С.И. Богомолов

Сети ЭВМ и средства коммуникаций

Лабораторный практикум №4

Министерство образования и науки РФ
Томский университет систем управления и радиоэлектроники

Радиотехнический факультет
Кафедра телекоммуникаций и основ радиотехники

«Утверждаю»
Зав. кафедрой ТОР
А.В Пуговкин
_____ 2010 г.

Лабораторный практикум №4
по дисциплине
«Сети ЭВМ и средства коммуникаций»
для студентов факультета вычислительных систем

Лабораторный практикум составил:
к.т.н., доцент С.И. Богомолов

Томск - 2010 г.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Лабораторный практикум по дисциплине «Программные средства систем связи» предназначен для закрепления теоретических сведений и получения практических навыков работы с модулями верхних уровней эталонной модели взаимодействия открытых систем на примере исследования отдельных характеристик стека протоколов TCP/IP. Задачи, решаемые студентом в ходе выполнения данного практикума, затрагивают следующие аспекты сетевого взаимодействия:

- диагностика сети;
- удаленный доступ.

В ходе выполнения лабораторной работы №1 «Элементы диагностики сети» исследуется функционирование основных сетевых утилит операционной системы семейства Linux, используемых для получения служебной информации о состоянии сетевых ресурсов на сетевом, транспортном и прикладном уровнях.

В рамках решения второй задачи выполняется лабораторная работа №2 "Удаленный доступ", в которой раскрываются основные черты удаленного доступа.

Лабораторные работы выполняются в режиме командной строки операционной системы Linux. Общие сведения о системе Linux представлены на сервере S локальной сети и доступны по адресу system:/home/server_S/1_Курсы Linux. Более подробную информацию о каждой из команд системы Linux можно получить непосредственно из справочной службы операционной системы. Так, в режиме командной строки эту информацию можно получить, набрав в командной строке запросы вида:

$ info –h

либо

$ man -h,

где символ $ означает приглашение командной строки.

Некоторые сведения о командах info и man приведены ниже. Опции можно набирать полным названием либо в сокращенном виде. Не забывайте, что при наборе команд имеет значение состояние регистра.

Команда **info** позволяет просматривать информацию в формате Info

Синтаксис команды:

info [*option*]... [*menu-item*...]

Здесь и далее курсивом выделены пункты команды, которые могут принимать более, чем одно значение (альтернативные аргументы команды, либо имена в текстовом или числовом формате и т.п.).

Ключи (опции):

--apropos=*subject*

- искать указанную тему *subject* во всех алфавитных указателях всех руководств.

-d, --directory=*dir*

- добавить указанный каталог *dir* к справочной системе (переменная INFOPATH).

--dribble=*filename*

- записать вводимые пользователем символы (последовательность нажимаемых клавиш) в файл *filename*.

-f, --file=*filename*

- показать указанный файл *filename*.

-h, --help

- показать эту справку и затем выйти.

--index-search=*string*

- перейти к разделу, на который ссылается заданный строкой *string* именной указатель.

-n, --node=*nodename*

- начать просмотр с заданного раздела *nodename*.

-o, --output=*filename*

- записать выбранные разделы в файл *filename*.

-R, --raw-escapes

- выводить escape-последовательности ANSI (по умолчанию).

--no-raw-escapes

- выводить escape-последовательности как буквенный текст.

--restore=*filename*

- прочитать последовательность нажатых клавиш из файла *filename*.

-O, --show-options, --usage

- перейти к разделу, описывающему опции (ключи) командной строки.

--subnodes

- рекурсивно выводить пункты меню.

--version

- показать информацию о версии и выйти.

Первый не являющийся ключом аргумент, если он есть, задает имя начального пункта меню; он ищется во всех файлах `dir' в INFOPATH. Если он не найден, info объединяет все файлы `dir' и показывает результат. Все остальные аргументы рассматриваются как имена пунктов меню относительно первого посещенного раздела.

Примеры:

info

- показать меню каталога верхнего уровня.

info *emacs*

- начать с раздела emacs из каталога верхнего уровня.

info *emacs buffers*

- начать с раздела buffers в руководстве emacs.

info --show-options *emacs*

- начать с раздела, описывающего ключи emacs.

info -f *./foo.info*

- показать файл *./foo.info*, не искать dir.

Команда **man** выводит страницы руководства по операционной системе Linux.

Синтаксис команды:

man [-adfhktwW] [*section*] [-M *path*] [-P *pager*] [-S *list*] [-m *system*] [-p *string*] *name* ...

Ключи (опции):

-a

- найти все совпадающие страницы руководства для указанной команды name (описание ряда команд находится в нескольких разделах руководства);

-c

- не использовать cat-файл (форматировать страницу руководства повторно);

-d

- выводить отладочную информацию; (при -D отображать страниц руководства);

-f

- то же самое, что и команда whatis;

-h

- вывести эту справку;

-k

- то же самое, что и команда apropos;

-K

- выполнить поиск команды на всех страницах руководства;

-t

- использовать troff, чтобы форматировать страницы для печати;

-w

- вывести расположение страниц руководства, которые будут отображены; (если название не указано, то будет выведены все каталоги поиска);

-W

- как для -w, но отображать только имена файлов.

-M *path*

- установить заданный в *path* маршрут поиска страниц руководства.

-P *pager*

- использовать указанную программу *pager* для отображения страниц руководства.

-S *list*

- разделенный двоеточиями список разделов для поиска.

-C *file*

- использовать указанный файл *file* в качестве конфигурационного.

-m *system*

- поиск страниц руководства альтернативной системы.

-p *string*

- строка, указывающая, какому препроцессор стартовать. Возможны сокращения:

e - [n]eqn;	p – pic;	t – tbl;
g – grap;	r – refer;	v – vgrind.

ЛАБОРАТОРНАЯ РАБОТА №1. ЭЛЕМЕНТЫ ДИАГНОСТИКИ СЕТИ

Цель работы

Целью данной работы является знакомство с командами операционной системы Linux, обеспечивающими диагностику сетевых компонентов, а также получение доступной рядовому пользователю (не обладающему правами root – администратора) информации о параметрах сети с помощью этих утилит.

Совместное функционирование сетевых компонентов обеспечивает сложный аппаратно-программный комплекс, как каждого отдельного узла, так и сети в целом. Для поддержания соответствующего качества работы сети используется мощный набор средств, отслеживающих как изменения топологии сети, так и изменения режимов работы узлов и отдельных программных модулей, связанные с неравномерным характером нагрузки. Кроме того, прикладным процессам доступен набор средств, обеспечивающих получение информации о сетевых компонентах.

Информация, выводимая в результате выполнения утилит

Информация, отображаемая утилитой **host**

HEADER
ЗАГОЛОВОК (поля выходных данных).
opcode
- код операции (например, запрос - QUERY).
status
- состояние (например, без ошибок - NOERROR).
id
- идентификатор.
flags
- флаги (например: qr, aa, rd, ra; соответственно: QUERY, ANSWER, AUTHORITY, ADDITIONAL).
QUESTION (ANSWER, AUTHORITY) SECTION
СЕКЦИЯ ОПРОСА (ОТВЕТА, ПОЛНОМОЧИЙ).
IN
- класс адресов - IP.
Типы запросов:
A
- преобразование имени в адрес (address) узла.
AAAA
- полное имя узла.
ANY
- отображение всех ресурсных записей.
CNAME
- псевдоним узла (canon name).
HINFO
- информация об аппаратном обеспечении узла.

MX
- информация о почтовом сервере (mail exchanger).
NS
- информация о сервере DNS.
PTR
- указатель преобразования IP адреса в имя узла.
SOA
- начало полномочий (Start of Authority).

Информация, отображаемая утилитой **netstat**

OUTPUT

ВЫВОД (поля выходных данных):

Active Internet connections (TCP, UDP, raw)

Активные соединения интернет (tcp, udp, raw):

Proto
- протокол (tcp, udp, raw) используемый сокетом.
Recv-Q
- количество байт, не скопированных пользовательской программой, подключенной к этому сокету.
Send-Q
- количество байт, не подтвержденных удаленным узлом.
Local Address
- адрес и номер порта сокета местного окончания. Если не указана опция --numeric (-n) — адрес сокета определен как каноническое имя узла (FQDN), а номер порта переведен в имя соответствующей службы.
Foreign Address
- адрес и номер порта сокета удаленного окончания. Аналогично с Local Address.
State
- состояние сокета. Пока не установлен статус в необработанном режиме и обычно нет состояний, используемых UDP, эта колонка может оставаться незаполненной. Обычно это может принимать одно из нескольких значений:
ESTABLISHED
- сокет имеет установленное соединение.
SYN_SENT
- сокет активно пытается установить соединение.
SYN_RECV
- из сети получен запрос на соединение.
FIN_WAIT1
- сокет закрыт и соединение разрывается.
FIN_WAIT2
- соединение закрыто и сокет ожидает закрытия от удаленного узла.
TIME_WAIT

- сокет после закрытия находится в состоянии ожидания пакетов, все еще находящихся в сети.

CLOSED

- сокет не может быть использован.

CLOSE_WAIT

- удаленный узел закрыт, сокет ожидает закрытия.

LAST_ACK

- удаленный узел закрыт, сокет закрыт и ожидает подтверждения.

LISTEN

- сокет на прослушивании поступающих соединений. Такие сокеты не включают в вывод (OUTPUT), если не установлены опции --listening (-l) или --all (-a).

CLOSING

- оба сокета закрыты, но все еще не получены все отправленные пакеты.

UNKNOWN

- состояние сокета неизвестно.

User

- имя или идентификатор (User ID) пользователя сокета.

PID/Program name

- разделенная слэшем пара - идентификатор процесса (Process ID) и имя процесса, которому принадлежит пакет. Опция --program заставляет включить эту колонку. Также нужно иметь привилегии суперпользователя, чтобы видеть информацию на сокете, который не принадлежит вам. Эта идентификационная информация все еще недоступна для IPX сокетов.

Timer

- пока не записано.

Active UNIX domain Sockets

Активные сокеты домена UNIX

Proto

- протокол (обычно unix)

RefCnt

- счетчик ссылок (т. е. подключенных процессов через этот сокет).

Flags

- отображены флаги SO_ACCEPTON (показаны как ACC), SO_WAITDATA (W) или SO_NOSPACE (N). SO_ACCECPTON использованы на неподключенных сокетах, если соответствующие им процессы ожидают запроса на соединение. Другие флаги обычно не представляют интереса.

Type

- есть несколько типов доступа сокетов:

SOCK_DGRAM

- сокет, используемый в дейтаграммном режиме (без установления соединения).

SOCK_STREAM

- потоковый сокет (с установлением соединения).

SOCK_RAW

- сокет используется как низкоуровневый сокет.

SOCK_RDM

- сокет обслуживания сообщений надежной доставки.

SOCK_SEQPACKET

- сокет последовательности пакетов.

SOCK_PACKET

- сокет доступа низкоуровневых интерфейсов.

UNKNOWN

- неизвестный режим.

State

Состояние

- содержит одно из ключевых слов.

FREE

- сокет не размещен.

LISTENING

- сокет, прослушивающий запросы на соединение. Такие сокеты включают в вывод (OUTPUT), если определены опции -listening (-l) или --all (-a).

CONNECTING

- сокет в состоянии установления подключения.

CONNECTED

- сокет подключен.

DISCONNECTING

- сокет в состоянии процесса разъединения.

(empty)

- (пустой) сокет ни к чему не подключен.

UNKNOWN

- такое состояние не должно никогда случаться.

PID/Program name:

- идентификатор и имя процесса, который имеет открытый сокет. Больше информации о секции активных соединений интернет написано выше.

Path

- имя маршрута, который соответствует процессу, подключенному к сокету.

Iface

- имя интерфейса.

MTU

- максимальный размер пакета в байтах (Maximum Transfer Unit).

RX-OK (TX-OK)

- количество принятых (отправленных) пакетов без ошибок.

RX-ERR (TX-ERR)

- количество принятых (отправленных) пакетов с ошибками.

RX-DRP (TX-DRP)

- количество отброшенных пакетов при приеме (отправке).

RX-OVR (TX-OVR)

- количество пакетов, потерянных при из-за переполнения при приеме (отправке).

Flg
- флаги (могут принимать значения):
B
- разрешить широковещательную передачу (broadcast);
L
- интерфейс обратной петли (loopback);
R
- интерфейс запущен;
U
- интерфейс активен.

Информация, выводимая утилитой **tracepath**

Первая колонка выводимой информации показывает TTL пробного пакета, сопровождаемая двоеточием. Значение TTL обычно получено из отклика сети, но иногда ответ не содержит необходимой информации и заставляет предполагать. В этом случае число сопровождается знаком вопроса (?).

Вторая колонка показывает показывает сетевые этапы, которые были ответом на пробные пакеты; это - любой адрес маршрутизатора или слово [LOCALHOST], если пробный пакет не был отправлен в сеть.

Строки показывают разнообразную информацию о пути и соответствующих сетевых этапах. Как правило, они содержат значение среднего времени кругооборота RTT. Дополнительно может быть показан путевой MTU, когда он изменен. Если путь асимметричен или пробный пакет финиширует до достижения заданного этапа, различие между числом этапов в прямом и обратном направлении показано, сопровождая ключевым словом async. Такая информация ненадежна. Третья строка показывает асимметрию с первой, так как первый пробник с TTL 2 был отброшен на на первом этапе раскрытия пути.

На последней строке просуммирована информация о всем пути к узлу назначения, показан определенный путевой MTU, число этапов до цели и наши предположения о числе этапов от пункта назначения до нас, которое может отличающимся при асимметричном пути.

Лабораторное задание

1. Ознакомиться с описанием сетевых утилит, используемых в лабораторной работе, по рекомендуемым в списке литературы документам и данному руководству.

2. Войти в режим командной строки. С помощью утилиты users определить список пользователей данного узла.

Открыть еще одну вкладку Konsole и повторить команду users. Зафиксировать изменения.

3. С помощью утилиты who установить время загрузки системы, исследовать процессы входа в систему, процессы, выполняемые на данном узле, и уже законченные (отобразить заголовки столбцов).

4. С помощью утилиты hostname определить короткое имя узла, имя домена. полное имя узла, IP адрес, псевдоним узла.

5. С помощью утилиты host вывести и проанализировать полную информацию для соседнего узла с именем tor0002X, в котором символ X принимает значение, отличающееся на единицу от последней цифры имени местного узла.

Провести анализ выводимой информации в зависимости от устанавливаемых флагов в пакетах запроса.

6. С помощью утилиты netstat проверить состояние сети. По таблице маршрутизации определить параметры подключения сети: адреса (маски) шлюзов, используемые интерфейсы.

По таблице сетевых интерфейсов определить максимальные размеры и статистики принятых пакетов всех подключений.

По таблицам активных соединений через сокеты UDP и TCP определить адреса и состояния соответствующих подключений.

Ознакомиться таблицей активных подключений через сокеты Unix. Определить режим доступа к сокету.

Ознакомиться со списком сокетов сервера, установленных на прослушивание.

Провести анализ устанавливаемых флагов в заголовках пакетов и выводимой информацией.

Просмотреть таблицу сокетов с дополнительной информацией о программе, использующей сокет. Определить идентификаторы программы и состояние подключения.

Определить групповые адреса узла для протоколов IPv4 и IPv6.

Ознакомиться с полной таблицей всех подключений (опции -v, -e). Установить различие выводимой информации для этих опций.

7. С помощью утилиты ping определить доступность узлов сети. Командой ping направить 3 пакета (не забывать ограничивать количество отправляемых пакетов) по петле обратной связи (по собственному адресу). Провести анализ полученной информации.

Повторить выполненный анализ для соседнего узла сети, для публичного сервера ТУСУРа а также для узлов, указанных преподавателем.

Изменить в пакете количество пересылаемых байт данных. Убедиться о выполнении команд по выводимой информации.

Определить маршрут следования пакета до сервера ТУСУРа. Установить значение поля "ttl" равным количеству узлов маршрута. Отправить пакеты и провести анализ полей принятого пакета.

Уменьшить значение поля "ttl" на единицу. Повторить запрос. Сравнить результаты последних этапов исследований.

Установить режим регистрации временных меток. Провести анализ полученных результатов.

8. С помощью утилиты tracepath определить маршруты до исследованных в п. 7 задания узлов сети. Сравнить результаты проведенных испытаний. Оценить пропускную способность исследованных маршрутов и их временные характеристики.

9. С помощью пунктов меню Konsole "Сеанс/Печать экрана" сохранить протокол работы в файле *user_name*_lab1.txt (печатать в файл), где - *user_name* - имя пользователя.

10. В отчете предоставить результаты исследований по тестированию сети и краткие выводы по каждому пункту задания. При защите работы уметь выполнить тестирование сетевых параметров, предложенных преподавателем, и проанализировать информацию, выводимую в результате выполнения соответствующей утилиты.

Контрольные вопросы

1 Исправить синтаксис команды $ users -help.

2. Для чего нужна утилита users?

3. В чем разница выполнения команд $ who -H и $ who -p?

4. Указать неверный параметр команды $ who -login.

5. Для чего нужна утилита who?

6. В чем разница выполнения команд $ hostname -d и $ dnsdomainname -v?

7. Указать неверный параметр команды $ hostname -login.

8. Для чего нужна утилита hostname?

9. В чем разница выполнения команд $ host -a и $ host -v?

10. Указать неверный параметр команды $ host -l.

11. Для чего нужна утилита host?

12. Вывод каких типов информации можно установить утилитой host?

13. Для чего нужна петля обратной связи?

14. Как проверить работоспособность петли обратной связи?

15. В чем разница выполнения команд $ netstat -V и $ netstat -v?

16. Выбрать формат команды netstat для определения групповых адресов.

17. Перечислить состояния сокетов, выводимые программой netstat.

18. Вывод каких режимы доступа к сокету предусмотрен программой netstat?

19. Какую информацию о статистике принятых пакетов предоставляет netstat?

20. Указать неверный параметр команды $ netstat -l.

21. Для чего нужна утилита netstat?

22. Вывод каких типов информации можно установить утилитой netstat?

23. Какой результат выдаст утилита netstat с параметром –r?

24. Какой протокол использован для работы с утилитой ping?

25. В чем разница выполнения команды ping при использовании ключей -R и -r?

26. Какая информация выводится в результате выполнения программы ping?

27. Какой протокол использован для работы с утилитой tracepath?

28. В чем разница выполнения команды ping -R и команды tracepath?

29. Какая информация выводится в результате выполнения программы tracepath?

30. Указать неверный параметр команды $ tracepath -h.

Литература

1. Ю.А. Семенов. Протоколы Internet. - М.: Горячая линия-Телеком, 2005. - 1100 с.

2. В.Г. Олифер, Н.А Олифер. Компьютерные сети: принципы, технологии, протоколы. СПб.: Изд-во «Питер», 2005. 863 с.

3.Таненбаум Э. Компьютерные сети. СПб.: Изд-во «Питер», 2002. – 848 с.

4. Д.Н. Колисниченко, Питер В. Аллен. Linux. Полное руководство.- СПб: Наука и техника, 2006.- 784 с.

5. С.Л. Скловская. Команды Linux. Справочник. СПб: ООО"ДиаСофтЮП", 2004. - 848 с.

Краткие сведения о сетевых утилитах

1. Утилита **users**

Команда **users** выводит пользовательские имена клиентов, зарегистрированных на данном узле к настоящему времени.

Синтаксис команды:

users [OPTION]... [*FILE*]

Опции:

--help
- отобразить эту справку и выйти

--version
- показать информацию о версии и выйти

Выводит список имен подключенных пользователей в соответствии с файлом *FILE*. Если *FILE* не указан, используется /var/run/utmp. Обычно как *FILE* используют /var/log/wtmp.

2. Утилита **who**

Команда **who** показывает пользователей, зарегистрированных на данный момент.

Синтаксис команды:

who [*OPTION*]... [*FILE* | *ARG1 ARG2*]

Опции:

-a, --all
- эквивалентно опциям -b -d --login -p -r -t -T -u.

-b, --boot
- время последней загрузки системы.

-d, --dead
- печатать мертвые процессы.

-H, --heading
- печатать строку с заголовками столбцов.

-l, --login

- печатать процессы входа в систему.

--lookup

- пытаться канонизировать имена узлов через DNS.

-m

- только имя узла и пользователя, подсоединенного стандартным вводом.

-p, --process

- печатать активные процессы, порожденные init.

-q, --count

- все имена и число подключенных пользователей.

-r, --runlevel

- печатать текущий уровень выполнения

-s, --short

- печатать только имя, линию и время. Установлено по умолчанию.

-t, --time

- печатать последнее изменение системного времени

-T, -w, --mesg

- добавлять статус приема сообщений как +, - или ?

-u, --users

- перечислить подключенных пользователей.

--message, -writable

- эквивалентно -T.

--help

отобразить эту справку и выйти.

--version

- показать информацию о версии и выйти.

Возможен также вариант команды

who am i

- эквивалентно команде who -m.

Если FILE не указан, используется /var/run/utmp. Если заданы аргументы ARG1 ARG2, предполагается использование ‘am i’ or ‘mom likes’.

3 Утилита **hostname**

Команда **hostname** устанавливает или показывает системное имя локального компьютера (хоста). Устанавливать системное имя имеет право только администратор.

Синтаксис команды:

hostname [-v] [-a] [--alias] [-d] [--domain] [-f] [--fqdn] [-i] [--ip-address] [--long] [-s] [--short] [-y] [--yp] [--nis]

hostname [-v] [-F *filename*] [--file *filename*] [*hostname*]

hostname [-v] [-h] [--help] [-V] [--version]

Опции:

-a, --alias

- отобразить псевдоним узла (если используется).

-d, --domain

- отобразить имя домена системы DNS.

-f, --fqdn, --long

- отобразить полное доменное имя FQDN (Fully Qualified Domain Name) системы DNS. FQDN состоит из короткого имени хоста и имени домена DNS.

-F, --file *filename*

- чтение имени узла из указанного файла *filename.*

-h, --help

- вывести сообщение об использовании справочной системы и выйти из нее.

-i, --ip-address

- отобразить IP-адрес узла.

-s, --short

- отобразить короткое имя узла (имя хоста до первой точки).

-V, --version

- вывести информацию о версии программы и выйти из нее.

-v, --verbose

- установить подробный формат вывода сообщения.

-y, --yp, --nis

- отобразить имя домена NIS.

Когда команда вызывается, без аргументов программа отображает текущее имя узла.

4. Утилита **dnsdomainname**

Команда **dnsdomainname** выводит доменное имя DNS.

Синтаксис команды:

dnsdomainname [-v] [-h] [--help] [-V] [--version]

Опции команды dnsdomainname соответствуют опциям утилиты hostname. Фактически dnsdomainname дублирует команду hostname –d.

5. Утилита **host**

Host – утилита для выполнения поисков в системе DNS. Обычно используется для преобразования системного имени в IP адрес.

Синтаксис команды:

host [-aCdlriTwv] [-c *class*] [-N *ndots*] [-R *number*] [-t *type*] [-W *wait*] [-m *flag*] [-4] [-6] {*name*} [*server*]

Опции:

-a

- эквивалентно -v или -t опции.

-C

- сравнить записи SOA (Start of Authority) для имен зоны на авторитетных серверах имен.

-d

- эквивалентно -v опции (различия были в прежних версиях).

-l

- указать вид списка (символьные имена и числовые адреса). В комбинации с -a выводить все записи - *список всех узлов домена, используя AXFR.*

r

- отмена режима рекурсии при запросе серверов имен, т. е. узлу разрешено имитировать поведение сервера имен без направления запросов другим серверам имен.

-i

- обратный поиск IP адреса по протоколу v6 должен использовать домен IP6.INT. По умолчанию использовать IP6.ARPA.

-T

- использовать TCP соединение для запроса. TCP автоматически выбирается для запросов, которые требуют это, таких как запросы передачи зоны (AXFR). По умолчанию использовать UDP.

-w

- всегда ожидать ответа на запрос.

-v

- разрешить подробный вывод.

-c *class*

- выполнить запрос класса *class* DNS. По умолчанию установлен класс IN (Internet).

-N *ndots*

- установить количество разделительных точек *ndots* в составном имени, для того, чтобы считать его абсолютным. По умолчанию это число определено спецификацией /etc/resolv.conf, либо равно 1, если ее нет. Имена с меньшим количеством точек интерпретируются как относительные и для их поиска используют дополнительные указатели в /etc/resolv.conf.

-t *type*

- выбрать тип запроса (CNAME, NS, SOA, SIG, KEY, AXFR, и т.д.). Хост автоматически выбирает требуемый тип запроса. По умолчанию представляются записи адреса.

-W *wait*

- установить время ожидания ответа на запрос *wait* секунд.

-R *number*

- установить количество попыток UDP запросов *number* для поиска. По умолчанию равно 1.

-m *flag*

установка памяти, используемой для отладочных флагов (record|usage| trace).

name

- под именем *name* понимается доменное имя, которое должно быть найдено. Это также может быть IP адрес в децимальной точечной нотации версии IPv4 либо с разделительными двоеточиями версии IPv6.

server

- дополнительный аргумент в виде имени или IP адреса сервера имен, который узел должен запросить вместо серверов, указанных в спецификации etc/resolv.conf.

Когда команда вызывается без аргументов и опций, программа отображает краткое описание строки аргументов и опций команды.

6. Утилита **netstat**

Программа **netstat** распечатывает информацию о сетевых соединениях, таблицах маршрутизации, статистиках интерфейсов, замаскированных соединениях и групповых вещаниях.

Синтаксис команды:

netstat [*address_family_options*] [--tcp|-t] [--udp|-u] [--raw|-w] [--listening|-l] [--all|-a] [--numeric|-n] [--numeric-hosts] [--numeric-ports] [--numeric-users] [--symbolic|-N] [--extend|-e[--extend|-e]] [--timers|-o] [--program|-p] [--verbose|-v] [--continuous|-c] [*delay*]

netstat {--route|-r} [*address_family_options]* [--extend|-e[--extend|-e]] [--verbose| -v] [--numeric|-n] [--numeric-hosts] [--numeric-ports] [--numeric-ports] [--continuous| -c] [delay]

netstat {--interfaces|-i} [iface] [--all|-a] [--extend|-e[--extend|-e]] [--verbose|-v] [--program|-p] [--numeric|-n] [--numeric-hosts] [--numeric-ports] [--numeric-ports] [--continuous|-c] [delay]

netstat {--groups|-g} [--numeric|-n] [--numeric-hosts][--numeric-ports][--numeric-ports] [--continuous|-c] [delay]

netstat {--masquerade|-M} [--extend|-e] [--numeric|-n] [--numeric-hosts] [--numeric-ports] [--numeric-ports] [--continuous|-c] [*delay*]

netstat {--statistics|-s} [--tcp|-t] [--udp|-u] [--raw|-w] [*delay*]

netstat {--version|-V}

netstat {--help|-h}

Описание команд:

Netstat распечатывает информацию о функционировании сетевых подсистем Linux. Тип выводимой информации определяет первый аргумент команды.

Без аргументов по умолчанию отображается перечень открытых сокетов. Если не указано никакое адресное семейство *address_family_options*, выводится список активных сокетов всех конфигураций.

Первый параметр команды может принимать следующие значения

address_family_options:

[--protocol={inet,unix,ipx,ax25,netrom,ddp}[,...]]
либо [--unix|-x] [--inet|--ip] [--ax25] [--ipx] [--netrom] [--ddp]
- указать адресные семейства, поддерживающие соединения.

--route, -r
- отобразить ядро таблицы маршрутизации.

--interface=*iface* , -i
- печатать таблицу всех сетевых интерфейсов или указанных параметром *iface*.

--groups, -g
- отобразить информацию о членах мульти-вещательных групп для IPv4 и IPv6.

--masquerade, -M
- отобразить список замаскированных соединений.
--statistics, -s
- отобразить итоговую статистику для каждого протокола.
--version, -V
- показать информацию о версии и выйти.
--help, -h
- отобразить эту справку и выйти.
Дополнительные аргументы:
--tcp, -t
- отобразить активные соединения через сокеты tcp.
--udp, -u
- отобразить активные соединения через сокеты udp.
--raw, -w
- отобразить активные соединения через низкоуровневые сокеты raw.
--listening, -l
- отображать только прослушивающие сокеты (опущено по умолчанию).
--all,-a
- показать как прослушающие, так и не слушающие сокеты. С опцией --interfaces показать интерфейсы, которые не отмечены.
--numeric, -n
- показать числовые адреса вместо попыток определения символьных имен узла, порта и пользовательского имени.
--numeric-hosts
- показать числовые адреса узлов, но не затрагивать разрешения порта или пользовательского имени.
--numeric-ports
- показать числовые номера портов, но не затрагивать разрешения имен узлов или пользовательских имен.
--numeric-users
- показать числовые идентификаторы пользователей, но не затрагивать разрешения имен узлов или пользовательских имен.
--symbolic, -N
- показать аппаратные имена.
--extend, -e
- отобразить дополнительную информацию. Для максимальной детализации используют эту опцию дважды.
--timers, -o
- включить информацию, относящуюся к сетевым таймерам.
--program, -p
- показать идентификационный номер и название программы, к которой принадлежит каждый сокет.
--verbose,-v

- подробно сообщать о подключениях, действующих на данный момент. Особенно печатать некоторую полезную информацию о несконфигурированных адресных семействах.

--continuous, -c

- печатать длительно выбранную информацию каждую секунду.

delay

- повторять цикл печати через каждые *delay* секунд.

--protocol=family, -A

- указать адресное семейство (возможно, лучше описанное как протокол нижнего уровня) для каждого подключения, которое будет показано. Под семейством понимается перечень ключевых слов семейства адресов, таких как inet, unix, ipx, ax25, netrom и ddp. Имеет тот же самый эффект, как использование опций --net, --unix (-), --ipx, --ax25, --netrom и —ddp.

Семейство адресов inet включает в себя сокеты tcp, udp и raw протоколов.

--fib, -F

- отображать маршрутную информацию из пересылаемой информационную базу (Forwarding Information Base) (по умолчанию)

--cache, -C

- отображать маршрутную информацию из из кэша

--notrim, -T

- остановить подгонку длинных адресов.

--context, -Z

- если SELinux доступен, печатать в контексте SELinux.

7. Утилита **ping (ping6)**

Утилита **ping (ping6)** посылает ICMP запросы сетевым узлам. Ping использует дейтаграммы ECHO_REQUEST протокола ICMP для вызова пакетов ECHO_RESPONSE протокола ICMP от хостов и шлюзов. Дейтаграммы ECHO_REQUEST (‘‘pings’’) IP и ICMP содержат заголовок, сопровождаемый структурой временных интервалов, и затем произвольным числом дополняющих байт, используемых для заполнения пакета.

Синтаксис команды:

ping [-LRUbdfnqrvVaAB] [-c count] [-i interval] [-l preload] [-p pattern] [-s packetsize] [-t ttl] [-w deadline] [-F flowlabel] [-I interface] [-M hint] [-Q tos] [-S sndbuf] [-T timestamp option] [-W timeout] [hop ...] destination

Опции:

-a

- звуковой ping.

-A

- адаптивный ping. Межпакетный интервал адаптируется к времени кругооборота пакетов. Межпакетный интервал адаптируется к времени кругооборота пакетов, так что не более одной (или более. если установлена опция -preload) без ответной попытки присутствует настоящий момент в сети. Для пользователей,

не имеющих привилегий, минимальный интервал составляет 200 мс. Для сети с низкоуровневым временем оборота этот режим по существу соответствует потоковому режиму.

-b

- позволять "пингование" широковещательных адресов.

-B

- не позволять ping изменять адреса источника попыток. Адреса ограничены одним, выбранным при старте ping.

-c *count*

- остановить после посылки *count* количества пакетов ECHO_REQUEST. С опцией deadline ping ожидает *count* пакетов ECHO_REPLY до тех пор, пока не закончится время ожидания.

-d

- установить опцию SO_DEBUG для используемого сокета. По существу эта опция сокета не используется ядром LINUX.

-F *flow label*

- разместить и установить 20 битовую метку потока в пакете эхо-запроса (только ping6). Если значение равно нулю, ядро размещает метку потока произвольно.

-f

- потоковый ping. Для каждого посланного эхо-запроса печатается точка ".", тогда как печатается каждый принятый возврат эхо-ответа. Это обеспечивает быстрое отображение потерянных пакетов. Если интервал не задан, он устанавливается в нуль, и пакеты выводятся непосредственно по возвращению, либо 100 раз за секунду (наибольшее из этих условий).

-i *interval*

- ожидает *interval* секунд между отправлениями каждого пакета. По умолчанию нормальный интервал ожидания между пакетами 1 секунда, или без ожидания в потоковом режиме. Устанавливать интервал менее 0,2 секунды может только администратор.

-I *interface address*

- установить определенный интерфейс адреса источника. Аргументом может быть числовой IP адрес либо имя устройства. Эта опция необходима для "пингования" IPv6 локальных адресов.

-l *preload*

- если определен , ping посылает столько пакетов, не ожидая ответа. Только администратор может выбрать *preload* более 3.

-L

- запретить кольцевание (loopback) широковещательных пакетов. Этот флаг применим для "пингования" широковещательных адресов назначения.

-n

- вывод только числовой. Не пытаться заставлять искать символические имена для адресов узлов.

-p *pattern*

- можно определить до 16 дополняющих байт для заполнения посылаемого пакета. Это полезно для диагностики проблем сети, зависящих от данных. Например, опция -p *ff* заставляет посылать пакеты, полностью заполненные единицами.

-Q *tos*

- установить биты дейтаграммы ICMP, относящиеся к назначению качества сервиса. Параметр *tos* может быть как десятичным, так и шестнадцатеричным числом. Традиционно (RFC1349) интерпретируется как: 0 - зарезервирован (в настоящее время переопределен для контроля перегруженности), 1-4 для установления типа сервиса и 5-7 для установления приоритета. Доступные установки типа сервиса: минимальная стоимость - 0x02, надежность - 0x04, пропускная способность - 0x08, низкая задержка - 0x10. Много бит качества не должно быть установлено одновременно. Возможные параметры, определенные для диапазона приоритета - от приоритета (0x02) до сетевого контроля (0xe0). Нужно быть администратором (CAP_NET_ADMIN возможности) для использования критических или более высоких значений приоритета). Нельзя установить бит 0x01 (зарезервированный), если в ядре недоступен ECN. В REF2474 эти поля переопределены как 8-битовый дифференцированный сервис (DS), состоящий из: бит 0-1 - для разделения данных (нужно использование ECN), и бит 2-7 - основные коды Codepoint (DSCP).

-q

- ограниченный вывод. Ничего не отображать, кроме строк старта и завершения.

-R

- записать маршрут. Включить опцию RECORD_ROUTE в пакет эхо-запроса и отображать маршрутный буфер возвращаемого пакета. Заметим, что размеры IP заголовка достаточны лишь для 9 пунктов маршрута. Многие узлы игнорируют или сбрасывают эту опцию.

-r

- миновать обычную таблицу маршрутизации и отправлять непосредственно узлу по подключенному интерфейсу. Если узел не находится в непосредственно подключенной сети, возвращается ошибка. Эта опция может быть использована для "пингования" местных узлов по интерфейсу, через который нет маршрута, при условии, что предусмотрено также использование опции -I.

-s *packetsize*

- отправляет количество байт данных для передачи. По умолчанию - это 56, которые транслируются в 64 байта данных IP, объединенных с 8 байтами заголовка ICMP.

-S *sndbuf*

- установить размер буфера сокета. Если не определено, выбирается буферизировать не более одного пакета.

-t *ttl*

- установить IP время жизни *ttl*.

-T *timestamp option*

- установить специальную IP опцию временная метка *timestamp*. Опция *timestamp* может выглядеть как *tsonly* (только временная метка), *tsandaddr* (временная метка и адрес) или *tsprespec host1 [host2 [host3 [host4]]]* (временная метка и предусмотренные узлы)

-M *hint*

- выбрать стратегию выбора пути для максимальной размера блока передачи данных (MNU). Аргумент *hint* может принимать значение *do* (запрещает фрагментацию, даже локальную), *want* (выполнять стратегию поиска, фрагментировать локально, когда размер пакета больше), или dont (не устанавливать флаг DF.

-U

- печатать полное время ожидания пользователь-пользователь. Обычно ping печатает сетевое время кругооборота.

-v

- выводить подробную информацию.

-V

- показать версию и выйти

-w *deadline*

- определить задержку в секундах до завершения функционирования ping независимо от количества принятых или отправленных пакетов. В этом случае ping не останавливается после получения count пакетов, а ожидает также истечения последнего срока или ответа счетчика попыток или извещения об ошибке из сети.

-W *timeout*

- время *timeout* ожидания ответа в секундах. Опция влияет только при отсутствии любых ответов, иначе ping ожидает двойное время кругооборота.

Дополнительные сведения по использованию программы ping

При использовании утилиты ping для выделения ошибок он сначала должен быть запущен на местные узлы, чтобы проверить что локальный сетевой интерфейс поднят и выполняется. Затем "пингуемые" хосты и маршрутизаторы должны быть продвинуты все дальше и дальше. Вычисляется время кругооборота и статистика потерь. Если получены копии пакетов, они не включаются в расчет потерь пакетов, хотя время кругооборота используется в вычислениях минимального-среднего-максимального значений времени кругооборота. Когда определенное количество пакетов будет отправлено (и получено) или если программа завершается с SIGINT, отображаются краткие итоги. Короче, текущая статистика может быть получена без завершения работы процесса с сигналом SIGQUIT.

Если ping не получает никаких ответных пакетов, он будет завершен с кодом 1. Если определены число пакетов *count* и последний срок *deadline*, а число принятых пакетов на интервале крайнего срока меньше, чем *count*, также будет выход с кодом 1. По другой ошибке - выход с кодом 2. В противном случае - выход с кодом 0. Это делает возможным использовать код выхода для того, чтобы видеть, хост работоспособен или нет.

Эта программа предназначена для использования при тестировании, измерении и управлении в сети. Поскольку ее загрузка может накладываться на сеть, неразумно использовать ping в течении обычного функционирования или из автоматизированных сценариев.

Детали пакета ICMP

Заголовок IP пакета без опций - 20 байт. ICMP пакет эхо-запроса содержит дополнительные важных 8 байт ICMP заголовка, сопровождаемых произвольным количеством байт данных. Когда задан размер пакета, он указывает размер этой дополнительной части данных (по умолчанию - это 56). Таким образом, количество принимаемых данных IP пакета типа ICMP ECHO_REPLY будет всегда на 8 байт больше, чем затребовано объемом данных ICMP заголовка.

Если объем данных менее размера структурного интервала. ping использует начальные байты этого пространства для включения метки времени, которая используется для вычислении времени кругооборота. Если объем данных короче, время кругооборота не задано.

Продублированные и поврежденные пакеты

ping сообщает о продублированных и поврежденных пакетах. Дублирования пакетов не должно происходить никогда и, видимо, вызвано несоответствующей ретрансляцией пакетов на сетевом уровне. Дублирование может случаться во многих ситуациях и редко (если когда-либо) - хороший знак, хотя наличие низкоуровневых копий не всегда может быть причиной тревоги.

Поврежденные пакеты очевидно, более серьезная причина для тревоги и часто указывают неисправность аппаратных средств где-то на пути пакета ping (в сети или на узлах).

Затруднения, вызванные различными форматами данных

Межсетевой уровень никогда не должен обрабатывать пакеты, по разному зависимые от информации, содержащейся в разделе данных. К сожалению известно, что проблемы зависимости от данных прокрадываются в сеть, продолжительный период времени оставаясь не обнаруженными. Во многих случаях особенности форматов, которые будут иметь проблемы, это такие, как недостаточное количество переключений, например, длинные серии нулей или единиц.

Это означает, что если есть проблемы, связанные с зависимостью от данных, вероятно, придется выполнить большой объем тестирования, чтобы найти их. В случае удачи можно устраивать поиск файла, который каждый не может быть послан через сеть, или который требует много дольше для передачи, чем другой файл подобной длины. Потом можно исследовать этот файл для повторения образца, который можно тестировать, используя опцию -p команды ping.

Детали поля TTL

Значение TTL IP пакета представляет максимальное число IP маршрутизаторов, через которые пакет может пройти до того, как будет отброшен. В на-

стоящей практике можно ожидать, что каждый маршрутизатор в интернет уменьшает поле TTL точно на 1.

Спецификация TCP/IP указывает, что поле TTL для пакета TCP должно быть установлено на 60, но многие системы имеют меньшие значения (4.3 BSD использует 30, 4.2 BSD использует 15).

Максимально возможное значение этого поля - 255 и наибольшая установка поля TTL Unix систем пакетов ICMP ECHO_REQUEST - 255. Вот почему можно "пинговать" некоторые хосты, но не достигать их по протоколам telnet или ftp.

При нормальном функционировании ping печатает значение TTL из принимаемых пакетов. Когда удаленная система получает пакет ping, она может делать одну из трех вещей с полем TTL в своем ответе:

● Не изменять его, как это делала система Berkeley Unix до выпуска 4.3BSD Tahoe release. В этом случае значение TTL в принятом пакете будет равно 255 минус количество маршрутизаторов в маршруте кругооборота.

● Установить его на 255, как это делают современные системы Berkeley Unix. В этом случае значение TTL в принятом пакете будет равно 255 минус количество маршрутизаторов в маршруте от удаленной системы до "пингующего" узла.

● Установить ему несколько другое значение. Некоторые машины используют то же самое значение для ICMP пакета, которое они используют для TCP пакета, например, 30 или 60. Другие могут использовать полностью неконтролируемые значения.

Проблемы

Многие хосты и маршрутизаторы игнорируют опцию RECORD_ROUTE.

Максимальная длина IP заголовка слишком мала, чтобы были полностью использованы опции типа RECORD_ROUTE.

Потоковое "пингование" не рекомендуется, в общем случае, а потоковое "пингование" широковещательных адресов должно выполняться только при тщательно контролируемых условиях.

8. Утилита **tracepath (tracepath6)**

Команда **tracepath (tracepath6)** трассирует путь к сетевому узлу, раскрывая MTU вдоль этого пути.

Синтаксис команды:

tracepath *destination* [*port*]

Утилита **tracepath** устанавливает маршрут к сетевому узлу *destination* с определением максимального размера пакета. Она использует UDP порт *port* или какой-нибудь произвольный порт. Программа работает подобно утилите traceroute, но не требует привилегий суперпользователя и не предполагает опций.

ЛАБОРАТОРНАЯ РАБОТА №2. УДАЛЕННЫЙ ДОСТУП

Цель работы

Целью данной работы является знакомство с протоколом SSH, обеспечивающими безопасное зашифрованное соединение двух ненадежных узлов через небезопасную сеть для регистрации пользователя на удаленном узле, а также получение навыков работы пользователей, не имеющих привилегий администратора, в режиме удаленного доступа.

Краткие сведения о пакете OpenSSH

Удаленный доступ традиционно является одним из основных компонентов сетевого взаимодействия. Например. в защищенных сетях в свое время широкое применение получил протокол telnet. Telnet позволяет пользователю установить TCP соединение с сервером и работать с ним по командам клиентской машины так, словно она является удаленным терминалом сервера. Однако проблемы безопасности ограничивают его использование в сетях, ненадежных с точки зрения безопасности.

В настоящее время в таких сетях широкое применение получает протокол ssh. Ssh (secure shell - безопасная оболочка) является протоколом для удаленного безопасного входа и других сетевых сервисов безопасности в недостаточно надежно защищенной сети. В общей архитектуре протокола ssh можно выделить три основных блока: протокол транспортного уровня, протокол аутентификации пользователя и протокол соединения.

Протокол транспортного уровня (SSH-TRANS) обеспечивает аутентификацию сервера, конфиденциальность и целостность соединения. Также может дополнительно обеспечивать сжатие данных. Протокол транспортного уровня обычно выполняется поверх соединения TCP, но может использоваться и поверх любого другого надежного соединения.

Протокол аутентификации пользователя (SSH-USERAUTH) аутентифицирует клиента для сервера. Он выполняется поверх протокола транспортного уровня.

С целью повышения безопасности соединения осуществляется как аутентификация клиента для сервера, так и аутентификация сервера для клиента. Клиент посылает запрос на обслуживание всякий раз, когда устанавливается безопасное соединение на транспортном уровне. Второй запрос сервиса посылается после выполнения аутентификации пользователя.

Протокол соединения (SSH-CONN), мультиплексирует несколько логических каналов в один зашифрованный туннель. Протокол выполняется поверх протокола аутентификации пользователя.

Протокол соединения создает каналы, которые могут использоваться для различных целей. Существуют стандартные методы установки безопасных сессий интерактивного shell и перенаправления ("туннелирования") произвольных портов TCP/IP и соединений X11.

На любом узле сети, использующем ssh, может выполняться как клиентская, так и серверная часть программы. Для работы с ssh каждый хост может

иметь не менее одного ключа хоста, причем в шифровании могут быть использованы различные криптографические алгоритмы. Хосты могут иметь несколько ключей, используемых различными алгоритмами. Несколько узлов могут использовать общий ключ хоста. Однако каждый сервер должен иметь, по крайней мере, один ключ для каждого обязательного алгоритма открытого ключа. В настоящее время требуется поддерживать алгоритм DSS (Digital Signature Standard).

Ключ хоста-сервера используется при обмене открытыми ключами для проверки истинности соединения с подлинным (а не подмененным) сервером. Для этого клиент должен предварительно знать об открытом ключе сервера. Это знание может быть реализовано в рамках одной из двух моделей.

В первой клиент имеет локальную базу данных (файл), в которой каждому имени сервера ставится в соответствие его открытый ключ. Этот метод не требует централизованной административной инфраструктуры и трехсторонней координации. В то же время, такую базу данных тяжело поддерживать при большом количестве клиентов и серверов, с которыми они должны взаимодействовать.

Во второй модели вводится понятие доверенного сертификационного агента, который и отвечает за проверку соответствия имени хоста его открытому ключу. При этом упрощается поддержка клиента (он должен знать открытый ключ только самого сертификационного агента), но предъявляются высокие требования к сертификационному агенту, который должен иметь открытые ключи всех хостов, к которым обращаются клиенты.

Предусмотрена также возможность отказа от проверки открытого ключа хоста сервера при первом соединении клиента с сервером. Это обеспечивает возможность взаимодействия без предварительного знания ключа сервера. Такое соединение также обеспечивает защиту от пассивного прослушивания; но оно уязвимо для активных атак типа встреча посередине (man-in-the-middle), то есть попытки временной подмены сервера. И такие соединения не должны быть допущены по умолчанию, если в сети допускается возможность активных атак. Однако, так как инфраструктура открытого ключа еще недостаточно широко распространена, данная опция делает протокол более приемлемым для взаимодействия сторон, обеспечивая более высокий уровень безопасности, чем такие предыдущие решения как telnet или rlogin.

В результате использования ssh, то есть выполнения процедуры проверки полномочий пользователей и последующего шифрования любой проходящей через сеть информации, от паролей до сеансов, обеспечивается более высокая степень защищенности удаленного доступа.

Компоненты службы удаленного доступа ssh (OpenSSH) реализуются различными командами. Например. клиентская программа ssh обеспечивает регистрацию и выполнение команд на удаленной машине. программа sshd (демон ssh) обеспечивает защищенное шифрованное соединение между двумя ненадежными узлами по небезопасной сети, программа ssh-keygen генерирует и управляет ключами идентификации. Кроме перечисленных, имеется и ряд иных утилит: scp, sftp и др.

Перенаправление произвольного TCP соединения через защищенный канал может быть определено как к командной строке, так и в файле конфигурации. одно возможное приложение перенаправления TCP - это защищенное соединение с почтовым сервером, другое действует через сетевой экран.

Например, рассмотрим кодированное соединение между клиентом и сервером IRC, не имеющих непосредственной поддержки шифрованного соединения. Это действует следующим образом. Пользователь подключается к удаленному узлу, используя ssh, определяя порт, который должен быть использован для перенаправления соединения на удаленный сервер. После этого имеется возможность запустить службу шифрования на клиентской машине, подключенной к этому самому порту и ssh будет кодировать и перенаправлять соединение.

В следующем примере прокладывается туннель сеанса IRC от клиентской машины “127.0.0.1” (localhost) до удаленного сервера “server.example.com”:

$ ssh -f -L 1234:localhost:6667 server.example.com sleep 10

$ irc -c ’#users’ -p 1234 pinky 127.0.0.1

Здесь используется туннельное соединения IRC сервера “server.example.com”, связанного каналом с пользователем “#users”, nickname “pinky” с применением порта 1234. Не существенно, какой номер порта использован, насколько больше, чем 1023 (напомним, что только root может открывать сокеты на привилегированных портах), и не создается конфликтов с любым уже используемым портом. Соединение перенаправлено на порт 6667 удаленного сервера, поскольку это стандартный порт для IRC сервиса.

Опция -f устанавливает фоновый режим ssh и дистанционная команда “sleep 10” определяет разрешенное количество времени (10 секунд в примере) до старта службы, которая выполняет туннелирование. Если в пределах заданного времени соединение не сделано, ssh завершается.

Лабораторное задание

1. Ознакомиться с описанием протокола ssh и утилит, используемых в лабораторной работе (упомянутых ниже в пунктах задания), по рекомендуемым в списке литературы документам, данному руководству и справочной системы Linux.

2. Войти в режим командной строки. С помощью команды ssh подключиться к узлу, с которым работали по п. 5 предыдущей работы. На соответствующий запрос сервера ввести пароль для регистрации на удаленной машине.

3. Ознакомиться с файловой системой сервера. С помощью команды pwd определите имя текущего каталога. С помощью команды ls определите содержимое текущего каталога.

С помощью команд cd и ls ознакомиться с содержанием соседних каталогов.

4. С помощью команды mkdir домашнем каталоге создать именной подкаталог. В этом подкаталоге с помощью команды touch создать текстовый файл с именем lab_1.

5. С помощью текстового редактора (например, ed или vim), войти в режим редактирования этого файла и внести в него следующую информацию:

Фамилию ИО (студента);

группу, факультет;

имя пользователя;

сетевое имя локального узла;

IP адрес и маску локального узла;

имя домена;

адрес шлюза по умолчанию.

6. С помощью утилиты users определить список пользователей удаленного узла.

7. С помощью утилиты who установить время загрузки системы, исследовать процессы входа в систему, процессы, выполняемые на данном узле, и уже законченные.

8. С помощью утилиты hostname определить короткое имя узла, имя домена. полное имя узла, IP адрес, псевдоним узла.

9. С помощью утилиты host вывести и проанализировать полную информацию для локального узла (на котором работаете непосредственно в данный момент).

10. С помощью утилиты netstat проверить состояние сети. По таблице маршрутизации определить параметры подключения сети: адреса (маски) шлюзов, используемые интерфейсы.

11. Наиболее внимательно проанализировать информацию по соединению ssh на ближнем и удаленном узле.

12. С помощью утилиты ping определить доступность узлов сети. Командой ping направить 3 пакета локальному узлу.

Повторить операцию для публичного сервера ТУСУРа.

13. С помощью утилиты tracepath определить маршрут до узла, заданного преподавателем. Оценить пропускную способность маршрута и временные характеристики.

14. С помощью пунктов меню Konsole "Сеанс/Печать экрана" сохранить протокол работы на удаленной системе в файле lab_2.txt (печатать в файл). В отчете также предоставить результаты редактирования текстового файла. При защите работы уметь выполнять команды навигации по удаленной системе, созданию и перемещению файлов и каталогов, редактирования текстовых файлов.

Контрольные вопросы

1 Для чего нужна программа ssh?

2. Что общего и в чем отличие ssh и telnet?

3. Основные достоинства и недостатки telnet.

4. Основные компоненты ssh.

5. Назначение протокола транспортного уровня ssh.

6. Основные характеристики протокола аутентификации пользователя ssh.

7. Назначение протокола соединения ssh.

8. Что понимается под ключом хоста?

9. Как проверяется подлинность сервера?

10. Что такое сертификационный агент?

11. Особенности режима работы ssh без проверки полномочий сервера.

12. Что понимается под TCP перенаправлением?

13. Как реализуется перенаправление TCP соединения?

14. Записать обязательные элементы команды ssh.

15. Привести формат команды ssh при заданных портах местного и удаленного узла.

16. Для чего нужна команда pwd?

17. Какие ключи используют с командой cd?

18. Что выполняется по команде mkdir?

19. Для чего используется команда ls?

20. Назначение утилиты ed.

21. Для чего нужна утилиты vim?

Литература

1. Д.В. Иртегов. Введение в сетевые технологии. Учебное пособие для вузов. СПб.: БХВ-Петербург, 2004.-539 с.

2 О.Р. Лапонина. Протокол SSH. Интернет ресурс http://www.intuit.ru/department/security/networksec2/10/.

3. Д.Н. Колисниченко, Питер В. Аллен. Linux. Полное руководство.- СПб: Наука и техника, 2006.- 784 с.

4. С.Л. Скловская. Команды Linux. Справочник. СПб: ООО"ДиаСофтЮП", 2004. - 848 с.

Краткие сведения по командам ssh

SSH client - программа удаленного входа в систему.

Синтаксис команды:

ssh [-1246AaCfgkMNnqsTtVvXxY] [-b *bind_address*] [-c *cipher_spec*] [-D [*bind_address*:]*port*] [-e *escape_char*] [-F *configfile*] [-i *identity_file*] [-L [*bind_address*:]port:host:*hostport*] [-l *login_name*] [-m *mac_spec*] [-O *ctl_cmd*] [-o *option*] [-p *port*] [-R [*bind_address*:]*port*:*host*:*hostport*] [-S *ctl_path*] [-w *local_tun*[:*remote_tun*]] [*user@*]*hostname* [*command*]

Описание программы:

SSH client это программа для регистрации на удаленном узле и для выполнения команд на удаленной машине. Это предполагает замену программ rlogin и rsh и обеспечивает безопасное зашифрованное соединение двух ненадежных узлов через небезопасную сеть. Соединения X11 и произвольные TCP порты также могут пересылать безопасно.

SSH подключает и регистрирует к определенному узлу (с опцией имени пользователя). Пользователь должен доказать свою подлинность удаленной

машине, используя один из несколько методов в зависимости от используемой версии протокола.

Если определена команда, она выполняется на уделенном узле, вместо входа в систему.

Опции

-1

- заставляет ssh испытывать протокол только версии 1.

-2

- заставляет ssh испытывать протокол только версии 2.

-4

- заставляет ssh использовать только адресацию IPv4.

-6

- заставляет ssh использовать только адресацию IPv6.

-A

- разрешить предварительное соединение агента аутентификации (это также может быть определено в файле конфигурации хоста).

-a

- запретить предварительное соединение агента аутентификации.

-b *bind_address*

- использовать *bind_address* на локальной машине как исходный адрес соединения. Использовать только в системах с более, чем одним адресом.

-C

- запрашивать сжатие всех данных, включая stdin, stdout, stderr, и данные для пересылки соединений X11 и TCP. Алгоритм компрессии тот же, который использует gzip.

-c *cipher_spec*

- выбрать спецификацию шифра для кодирования сессии. Протокол версии 1 допускает единственный перечень шифра. Поддерживаемые значения: “3des”, “blowfish”, “des”. По умолчанию используется “3des”. Для протокола версии 2 перечень *cipher_spec* представляет собой разделенный запятыми список в порядке предпочтения: 3des-cbc, aes128-cbc, aes192-cbc, aes256-cbc, aes128-ctr, aes192-ctr, aes256-ctr, arcfour128, arcfour256, arcfour, blow-fish-cbc, and cast128-cbc. По умолчанию: aes128-cbc,3des-cbc,blowfish-cbc,cast128-cbc,arcfour128, arcfour256,arcfour,aes192-cbc,aes256-cbc,aes128-ctr, aes192-ctr,aes256-ctr.

-D [*bind_address*:]*port*

- определить местный динамически перенаправляемый порт уровня приложений (привилегия root).

-e *escape_char*

- установить escape-символ для сеансов с pty (псевдо-терминальный интерфейс) (по умолчанию "~"). Escape-символ распознается только в начале строки. Escape-символ, следующий за точкой ("."), закрывает соединение, следующий за control-Z - приостанавливает соединение, и следующий непосредственно за собой, пересылает один escape-символ. Установка символа "none" отменяет любые escape-символы и делает сеанс полностью прозрачным.

-F *configfile*

- определить альтернативный файл конфигурации пользователя. Если файл конфигурации задан в командной строке, общесистемный файл файл конфигурации (/etc/ssh/ssh_config) будет проигнорирован. По умолчанию: ~/.ssh/config.

-f

- запросить ssh перейти в фоновый режим перед выполнением команды. Это полезно, если ssh выполняет запрос пароля или передаваемой фразы.

-g

- позволять удаленному узлу подключить местный перенаправляемый порт.

-I *smartcard_device*

- определить устройство ssh, которое должно быть использовано для связи со смарткартой, используемой для записи приватный RSA ключей пользователя (если доступна поддержка таких устройств).

-i *identity_file*

- выбрать файл, из которого считывать идентичность (приватные ключи) для RSA или DSA аутентификации. По умолчанию ~/.ssh/identity для протокола версии 1 и ~/.ssh/id_rsa и ~/.ssh/id_dsa для протокола версии 2. Эти файлы могут быть также определены в файле конфигурации.

-k

- запретить перенаправление (делегирование) удостоверения GSSAPI серверу.

-L [*bind_address*:]*port:host:hostport*

- определить, что данный порт *port* на локальном (клиентском) узле пересылает информацию данным хосту *host* и порту *hostport* на удаленной стороне. Это работает распределением сокетов на прослушивание портов на ближней стороне, дополнительно связанных с определенным *bind_address*. Всякий раз, когда выполняется подключение к этому порту, соединение перенаправляется по защищенному каналу и соединяет с портом *hostport* узла *host* удаленной машины. Порт перенаправления также может быть определен в файле конфигурации.

-l *login_name*

- определить имя пользователя для регистрации на удаленной машине. Это также может быть определено в файле конфигурации хоста.

-N

- не выполнять удаленные команды (полезно только для перенаправления по протоколу версии 2).

-n

- переназначить stdin из /dev/null (предотвратить чтение из stdin).

-O *ctl_cmd*

- управлять мультиплексированием активных соединений основного процесса. Когда опция -O определена аргумент *ctl_cmd* интерпретируется и проходит в основной процесс. Правильные команды: “check” (проверять, что выполняет основной процесс) и “exit” (запрос выхода).

-o *option*

- задать опции в формате, используемом в файле конфигурации. Это полезно для определения опций, у которых нет отдельного флага в командной строке.

-p *port*

- задать номер порта *port* для соединения на удаленный хост. Это может определено в файле конфигурации хоста.

-q

- установить режим тишины. Заставляет запретить все предупреждающие и диагностические сообщения.

-R [*bind_address*:]*port:host:hostport*

- определить, что данный порт на удаленном (серверном) хосте перенаправлен на данные хост и порт на ближней стороне. Это работает локализацией сокета на прослушивание на удаленной стороне, и всякое подключение к этому порту перенаправляется по секретному каналу и подключается к порту *port* узла *host* локальной машины.

Порт перенаправления также может быть задан в конфигурационном файле. Привилегированные порты могут быть перенаправлены только когда на удаленной машине зарегистрирован суперпользователь. Адресация IPv6 может быть определена заключением в квадратные скобки, или использованием альтернативного синтаксиса: [bind_address/]host/port/hostport.

По определению сокеты на прослушивание на сервере ограничены только интерфейсом обратной связи. Это может быть проигнорировано определением *bind_address*. Пустой *bind_address* или адрес ‘*’ указывают, что удаленный сокет может прослушивать все интерфейсы. Указание на удаленный *bind_address* будет успешным, если только доступна опция сервера GatewayPorts.

-S *ctl_path*

- определить позицию управляющего сокета для разделения соединения. Ссылки на описание *ControlPath* и *ControlMaster* смотреть на ssh_config(5).

-s

- разрешить использование запроса вызова подсистемы на удаленной системе. Подсистемы представлены протоколом ssh2, которые способствуют использованию ssh как безопасного транспорта для других приложений (например, sftp). Подсистема определяется как дистанционная команда.

-T

- запретить назначение псевдо терминала.

-t

- назначить псевдотерминал. Это может быть использовано для выполнения произвольных экранных программ на удаленной машине, которое может быть очень полезным, в том числе, когда обслуживается выполнение меню. Многократные опции -t усиливают назначение псевдо терминала, даже если ssh не имеет локального tty.

-V

- отобразить номер версии и выйти.

-v

Режим подробного вывода. Заставляет ssh выводить отладочные сообщения о ходе выполнения. Они полезны для отладки проблем соединения, аутентификации и конфигурации. Многократные опции -v увеличивают многословие. Максимум - 3.

-w *local_tun*[:*remote_tun*]

- запросить перенаправление туннельных устройств, определенных командой tun между клиентом (local_tun) и сервером (remote_tun).

-X

- Разрешить перенаправление X11. Это также может быть определено в файле конфигурации. Пересылка X11 должно быть доступна с осторожностью. Пользователи, полномочные миновать разрешающий файл на удаленном узле (для базы данных разрешенных X-пользователей) могут иметь доступ к локальному X11 дисплею через пересылаемое соединение. Атакующий может затем выполнить такие действия, как мониторинг нажатия клавиш. По этой причине пересылка X11 подвергается преобразованию X11 SECURITY.

-x

- запретить перенаправление X11.