Министерство образования и науки Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО

Кафедра дискретной математики и
информационных технологий

Организация безопасности корпоративной сети с использованием операционной системы Linux.

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

Студента 4 курса 421 группы
направления 09.03.01 – Информатика и вычислительная техника
факультета КНиИТ
Краснобельмова Вячеслава Алексеевича

Научный руководитель _____ В.А Поздняков
доцент, к.ф.-м.н., доцент

Заведующий кафедрой _____ Л.Б. Тяпаев
доцент, к.ф.-м.н., доцент

Саратов 2017

Безопасность сети важна и актуальна, поскольку на сегодняшний день существует много угроз данным и информации.

Целью бакалаврской работы является организация безопасности корпоративной сети с использованием операционной системы Linux. Для достижения этой цели были сформулированы следующие задачи:

1. Анализ и выбор топологии;
2. Анализ и выбор технологии сети;
3. Анализ и выбор средств безопасности для Linux;
4. Практическая реализация.

Для того, чтобы соединить ПК нашей ЛС нам нужно выбрать топологию подключения компьютеров. Термин «топология», или «топология сети», характеризует физическое расположение компьютеров, кабелей и других компонентов сети. Топология сети обслуживает ее характеристики. В частности, выбор той или иной топологии влияет:

- на состав сетевого оборудования;
- характеристики сетевого оборудования;
- возможности расширения сети;
- способ управления сетью.

Чтобы совместно использовать ресурсы или выполнять другие сетевые задачи, компьютеры должны быть подключенные друг к другу. Для этой цели в большинстве используется кабель.

Однако просто подключить компьютер к кабелю, который соединяет другие компьютеры, не достаточно. Разные типы кабелей в сочетании с разной сетевой платой, сетевыми операционными системами и другими компонентами требуют и разного взаимного расположения компьютеров. Все сети строятся на базовых топологий: Шина, Кольцо, Звезда.

Для того, чтобы выбрать нужную топологию, было проанализировано 3 основных вида, такие как звезда, кольцо и шина. Была выбрана топология звезда, поскольку она имеет больше плюсов над всеми другими топологиями. Например, возьмем топологию шина, в случаи выхода из строя одного фрагмента кабеля, из строя выйдет вся сеть предприятия. Но если сеть относительно большая, то обнаружить такой фрагмент будет достаточно трудно. При этом нельзя в сеть ставить больше 10 ПК, если ставить больше - скорость значительно упадет. В топологии звезда при выходе из строя одного фрагмента, с сетью ничего не случится.

Самым популярным стандартом физического уровня сетей Fast Ethernet, есть 100BASE-Tx, используя в качестве физической среды передачи данных кабель UTP-5 категории, и 100BASE-Fx, использующий многомодовое оптоволокно.

На эффективность работы локальных сетей Ethernet негативно влияют следующие факторы:

-широковещательный характер передачи кадров данных;

-увеличение задержки распространения кадров при использовании сетевых устройств;

Для корпоративной сети были избраны две технологии: Fast Ethernet и Gigabit Ethernet(1000BASE-T). Они будут реализовываться на витой паре категории 5E.

Для настройки сетевого экрана придерживаемся следующей политики: запретить все, а потом то, что нужно разрешить. Так и поступим в данном случае. Правила будут иметь примерно следующую картину:

```
netfilter:~# iptables -L
Chain INPUT (policy ACCEPT)target prot opt source destination
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
```

Это значит, что политика по умолчанию для таблицы filter во всех цепочках - ACCEPT и нет никаких других правил, что-либо запрещающих. Поэтому давайте сначала запретим входящие, исходящие и проходящие пакеты (не вздумайте это делать удаленно-тут же потеряете доступ):

```
netfilter:~# iptables -P INPUT DROP
netfilter:~# iptables -P OUTPUT DROP
netfilter:~# iptables -P FORWARD DROP
```

Этими командами мы устанавливаем политику DROP по умолчанию. Это значит, что любой пакет, для которого явно не задано правило, которое его

разрешает, автоматически отбрасывается. Поскольку пока еще у нас не задано ни одно правило - будут отвергнуты все пакеты, которые придут на ваш компьютер, равно как и те, которые вы попытаетесь отправить в сеть.

В первую очередь в обязательно разрешим передачу пакетов через входящий петлевой интерфейс и исходящий петлевой интерфейс в таблицах INPUT (для возможности получения отправленных пакетов) и OUTPUT (для возможности отправки пакетов) соответственно. Итак, обязательно выполняем:

```
netfilter:~# iptables -A INPUT -i lo -j ACCEPT
netfilter:~# iptables -A OUTPUT -o lo -j ACCEPT
```

После этого пинг на локалхост заработает:

```
netfilter:~# ping -c1 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1 (127.0.0.1): icmp_seq=1 ttl=64 time=0.116 ms
--- 127.0.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 116ms
rtt min/avg/max/mdev = 0.116/0.116/0.116/0.116 ms
```

Данная команда разрешит типы ICMP пакета эхо-запрос и эхо-ответ, что повысит безопасность.

iptables:

```
netfilter:~# iptables -A INPUT -p icmp --icmp-type 0 -j ACCEPT
netfilter:~# iptables -A INPUT -p icmp --icmp-type 8 -j ACCEPT
netfilter:~# iptables -A OUTPUT -p icmp -j ACCEPT
```

Зная, что наш компьютер не заражен и он устанавливает только безопасные исходящие соединения, а также, зная, что безопасные соединения - это соединения из так называемого эфимерного диапазона портов, который задается ядром в файле /proc/sys/net/ipv4/ip_local_port_range, можно разрешить исходящие соединения с этих безопасных портов:

```
netfilter:~# cat /proc/sys/net/ipv4/ip_local_port_range
32768 61000
netfilter:~# iptables -A OUTPUT -p TCP --sport 32768:61000 -j ACCEPT
```

```
netfilter:~# iptables -A OUTPUT -p UDP --sport 32768:61000 -j ACCEPT
```

Разрешаем попадание на наш компьютер только тех TCP- и UDP-пакетов, которые были запрошены локальными приложениями:

```
netfilter:~# iptables -A INPUT -p TCP -m state --state ESTABLISHED,RELATED -j ACCEPT
netfilter:~# iptables -A INPUT -p UDP -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Для работы ssh-сервера, который принимает и отправляет запросы на 22 TCP-порту, необходимо добавить следующие iptables-правила:

```
netfilter:~# iptables -A INPUT -i eth0 -p TCP --dport 22 -j ACCEPT
netfilter:~# iptables -A OUTPUT -o eth0 -p TCP --sport 22 -j ACCEPT
```

Т.е. для любого сервиса нужно добавить по одному правилу в цепочки INPUT и OUTPUT, разрешающему соответственно прием и отправку пакетов с использованием этого порта для конкретного сетевого интерфейса (если интерфейс не указывать, то будет разрешено принимать/отправлять пакеты по любому интерфейсу).

Настройка netfilter/iptables для подключения нескольких клиентов к одному соединению.

Теперь рассмотрим наш Linux в качестве шлюза для локальной сети во внешнюю сеть Internet. Предположим, что интерфейс eth0 подключен к интернету и имеет IP 198.166.0.200, а интерфейс eth1 подключен к локальной сети и имеет IP 10.0.0.1. По умолчанию, в ядре Linux пересылка пакетов через цепочку FORWARD (пакетов, не предназначенных локальной системе) отключена. Чтобы включить данную функцию, необходимо задать значение 1 в файле `/proc/sys/net/ipv4/ip_forward`: `netfilter:~# echo 1 > /proc/sys/net/ipv4/ip_forward`

Чтобы форвардинг пакетов сохранился после перезагрузки, необходимо в файле /etc/sysctl.conf раскомментировать (или просто добавить) строку `net.ipv4.ip_forward=1`.

Необходимо подменить адрес отправителя, на внешний адрес шлюза Linux. Это достигается за счет действия MASQUERADE (маскарадинг) в цепочке POSTROUTING, в таблице nat.

```
netfilter:~# iptables -A FORWARD -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
netfilter:~# iptables -A FORWARD -m conntrack --ctstate NEW -i eth1 -s 10.0.0.1/24 -j ACCEPT
netfilter:~# iptables -P FORWARD DROP
netfilter:~# iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

Необходимо добавить запрещающие и разрешающие правила для входящих и исходящих соединений:

```
netfilter:~# iptables -P INPUT DROP
netfilter:~# iptables -P OUTPUT DROP
netfilter:~# iptables -A INPUT -i lo -j ACCEPT
netfilter:~# iptables -A OUTPUT -o lo -j ACCEPT
netfilter:~# iptables -A INPUT -p icmp --icmp-type 0 -j ACCEPT
netfilter:~# iptables -A INPUT -p icmp --icmp-type 8 -j ACCEPT
netfilter:~# iptables -A OUTPUT -p icmp -j ACCEPT
netfilter:~# cat /proc/sys/net/ipv4/ip_local_port_range
32768 61000
netfilter:~# iptables -A OUTPUT -p TCP --sport 32768:61000 -j ACCEPT
netfilter:~# iptables -A OUTPUT -p UDP --sport 32768:61000 -j ACCEPT
netfilter:~# iptables -A INPUT -p TCP -m state --state ESTABLISHED,RELATED -j ACCEPT
netfilter:~# iptables -A INPUT -p UDP -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Примечание: желательно негласно принято, перед всеми командами iptables очищать цепочки, в которые будут добавляться правила:

netfilter:~# iptables -F ИМЯ_ЦЕПОЧКИ

Предоставление доступа к сервисам на шлюзе.

Предоставление доступа к сервису будет осуществляться с помощью следующих разрешающих правил:

```
netfilter:~# iptables -A INPUT -p TCP --dport nn -j ACCEPT
netfilter:~# iptables -A OUTPUT -p TCP --sport nn -j ACCEPT
```

Данные правила разрешают входящие соединения по протоколу tcp на порт nn и исходящие соединения по протоколу tcp с порта nn. Кроме этого, можно добавить дополнительные ограничивающие параметры, например разрешить входящие соединения только с внешнего интерфейса eth0 (ключ -i eth0) и т.п.

.

Предоставление доступа к сервису будет осуществляться с помощью следующих разрешающих правил:

```
netfilter:~# iptables -t nat -A PREROUTING -p tcp -d 198.166.0.200
--dport xxx -j DNAT --to-destination X.Y.Z.1
netfilter:~# iptables -A FORWARD -i eth0 -p tcp -d X.Y.Z.1 --dport
xxx -j ACCEPT
```

Сохранение введенных правил при перезагрузке.

Настраиваем сетевой экран под свои нужды с помощью команды iptables

Создаем дамп созданный правил с помощью команды iptables-save > /etc/iptables.rules

Создаем скрипт импорта созданного дампа при старте сети (в каталоге /etc/network/if-up.d/) и не забываем его сделать исполняемым:

```
# cat /etc/network/if-up.d/firewall
#!/bin/bash
/sbin/iptables-restore < /etc/iptables.rules
exit 0
# chmod +x /etc/network/if-up.d/firewall
```

Для начала настроим ядро. В файле /etc/sysctl.conf хранятся настройки ядра, которые загружаются и применяются во время запуска системы . Необходимо:

- Включить защиту от переполнения буфера exec shield:
- `kernel.exec-shield=1`
  `kernel.randomize_va_space=1`
- Включить защиту от подделывания IP:
- `net.ipv4.conf.all.rp_filter=1`

- Отключить перенаправление IP адресов:
- `net.ipv4.conf.all.accept_source_route=0`
- Игнорировать широковещательные запросы:
- `net.ipv4.icmp_echo_ignore_broadcasts=1`
  `net.ipv4.icmp_ignore_bogus_error_messages=1`
- Логгировать все поддельные пакеты:
- `net.ipv4.conf.all.log_martians = 1`

Далее отключим все неиспользуемые SUID и GUID файлов, все исполняемые файлы, для которых включен флаг SUID или SGID потенциально опасны. Этот флаг означает, что программа будет выполняться с правами суперпользователя. А это означает, что если в программе есть какая-нибудь уязвимость или баг, то локальный или удаленный пользователь сможет использовать этот файл. Ищем все такие файлы с помощью следующей команды :

```
$ find / -perm +4000
```

Далее необходимо найти файлы с установленным SGID флагом:

```
$ find / -perm +2000
```

Или скомбинируем все это в одной команде:

```
$ find / \( -perm -4000 -o -perm -2000 \) -print
$ find / -path -prune -o -type f -perm +6000 -ls
```

Подробно изучаем каждый найденный файл, чтобы понять насколько нужен тот или иной файл.

Следом необходимо удалить все общедоступные файлы, которые могут изменять все пользователи, найти их можно так :

```
$ find /dir -xdev -type d \( -perm -0002 -a ! -perm -1000 \) -
print
```

Далее удаляем x-сервер, и все что с ним связано этой командой:

```
 $ sudo apt remove —purge xserver-xorg
```

Также делаем блокировку пользователей за превышение лимита неправильно введенных пар логин/пароль : $ faillog -r -u пользователь [20].

Отключаем все ненужные сервисы и обновления, отключаем вход для суперпользователя, настраиваем pam_cracklib.so, чтобы обеспечить соблюдение

политики паролей, и настраиваем iptables.

ЗАКЛЮЧЕНИЕ

Поставленная передо мной цель по организации безопасности корпоративной сети с использованием операционной системы Linux была успешно достигнута. Для достижения этой цели были выполнены следующие задачи:

1. Анализ и выбор топологии;
2. Анализ и выбор технологии сети;
3. Анализ и выбор средств безопасности для Linux;
4. Практическая реализация.

Таким образом, все поставленные задачи выполнены, цель бакалаврской работы достигнута.