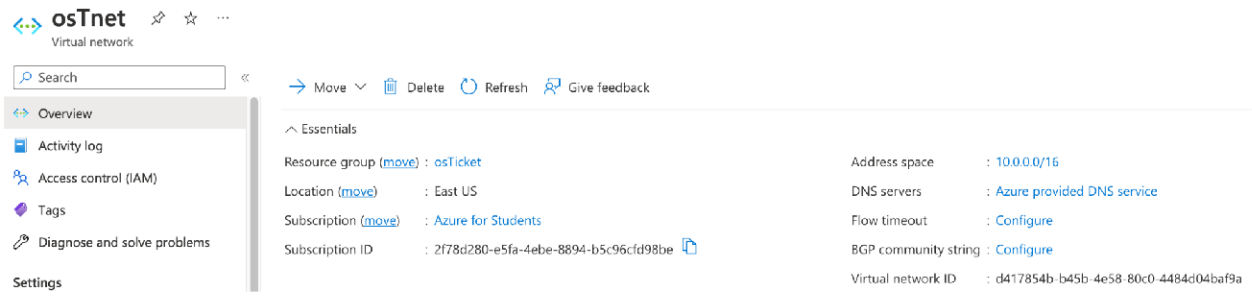
Technical Report of osTicket Deployment in Azure

Group 42

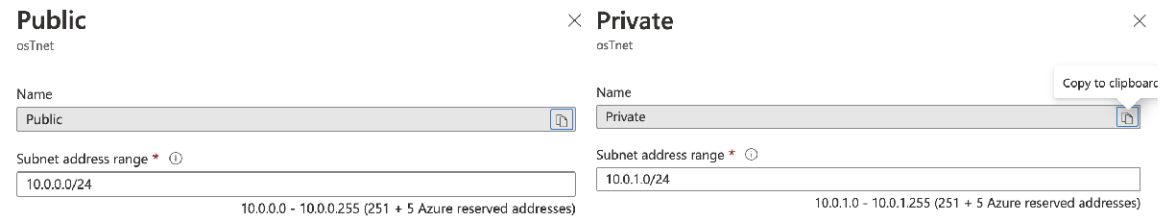
Creating the Azure Cloud Network Infrastructure

We began by creating a resource group for our project. We called it **osTicket** to differentiate it from other projects we are working on.

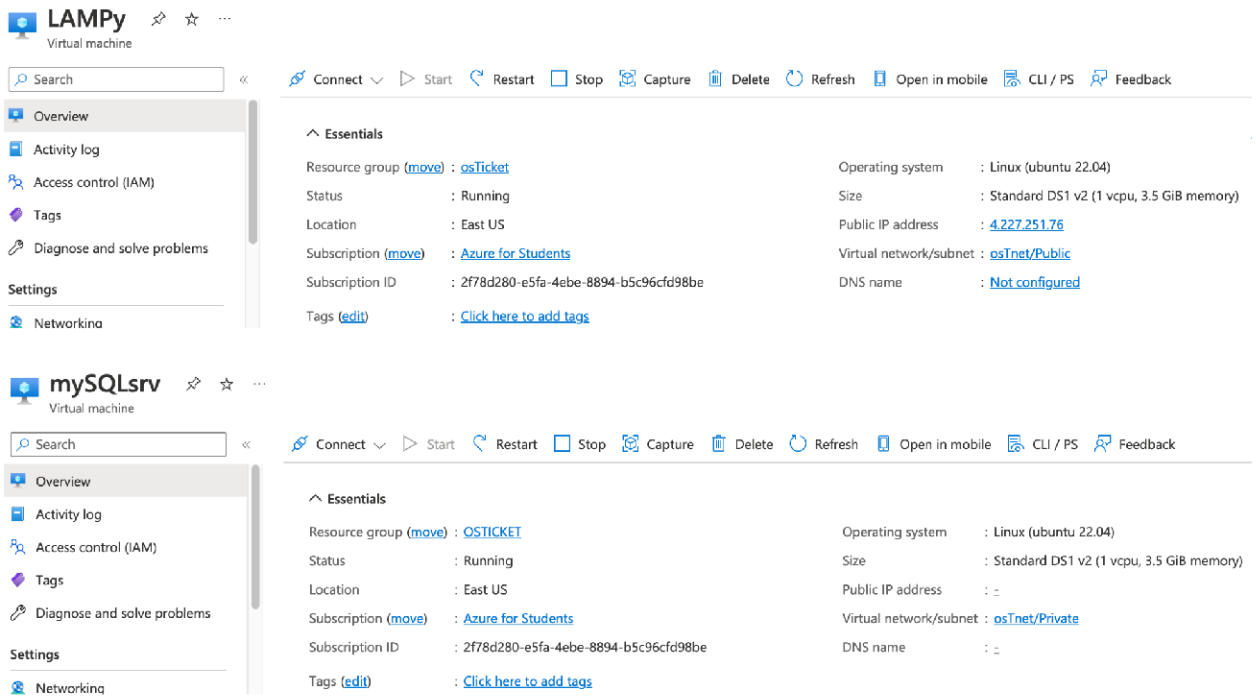
We then created a subnet, **ostnet**, to allow multiple servers to communicate both locally and externally. **Ostnet** has an address space of 10.0.0.0/16 to allow the creation of two subnets for security purposes.



We created two subnets on **ostnet**, one called **Public** with address space of 10.0.0.0/24 and one called **Private** with address space of 10.0.1.0/24, to keep the information in our database secure.



We then provisioned two different **Ubuntu 22.04** machines, **LAMPy** and **mySQLsrv**, one on each subnet to respectively serve as our **Public** web server and **Private** database server. We configured them initially to allow ssh access and saved the public keys.



We configured the firewall rule on both machines to allow traffic through port 22 and 80 so that we could connect with them and access the internet.

Priority	Name	Port	Protocol	Source	Destination	Action
300	SSH	22	TCP	Any	Any	Allow
310	AllowAnyCustom80Inbound	80	Any	Any	Any	Allow

We confirmed LAN functionality via ICMP:

```
lampyadmin@LAMPy:~$ ping -c4 10.0.1.4
PING 10.0.1.4 (10.0.1.4) 56(84) bytes of data.
64 bytes from 10.0.1.4: icmp_seq=1 ttl=64 time=1.05 ms
64 bytes from 10.0.1.4: icmp_seq=2 ttl=64 time=1.03 ms
64 bytes from 10.0.1.4: icmp_seq=3 ttl=64 time=1.06 ms
64 bytes from 10.0.1.4: icmp_seq=4 ttl=64 time=1.20 ms

--- 10.0.1.4 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 1.032/1.084/1.201/0.068 ms
lampyadmin@LAMPy:~$
```

```
mysqlasrvadmin@mySQLsrv:~$ ping -c4 10.0.0.4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=1.00 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=64 time=1.00 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=64 time=1.12 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=64 time=0.973 ms

--- 10.0.0.4 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 0.973/1.024/1.123/0.057 ms
mysqlasrvadmin@mySQLsrv:~$
```

At this point the infrastructure setup was complete and we could begin working on the servers themselves.

Setting Up the LAMP Server

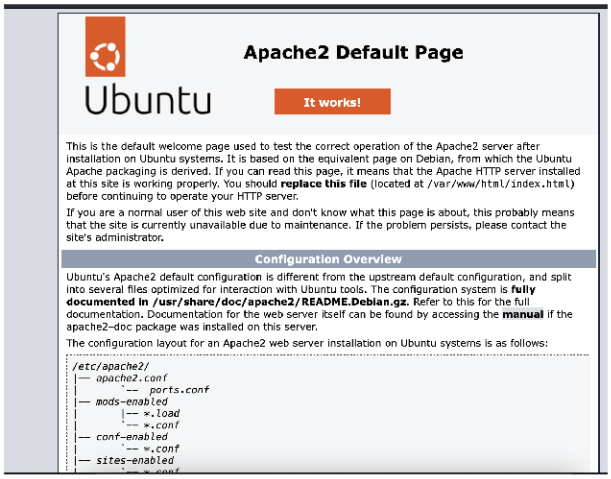
We began by updating the os to the latest version to protect it from known vulnerabilities with the commands:

sudo apt-get update
sudo apt-get upgrade

We then installed **Apache2** with the command

apt-get install apache2 -y

Confirming the functionality of **Apache2** by navigating to our web server's IP address:



We installed the required repository for **PHP**

apt-get install ca-certificates apt-transport-https software-properties-common add-apt-repository ppa:ondrej/php

We then updated the repository with:

apt-get update -y

We then installed PHP 8 (the version needed for osTicket) with:

apt-get install php8.0 libapache2-mod-php8.0 php8.0-common php8.0-fpm php8.0-cgi php8.0-bcmath php8.0-gd php8.0-imap php8.0-intl php8.0-apcu php8.0-cli php8.0-mbstring php8.0-curl php8.0-mysql php8.0-xml unzip -y

We confirmed the installation and version with:

php -v

After restarting apache with:

sudo systemctl restart apache2

We created a a phpinfo.php test page and verified successful installation with our brower:

echo '<?php phpinfo(); ?>' | sudo tee -a /var/www/html/phpinfo.php > /dev/null

Installing MySQL on the LAMP Stack

Initially, we installed mysql-client on the LAMP server, but later installed the full version due to osTicket's requirement.

We began by downloading the **MySQL .deb** file:

Next, we installed the repositories for **MySQL**:

```
mysqlasrvadmin@mySQLsrv:~$ sudo dpkg -i mysql-apt-config_0.8.24-1_all.deb
Selecting previously unselected package mysql-apt-config.
(Reading database ... 60622 files and directories currently installed.)
Preparing to unpack mysql-apt-config_0.8.24-1_all.deb ...
Unpacking mysql-apt-config (0.8.24-1) ...
Setting up mysql-apt-config (0.8.24-1) ...
Warning: apt-key should not be used in scripts (called from postinst maintainerscript of the package mysql-apt-config)
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
mysqlasrvadmin@mySQLsrv:~$ sudo apt-get update
Hit:1 http://azure.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://azure.archive.ubuntu.com/ubuntu jammy-updates InRelease [114 kB]
Get:3 http://azure.archive.ubuntu.com/ubuntu jammy-backports InRelease [99.8 kB]
Get:4 http://azure.archive.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:5 http://azure.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [698 kB]
Get:6 http://azure.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [417 kB]
Get:7 http://repo.mysql.com/apt/ubuntu jammy InRelease [15.2 kB]
```

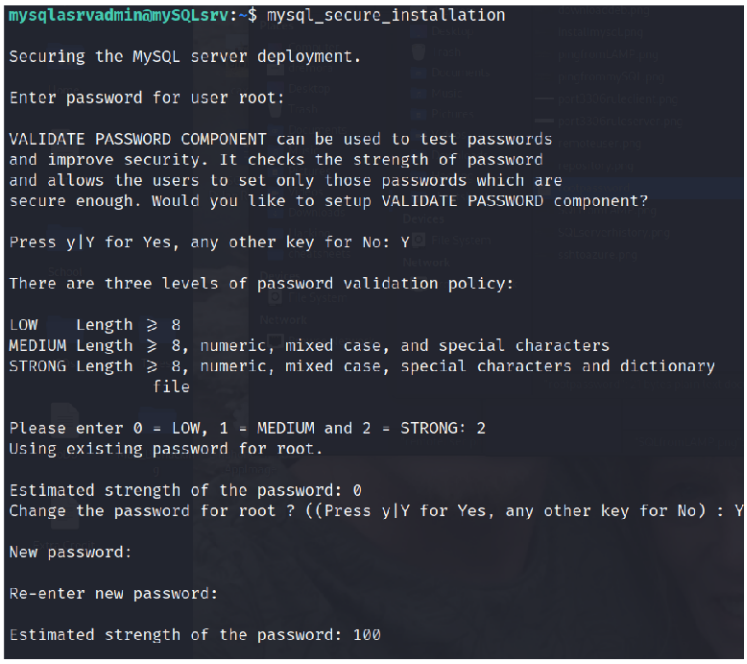
Installing **MySQL**:

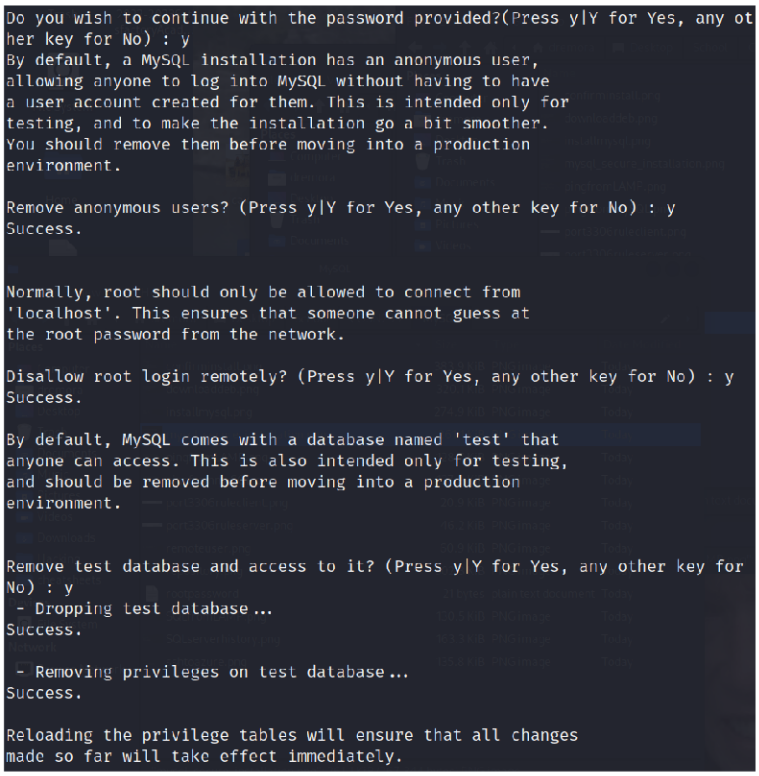
```
mysqlasrvadmin@mySQLsrv:~$ sudo apt-get install mysql-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libmecab2 mecab-ipadic mecab-ipadic-utf8 mecab-utils mysql-client mysql-common mysql-community-client mysql-community-client-core
  mysql-community-client-plugins mysql-community-server mysql-community-server-core
The following NEW packages will be installed:
  libmecab2 mecab-ipadic mecab-ipadic-utf8 mecab-utils mysql-client mysql-common mysql-community-client mysql-community-client-core
  mysql-community-client-plugins mysql-community-server mysql-community-server-core mysql-server
0 upgraded, 12 newly installed, 0 to remove and 0 not upgraded.
Need to get 37.9 MB of archives.
After this operation, 272 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 libmecab2 amd64 0.996-14build9 [199 kB]
Get:2 http://azure.archive.ubuntu.com/ubuntu jammy/main amd64 mecab-utils amd64 0.996-14build9 [4850 B]
```

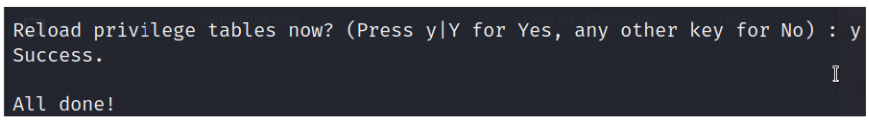
Confirming installation by checking the status of the **MySQL** service:

```
Running kernel seems to be up-to-date.

Restarting services...
Service restarts being deferred:
 systemctl restart networkd-dispatcher.service
 systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
mysqlasrvadmin@mySQLsrv:~$ systemctl status mysql.service
● mysql.service - MySQL Community Server
     Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2022-11-12 14:54:03 UTC; 27s ago
       Docs: man:mysqld(8)
             http://dev.mysql.com/doc/refman/en/using-systemd.html
   Main PID: 7091 (mysqld)
     Status: "Server is operational"
      Tasks: 39 (limit: 4095)
     Memory: 368.2M
        CPU: 1.138s
     CGroup: /system.slice/mysql.service
             └─7091 /usr/sbin/mysqld

Nov 12 14:54:01 mySQLsrv systemd[1]: Starting MySQL Community Server...
Nov 12 14:54:03 mySQLsrv systemd[1]: Started MySQL Community Server.
```

We ran **mysql_secure_installation** to configure better database security. This prompts the user for various security policies to harden the MySQL implementation: enforce password strength, update the root password, disable anonymous users, restrict root login to localhost, disable the test database, and reload privilege tables:







Setting Up the Database Server

In order to create a backend SQL database service for osTicket's storage and query needs, we commissioned an Ubuntu 22.04 server running MySQL on an internal subnet with a restrictive firewall.

For MySQL installation procedure, refer to "Installing MySQL on the LAMP Stack"

Create firewall rules in azure networking allowing traffic over port 3306 inbound for mySQLsrv and outbound for LAMPy:

Priority	Name	Port	Protocol	Source	Destination	Action
100	AllowAnyMySQLOutbound	3306	TCP	Any	10.0.0.4	Allow
65000	AllowVnetOutBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowInternetOutBound	Any	Any	Any	Internet	Allow
65500	DenyAllOutBound	Any	Any	Any	Any	Deny

Priority	Name	Port	Protocol	Source	Destination	Action
310	AllowAnyMySQLInbound	3306	TCP	10.0.0.4	Any	Allow
320	AllowCidrBlockSSHInbound	22	TCP	10.0.0.4	Any	Allow
65000	AllowVnetInBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalancerInBound	Any	Any	AzureLoadBalancer	Any	Allow
65500	DenyAllInBound	Any	Any	Any	Any	Deny

The firewall rules began in a more permissive state, with port 22, 3306, and others open to all traffic. As we became comfortable with our route to access the backend server, we confined access to what we knew to be necessary.

We created remote user account named **lampy** with root privileges for access via web server:

```
mysql> CREATE USER 'lampy'@'10.0.0.4';
ERROR 1819 (HY000): Your password does not satisfy the current policy require
ments
mysql> CREATE USER 'lampy'@'10.0.0.4' IDENTIFIED BY 'post&$^098GREEN';
Query OK, 0 rows affected (0.03 sec)

mysql> GRANT ALL ON *.* TO 'lampy'@'10.0.0.4' WITH GRANT OPTION;
Query OK, 0 rows affected (0.04 sec)

mysql> show grants for 'lampy'@'10.0.0.4';
+-----------------------------------------------------------------------------
-----------------------------------------------------------------------------
-----------------------------------------------------------------------------
```

Confirm account creation and priviledges:

```
mysql> show grants for 'lampy'@'10.0.0.4';
+-----------------------------------------------------------------------------
-----------------------------------------------------------------------------
-----------------------------------------------------------------------------
```

```
| GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, RELOAD, SHUTDOWN, PROCE
SS, FILE, REFERENCES, INDEX, ALTER, SHOW DATABASES, SUPER, CREATE TEMPORARY T
ABLES, LOCK TABLES, EXECUTE, REPLICATION SLAVE, REPLICATION CLIENT, CREATE VI
EW, SHOW VIEW, CREATE ROUTINE, ALTER ROUTINE, CREATE USER, EVENT, TRIGGER, CR
EATE TABLESPACE, CREATE ROLE, DROP ROLE ON *.* TO `lampy`@`10.0.0.4` WITH GRA
NT OPTION

                                                                            |
| GRANT APPLICATION_PASSWORD_ADMIN,AUDIT_ABORT_EXEMPT,AUDIT_ADMIN,AUTHENTICAT
ION_POLICY_ADMIN,BACKUP_ADMIN,BINLOG_ADMIN,BINLOG_ENCRYPTION_ADMIN,CLONE_ADMI
N,CONNECTION_ADMIN,ENCRYPTION_KEY_ADMIN,FIREWALL_EXEMPT,FLUSH_OPTIMIZER_COSTS
,FLUSH_STATUS,FLUSH_TABLES,FLUSH_USER_RESOURCES,GROUP_REPLICATION_ADMIN,GROUP
_REPLICATION_STREAM,INNODB_REDO_LOG_ARCHIVE,INNODB_REDO_LOG_ENABLE,PASSWORDLE
SS_USER_ADMIN,PERSIST_RO_VARIABLES_ADMIN,REPLICATION_APPLIER,REPLICATION_SLAV
E_ADMIN,RESOURCE_GROUP_ADMIN,RESOURCE_GROUP_USER,ROLE_ADMIN,SENSITIVE_VARIABL
ES_OBSERVER,SERVICE_CONNECTION_ADMIN,SESSION_VARIABLES_ADMIN,SET_USER_ID,SHOW
_ROUTINE,SYSTEM_USER,SYSTEM_VARIABLES_ADMIN,TABLE_ENCRYPTION_ADMIN,XA_RECOVER
_ADMIN ON *.* TO `lampy`@`10.0.0.4` WITH GRANT OPTION |
```

Confirm remote account and firewall access by connecting from the default 3306 from LAMP server:

```
lampyadmin@LAMPy:~$ mysql -h 10.0.1.4 -u lampy -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 17
Server version: 8.0.31 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement
.

mysql>
mysql> exit
Bye
```

Installing osTicket

To create a front-end web-app for serving help desk tickets, we installed **osTicket v1.17** on our **Apache2** web server.

Initially we attempted the installation using version 1.17.1.
This version results in a white screen and failed installation during the browser installation phase. As a workaround, we removed all traces of osTicket created throughout the following installation process, and restarted with version 1.17.

We began by using **ssh** to log in to the **LAMPy** web server:

```
┌──(dremora㉿Daemon)-[~]
└─$ ssh -i ~/.ssh/lampykey.pem lampyadmin@4.227.251.76
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-1022-azure x86_64)
```

Downloading **osTicket-v1.17.zip**:

```
lampyadmin@LAMPy:~$ wget https://github.com/osTicket/osTicket/releases/downlo
ad/v1.17/osTicket-v1.17.zip
--2022-11-13 18:03:52--  https://github.com/osTicket/osTicket/releases/downlo
ad/v1.17/osTicket-v1.17.zip
Resolving github.com (github.com)... 140.82.112.3
Connecting to github.com (github.com)|140.82.112.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-ass
```

```
HTTP request sent, awaiting response... 200 OK
Length: 54221928 (52M) [application/octet-stream]
Saving to: 'osTicket-v1.17.zip'

osTicket-v1.17.zip  100%[===================>]  51.71M   147MB/s    in 0.4s

2022-11-13 18:03:53 (147 MB/s) - 'osTicket-v1.17.zip' saved [54221928/5422192
8]
```

After creating **/var/www/html/osTicket,** we moved the .zip file to this new location, and extracted it with **7zz**, revealing the folders, **scripts** and **upload**:

```
lampyadmin@LAMPy:/var/www/html$ sudo mkdir osticket
```

```
lampyadmin@LAMPy:/var/www/html$ sudo 7zz x osTicket-v1.17.zip
```

```
lampyadmin@LAMPy:/var/www/html$ ls
index.html  osTicket-v1.17.zip  phpinfo.php  scripts  upload
```

```
lampyadmin@LAMPy:/var/www/html/osticket$ ls
scripts  upload
```

Next, from the **osticket** directory, we copied the **./upload/include/ost-sampleconfig.php** with a new name **./upload/include/ost-config.php**:

```
lampyadmin@LAMPy:/var/www/html/osticket$ sudo cp upload/include/ost-samplecon
fig.php upload/include/ost-config.php
```

We grant the appropriate privileges to **/var/www/html/osticket/** using **chown** and **chmod** as is required for functionality:

```
chown: changing ownership of 'osticket/': Operation not permitted
lampyadmin@LAMPy:/var/www/html$ sudo !!
sudo chown -R www-data:www-data osticket/
lampyadmin@LAMPy:/var/www/html$ sudo chmod 766 -R osticket/
```

We created a new database for the osTicket implementation on the backend server by connecting to with **mysql -h 10.0.1.4 -u osticketuser1 -p** and typing in the password.

After accessing the MySql server, we used the following commands to create and confirm the **osticketdb** database:

CREATE DATABASE osticketdb;

SHOW DATABASES;

Next, we created the configuration file called **osticket.conf** in **/etc/apache2/sites-available/** for **apache2** to make it interoperable with **osTicket**:

```
lampyadmin@LAMPy:~$ vim /etc/apache2/sites-available/osticket.conf
lampyadmin@LAMPy:~$ sudo !!
sudo vim /etc/apache2/sites-available/osticket.conf
```

```
<VirtualHost *:80>
    ServerAdmin ostadmin@allenkarel.rocks
     DocumentRoot /var/www/html/osticket/upload
     ServerName allenkarel.rocks

     <Directory /var/www/html/osTicket/>
          Options FollowSymlinks
          AllowOverride All
          Require all granted
     </Directory>

     ErrorLog ${APACHE_LOG_DIR}/osTicket_error.log
     CustomLog ${APACHE_LOG_DIR}/osTicket_access.log combined
RewriteEngine on
RewriteCond %{SERVER_NAME} =allenkarel.rocks
RewriteRule ^ https://%{SERVER_NAME}%{REQUEST_URI} [END,NE,R=permanent]
</VirtualHost>
```

This file was originally written with a modified template from an online walkthrough. The initial installation failed because it had capitalized the "T" in osTicket, when the actual directory had a lower-case "t". This was discovered in the systemctl status output during troubleshooting:

```
ting The Apache HTTP Server ...
 AH00112: Warning: DocumentRoot [/var/www/html/osTicket/upload] does not exi>
ted The Apache HTTP Server.
ading The Apache HTTP Server ...
 AH00112: Warning: DocumentRoot [/var/www/html/osTicket/upload] does not exi>
aded The Apache HTTP Server.
ading The Apache HTTP Server ...
aded The Apache HTTP Server.
ading The Apache HTTP Server ...
aded The Apache HTTP Server.
```

We used the **a2ensite** tool to enable the web-app on our webserver, and enabled the **apache2 rewrite module** using **a2enmod rewrite.** We then reloaded **apache2** in order to apply the new configuration files, and checked the status of the web service using **systemctl**:

```
lampyadmin@LAMPy:~$ sudo a2ensite osticket.conf
Enabling site osticket.
To activate the new configuration, you need to run:
  systemctl reload apache2
lampyadmin@LAMPy:~$ a2enmod rewrite
Module rewrite already enabled
lampyadmin@LAMPy:~$ sudo systemctl reload apache2
```

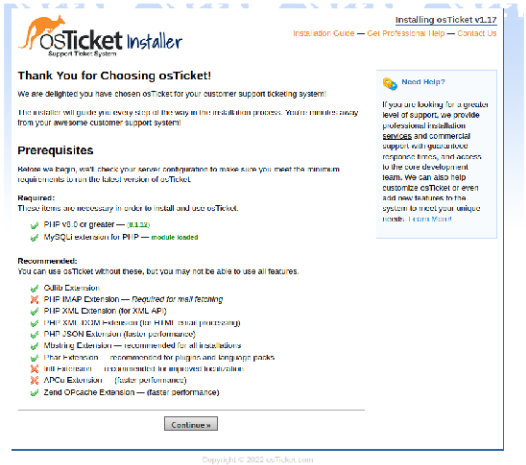
```
lampyadmin@LAMPy:~$ systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor pr>
     Active: active (running) since Sun 2022-11-13 18:22:20 UTC; 9s ago
```

The remaining portion of osTicket installation will be conducted via web browser.

We used **Name.com** to add a record to link the IP to a domain name we control for easy access:

TYPE	HOST	ANSWER	TTL	PRIO	ACTIONS
A	allenkarel.rocks	4.227.251.76	300	N/A	EDIT DELETE CREATED: 2022-04-20
A	osticket.allenkarel.rocks	4.227.251.76	300	N/A	EDIT DELETE CREATED: 2022-11-13
A	www.allenkarel.rocks	4.227.251.76	300	N/A	EDIT DELETE CREATED: 2022-11-12

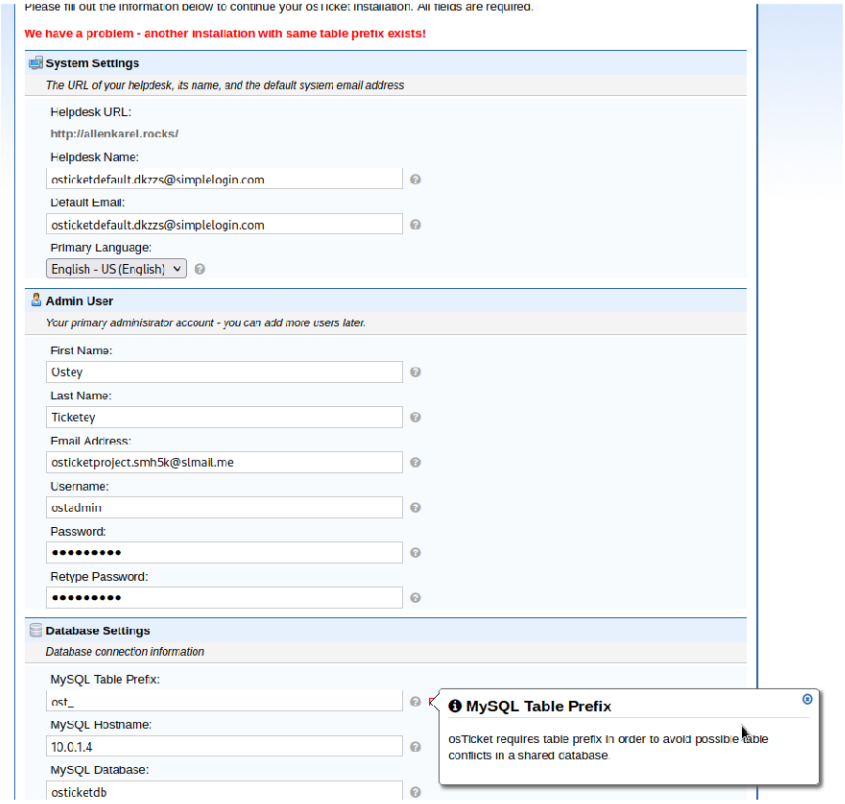
We navigated to the url of our web server, **allenkarel.rocks:**



To ensure full functionality, **php-imap php-intl** and **php-apcu** were installed on **LAMPy:**

```
lampyadmin@LAMPy:~$ sudo apt install php-imap php-intl php-apcu
```

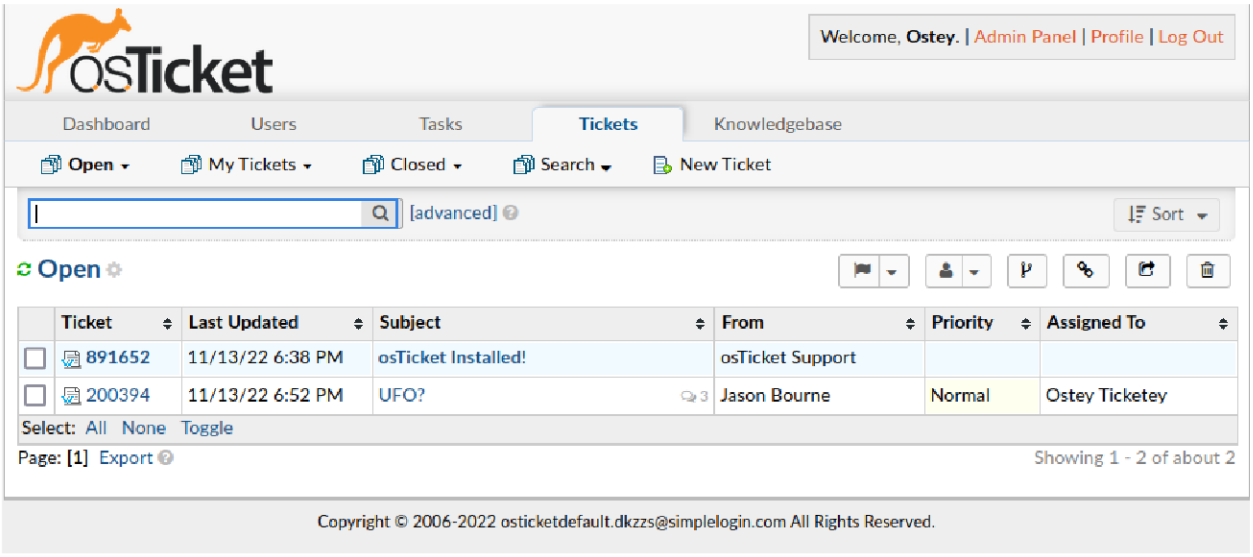
After all check marks were green, we clicked **continue**, which led to the next installation configuration screen:



After submitting, the prior failed installation's database conflicted with our new installation due to the ost_ MySQL table prefix. We could have used a different prefix, however that would have left the old database, so we connected to the MySQL server and used DROP and CREATE commands to create a new osticketdb database:

```
DROP\040DATABASE\040osticketdb
;
SHOW\040DATABASES
;
CREATE\040DATABASE\040osticketdb;
exit
SHOW\040DATABASES
;
SELECT\040*\040FROM\040osticketdb;
USE\040osticketdb;
SELECT\040*\040FROM\040osticketdb;
show\040databases;
SHOW\040TABLES;
SHOW\040*\040FROM\040ost_ticket;
SELECT\040*\040FROM\040ost_ticket;
SELECT\040*\040FROM\040ost_ticket__cdata;
SELECT\040*\040FROM\040ost_team;
show\040tables;
SELECT\040*\040FROM\040ost_staff;
SELECT\040*\040FROM\040ost_form_entry;
SELECT\040*\040FROM\040ost_form_entry_values;
SELECT\040*\040FROM\040ost_task;
SELECT\040*\040FROM\040ost_task__cdata;
```

After fixing this problem, we were able to successfully install osTicket.



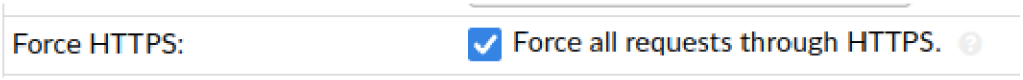
For security, we secured the domains with **SSL** by first installing the **certbot client**:

apt-get install python3-certbot-apache -y

Then we requested a certificate from **Let's Encrypt SSL** with:

certbot --apache -d osticket.allenkarel.rocks

We then added a firewall rule to **allow port 443** and **removed the rule for port 80** to ensure all traffic is secure. We also forced all requests through HTTPS via the admin panel in osTicket's GUI web-based interface:



After ensuring that ticketing and login systems were functional, we had successfully completed our osTicket deployment: