**Day 2 任务书 &Day 3 任务书**

**Day 2 任务书**

**【任务目标】：**

任务 1：完成 BOA 服务下的 shell 脚本 CGI 程序功能；

任务 2：完成 C 语言下 CGI 基础动态数据展示效果；

任务 3：熟悉 BOA 下 CGI 配置步骤和工作原理；

**【任务说明】：**

l **任务 1 描述：**

1. 什么是动态网页，什么是静态网页？

动态网页：同一个地址，解析请求后，执行了一个程序代码，这个程序代码产生了我们需要的文本文件

静态网页：同一个地址，解析请求后，打开的是一个固定的文本文件

1. CGI 的工作原理介绍？

CGI提供了一个web服务器和程序动态生成数据之间的一个通用接口接口

1. 使用 shell 脚本快速实现一个动态 cgi 访问的步骤；
2. 首先在/web-prj/works/cgi/time的shell脚本下添入可以动态生产数据的代码
3. 然后给time文件增加一个可执行文件
4. 打开boa.conf文件在196行调整虚拟路径和目录的关系
5. 重启boa服务
6. 用netstat -tan,检查没有80，就证明没有问题
7. 在浏览器上运行

l **任务 2 描述：**

1. C 语言编译环境搭建、Makefile 工程管理文件的编写；

C语言中：

1. #include <stdio.h>

int main(){

    char \*head\_title="这是一个CGI测试程序";

    printf("Content-Type:text/html;charset=utf-8\r\n");

    printf("\r\n");

    printf("<h2>%s</h2>",head\_title);

    return 0;

}

Makefile ：

TARGET := cgi\_test

CROSS\_COMPILE :=

CC = $(CROSS\_COMPILE)gcc

LD = $(CROSS\_COMPILE)ld

OBJS :=cgi\_test.o

CFLAGS += -Wall -I./inc

LDFLAGS +=

OUTPUT\_DIR = /home/ltt/web-prj/works/cgi/

all:$(TARGET)

$(TARGET):$(OBJS)

$(CC) $(LDFLAGS) $^ -o $@

cp $@ $(OUTPUT\_DIR)

%.o:%.c

$(CC) $(CFLAGS) -c $^ -o $@

clean:

rm -f $(TARGET) $(OBJS)

1. C 语言实现动态网页 CGI 程序方法；
2. 首先创建一个cgi\_test.c和Makefile的文件
3. 在test目录下创建一个day2的文件夹，把前面两个文件拖入day2的文件里面
4. 再make一下
5. 再开一个终端，检查cgi文件夹里面有cgi\_test文件
6. 最后在浏览器上运行(要主意IP的变化)

l **任务 3 描述：**

1. BOA 配置文件中有关 CGI 功能的配置选项有哪些？

在/etc/boa/boa.conf文件里面

1. 行CGIPath,相当于CGI运行了，添一点东西，多添一点，能在某个路径上能找到
2. 行虚拟路径和目录的关系

看见/cgi-bin/,系统就认为是虚的，然后去一个地方找，到这个路径去找对应的程序/home/ltt/web-prj/works/cgi/

1. 解释 BOA 实现 CGI 功能的步骤？
2. 首先要移植boa，再修改里面的一些配置文件
3. 建立自己放文件的地方
4. 把写好的程序放进去，然后再make
5. 最后在浏览器上运行

**Day3 任务书**

**【任务目标】：**

Ø 任务 1：掌握浏览器访问 Web 服务器的数据流程原理；

Ø 任务 2：分析 html 加载网页和异步动态数据加载网页的区别和特点；

Ø 任务 3：掌握 HTTP 协议常见请求和响应数据包组成；

**【任务说明】：**

l **任务 1 描述：**

1. 用户在浏览器上输入 URL 网址回车后，经过了什么样的网络流程那？

把URL地址进行请求和解析

2、Chrome 浏览器上如何实现网站数据的抓包分析？

l **任务 2 描述：**

1、动态数据加载网页的优势？

2、动态数据加载网页的大体流程？

3、分析一下京东网站加载的过程（选做）。

l **任务 3 描述：**

1、HTTP 请求数据格式分为哪些部分，每个部分的数据格式要求是什么？

2、HTTP 响应数据格式分为哪些部分，每个部分的数据格式要求是什么？