**实验1：Linux Firewall Exploration Lab**

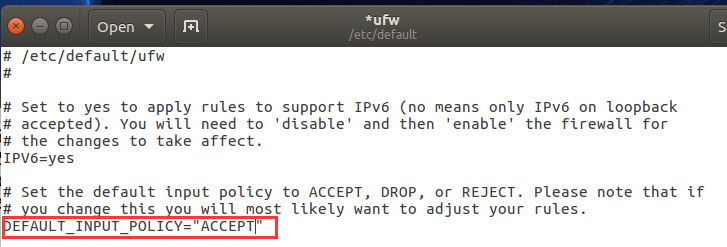
## Task 1: Using Firewall

主机A：10.0.2.8；主机B：10.0.2.11

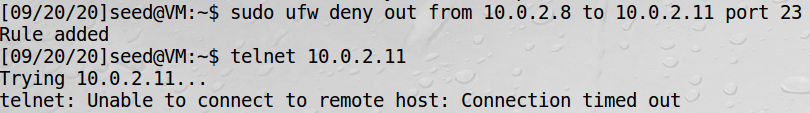
我们在主机A上部署ufw







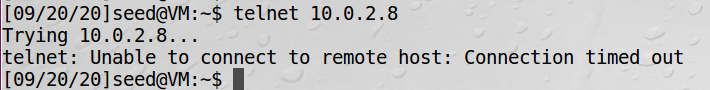
① 阻止主机A telnet主机B



拦截成功

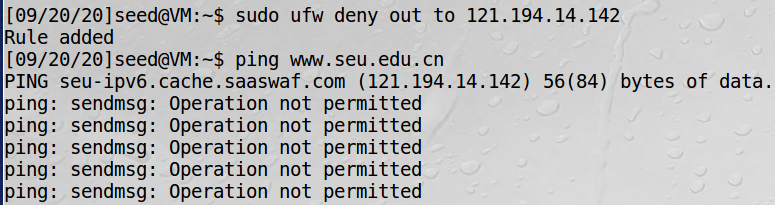
② 阻止主机B telnet主机A





拦截成功

③ 阻止A访问一个外部网站



## Task 2: Implementing a Simple Firewall

① 过滤A telnet B

// filter 1: A telnet B

     if (dst\_port == 23){

        print\_address(ip\_header); // print SRC and DST

        if (!check\_address\_src(ip\_header, 10, 0, 2, 8){

           printk(KERN\_INFO "filter 1: src not match\n");

           return NF\_ACCEPT;

        }

        if (!check\_address\_dst(ip\_header, 10, 0, 2, 11)){

           printk(KERN\_INFO "filter 1: dst not match\n");

           return NF\_ACCEPT;

        }

        printk(KERN\_INFO "filter 1: A telnet B\n");

printk(KERN\_INFO "filter 1: SRC\_PORT: %d DST\_PORT: %d\n", src\_port, dst\_port);

        return NF\_DROP;

    }

② 过滤B telnet A

// filter 2: B telnet A

     if (dst\_port == 23){

        print\_address(ip\_header);

        if (!check\_address\_src(ip\_header, 10, 0, 2, 11)){

           printk(KERN\_INFO "filter 2: src not match\n");

           return NF\_ACCEPT;

        }

        if (!check\_address\_dst(ip\_header, 10, 0, 2, 8)){

           printk(KERN\_INFO "filter 2: dst not match\n");

           return NF\_ACCEPT;

        }

        printk(KERN\_INFO "filter 2: B telnet A\n");

        printk(KERN\_INFO "filter 2: SRC\_PORT: %d DST\_PORT: %d\n", src\_port, dst\_port);

        return NF\_DROP;

    }

③ 过滤A访问www.seu.edu.cn

// filter 3: A http www.seu.edu.cn

     if (dst\_port == 80){

        print\_address(ip\_header);

        if (!check\_address\_src(ip\_header, 10, 0, 2, 8)){

           printk(KERN\_INFO "filter 3: src not match\n");

           return NF\_ACCEPT;

        }

        if (!check\_address\_dst(ip\_header, 121, 194, 14, 142)){

           printk(KERN\_INFO "filter 3: dst not match\n");

           return NF\_ACCEPT;

        }

        printk(KERN\_INFO "filter 3: A http fudan.edu.cn\n");

        printk(KERN\_INFO "filter 3: SRC\_PORT: %d DST\_PORT: %d\n", src\_port, dst\_port);

        return NF\_DROP;

    }

④ 过滤A ping B

// filter 4: A ping B

     if (icmp\_header->type == 8){

        print\_address(ip\_header);

        if (!check\_address\_src(ip\_header, 10, 0, 2, 8)){

           printk(KERN\_INFO "filter 4: src not match\n");

           return NF\_ACCEPT;

        }

        if (!check\_address\_dst(ip\_header, 10, 0, 2, 11)){

           printk(KERN\_INFO "filter 4: dst not match\n");

           return NF\_ACCEPT;

        }

        printk(KERN\_INFO "filter 4: A ping B\n");

        printk(KERN\_INFO "filter 4: SRC\_PORT: %d DST\_PORT: %d\n", src\_port, dst\_port);

        return NF\_DROP;

    }

⑤ 过滤A ssh B

// filter 5: A ssh B

     if (dst\_port == 22){

        print\_address(ip\_header);

        if (!check\_address\_src(ip\_header, 10, 0, 2, 8)){

           printk(KERN\_INFO "filter 5: src not match\n");

           return NF\_ACCEPT;

        }

        if (!check\_address\_dst(ip\_header, 10, 0, 2, 11)){

           printk(KERN\_INFO "filter 5: dst not match\n");

           return NF\_ACCEPT;

        }

        printk(KERN\_INFO "filter 5: A ssh B\n");

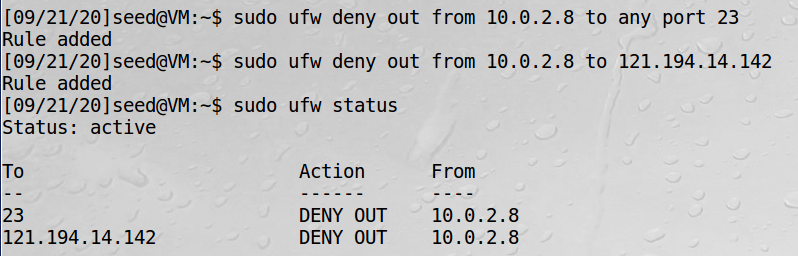
        printk(KERN\_INFO "filter 5: SRC\_PORT: %d DST\_PORT: %d\n", src\_port, dst\_port);

        return NF\_DROP;

    }

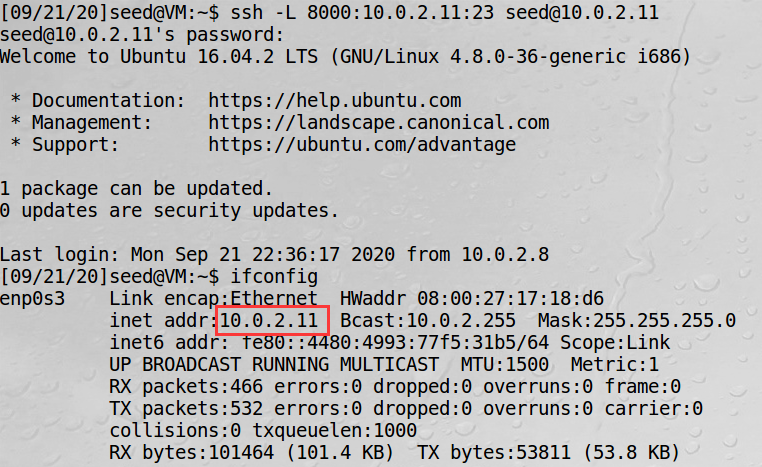
## Task 3: Evading Egress Filtering

设置过滤器

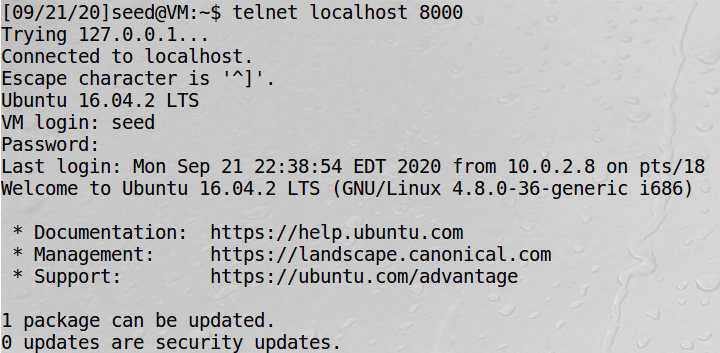


## Task 3.a: Telnet to Machine B through the ﬁrewall

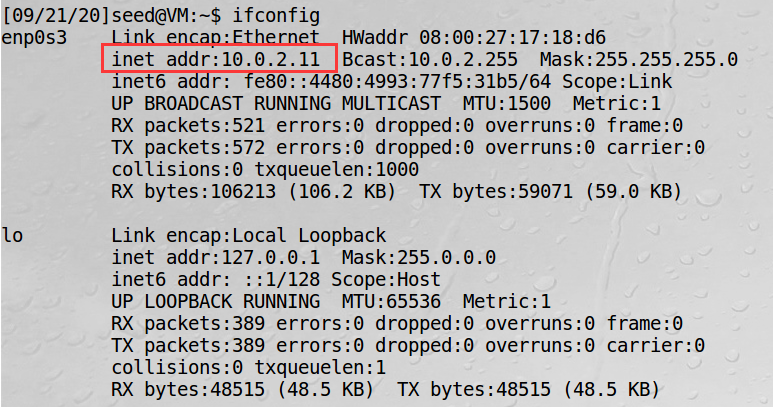
在A和B间建立ssh连接



在新终端中连接8000端口

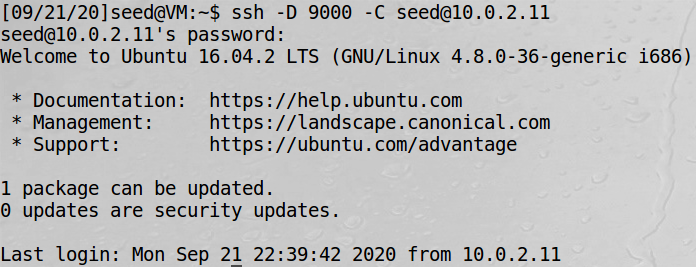


可以发现我们成功的绕过了防火墙

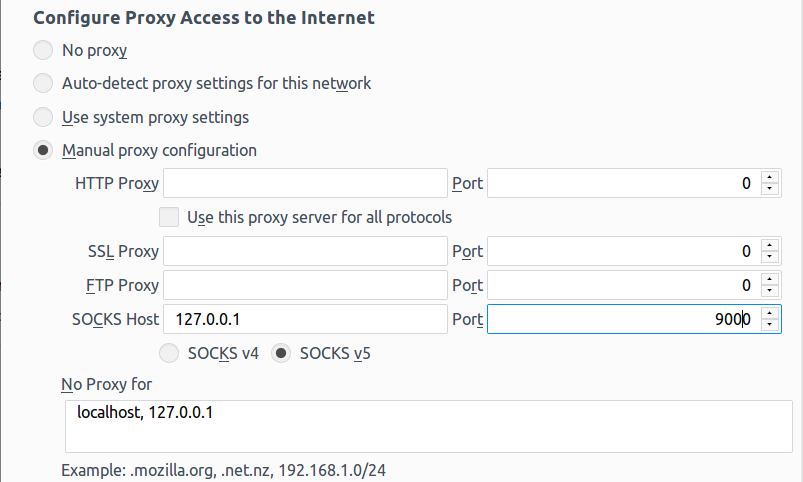


## Task 3.b: Connect to Facebook using SSH Tunnel

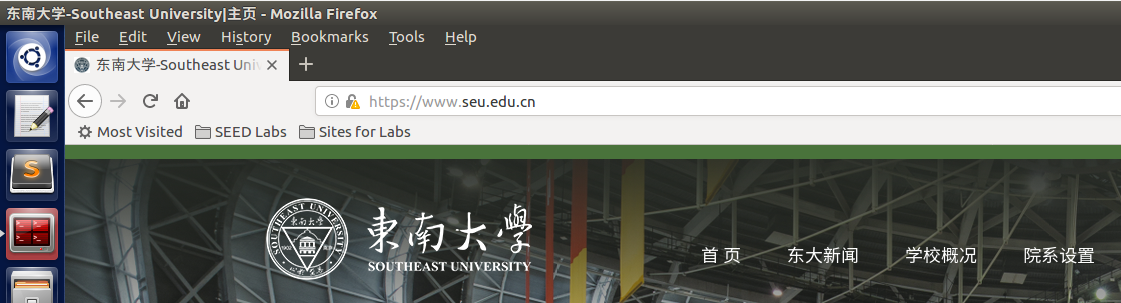
首先通过ssh连接到B



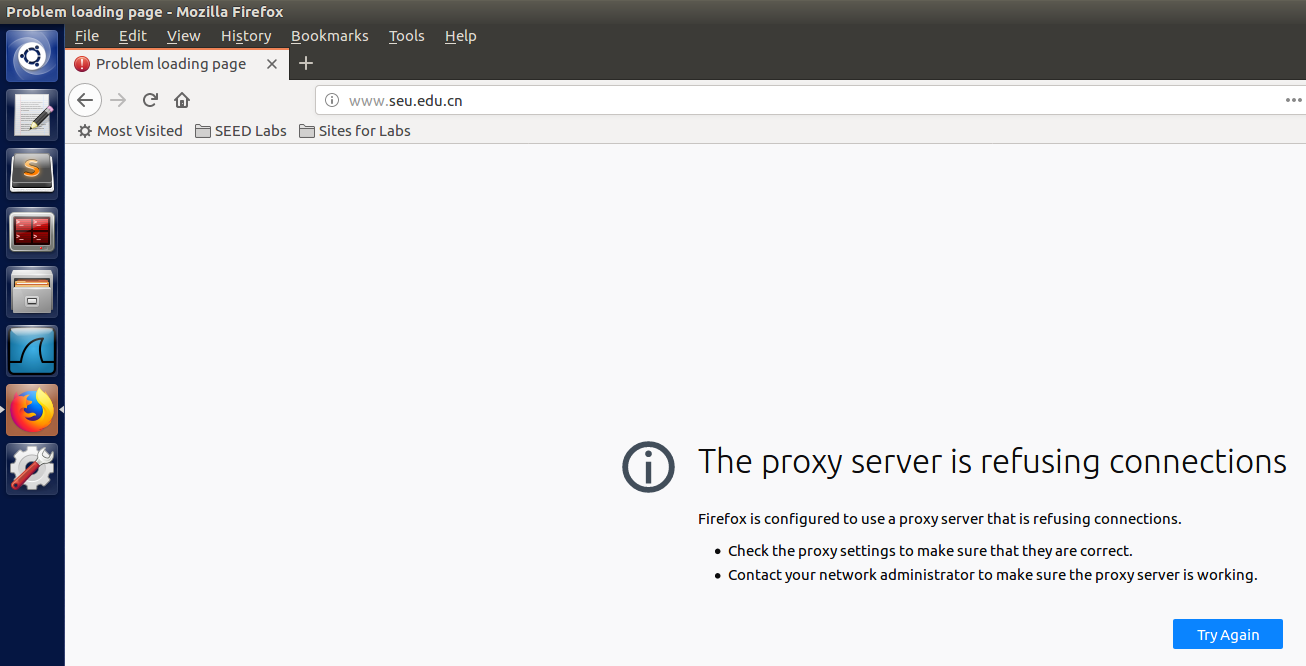
firefox设置代理



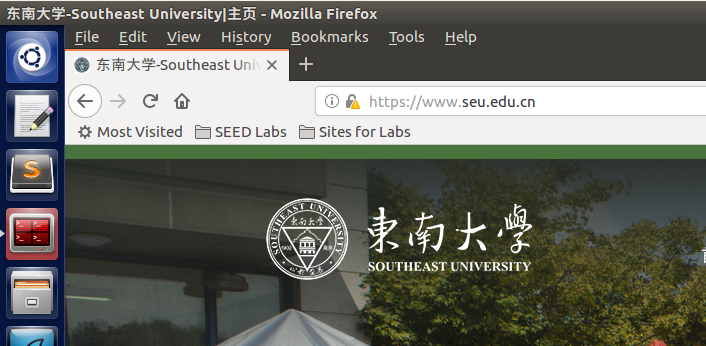
可以访问东南大学主页



断开ssh连接后，无法访问主页



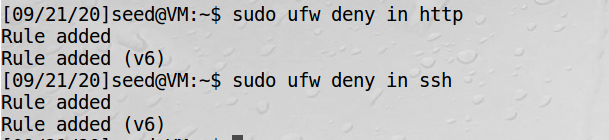
恢复ssh连接后，又可以访问主页



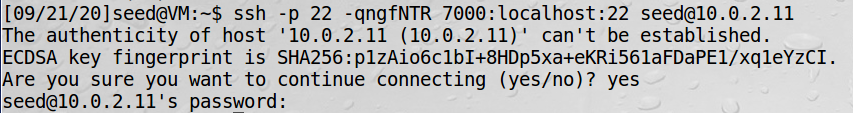
当A与B建立ssh连接时，B作为A的代理访问外界网页，将返回的数据传给A，这样就可以绕过防火墙；当A与B断开连接后失去代理，因此无法成功访问

## Task 4: Evading Ingress Filtering

首先，配置A上的ufw，禁止外界对A的http和ssh连接



在A上与B建立反向ssh连接



B可以通过反向ssh连接到A

