot.conf.

Run the commands below to open its default configuration file:

```
sudo nano /etc/dovecot/dovecot.conf
```

Then make your changes and save the file..

```
#uncomment line 30
listen = *, ::
```

There are three Dovecot configuration options that are mostly configured in a live environment.. listen, protocols, and mail_location.

To specify the IP addresses that Dovecot can listen on, you use the listen configuration option.. For all IPv4 address, use the wildcard value (*). IPv6 is represented as (::)

Example:

```
listen = *, ::
```

The protocol option specifies which protocol Dovecot communicate over.. Dovecot support IMAP, POP3 and few others..

Example:

```
protocols = imap, pop3, pop3s, lmtp
```

f you like, you can enable all the above protocols, or you can choose to enable just one or any number of them.

The mail_location option defines where mails are pick up from…

The mail location can be configured at:

```
sudo nano /etc/dovecot/conf.d/10-mail.conf
```

Edit and save your changes

```
# At line 30: change to Maildir
mail_location = maildir:~/Maildir
```

Besides the configuration options above, Dovecot also comes with a self-signed certificate settings that come configured on the system.

Dovecot self-signed SSL certificate file is located at /etc/dovecot/dovecot.conf.

Many of other Dovecot configurations are located in the /etc/dovecot directory… most protocols are setup in thy conf.d folder of the directory.

When connecting to the service for the first time, you will receive a warning message because they are self-signed and not CA certificates..

Install Dovecot & configure

Tuesday, March 17, 2020 1:26 PM

Install Dovecot to configure POP/IMAP server. POP uses 110/TCP, IMAP uses 143/TCP.

[1] This example shows to configure to provide SASL function to Postfix.

```
root@mail:~# apt -y install dovecot-core dovecot-pop3d dovecot-imapd

root@mail:~# vi /etc/dovecot/dovecot.conf

# line 30: uncomment
listen = *, ::

root@mail:~# vi /etc/dovecot/conf.d/10-auth.conf

# line 10: uncomment and change ( allow plain text auth )
disable_plaintext_auth = no

# line 100: add
auth_mechanisms = plain login
root@mail:~# vi /etc/dovecot/conf.d/10-mail.conf

# line 30: change to Maildir
mail_location = maildir:~/Maildir

root@mail:~# vi /etc/dovecot/conf.d/10-master.conf

# line 96-98: uncomment and add
# Postfix smtp-auth
unix_listener /var/spool/postfix/private/auth {
   mode = 0666
   user = postfix
   group = postfix
}

root@mail:~# systemctl restart dovecot
```

Postfix E-Mail Server w/ Dovecot

Tuesday, March 17, 2020 1:47 PM

Postfix Introduction

This tutorial will tell you how to setup a basic mail server and teach you a bit about the Postfix MTA (Mail Transfer Agent) in the process.

Postfix is extremely flexible. Its architecture is based on a loose composition of services that receive emails and pass them on to other services (with services like “smtp” on the receiving outer edge, and “local” and “virtual” on the delivering outer edge, if you’re looking at receiving mail). Postfix itself implements the core requirements to receive, route, and deliver mail, and relies on third-party extensions to do the rest.

Postfix has several hundred configuration parameters. If you want to administer a mail server that reliably delivers business requirements to a sizable organization, you should make yourself intimate with all of them (man 5 postconf). This tutorial will *not* be enough, on its own, to make you a competent professional email admin. However, if you want to become familiar with postfix or set up a mail server for yourself and a few friends, this tutorial, and the ones to follow, will be your friend.

Dovecot Introduction

I’m not going to spend a lot of introductory words on dovecot. Dovecot is also huge (here is the wiki for dovecot 2), but we only want a very small set of features from dovecot.

This article explains almost every single setting to be set in detail. You can go ahead and skim over the explanations if you want - *at your own risk*.

This tutorial assumes (and was built using) the following setup:

- Ubuntu 18.04 | 16.04
- Postfix 2.9.6-2
- dovecot 2.2.33.2

While any Debian-based OS should be fine, Postfix is in use in a wide array of versions, including Postfix 1.x, Postfix 2.9, and Postfix 2.10, which have some mutually incompatible settings and features - and using Postfix 2.9, this tutorial is not on the bleeding edge.

This tutorial also assumes a few things about you:

- That you are comfortable on a GNU/Linux commandline and with the general layout and working principles of a GNU/Linux system like Debian.
- That your local system is a GNU/Linux or reasonably compatible (MinGW, Cygwin, Mac OS X, *BSD)
- That you know how to get a root shell on your droplet
- That you know how to use a text editor (e.g. vim, nano, emacs, or the standard editor, ed) on Linux

System Setup

The following ingredients are necessary to make your droplet ready to be a mail server:

- A domain, let’s assume it is “mydomain.com”
- A hostname for your mail server, let’s assume “mail.mydomain.com”
- An SSL certificate that is valid for “mail.mydomain.com”

Setting up SSL certificate

For SSL, you need a certificate and a private key. In this tutorial, we’re going to assume that the certificate is saved in /etc/ssl/certs/mailcert.pem and the key is saved in /etc/ssl/private/mail.key. Make sure the key is only readable by the root user!

How to set up SSL certificates for your website and e-mail depends on your website structure and the CA you use (self-signed, organisational (sub)-ca, or commercial ca for example). Creating a self-signed test certificate is as easy as executing

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/mail.key -out /etc/ssl/certs/mailcert.pem
```

and leaving the default values in by just hitting enter on all questions asked. Don’t use this certificate in production!

Most CAs will require you to submit a certificate signing request. (CSR) You can generate one like this:

```
sudo openssl req -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/mail.key -out mailcert.csr
```

Fill in the information queried properly, like in this transcript: (Check with the CA you intend to use on what information needs to be in the CSR)

```
Generating a 2048 bit RSA private key
.............................+++
...............+++
writing new private key to 'mail.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]: US
State or Province Name (full name) [Some-State]: Virginia
Locality Name (eg, city) []: Langley
Organization Name (eg, company) [Internet Widgits Pty Ltd]: Network Services Association
Organizational Unit Name (eg, section) []: Infrastructure Services
Common Name (e.g. server FQDN or YOUR name) []: mail.mydomain.com
Email Address []: postmaster@mydomain.com
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

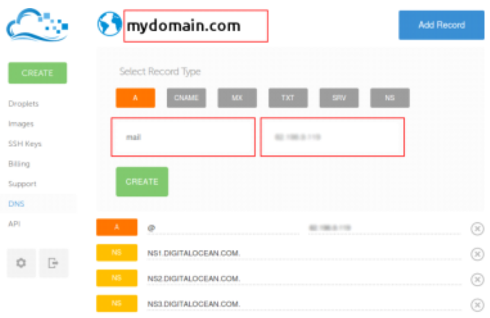
(Note that this way you cannot create a certificate valid for more than one domain using the subjectAltName field without some additional work - again, check the CA’s documentation!)

Setting up DNS

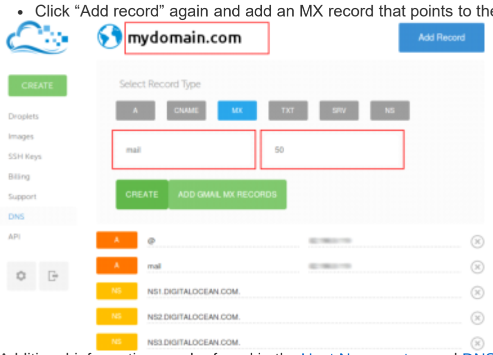
You have to set up your DNS with an A record that points to your mail server IP and an MX record that points to the mail servers hostname.

Here is how to do it if you’re using DigitalOcean’s DNS:

- Go to the “DNS” area in your DigitalOcean panel
- Create a new domain or select one you’ve created before
- Click the “Add record” button in the top right
- Add an A record:



- Click “Add record” again and add an MX record that points to the A record:



Additional information can be found in the Host Name setup and DNS tips and tricks articles.

Verify DNS

DNS will take a few hours to propagate all over the internet, but it should be set on your DNS server after a few minutes. You can check with **dig** & **host**:

```
[root@yourbase] ~# dig MX mydomain.com +short @ns1.digitalocean.com
50 mail.mydomain.com.
[root@yourbase] ~# host mail.mydomain.com ns1.digitalocean.com
Using domain server:
Name: ns1.digitalocean.com
Address: 198.199.120.125#53
Aliases:
mail.mydomain.com has address 82.196.9.119
```

Postfix

We will now set up Postfix to receive and deliver mail for local users.

Packages

The default MTA on Debian is exim. Off with it! We’ll also stop postfix after it has been installed, because we don’t want it to be running yet.

apt install postfix
Pop up window "**No Configuration**"

A small insert: Postfix manages its daemons by itself and doesn’t need the service (init.d) system. postfix start, postfix stop, and postfix reload are equivalent to service postfix start, service postfix stop and service postfix reload.

Postfix Configuration

Postfix has two main config files: **main.cf**, which specifies what you would think of as config options, and **master.cf**, which specifies the services postfix should run.

First, configure the **master.cf** file (in /etc/postfix/). Add an extra “smtpd” instance called “submission” that will take mail from trusted clients for delivery to the world at large, which we don’t allow for anyone else.

To do that, open master.cf (take a look at man 5 master if you want to understand what’s going on) and uncomment the submission config and add options to enable SASL:

```
submission inet n       -       -       -       -       smtpd
  -o syslog_name=postfix/submission
  -o smtpd_tls_wrappermode=no
  -o smtpd_tls_security_level=encrypt
  -o smtpd_sasl_auth_enable=yes
  -o smtpd_recipient_restrictions=permit_mynetworks,permit_sasl_authenticated,reject
  -o milter_macro_daemon_name=ORIGINATING
  -o smtpd_sasl_type=dovecot
  -o smtpd_sasl_path=private/auth
```

This warrants a bit of explanation. The -o ... options override the settings that are taken from defaults or define in the config, which we’ll set later.

In a nutshell what happens here is that this enables the “submission” daemon with TLS to secure the outer connection, and dovecot-mediated SASL to check the username and password of connecting clients. (We will set that up in dovecot later).

The important detail is one that can’t be seen: The smtpd_recipient_restrictions is missing reject_unauth_destination, which is present as a default and restricts relaying.

Then we move on to **main.cf**. We’ll start with a clean slate here - run cp /etc/postfix/main.cf /etc/postfix/main.cf.orig if you want to save the default config file (it’s also in /usr/share/postfix/main.cf.dist though), then open it and clear it out!

Let’s first set the network information: (information about the domains postfix is handling mail for, and a bit of extra info)

```
myhostname = mail.domain.com
myorigin = /etc/mailname
mydestination = mail.domain.com, domain.com, localhost, localhost.localdomain
relayhost =
mynetworks = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128
mailbox_size_limit = 0
recipient_delimiter = +
inet_interfaces = all
```

We set the hostname and the default origin, which is sourced from **/etc/mailname** by debian convention. You can set it explicitly if you don't have **/etc/mailname**. The default origin is used to construct the 'From' address for local users. **mydestination** sets the domains that postfix accepts emails for as final destination, and we set "relayhost" empty to disable relaying mail (relaying means accepting mail and then forwarding to a mail server that is not the final destination for the mail and we have no need for that; that is useful e.g. in a corporate intranet where a central mail server should check mail before it leaves the network.)

Additional note: This has nothing to do with the term "open relay", which is a mail server that accepts email from anybody without authentication and sends it to MTAs for domains that aren't in their own network - for this the other `relay_` settings are used, which we leave on default and disabled)

Let's now set the local alias maps. We don't have to set this setting since we're just keeping the default setting, but it's good to make it explicit in case we later want to add another method of defining alias maps. (like a real DBMS)

```
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
```
Then we set up SSL:

```
smtpd_tls_cert_file=/etc/ssl/certs/mailcert.pem
smtpd_tls_key_file=/etc/ssl/private/mail.key
smtpd_use_tls=yes
smtpd_tls_session_cache_database = btree:${data_directory}/smtpd_scache
smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
smtpd_tls_security_level=may
smtpd_tls_protocols = !SSLv2, !SSLv3
```
We set the cert file and the key for it, enable tls, and set the cache files. Then we make TLS optional, because we're not allowed to make TLS required on a public smtp server per RFC2487. We also disallow SSLv2 and SSLv3, so that only TLSv1.0 and higher is allowed (read a SSL tutorial if you want to know why - in a nutshell, SSLv2 and SSLv3 are obsolete).

Another setting that is fine as default but should be specified explicitly in case you want to add to it later is the **local*recipient*maps**:

```
local_recipient_maps = proxy:unix:passwd.byname $alias_maps
```
This setting tells Postfix to check a lookup table and reject email to users that cannot be found in the table. This is important because the alternative behaviour, if local_recipient_maps is unset, is to accept mail first and then bounce it later. This causes "backscatter": If postfix cannot determine all valid users immediately (in the smtpd service), like when local_recipients_maps is unset, it will accept mail and then send a non-delivery notice later (when it finds out the mail is undeliverable after it has been handed off by smptd). These non-delivery notices usually hit innocent people whose addresses have been spoofed in spam and scam mails and contribute to the spam problem.

Sane Alias Config

There are a few mail accounts you should set up in your alias config that are important. For example the SMTP RFC mandates that any publicly accessible mailserver that accepts any mail at all must also accept mail to the 'postmaster' account, and some people might expect "hostmaster", "abuse", "webmaster", and other mailboxes to be present. You can either redirect those mail addresses to root, or to a specific user. Here is a sane default for /etc/aliases, presuming that you check email for **root**:

```
mailer-daemon: postmaster
postmaster: root
nobody: root
hostmaster: root
usenet: root
news: root
webmaster: root
www: root
ftp: root
abuse: root
```
If you want to redirect all of that to a specific local user, say, "yourname" just add

```
root: yourname
```
Postfix will resolve the entire chain of aliases for you and forward all those mail addresses to "yourname". (This is done by the local daemon using the aliases specification.)

As "aliases" says, after updating the **/etc/aliases** file, you have to run

```
newaliases
```
to compile the file into the database Postfix uses for fast lookup.

Dovecot

This one will be less wall-of-text-y! Take a deep breath, we're almost done.

Packages

```
apt install dovecot-core dovecot-imapd
```
Should do it. If you want all the default packages, run

```
apt install dovecot-common
```
Then go into /etc/dovecot/dovecot.conf and clear out the file again. (that's important this time - the default config includes a bunch of subordinate config files in /etc/dovecot/conf.d that we don't want).

Now enter the following config:

```
disable_plaintext_auth = no
mail_privileged_group = mail
mail_location = mbox:~/mail:INBOX=/var/mail/%u
userdb {
  driver = passwd
}
passdb {
  args = %s
  driver = pam
}
protocols = " imap"
```
This enables plaintext auth (The "plaintext" authentication will be tunneled through TLS), tells dovecot to use the mail system group for accessing the local mailboxes (plus the location of the mailboxes), use the unix authentication system to authenticate users, and enable imap only.

If you want, you can have dovecot automatically add a Trash and Sent folder to mailboxes:

```
protocol imap {
  mail_plugins = " autocreate"
}
plugin {
  autocreate = Trash
  autocreate2 = Sent
  autosubscribe = Trash
  autosubscribe2 = Sent
}
```
Next, we need to open a socket that postfix can use to piggy-back on dovecot's authentication:

```
service auth {
  unix_listener /var/spool/postfix/private/auth {
    group = postfix
    mode = 0660
    user = postfix
  }
}
```

And finally the ssl config:

```
ssl=required
ssl_cert = </etc/ssl/certs/mailcert.pem
ssl_key = </etc/ssl/private/mail.key
```

Note the angle brackets! They tell dovecot to read from a file.

The End

Save and close all the config files, and execute

```
newaliases
postfix start
service dovecot restart
```

And you should be good to go. Test your config with a mail client, e.g. Mozilla Thunderbird. You should be able to send and receive mails from everywhere and to everywhere!

Continued

If you want to add virtual mailboxes (mail boxes that are not tied to a local user account, but can instead be configured using a local database) continue with Part 2.***

You can now test that sending e-mail both ways works, from a terminal on the droplet:

```
~# mail someotheremail@gmail.com
Subject: test email from postfix
this is a test
.
EOT
```

The mail from "root@yourdomain.com" should shortly arrive at "someotheremail@gmail.com" (fill in an email adress you control, obviously). If you reply to it and call **mail** again, you should see this: (it might take a minute for the mail to arrive).

```
~# mail
Heirloom mailx version 12.5 6/20/10.  Type ? for help.
"/var/mail/root": 1 message
>N  1 Your Name         Wed Nov 13 23:45   41/1966  Re: test email from postf
```

And if you hit the Enter key, it will show the message. (then type **q** and hit **Enter** to leave the mail client)

The same thing will work with a local e-mail client. Set up a new system user:

```
~# adduser joe
Adding user `joe' ...
Adding new group `joe' (1001) ...
Adding new user `joe' (1001) with group `joe' ...
Creating home directory `/home/joe' ...
Copying files from `/etc/skel' ...
Enter new UNIX password: Enter password here
Retype new UNIX password: Enter password here
passwd: password updated successfully
Changing the user information for joe
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
```

The password you entered here is the password to use for e-mail. Joe can now use the address joe@yourdomain.com with a local mail client like Thunderbird. In Thunderbird, just add a new Account (File -> New -> Existing Mail Account) and enter joe@yourdomain.com and the password in the dialog.

If your mail client doesn't auto-detect the necessary settings: The username for the IMAP connection is joe, the port is 143, and the authentication method is unencrypted password via STARTTLS. For SMTP it's the same, but port 587.

If anything isn't working, check for error messages in the system log with tail -n 50 /var/log/syslog and in the mail log with tail -n 50 /var/log/mail.log.

Mail Server using Postfix Dovecot on Ubuntu 18.04

Tuesday, March 17, 2020 3:15 PM

1. lsb_release -a ; getconf LONG_BIT ; hostname -I

2. hostnamectl set-hostname mail.example.com
3. gedit /etc/hosts &>/dev/null
   a. "change out 127.0.1.1 www.example.com 192.168.213.165 mail.example.com
4. reboot

6. apt -y install postfix sas12-bin dovecot-core dovecot-pop3d dovecot-imapd mailutils
7. cp /usr/share/postfix/main.cf.dist /etc/postfix/main.cf

8. gedit /etc/postfix/main.cf &>/dev/null
   edit line
   78 mail_owner = postfix
   94 myhostname = mail.example.com
   102 mydomain = example.com
   123 myorigin = $mydomain
   137 inet_interfaces = all
   185 mydestination = $myhostname, localhost.$mydomain, localhost, $mydomain
   228 local_recipients_maps = unix:passwd.byname $alias_maps
   270 mynetworks_style = subnet
   287 mynetworks = 127.0.0.0/8. 192.168.213.165/24
   407 alias_maps = hash:/etc/aliases
   418 alias_database = hash:/etc/aliases
   440 home_mailbox = Maildir /
   576 #smtpd_banner = $myhostname ESMTP $mail_name (Ubuntu)
       smtpd_banner = $myhostname ESMTP
   650 sendmail_path = /usr/sbin/postfix
   655 newaliases_path = /usr/bin/newaliases
   660 mailq_path = /usr/bin/mailq
   666 setgid_group = postdrop
   670 #html_directory =
   674 #manpage_directory =
   679 #sample_directory
   683 #readme_directory =

   insert at end of document
   message_size_limit = 10485760
   mailbox_size_limit = 1073741824
   #SMTP-Auth setting
   smtpd_sasl_type = dovecot
   smtpd_sasl_path = private/auth
   smtpd_sasl_auth_enable = yes
   smtpd_sasl_security_options = noanonymous
   smtpd_sasl_local_domain = $myhostname
   smtpd_recipient_restrictions =
   permit_mynetworks,permit_auth_destination,permit_sasl_authenticated,reject

newaliases

```
gedit /etc/dovecot/dovecot.conf &>/dev/null
30 listen = *. ::

gedit /etc/dovecot/conf.d/10-auth.conf $>/dev/null
10  disable_plaintext_auth = no
100  auth_mechanisms = plain login

gedit /etc/dovecot/conf.d/10-mail.conf $>/dev/null
30   mail_location =maildir:~/Maildir

gedit /etc dovecot/conf.d/10-master.conf $>/dev/null
96-98 # Postfix smtp-auth
unix_listener /var/spool/postfix/private/auth  {
   mode = 0666
   user = postfix
   group = postfix
}

systemctl restart dovecot postfix ; systemctl status dovecot postfix
echo 'export MAIL=$HOME/Maildir/' >> /etc/profile.d/mail.sh
less /etc/passwd


reboot
```

SFTP/FTP

Tuesday, March 17, 2020 1:27 PM

FTP server on Ubuntu

Tuesday, March 17, 2020 1:27 PM

Introduction

If you are looking to install an FTP server on Ubuntu, you can't beat the simplicity of vsftpd .

FTP stands for File Transfer Protocol. It is similar to HTTP (HyperText Transfer Protocol), in that it specifies a language for transferring data over a network.
FTP is unencrypted by default, so by itself, it is not a good choice for secure transmission of data.

This guide will help you install and configure an FTP server (vsftpd) on Ubuntu 18.04.

Prerequisites
Access to a user account with sudo privileges
Access to a terminal window/command line (Ctrl-Alt-T)
The apt package manager, included by default

Step 1: Update System Packages
Start by updating your repositories – enter the following in a terminal window:

```
sudo apt-get update
```

The system proceeds to update the repositories.

System Updating Repositories.

Step 2: Backup Configuration Files
Before making any changes, make sure to back up your configuration files.

1. Create a backup copy of the default configuration file by entering the following:

```
sudo cp /etc/vsftpd.conf  /etc/vsftpd.conf_default
```

This command creates a copy of the default configuration file.

2. Create a new vsftpd configuration file /etc/vsftpd.conf using your preferred text editor:

```
$ sudo nano /etc/vsftpd.conf
```

Step 3: Install vsftpd Server on Ubuntu
A common open-source FTP utility used in Ubuntu is vsftpd. It is recommended for its ease of use.

1. To install vsftpd, enter the command:

```
sudo apt install vsftpd
```

This is an example of the output in Ubuntu.

vsftpd installation process.
2. To launch the service and enable it at startup:

```
sudo systemctl start vsftpd

sudo systemctl enable vsftpd
```

Note: Instruction for setting up and configuring FTP with Vsftpd is also available for CentOS 7.

Step 4: Create FTP User
Create a new FTP user with the following commands:

```
sudo useradd –m testuser

sudo password testuser
```

The system should ask you to create a password for the new testuser account. Create a sample file in the new user's home account:

```
sudo mkdir /home/testuser
```

Step 5: Configure Firewall to Allow FTP Traffic
If you are using UFW that comes standard with Ubuntu, it will block FTP traffic by default. Enter the following commands to open Ports 20 and 21 for FTP traffic:

```
sudo ufw allow 20/tcp

sudo ufw allow 21/tcp
```

Note: If you are using a different firewall, refer to the instructions to allow access on Port 20 and Port 21. These are the listening ports for the FTP service.

Step 6: Connect to Ubuntu FTP Server
Connect to the FTP server with the following command:

```
sudo ftp ubuntu-ftp
```

Replace ubuntu-ftp with the name of your system (taken from the command line).

Log in using the testuser account and password you just set. You should now be successfully logged in to your FTP server.

Configuring and Securing Ubuntu vsftpd Server
Change Default Directory
By default, the FTP server uses the /srv/ftp directory as the default directory. You can change this by creating a new directory and changing the FTP user home directory.

To change the FTP home directory, enter the following:

```
sudo mkdir /srv/ftp/new_location

sudo usermod –d /srv/ftp/new_location ftp
```

Restart the vsftpd service to apply the changes:

```
sudo systemctl restart vsftpd.service
```
Now, you can put any files you want to share via FTP into the /srv/ftp folder (if you left it as the default), or the /srv/ftp/new_location/ directory (if you changed it).

Authenticate FTP Users
If you want to let authenticated users upload files, edit the vsftpd.conf file by entering the following:

```
sudo nano /etc/vsftpd.conf
```
Find the entry labeled write_enable=NO, and change the value to "YES."

Write_Enable option in vsftpd.config file.
Save the file, exit, then restart the FTP service with the following:

```
sudo systemctl restart vsftpd.service
```
This allows the user to make changes inside their home directory.

Securing FTP
Limit User Access
Numerous exploits take advantage of unsecured FTP servers. In response, there are several configuration options in vsftpd.conf that can help secure your FTP server.

One method is to limit users to their home directory. Open vsftpd.conf in an editor and uncomment the following command:

```
chroot_local_user=YES
```
This is an example of the file in nano:

Command to limit users to their home directory.
Create a User List File
To create a list file, edit /etc/vsftpd.chroot_list, and add one user per line.

Instruct your FTP server to limit this list of users to their own home directories by editing vsftpd.conf:

```
chroot_local_user=YES

chroot_list_file=/etc/vsftpd.chroot_list
```
The image ilustrates the edits that were made:

Chroot list to limit user access.
Restart the vsftpd service:

```
sudo systemctl restart vsftpd.service
```
By default, the list of blocked users from FTP access is stored in /etc/ftpusers. To add blocked users, edit this file and add one user per line.

Encrypt Traffic With FTPS
Another method to secure your FTP server is to encrypt the traffic. This is done by using FTPS – File Transfer Protocol over SSL (Secure Socket Layer).

For this to work, users need to be set up with a shell account on the FTP server. This will add a layer of secure encryption to your FTP traffic. To set up FTPS, open your vsftpd.conf file in an editor, and add the following lines:

```
ssl_enable=YES

rsa_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem

rsa_private_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
```
The file shoullook like this:

Enable SSl for vsftpd on Ubuntu.
Restart the service to apply the changes:

```
sudo systemctl restart vsftpd.service
```

SFTP Server on Ubuntu

Tuesday, March 17, 2020 1:28 PM

How to Setup SFTP Server on Ubuntu 18.04
TABLE OF CONTENTS
## Table of Contents

## Introduction

FTP stands for “File Transfer Protocol” is a popular method of transferring files between two remote systems. SFTP stands for SSH File Transfer Protocol, or Secure File Transfer Protocol is a separate protocol packaged with SSH that works similarly over a secure connection.

## Prerequisites

SFTP Configuration assumes that you have to configure your FTP server by following the FTP configuration guide. Privileged ac cess to the system as root or via sudo command is required.

## Configure FTP Server

This guide describes FTP over secure SSH protocol. Before starting this guide make sure that you have already configured your FTP server using below link.

## Configure SSH Daemon

If you have not configured till now, you have to install SSH server:
“`
$ sudo apt install ssh
“`
If you want to configure FTP over OpenSSH server, you have to edit the existing SSHD configuration file as below.

“`
$ sudo nano /etc/ssh/sshd_config
“`
Add this below file at the end of the file as below.
“`
Match group sftp
ChrootDirectory /home
X11Forwarding no
AllowTcpForwarding no
ForceCommand internal-sftp
“`
These lines say that users related to sftp group will be able to access their home directories, even though they will be deni ed SSH shell access.

![configuressgdaemon](https://grid.media/assets/images/configure-ssh-daemon.png

To apply the new changes you have to restart the SSH server.
“`
$ sudo service ssh restart
“`
## Create SFTP User Account
Now you have to create a new user account which is specific to SFTP service. Now you have to create a new group called sftp:
“`
$ sudo addgroup sftp
“`

![addgroupsftp](https://grid.media/assets/images/addgroup-sftp.png

Now you have to create a new user sftpuser assign him to the sftp group by using the below command.
“`
$ sudo useradd -m sftpuser -g sftp
“`
You have to set a new password for sftpuser user:
“`
$ sudo passwd sftpuser

“`

![passwordsftpuser](https://grid.media/assets/images/passwd-sftpuser.png

At last change access permissions to the user’s home to deny access to it from any others on the same system. To do it use th e below command.
“`
$ sudo chmod 700 /home/sftpuser/

“`
## User Login via SFTP

Now the new user called sftpuser can log in to the new sftp server via sftp:// protocol. Now SFTP server can be resolved via eg. hostname ubuntu-sftp use sftp command to create new SFTP connection.
“`
$ sftp sftpuser@testsftp

“`

![sftpusertestsftp](https://grid.media/assets/images/sftpuser-testsftp.png

Now navigate to your home directory and confirm write access by creating a new directory.
“`
sftp> cd sftpuser
sftp> mkdir sftp-test
sftp> ls
“`

![sftptest](https://grid.media/assets/images/ls.png

## Conclusion
Now you are connected to SFTP

Security Tools

Tuesday, March 17, 2020 4:44 PM

Lynis

Tuesday, March 17, 2020 4:44 PM

```
sudo apt install lynis -y
cd
wget http://sable.madmimi.com/c/6938?id=44150.2674.1.a12c46882ca668ab69e63acbe670c747 -O  lynis-community-plugins.tar.gz
sudo tar -zxvf lynis-community-plugins.tar.gz --strip-components=1 -C /usr/share/lynis/plugins
"if it fails then do following step"
mkdir -p /usr/share/lynis/plugins
sudo chown root:root /usr/share/lynis/plugins/plugin_*
sudo chmod 600 /usr/share/lynis/plugins/plugin_*
sudo grep plugin= /etc/lynis/default.prf

plugin=compliance
plugin=configuration
plugin=control-panels
plugin=crypto
plugin=dns
plugin=docker
plugin=file-integrity
plugin=file-systems
plugin=firewalls
plugin=forensics
plugin=intrusion-detection
plugin=intrusion-prevention
plugin=kernel
plugin=malware
plugin=memory
plugin=nginx
plugin=pam
plugin=processes
plugin=security-modules
plugin=software
plugin=system-integrity
plugin=systemd
plugin=users
```

Debian, Ubuntu
Are you are a customer with an active subscription to Lynis Enterprise? Have a look in the customers section.

Import key
For these commands root access may be needed. Use sudo or run as root user.

Download the key from a central keyserver:

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys C80E383C3DE9F082E01391A0366C67DE91CA5D5F
```

Or manually import it:

```
sudo wget -O - https://packages.cisofy.com/keys/cisofy-software-public.key | sudo apt-key add -
```

Add software repository

The software repository uses preferably HTTPS for secure transport. Install the 'https' method for APT, if it was not available yet.

```
sudo apt install apt-transport-https
```

Using your software in English? Then configure APT to skip downloading translations. This saves bandwidth and prevents additional load on the repository servers.

```
echo 'Acquire::Languages "none";' | sudo tee /etc/apt/apt.conf.d/99disable-translations
```

Next step is adding the repository:

```
echo "deb https://packages.cisofy.com/community/lynis/deb/ stable main" | sudo tee /etc/apt/sources.list.d/cisofy-lynis.list
```

Install Lynis

Refresh the local package database with the new repository data and install Lynis:

```
apt update
```

RKhunter

Tuesday, March 17, 2020 7:22 PM

RKhunter

Don't be afraid of the RKhunter warnings in the terminal.

Using RKhunter is always a work in progress.

To install RKhunter:
sudo apt-get install rkhunter

Before running RKhunter you will need to fill the file properties database by running the following command: **rkhunter --propupd** Do no forget to set rkhunter in sysconfig to run the --propupd every time new software is installed or else you will get "false positives" after every software and system update.

sudo rkhunter --propupd

To run **rkhunter --propupd**, automatic after software updates, add the line **APT_AUTOGEN="yes"** to */etc/default/rkhunter* (this gets read by */etc/apt/apt.conf.d/90rkhunter*).

Wait till it completes gathering the new values, then exit. This should eliminate all the warnings except the hidden files related to the /dev folder. They show up occassionally and disappear with a next reboot of your system.

Additionally, the **--versioncheck** option of rkhunter itself will indicate if a new version is available.

sudo rkhunter --versioncheck

The first run of 'rkhunter' after installation may give some warning messages. They are is some way normal. Even on clean installed system, with no additional software installed, these warnings occur. You could take a at the FAQ of RKhunter. I got these warnings on Xubuntu beta, clean install:

sudo rkhunter --checkall

warnings:

/usr/bin/mail
/usr/bin/bsd/mail-x
checking /dev for susp. files
checking hidden files and direct
/usr/bin/lwp-request

It is possible for a package manager database to become maliciously corrupted. RKhunter can only report on changes, but not on what has caused the change, it is reactive.

Help Rootkit Hunter users on the rkhunter-users mailing list. the rkhunter mailinglist It is also a source of information on "false positives".

"Intruder Detection Checklist". This list is available via the intruder detection list

What to do with "common" warnings as:

Warning: Hidden directory found: /dev/.static
Warning: Hidden directory found: /dev/.udev
Warning: Hidden directory found: /dev/.initramfs

To avoid these warnings, you can reconfigure rkhunter to ignore these files via whitelisting these warnings. Edit the rkhunter.conf file: **gedit /etc/rkhunter.conf** and remove the # in front of these lines:

#ALLOWHIDDENDIR=/dev/.udev
#ALLOWHIDDENDIR=/dev/.static
#ALLOWHIDDENDIR=/dev/.initramfs

ALLOWHIDDENDIR=/dev/.udev
ALLOWHIDDENDIR=/dev/.static
ALLOWHIDDENDIR=/dev/.initramfs

usbguard

Tuesday, March 17, 2020 7:29 PM

Installation -

```
apt install -y usbguard

 systemctl enable usbguard
 systemctl start usbguard
 systemctl status usbguard
 systemctl stop usbguard
```

Configuration

usbguard-daemon.conf – USBGuard daemon configuration file

The `usbguard-daemon.conf` file is loaded by the USBGuard daemon after it parses its command-line options and is used to configure runtime parameters of the daemon. The default search path is */etc/usbguard/usbguard-daemon.conf*. It may be overridden using the `-c` command-line option, see usbguard-daemon(8) for further details.

Options

- `RuleFile=<path>` The USBGuard daemon will use this file to load the policy rule set from it and to write new rules received via the IPC interface.

- `ImplicitPolicyTarget=<target>` How to treat devices that don't match any rule in the policy.
  - allow - authorize the device
  - block - deauthorize the device
  - reject - logically remove the device node from the system
- `PresentDevicePolicy=<policy>` How to treat devices that are already connected when the daemon starts:
  - allow - authorize every present device
  - block - deauthorize every present device
  - reject - remove every present device
  - keep - just sync the internal state and leave it
  - apply-policy - evaluate the ruleset for every present device
- `PresentControllerPolicy=<policy>` How to treat USB controllers that are already connected when the daemon starts:
  - allow - authorize every present device
  - block - deauthorize every present device
  - reject - remove every present device
  - keep - just sync the internal state and leave it
  - apply-policy - evaluate the ruleset for every present device
- `InsertedDevicePolicy=<policy>` How to treat USB devices that are already connected after the

daemon starts. One of block, reject, apply-policy.

- RestoreControllerDeviceState=<boolean> The USBGuard daemon modifies some attributes of controller devices like the default authorization state of new child device instances. Using this setting, you can control whether the daemon will try to restore the attribute values to the state before modification on shutdown.

- DeviceManagerBackend=<backend> Which device manager backend implementation to use. Backend should be one of uevent (default) or umockdev.

- IPCAllowedUsers=<username> [<username> ...] A space delimited list of usernames that the daemon will accept IPC connections from.

- IPCAllowedGroups=<groupname> [<groupname> ...] A space delimited list of groupnames that the daemon will accept IPC connections from.

- IPCAccessControlFiles=<path> The files at this location will be interpreted by the daemon as IPC access control definition files. See the <<ipc-access-control,IPC ACCESS CONTROL>> section for more details.

- DeviceRulesWithPort=<boolean> Generate device specific rules including the “via-port” attribute.

- AuditBackend=<backend> SBGuard audit events log backend. The backend value should be one of FileAudit or LinuxAudit.

- AuditFilePath=<filepath> USBGuard audit events log file path. Required if AuditBackend is set to FileAudit.

Security Considerations

IPC

The daemon provides the USBGuard public IPC interface. Depending on your distribution defaults, access to this interface is limited to a certain group or a specific user only. Please set either the *IPCAllowedUsers*, *IPCAllowedGroups* or *IPCAccessControlFiles* options to limit access to the IPC interface. *Do not leave the ACL unconfigured as that will expose the IPC interface to all local users and will allow them to manipulate the authorization state of USB devices and modify the USBGuard policy.*

RestoreControllerDeviceState configuration option

If set to true, the USB authorization policy could be bypassed by performing some sort of attack on the daemon (via a local exploit or via a USB device) to make it shutdown and restore to the operating-system default state (known to be permissive).

IPC ACCESS CONTROL

Access to the USBGuard IPC interface can be limited per user or group. Furthermore, by using the IPC Access Control files, it is possible to limit the access down to the level of Sections and Privileges as explained below.

Recommended: *IPCAccessControlFiles*

When you set *IPCAccessControlFiles* option, the daemon will look for IPC access control files in the directory specified by the setting value. Each file in the directory is processed as follows:

1. The basename of the file is interpreted as an username, UID, groupname or GID. If the names starts with `:` (colon), it is assumed that the rest of the name represents a group identifier (groupname or GID in case of a numeric-only string). Otherwise, it is interpreted as an user identifier (username or UID in case of numeric-only string).

2. The contents of the file are parsed as `Section=privilege [privilege ...]` formatted lines which specify the section privileges. If a section is omitted, it is assumed that no privileges are given for that section.

Available sections and privileges:

- `Devices`
  - modify: Change authorization state of devices including permanent changes (i.e. modification of device specific rules in the policy).
  - list: Ability to get a list of recognized devices and their attributes.
  - listen: Listen to device presence and device policy changes.
- `Policy`
  - modify: Append rules to or remove any rules from the policy.
  - list: Ability to view the currently enforced policy.
- `Exceptions`
  - listen: Receive exception messages.
- `Parameters`
  - modify: Set values of run-time parameters.
  - list: Get values of run-time parameters.

The following is a generally usable and reasonably safe example of an access control file. It allows to modify USB device authorization state (`Devices=modify`), list USB devices (`Devices=list`), listen to USB device related events (`Devices=listen`), list USB authorization policy rules (`Policy=list`) and listen to exception events (`Exceptions=listen`):

```
Devices=modify list listen
Policy=list
Exceptions=listen
```

Instead of creating the access control files by yourself, you can use the `usbguard add-user` or `usbguard remove-user` CLI commands. See usbguard(1) for more details.

Legacy: *IPCAllowedUsers* and *IPCAllowedGroups*

Example configuration allowing full IPC access to users *root*, *joe* and members of the group *wheel*:

```
IPCAllowedUsers=root joe
IPCAllowedGroups=wheel
```

Java

Tuesday, March 17, 2020 7:59 PM

Installing Java

```
sudo apt install default-jdk
```

Installing Openjdk Java

```
sudo apt install openjdk-8-jdk
```

Installing Oracle Java

The following steps describe how to install Oracle Java 11 on Ubuntu 18.04:

1. Install the dependencies necessary to add a new repository:
   sudo apt install software-properties-common
2. Enable the Linux Uprising PPA by running the following commands:
   sudo add-apt-repository ppa:linuxuprising/java
3. Once the repository is added, update the packages list and install the oracle-java11-installer package by typing:
   sudo apt updatesudo apt install oracle-java11-installer
   You will be prompted to accept the Oracle license.
4. Verify the installation by running the following command which will print the R version:
   ```
   java -version
   java version "11.0.2" 2019-01-15 LTS
   Java(TM) SE Runtime Environment 18.9 (build 11.0.2+9-LTS)
   Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.2+9-LTS, mixed mode)
   ```

Set the Default Java Version

To check the default Java version you would use the following command:

```
java -version
openjdk version "11.0.2" 2019-01-15
OpenJDK Runtime Environment (build 11.0.2+9-Ubuntu-3ubuntu118.04.3)
OpenJDK 64-Bit Server VM (build 11.0.2+9-Ubuntu-3ubuntu118.04.3, mixed mode, sharing)
```

If you have multiple Java installations to change the default version, use the update-alternatives tool as shown below:

```
sudo update-alternatives --config java

There are 3 choices for the alternative java (providing /usr/bin/java).
Selection    Path                                            Priority   Status
------------------------------------------------------------
* 0          /usr/lib/jvm/java-11-openjdk-amd64/bin/java      1111      auto mode
  1          /usr/lib/jvm/java-11-openjdk-amd64/bin/java      1111      manual mode
  2          /usr/lib/jvm/java-8-openjdk-amd64/jre/bin/java   1081      manual mode
Press <enter> to keep the current choice[*], or type selection number:
```

To change the default Java version just enter the version number (the number in the Selection column) and press Enter.

Set the JAVA_HOME Environment Variable

Some applications written in Java are using the JAVA_HOME environment variable to determine the Java installation location.

To set the JAVA_HOME environment variable, first, you need to find out the Java installation paths using the update-alternatives command

```
sudo update-alternatives --config java
```

In our case, the installation paths are as follows:

- OpenJDK 11 is located at /usr/lib/jvm/java-11-openjdk-amd64/bin/java
- OpenJDK 8 is located at /usr/lib/jvm/java-8-openjdk-amd64/jre/bin/java

Copy the installation path of your preferred installation. Next, open the /etc/environment file:

```
sudo nano /etc/environment
```

Add the following line, at the end of the file:

```
/etc/environment
JAVA_HOME="/usr/lib/jvm/java-11-openjdk-amd64"
```

Copy
Make sure you replace the path with the path to your preferred Java version.
You can either log out and log in or run the following source command to apply the changes to your current session:

```
source /etc/environment
```

To verify that the JAVA_HOME environment variable is correctly set, run the following echo command:

```
echo $JAVA_HOME
```

```
/usr/lib/jvm/java-11-openjdk-amd64
```

/etc/environment is a system-wide configuration file, which is used by all users. If you want to set the JAVA_HOME variable on a per-user basis, add the line to the .bashrc or any other configuration file which is loaded when the user logs in.

Uninstall Java

If for any reason you want to uninstall the Java package, you can uninstall it like any other package installed with apt.
For example, if you want to uninstall the openjdk-8-jdk package run:

```
sudo apt remove openjdk-8-jdk
```