# 第一章绪论

随着计算机技术的迅速发展，各种程序渗透到不同的各行各业，成为几乎为人们生活中不可或缺的工具。软件行业在快速发展，于此同时对软件的管理也显得越来越麻烦，大量的各种各样的新的服务程序不断产生，网络管理员对于他们的管理十分困难，是一件十分头疼的事情。幸运的是，随着计算机虚拟化和容器化技术的不断成长，现有的虚拟化技术和容器化技术可以很好的解决这种问题，网络管理员可以通过将网络服务制作成一个个的容器，这样不但减少了网络服务迁移的各种配置，还有效地解决了环境污染的问题。

1.1研究背景与意义

随着软件虚拟化技术的快速发展和进步，对于大量网络服务对于虚拟化技术的依赖不断提高。虚拟化和容器化网络服务已经成为一种普遍现象，docker虚拟化和容器化技术无疑是做的最成功的。容器化服务可以有效提高开发出者的效率，可以避免服务从一个环境移植到另一个环境所带来的各种依赖和版本问题。将程序置于docker容器中，可以有效的将服务隔离至docker容器中，也可以有效隔离服务里面的恶意软件带来的攻击，从而提高网络安全。

1.2本文的研究内容

1. 本文研究Linux平台上基于docker容器技术搭建Apache，mysql和php服务。

# 第二章环境介绍

介绍了LAMP的各种组件。

* 1. Docker

Docker是一种轻量级的虚拟化技术，同时是一个开源的应用容器运行环境搭建平台，可以让开发者以便捷方式打包应用到一个可移植的容器中，然后安装至任何运行Linux或Windows等系统的服务器上。相较于传统虚拟机，Docker容器提供轻量化的虚拟化方式、安装便捷、启停速度快。

* + 1. Linux

Linux，全称GNU/Linux，是一种免费使用和自由传播的[类UNIX](https://baike.baidu.com/item/%E7%B1%BBUNIX/9032872" \t "/Users/coolder/Documents\\x/_blank)操作系统，其内核由[林纳斯·本纳第克特·托瓦兹](https://baike.baidu.com/item/%E6%9E%97%E7%BA%B3%E6%96%AF%C2%B7%E6%9C%AC%E7%BA%B3%E7%AC%AC%E5%85%8B%E7%89%B9%C2%B7%E6%89%98%E7%93%A6%E5%85%B9/1034429" \t "/Users/coolder/Documents\\x/_blank)于1991年10月5日首次发布，它主要受到[Minix](https://baike.baidu.com/item/Minix/7106045" \t "/Users/coolder/Documents\\x/_blank)和Unix思想的启发，是一个基于[POSIX](https://baike.baidu.com/item/POSIX" \t "/Users/coolder/Documents\\x/_blank)的多用户、[多任务](https://baike.baidu.com/item/%E5%A4%9A%E4%BB%BB%E5%8A%A1/1011764" \t "/Users/coolder/Documents\\x/_blank)、支持[多线程](https://baike.baidu.com/item/%E5%A4%9A%E7%BA%BF%E7%A8%8B/1190404" \t "/Users/coolder/Documents\\x/_blank)和多[CPU](https://baike.baidu.com/item/CPU" \t "/Users/coolder/Documents\\x/_blank)的操作系统。它能运行主要的[Unix](https://baike.baidu.com/item/Unix/219943" \t "/Users/coolder/Documents\\x/_blank)工具软件、应用程序和网络协议。它支持[32位](https://baike.baidu.com/item/32%E4%BD%8D/5812218" \t "/Users/coolder/Documents\\x/_blank)和[64位](https://baike.baidu.com/item/64%E4%BD%8D" \t "/Users/coolder/Documents\\x/_blank)硬件。Linux继承了Unix以网络为核心的设计思想，是一个性能稳定的多用户网络操作系统。Linux有上百种不同的发行版，如基于社区开发的[debian](https://baike.baidu.com/item/debian/748667" \t "/Users/coolder/Documents\\x/_blank)、[archlinux](https://baike.baidu.com/item/archlinux/10857530" \t "/Users/coolder/Documents\\x/_blank)，和基于商业开发的[Red Hat Enterprise Linux](https://baike.baidu.com/item/Red Hat Enterprise Linux/10770503" \t "/Users/coolder/Documents\\x/_blank)、[SUSE](https://baike.baidu.com/item/SUSE/60409" \t "/Users/coolder/Documents\\x/_blank)、[Oracle Linux](https://baike.baidu.com/item/Oracle Linux/6876458" \t "/Users/coolder/Documents\\x/_blank)等，本次实验采用的Linux发行版本是Ubuntu 20.04 X86\_64架构。

* + 1. Mysql

MySQL是一个[关系型数据库管理系统](https://baike.baidu.com/item/%E5%85%B3%E7%B3%BB%E5%9E%8B%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86%E7%B3%BB%E7%BB%9F/696511" \t "/Users/coolder/Documents\\x/_blank)，由瑞典[MySQL AB](https://baike.baidu.com/item/MySQL AB/2620844" \t "/Users/coolder/Documents\\x/_blank) 公司开发，属于 [Oracle](https://baike.baidu.com/item/Oracle" \t "/Users/coolder/Documents\\x/_blank) 旗下产品。MySQL 是最流行的[关系型数据库管理系统](https://baike.baidu.com/item/%E5%85%B3%E7%B3%BB%E5%9E%8B%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86%E7%B3%BB%E7%BB%9F/696511" \t "/Users/coolder/Documents\\x/_blank)之一，在 [WEB](https://baike.baidu.com/item/WEB/150564" \t "/Users/coolder/Documents\\x/_blank) 应用方面，MySQL是最好的 [RDBMS](https://baike.baidu.com/item/RDBMS/1048260" \t "/Users/coolder/Documents\\x/_blank) (Relational Database Management System，关系数据库管理系统) 应用软件之一。

1. PHP

PHP（PHP: Hypertext Preprocessor）即“[超文本](https://baike.baidu.com/item/%E8%B6%85%E6%96%87%E6%9C%AC" \t "/Users/coolder/Documents\\x/_blank)[预处理器](https://baike.baidu.com/item/%E9%A2%84%E5%A4%84%E7%90%86%E5%99%A8" \t "/Users/coolder/Documents\\x/_blank)”，是在[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8/100571" \t "/Users/coolder/Documents\\x/_blank)端执行的[脚本语言](https://baike.baidu.com/item/%E8%84%9A%E6%9C%AC%E8%AF%AD%E8%A8%80/1379708" \t "/Users/coolder/Documents\\x/_blank)，尤其适用于[Web](https://baike.baidu.com/item/Web/150564" \t "/Users/coolder/Documents\\x/_blank)开发并可嵌入[HTML](https://baike.baidu.com/item/HTML/97049" \t "/Users/coolder/Documents\\x/_blank)中。PHP语法学习了[C语言](https://baike.baidu.com/item/C%E8%AF%AD%E8%A8%80/105958" \t "/Users/coolder/Documents\\x/_blank)，吸纳[Java](https://baike.baidu.com/item/Java/85979" \t "/Users/coolder/Documents\\x/_blank)和[Perl](https://baike.baidu.com/item/Perl/851577" \t "/Users/coolder/Documents\\x/_blank)多个语言的特色发展出自己的特色语法，并根据它们的长项持续改进提升自己，例如java的[面向对象](https://baike.baidu.com/item/%E9%9D%A2%E5%90%91%E5%AF%B9%E8%B1%A1/2262089" \t "/Users/coolder/Documents\\x/_blank)编程，该语言当初创建的主要目标是让开发人员快速编写出优质的web[网站](https://baike.baidu.com/item/%E7%BD%91%E7%AB%99/155722" \t "/Users/coolder/Documents\\x/_blank)。 [1-2]  PHP同时支持面向对象和[面向过程](https://baike.baidu.com/item/%E9%9D%A2%E5%90%91%E8%BF%87%E7%A8%8B/9957246" \t "/Users/coolder/Documents\\x/_blank)的开发，使用上非常灵活。

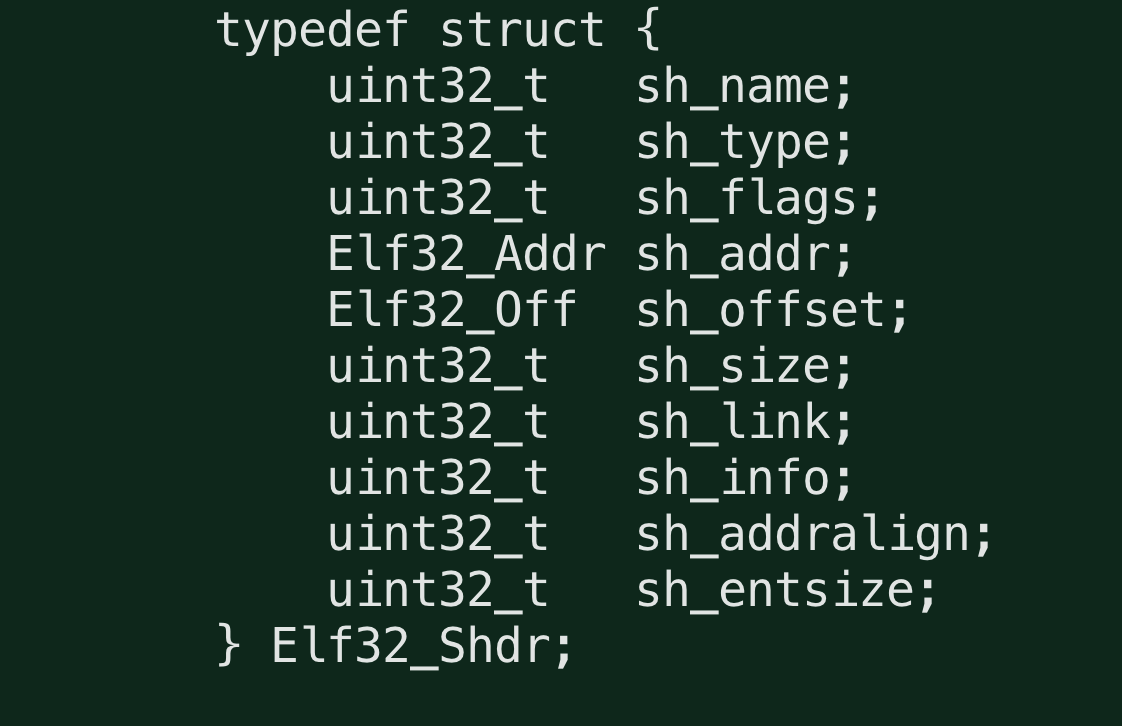


图2-4 ELF程序头结构

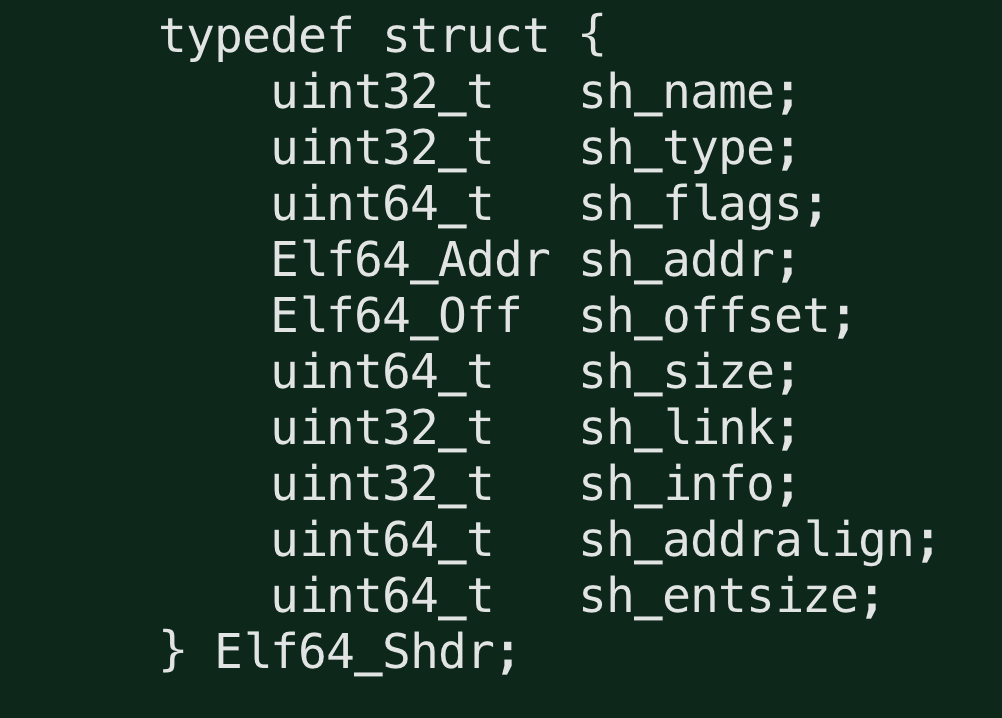


图2-5 ELF程序头结构

sh\_addralign节

指明本节内容在进程地址空间中的对齐粒度，如果该成员为0或1，则表示没 有字节对齐要求。

sh\_entsize节

对于像符号表这样的节，节的大小固定，该成员指定这种节的大小。如果当 前节不是固定大小类型的节，则该成员为零。

如果ELF文件中没有节头表，并不意味这没有节存在，只是不能通过节头来 对节进行引用。没有节头表的ELF文件对调试器或者反编译程序来说，也只是可 以参考的信息变少了而己，仍然能够正常进行反编译和调试工作。在实际的应用中，一般会通过删除节头表来增加逆向的难度。但是对于有经验的逆向分析人员， 仍然可以从程序头中获取相关信息来重构节头表，甚至能重构出部分或者完整的 符号表与节头表。一个正常的ELF文件一般会包含三十多个节，下面就其中一些比较常见且重 要的节进行简单介绍。

.text节

该节保存程序的代码指令，是ELF文件中最重要的节之一，该节一般存在于 text段中。

.rodata节

保存了程序中的只读数据，在程序装载时，会被装载进入text段中。

.plt 节

.plt节中包含了动态链接器调用从共享库导入的函数所必需的相关代码。

.data 节

存放程序中被初始化的全局变量，该节存放在data段中。

.bss 节

保存文件中未初始化的全局变量，存放在data段中。程序在加载时这个节的 数据会被初始化为0。该节在文件中不占用实际空间。

.got 节

保存程序的全局偏移表。

.dynsym 节

保存需要从共享库导入的动态符号信息。

.symtab 节

ELF文件的符号表。如果文件有可加载段包括这个符号表，则该节的属性将 包含SHF\_ALL0C比特位；否则，节的属性比特位为零。

.strtab 节

本节包含ELF文件中使用到的字符串，最常见的是表示与符号表入口相关 联的名称。如果文件中有可加载段包括这个符号字符串表，则节的属性会包括 SHF\_ALLOC比特位，相应比特位为0。

.ctors 节

保存指向C++构造函数的初始化指针，这个节在main函数之前被执行。这个 节常被用来做反调试和反跟踪检查，也常常被攻击者利用进行恶意攻击。

.dtors 节

保存指向C++析构函数的初始化指针，这个节在main函数结束后被执行。

1. 本章小结

本章首先详细介绍了Docker容器化虚拟技术，Linux操作系，Mysql关系型数据库和Php后端编程语言的基础概念，下面将介绍相关环境的搭建。

# 第三章LAMP环境的搭建

本章通过在Ubuntu 20.04 x86\_64位操作系统下使用apt命令进行相关环境的安装。使用Docker，DockerFile和docker-compose进行docker lamp环境的部署。

1. Docker Docker-Compose的安装

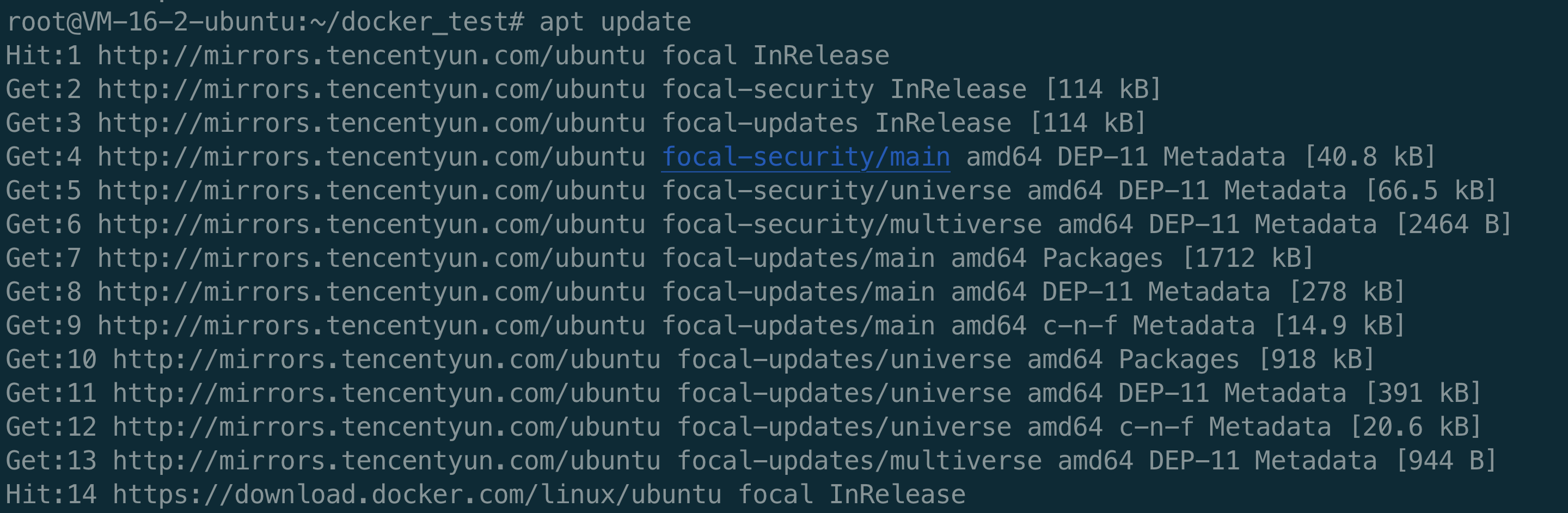


图3-1 更新软件源



图3-2 GPG证书安装

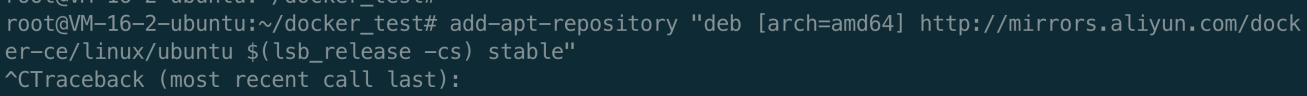


图3-3 添加仓库

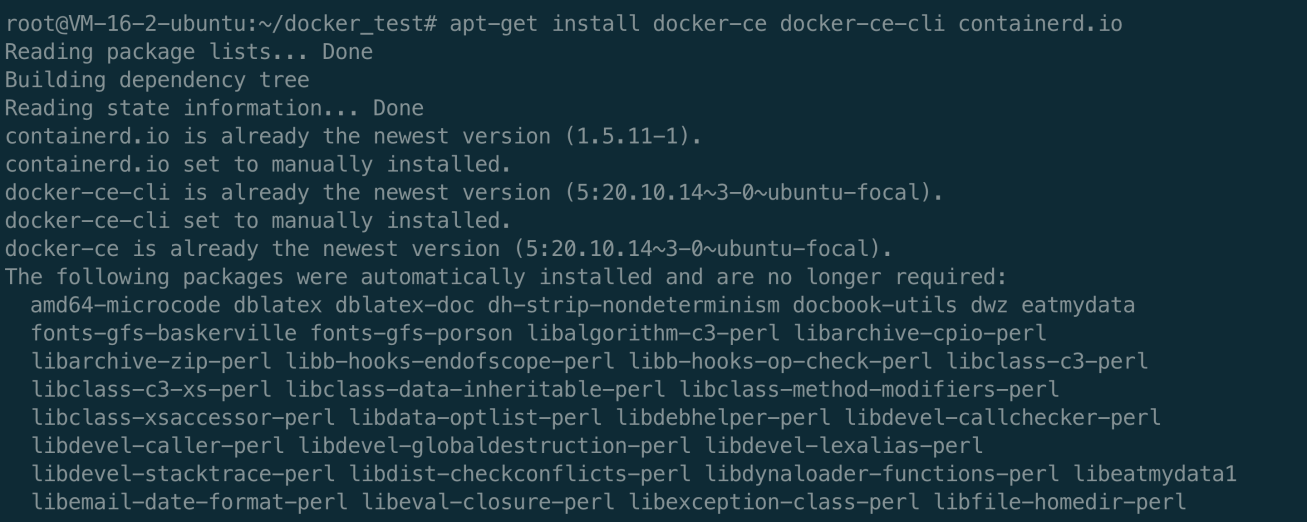


图3-4 docker-compose的安装

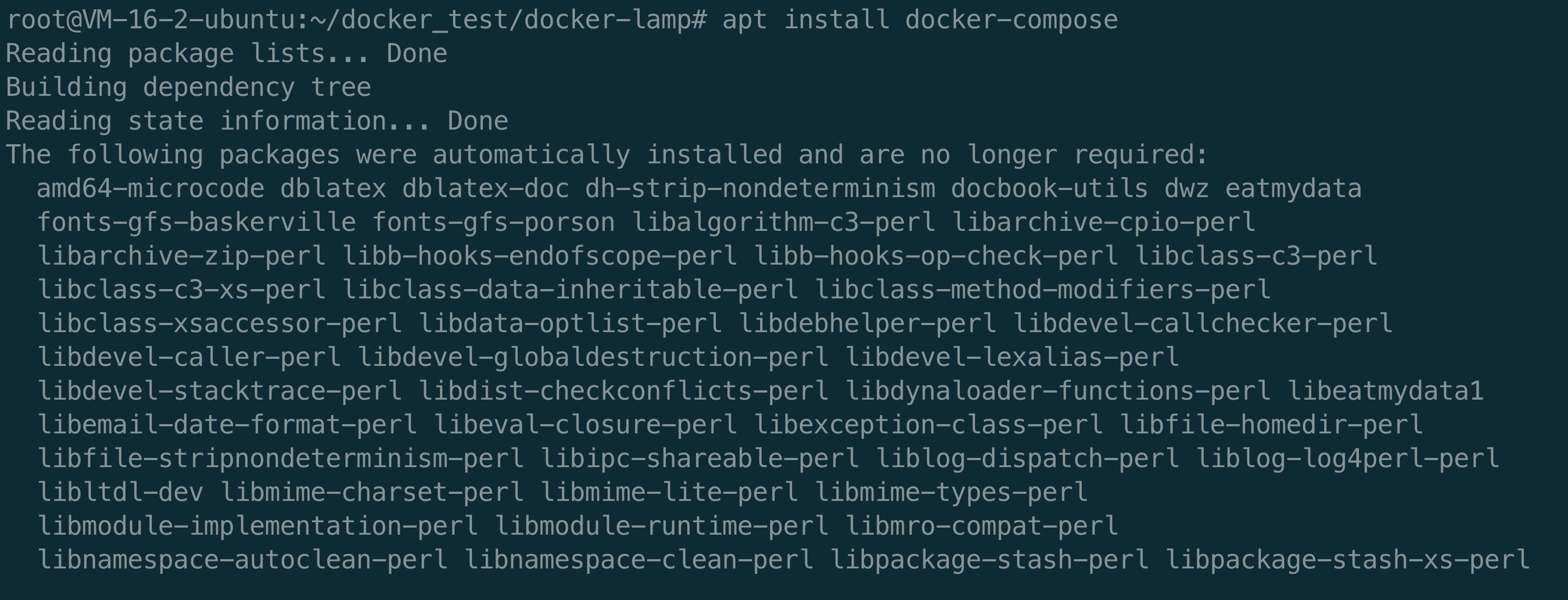


图3-5 docker-compose的安装

1. 配置文件的编写

Dockerfile的编写

FROM php:7.2-apache

RUN a2enmod rewrite

RUN docker-php-ext-install pdo pdo\_mysql

RUN apt-get update \

&& apt-get install -y libzip-dev \

&& apt-get install -y zlib1g-dev \

&& rm -rf /var/lib/apt/lists/\* \

&& docker-php-ext-install zip

在容器里面使用apt-get安装相应库的安装，随后使用docker-php-ext-install进行php扩展服务的安装。

Docker-compose.yml文件的编写

使用services关键字指定服务名称，服务器www。使用ports关键字映射端口号为容器的80端口到本机的8001端口。使用volumes关键字挂载文件夹为./www:/var/www/html/。使用networks关键字绑定网络到docker的默认网络。

services:

www:

build: .

ports:

- "8001:80"

volumes:

- ./www:/var/www/html/

links:

- db

networks:

- default

使用image指定镜像的名称，这里为mysql的8.0版本。使用ports进行端口映射，从容器的3306到本地的3306端口。使用environment，指定环境变量，默认账户为user，默认密码为test。使用volumes进行mysql服务配置文件的挂载。使用的是docker的默认网络。

db:

image: mysql:8.0

ports:

- "3306:3306"

command: --default-authentication-plugin=mysql\_native\_password

environment:

MYSQL\_DATABASE: myDb

MYSQL\_USER: user

MYSQL\_PASSWORD: test

MYSQL\_ROOT\_PASSWORD: test

volumes:

- ./dump:/docker-entrypoint-initdb.d

- ./conf:/etc/mysql/conf.d

- persistent:/var/lib/mysql

networks:

- default

Index.php

$conn = mysqli\_connect('db', 'user', 'test', "myDb");

$query = 'SELECT \* From Person';

$result = mysqli\_query($conn, $query);

echo '<table class="table table-striped">';

echo '<thead><tr><th></th><th>id</th><th>name</th></tr></thead>';

while($value = $result->fetch\_array(MYSQLI\_ASSOC)){

echo '<tr>';

echo '<td><a href="#"><span class="glyphicon glyphicon-search"></span></a></td>';

foreach($value as $element){

echo '<td>' . $element . '</td>';

}

echo '</tr>';

}

echo '</table>';

在php文件使用mysqli\_connect('db', 'user', 'test', "myDb")函数连接数本地据库，用户名为user， 密码为test，连接的数据库为myDb。

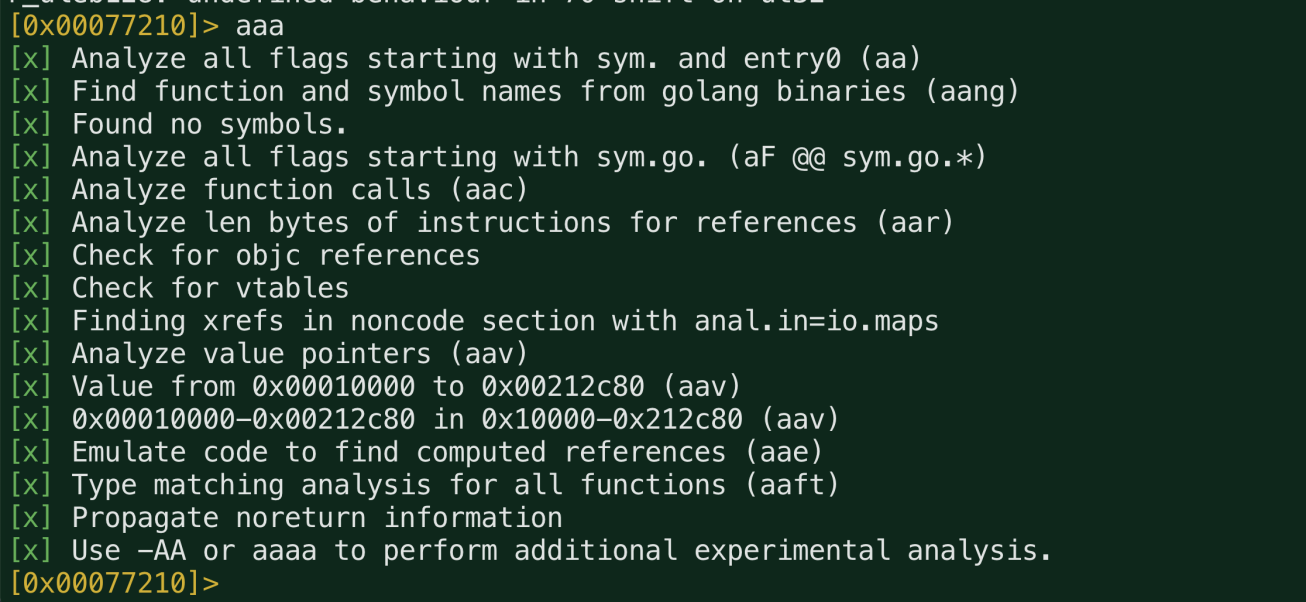


图3-6 分析程序

通过使用alf命令加上管道和grep，搜索我们想要的关键函数

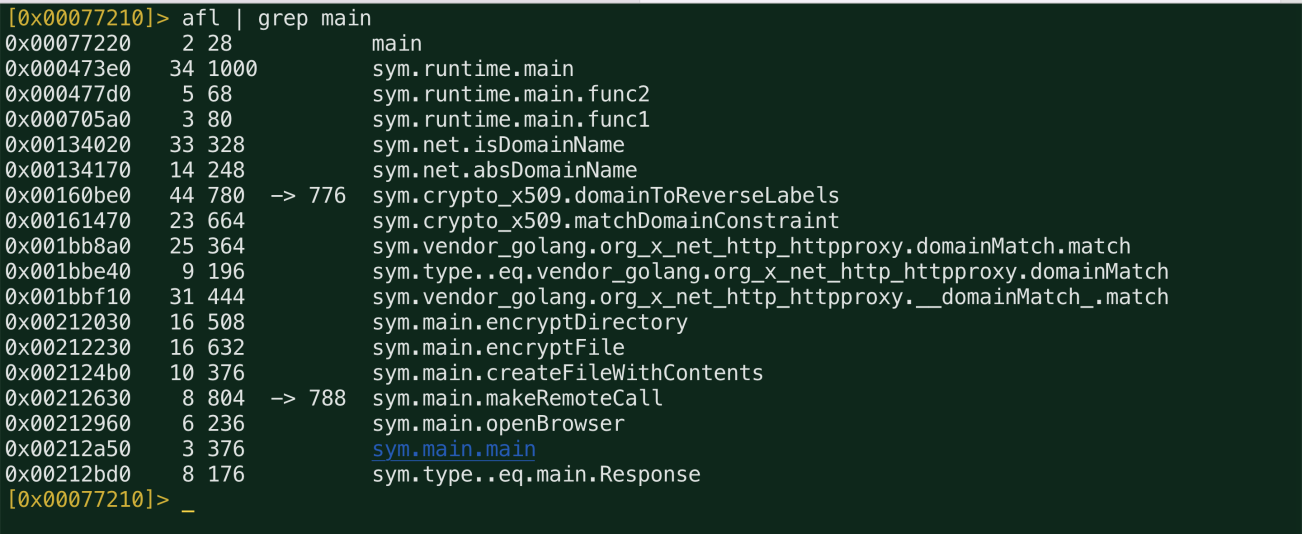


图3-7 查找关键函数

使用pdf命令反汇编sym.main.main函数

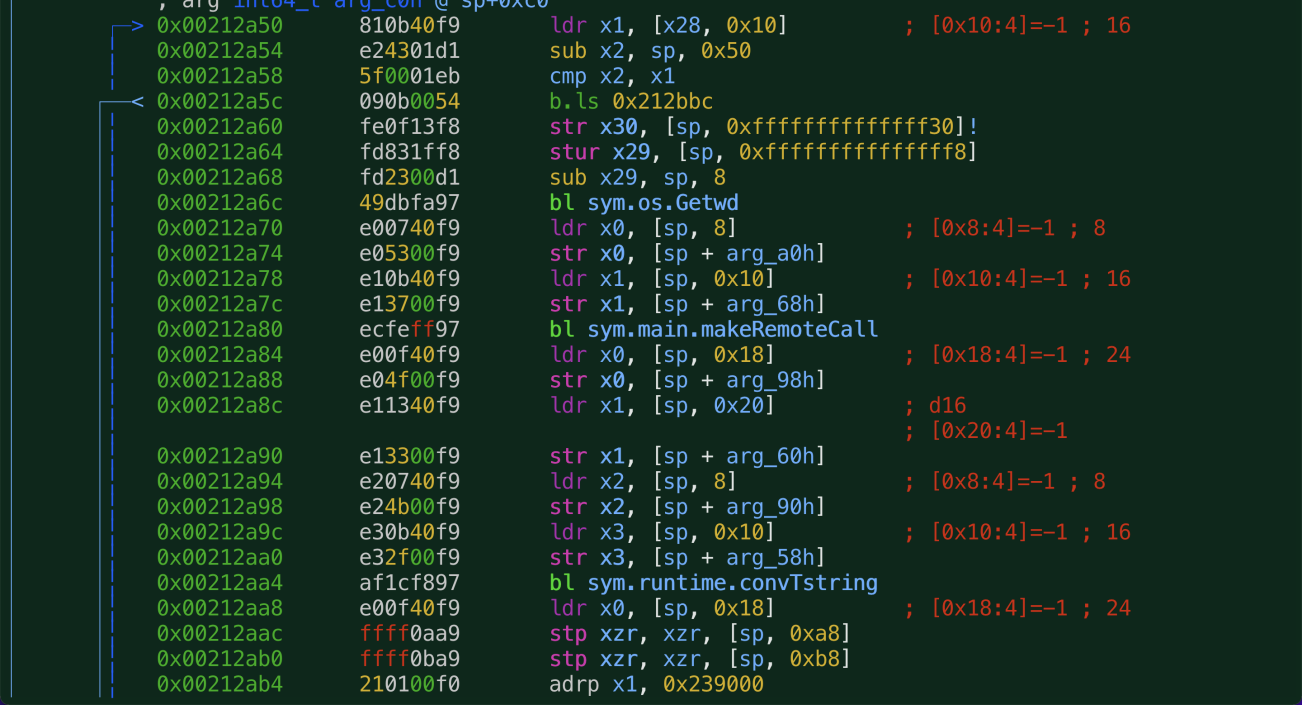


图3-8 反编译main函数



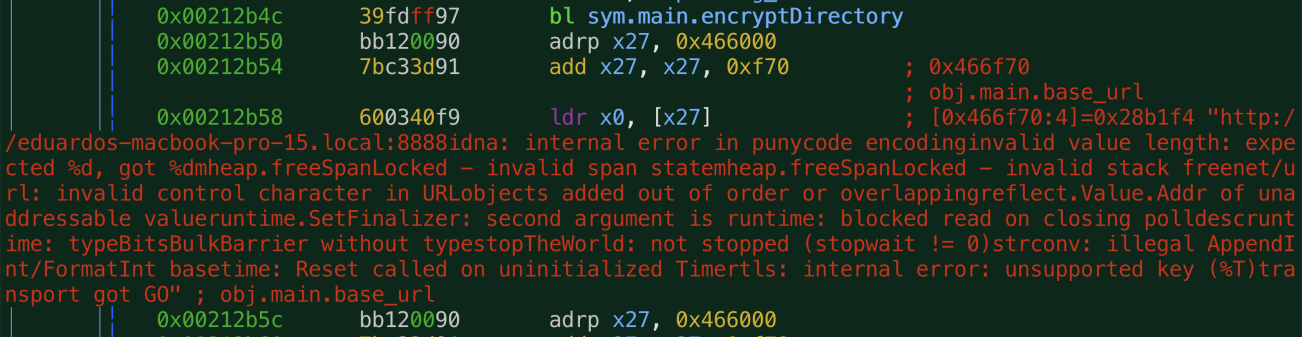


表3-9 main函数汇编代码



表3-10 main函数汇编代码

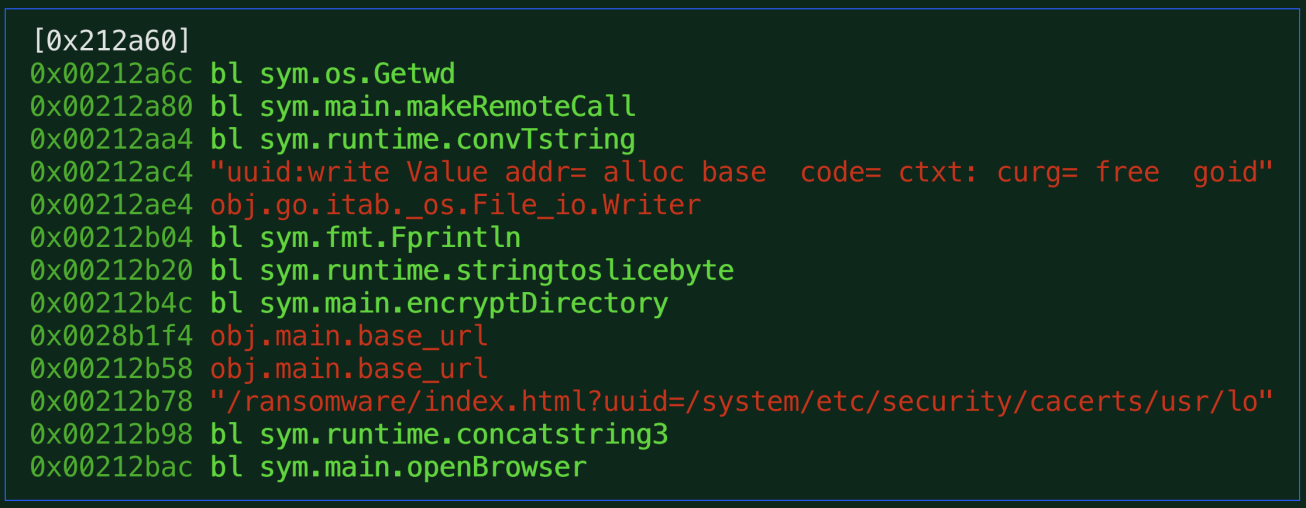


表3-11整理后的 main函数汇编代码

通过汇编代码可以发现，该程序通过使用remoteCall发起远程访问z，然后使用remoteCall返回的数据作为加密的一个重要点，最后通过调用浏览器，让受害者浏览攻击者想要的网页，可以看到疑似URL的字符串/ransomware/index.html。

# 第四章运行结果

4.1 容器服务情况

使用docker-compose -d开始容器的构建

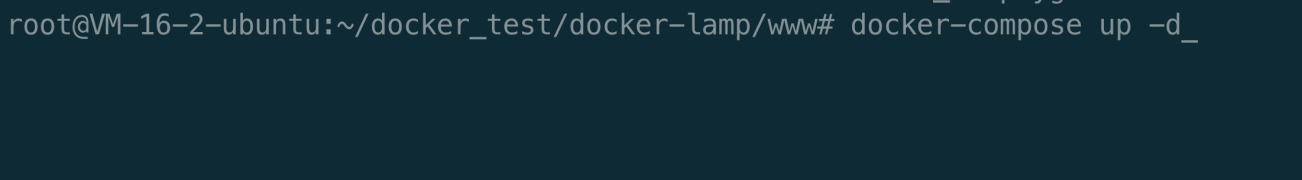


图4-1 构建容器

使用docker ps查看运行的进程

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

baf4ed644d30 phpmyadmin/phpmyadmin "/docker-entrypoint.…" 25 hours ago Up 25 hours 0.0.0.0:8000->80/tcp, :::8000->80/tcp docker-lamp\_phpmyadmin\_1

b9e55dfe2a86 docker-lamp\_www "docker-php-entrypoi…" 25 hours ago Up 25 hours 0.0.0.0:8001->80/tcp, :::8001->80/tcp docker-lamp\_www\_1

9afb4b4e1aba mysql:8.0 "docker-entrypoint.s…" 25 hours ago Up 25 hours 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp docker-lamp\_db\_1

e55f204f9624 ubuntu "/bin/bash" 2 days ago Up 2 days determined\_chaplygin

可以看到已经有三个容器启动了，人别事phpmyadmin，apache和mysql，端口映射也没有问题。

使用ss -ntl查看开放的端口号



图4-2 使用ss -ntl获取端口号

访问网站8001端口

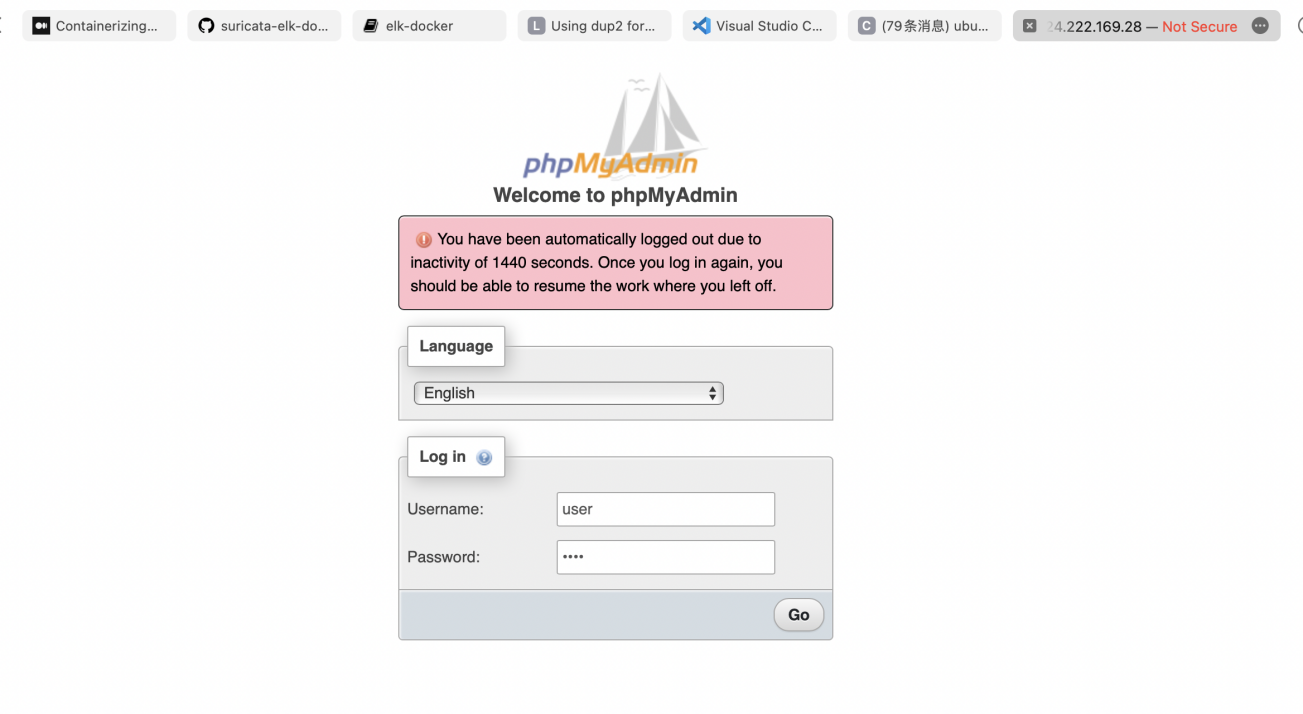


图4-3 访问登陆界面

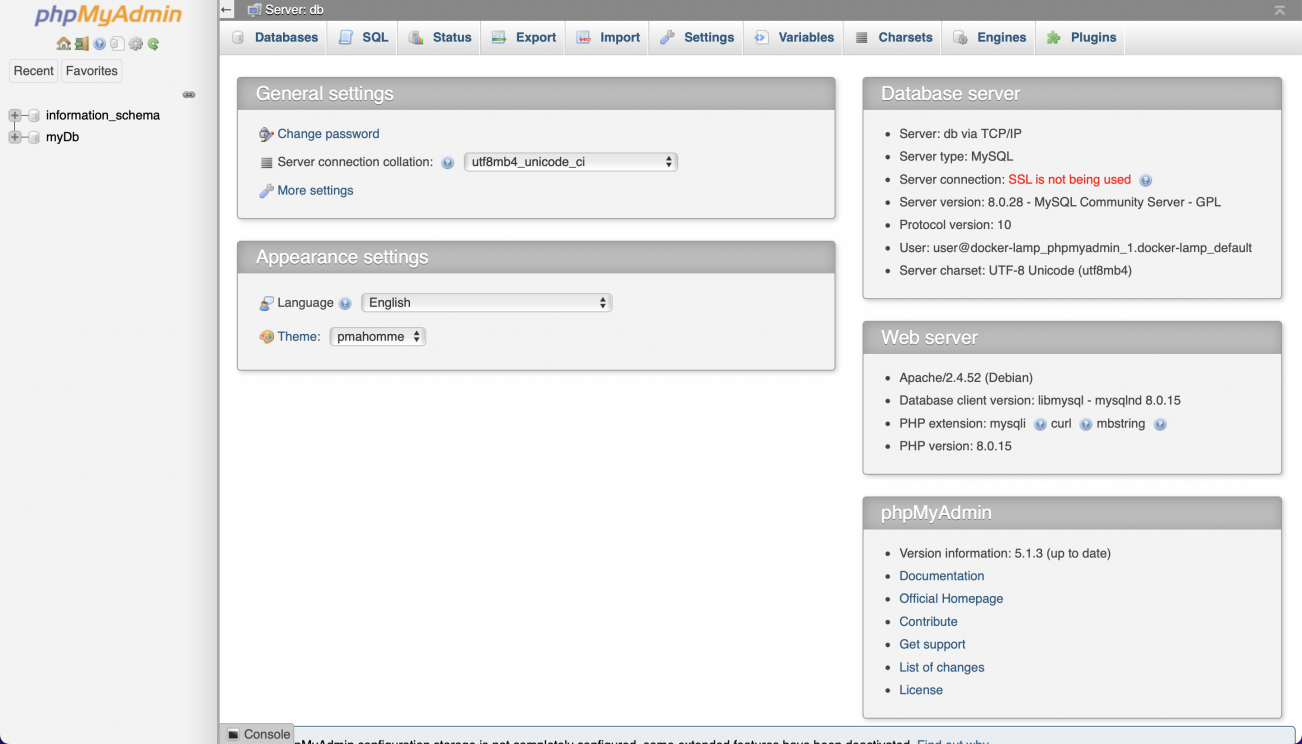


图4-4 成功访问后台

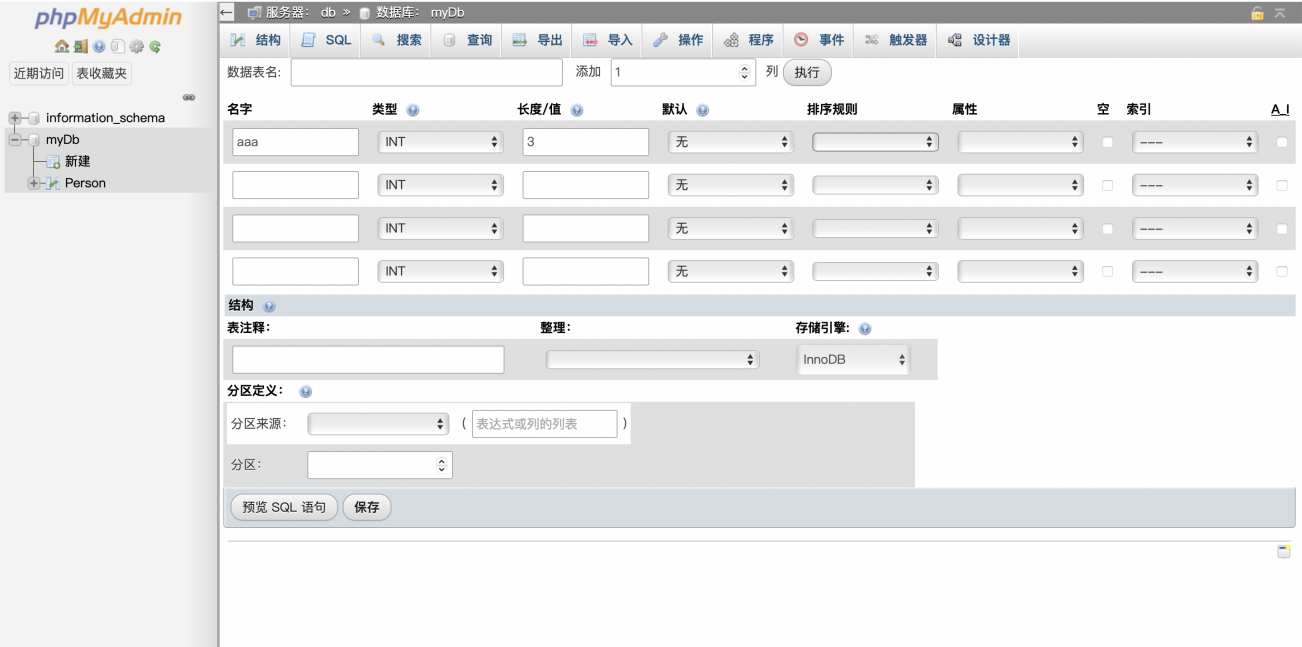


图4-5 成功管理后台mysql数据库

全文总结

随着大数据与网络时代的不断普及，大量新型设备和网络服务的添加，网络管理员对资源的维护将会变得越来越困难。幸运的是我们可以使用docker容器化和虚拟化技术进行各种服务的管理，每个容器都可以脱离环境在不同的平台上面运行可以轻松实现跨平台的操作。在有大量容器时，我们也可以使用docker-compose进行批量管理。这极大地简化了网络管理员的管理。

使用Dockerfile进行容器的单独管理，将所有配置文件全部写入Dockerfile里面，使用者只需使用docker build命令即可自动从网络构建容器。使用Phpadmin方便后端php服务的管理，可视化mysql数据库，方便网络管理员的管理。