准备 2 台虚拟机，配置如下：
vm1（eth1:192.168.2.20）,vm2(eth1:192.168.2.30)

vm1 创建账户 netadm,softadm,uesradm,procadm;为所有的账户设置初始密码 123456

vm1 创建账户 devel1,devel2,devel3,test1,test2,test3;为所有账户设置初始密码 passwd

参考答案

```
# for i in netadm softadm useradm    procadm
do
        useradd $i
        echo 123456 | passwd --stdin $i
done
# for i in devel1 devel2 devel3 test1 test2 test3
do
        useradd $i
        echo passwd | passwd --stdin $i
done
```

vm1 设置 test3 的账户过期时间为 2019-12-12.

参考答案：

```
#chage -E 2019-12-12 test3
```

vm1 使用 passwd 命令临时锁定 devel3 账户

参考答案：

```
#passwd -l devel3                    #注意是小写的 L 选项
```

vm1 为/etc/resolv.conf 文件添加 i 锁定属性，为/etc/hosts 添加 a 仅可追加属性

参考答案：

```
#chattr +i   /etc/resolv.conf
#chattr +a   /etc/hosts
```

vm1 用 test1 用户登陆系统，使用 su 命令切换为 test2 账户在 tmp 下创建一个文件(非交互模式)

参考答案：

```
#su - test2   -c   "touch /tmp/txt"
```

vm1 使用 root 登陆系统，设置 sudo 权限，要求如下：

vm1 让 netadm 能以 root 的身份执行网络管理的任务（参考 sudo 命令别名）

vm1 让 softadm 能以 root 的身份执行软件管理的任务

vm1 让 useradm 能以 root 的身份执行账户管理的任务（不能修改 root 密码）

vm1 让 procadm 能以 root 的身份执行进程管理任务（如杀死进程）

vm1 设置虚拟机 ssh 配置，进入 root 远程本机，设置 sshd 黑名单，禁止 test3 从任何主机远程本机

参考答案：

```
#vim /etc/sudoers
Cmnd_Alias   NETWORKING   =   /sbin/route,   /sbin/ifconfig,   /bin/ping,   /sbin/dhclient,
```

/usr/bin/net, /sbin/iptables, /usr/bin/rfcomm, /usr/bin/wvdial, /sbin/iwconfig, /sbin/mii-tool
Cmnd_Alias SOFTWARE = /bin/rpm, /usr/bin/up2date, /usr/bin/yum
Cmnd_Alias PROCESSES = /bin/nice, /bin/kill, /usr/bin/kill, /usr/bin/killall

netadm ALL=(ALL) NETWORKING
softadm ALL=(ALL) SOFTWARE
useradm ALL=(ALL) /usr/bin/passwd,!/usr/bin/passwd root,/usr/bin/user* *
procadm ALL=(ALL) PROCESSES

vm1 真实主机创建一对 ssh 密钥，让真实机可以无密码远程虚拟机，观察密钥在虚拟机中的位置

参考答案：

ssh-keygen -N '' -f /root/.ssh/id_rsa
ssh-copy-id 192.168.2.20

在 vm1 主机使用 gpg 软件对/etc/rc.d/rc.local 文件进行对称加密，并将加密文件传给 vm2

参考答案：

vm1# gpg -c /etc/rc.d/rc.local
vm1#scp /etc/rc.d/rc.local.gpg 192.168.2.30:/root/

在 vm2 对主机 vm1 传来的加密文件进行解密

参考答案：

vm2# gpg -d rc.local.gpg

在 vm1 上使用 gpg 创建非对称密钥对，并将公钥到处传给 vm2

参考答案：

vm1# gpg --gen-key
vm1# gpg --export 密钥 > my.key
vm2# scp my.key 192.168.2.30:/root/

在 vm2 主机将 vm1 传过来的公钥导入，并使用公钥对/etc/sysctl.conf 文件加密，并将加密文件传给 vm1，在 vm1 主机使用自己的私钥解密该文件

参考答案：

vm2# gpg --import my.key
vm2# gpg -e -r 密钥 /etc/sysctl.conf
vm2# scp /etc/sysctl.conf.gp 192.168.2.20:/root/
vm1# gpg -d /etc/sysctl.conf.gpg

在 vm1 主机使用私钥给文件/etc/sysctl.conf 文件签名，在 vm2 主机验证签名

参考答案：

```
vm1# gpg -b /etc/sysctl.conf
vm1#scp /etc/sysctl.con*  192.168.2.30:/root
vm2#gpg –verity sysctl.conf.sig sysctl.conf
```

使用 aide 软件对/bin/和/sbin/目录进行入侵检测

```
# yum -y install aide
#vim /etc/aide.conf
/bin/   DATAONLY
/sbin/ DATAONLY
#aide --init
#cp   aide.db.new.gz   aide.db.gz
#aide --check
```

在 vm2 上安装 nginx,vsftpd,mariadb,mariadb-server,并启动所有对应的服务

```
vm2# yum -y install gcc pcre-devel openssl-devel
vm2# tar -xf nginx-1.12.2.tar.gz
vm2# cd nginx-1.12.2
vm2# ./configure
vm2# make && make install
vm2# yum -y install vsftpd   mariadb   mariadb-server
vm2# /usr/local/nginx/sbin/nginx
vm2# systemctl start vsftpd
vm2# systemctl start mariadb
```

在 vm1 上使用 nmap 扫描 vm2 主机的所有 TCP 服务

参考答案：

```
vm1# nmap -sT 192.168.2.30
vm1# namp -sS 192.168.2.30
```

在 vm2 上配置 nginx 用户认证，并使用 tcpdump 抓取 80 端口相关的数据包，注意默认抓取的是第一个网卡的数据，抓取其他网卡可以使用-i 选项

参考答案：

```
vm2# vim /usr/local/nginx/conf/nginx.conf
……
server {
    listen 80;
    server_name localhost;
auth_basic "xx";
auth_basic_user_file "/usr/local/nginx/pass";
… …
```

```
vm2# /usr/local/nginx/sbin/nginx -s reload
vm2# yum -y install httpd-tools
vm2# htpasswd -c /usr/local/nginx/pass tom
>123456
vm2# tcpdump -A -i   eth1 tcp port 80
```

在 vm1 上使用 firefox 访问 vm2 的页面，输入账户与密码，到 vm2 观察数据包

参考答案：

```
vm1# firefox http://192.168.2.30                    （注意远程 SSH 需要使用-X 选项）
```

在 vm2 主机，使用 limit 模块对 nginx 限制并发，限制并发数量为 10，burst 为 10

参考答案：

```
vm2# vim /usr/local/nginx/conf/nginx.conf
    limit_req_zone $binary_remote_addr zone=one:10m rate=1r/s;
    server {
        listen 80;
        server_name localhost;
        limit_req zone=one burst=5;
            }
```

在 vm2 主机，设置 nginx 拒绝所有非 POST 或 GET 的请求

参考答案：

```
vm2# vim /usr/local/nginx/conf/nginx.conf
        server {
        ... ...
        listen 80;
                if ($request_method !~ ^(GET|POST)$ ) {
                        return 444;
                }
        ... ...
         }
```

在 vm2 主机，设置 nginx 防止 buffer 数据溢出

参考答案：

```
vm2# vim /usr/local/nginx/conf/nginx.conf
http{
client_body_buffer_size   1K;
client_header_buffer_size 1k;
client_max_body_size 1k;
large_client_header_buffers 2 1k;
 ... ...
}
```

在 vm2 主机登陆 mariadb 服务器，创建一个可以从远程登陆的数据库账户

参考答案：

```
vm2# mysql
>grant all on *.* to tom@'%' identified by '123456'
>exit
```

在 vm2 主机使用 tcpdump 对 3306 进行抓包，在 vm1 连接 vm2 的数据库，进行查询操作，回到 vm2 观察抓取的数据包信息

参考答案：

```
vm2# tcpdump -A tcp port 3306
vm1# yum -y install mariadb
vm1# mysql -h192.168.2.30 -utom -p123456
vm1#select * from mysq.user
```

在 vm2 安装 tomcat，并以 tomcat 身份降级启动 tomcat 服务

参考答案：

```
vm2# tar -xf apache-tomcat-8.0.30.tar.gz
vm2# mv apache-tomcat-8.0.30 /usr/local/tomcat
vm2# useradd tomcat
vm2# chown tomcat.tomcat   /usr/local/tomcat
vm2# su - tomcat /usr/local/tomcat/bin/startup.sh
```

在 vm2 主机执行如下命令:

```
mkdir -p /root/{source1.0,surce2.0}/test/
echo “hehe” > /root/source1.0/test.conf
echo “haha” > /root/source2.0/test.conf
echo “hello” > /root/source1.0/test/hello.sh
echo “hello world” > /root/source2.0/test/hello.sh
cp /bin/find   /root/source1.0/
cp /bin/find   /root/source2.0/
echo “xyz” >> /root/source2.0/find
```

在 vm2 主机对/root/source1.0 和/root/source2.0 生成补丁文件，并使用 patch 工具对 source1 目录下的所有代码打补丁

参考答案：

```
vm2# diff -Nura   source1.0   source2.0   > patch.txt
vm2# yum -y install patch
vm2# cd source1.0
vm2# patch -p1   < ../patch.txt
```

准备 2 台虚拟机，配置如下：

vm1(eth1:192.168.2.20，设置网关为 192.168.2.30)

vm2(eth1:192.168.2.30,eth2:201.1.1.30)，开启路由转发

vm3(eth2: 201.1.1.40,设置网关为 201.1.1.30)

在 vm2 主机设置防火墙规则：

禁止任何其他主机 ping 本机，但本机可以 ping 其他主机

禁止 192.168.2.20 通过 ssh 远程本机

通过 mac 地址禁止 vm3 访问本机的 ftp 服务

通过一条规则设置允许访问本机的 80,3306,25 端口

禁止任何主机通过本机的 eth2 网卡访问本机的 53 端口

禁止 vm2 转发任何与 ftp 有关的数据包

设置 SNAT 规则，来源于 192.168.2.0 网络的数据包，修改源地址为 201.1.1.30

参考答案：

```
vm2# iptables -I INPUT -p icmp --icmp-type echo-request -j DROP
vm2# iptables -I INPUT -p tcp –dport 22 -s 192.168.2.20 -j REJECT
vm2# iptables -I INPUT -m mac --source-mac vm1 的 mac 地址   -j REJECT
vm2# iptables -I INPUT -p tcp -m multiport --dports 80,3306,25 -j ACCEPT
vm2# iptables -I INPUT -i eth2 -p tcp --dport 53   -j REJECT
vm2# iptables -I INPUT -i eth2 -p udp --dport 53   -j REJECT
vm2# iptables -I FORWARD -p tcp --dport 22 -j REJECT
vm2# iptables -t nat -s 192.168.2.0/24 -j SNAT --to-source 201.1.1.30
```