



<https://github.com/al-sapsan>

email: al.sapsan@mail.ru

ПРАКТИЧЕСКАЯ РАБОТА № 3

модуля «Работа с сетью и аппаратным обеспечением роботов в Linux».
курса «Linux для робототехников».

Основание:

изучение модуля «Работа с сетью и аппаратным
обеспечением роботов в Linux»
курса «Linux для робототехников».

Исполнитель:

Соколов Олег Вадимович

СОДЕРЖАНИЕ:

Задание 1. Работа с SSH.

1.1. Объекты.

1.2. Выполненная работа.

1.3. Результат.

Задание 2. Использование `tmux` для управления терминалами и сеансами

2.1. Объекты.

2.2. Выполненная работа.

2.3. Результат.

Задание 3. Создание виртуального последовательного порта и работа с ним

3.1. Объекты

3.2. Выполненная работа.

3.3. Дополнительное задание.

3.4. Результат.

Приложение 1. Иллюстративный материал.

Задание 1. Работа с SSH.

1.1. Объекты [\[ссылка на содержание\]](#).

- в качестве сервера был использован ноутбук MacBook Pro (A1286), сетевое имя `sapsan@UbuntuBookPro`, с предустановленной ОС `Ubuntu 20.04.6` (далее – хост);

- в качестве клиентского устройства был использован ноутбук MacBook Air (A1466), сетевое имя `sapsan@UbuntuAir`, с предустановленной ОС `Ubuntu 20.04.6` (далее – клиент).

1.2. Выполненная работа [\[ссылка на содержание\]](#):

- с помощью команды `sudo apt-get install openssh-server` было установлено соответствующее приложение на хост и клиент [\[см. иллюстрацию 1\]](#) и [\[см. иллюстрацию 2\]](#);

- с помощью команды `sudo systemctl start ssh` была установлена возможность подключения по протоколу SSH к клиенту [\[см. иллюстрацию 3\]](#);

- с помощью команды `ls` в иллюстративных целях производился вывод содержимого корневого каталога хоста, после чего с помощью команды `ssh sapsan@UbuntuAir` было установлено соединение с клиентом и с помощью команды `ls` произведена иллюстрация содержимого его корневого каталога [\[см. иллюстрацию 4\]](#);

- с помощью команды `ssh-keygen` на хост-компьютере были созданы SSH-ключи, после чего с помощью команды `ssh-copy-id sapsan@UbuntuAir` открытый ключ был скопирован на клиент компьютер; дальнейшее подключение производилось с использованием открытого ключа (без введения пароля) [\[см. иллюстрацию 5\]](#) и [\[см. иллюстрацию 6\]](#).

1.3. Результат [\[ссылка на содержание\]](#).

В результате вышеописанных действий была выполнена следующая работа:

- установлено SSH-соединение хоста и клиента;
- создана пара SSH-ключей, открытый ключ которой был передан клиенту;
- осуществлено беспарольное подключение с помощью SSH-ключей.

Задание 2. Использование `tmux` для управления терминалами и сеансами.

2.1. Объекты [\[ссылка на содержание\]](#).

- в качестве сервера был использован ноутбук MacBook Pro (A1286), сетевое имя `sapsan@UbuntuBookPro`, с предустановленной ОС `Ubuntu 20.04.6`;

- в качестве клиентского устройства был использован ноутбук MacBook Air (A1466), сетевое имя `sapsan@UbuntuAir`, с предустановленной ОС `Ubuntu 20.04.6`.

2.2. Выполненная работа [ссылка на содержание]:

- с помощью команды ``sudo apt-get install -y tmux`` было установлено данное приложение на хост и клиент [см. иллюстрацию 7];
- путем создания конфигурационного файла `~/tmux.conf1` были перенастроены горячие клавиши программы (чтоб слишком далеко пальцами не тянуться, ибо я не пианист ☺) и подключена мышь;
- с помощью команды ``tmux new -s mysession``, с использованием SSH-соединения, была начата новая сессия ``tmux`` на клиенте [см. иллюстрацию 8];
- далее в отдельной панели был запущен скрипт ``time_loop.sh2`, с бесконечным циклом, выводящим текущее время и дату каждую секунду [см. иллюстрацию 9];
- далее была проведена имитация разрыва SSH-соединения путем закрытия окна терминала [см. иллюстрацию 10];
- после чего с помощью команды ``tmux attach -t mysession`` было произведено повторное подключение к ранее закрытой сессии, чем установлено, что запущенный скрипт всё ещё выполняется [см. иллюстрацию 11] и [см. иллюстрацию 12].

2.3. Результат [ссылка на содержание].

В результате вышеописанных действий была выполнена следующая работа:

- подключение к SSH-серверу и установка ``tmux``;
- запуск новой сессии ``tmux`` и работа в ней;
- уяснение возможностей ``tmux`` по его настройке, работе с окнами, панелями, использовании других опций и горячих клавиш;
- уяснение возможностей ``tmux`` по работе в сессиях, их начало, отключение, подключение, восстановление.

Задание 3. Создание виртуального последовательного порта и работа с ним.

3.1. Объекты [ссылка на содержание].

- в качестве сервера был использован ноутбук MacBook Pro (A1286), сетевое имя ``sapsan@UbuntuBookPro``, с предустановленной ОС ``Ubuntu 20.04.6``;
- в качестве клиентского устройства был использован ноутбук MacBook Air (A1466), сетевое имя ``sapsan@UbuntuAir``, с предустановленной ОС ``Ubuntu 20.04.6``.

3.2. Выполненная работа [ссылка на содержание]:

- с помощью команды ``sudo apt-get install -y socat`` было установлено данное приложение на хост и клиент [см. иллюстрацию 13];

¹ https://drive.google.com/open?id=10tvR8X4Ez8gICuftLN5TZ030grsbWlW6&usp=drive_fs

² https://drive.google.com/open?id=10kEqkMdXhm0i9XhutW2ry-tev_cdb-wU&usp=drive_fs

- с помощью команды ``socat -d -d pty,raw,echo=0 pty,raw,echo=0`` на хосте была создана пара последовательных виртуальных соединений между собой портов [\[см. иллюстрацию 14\]](#), после чего была проведена проверка их работоспособности путем отправки и получения сообщений;

- с помощью команды ``socat TCP4-LISTEN:1234,reuseaddr,fork FILE: /dev/pts/2,b9600,raw`` на хосте было установлено TCP-соединение порта ``1234`` с виртуальным портом ``/dev/pts/2``, который, в свою очередь, подключен к порту ``/dev/pts/3`` [\[см. иллюстрацию 14\]](#);

- после чего с помощью команды ``ssh -L 2345:localhost:1234 sapsan@UbuntuBookPro`` был создан SSH-туннель между клиентским портом ``2345`` до порта ``1234`` хоста [\[см. иллюстрацию 14\]](#);

- далее с помощью команды ``socat STDIO TCP4:localhost:2345`` на клиенте и последовательного применения команд ``echo`` и ``cat`` на хосте, была проведена успешная проверка установленного соединения [\[см. иллюстрацию 14\]](#).

3.3. Дополнительное задание [\[ссылка на содержание\]](#):

В целях выполнения данного задания был создан `bash`-скрипт ``socat_port_number.sh``³, который извлекает номера созданных виртуальных портов из вывода программы ``socat``.

3.3.1. Описание скрипта:

Скрипт создает виртуальные порты с помощью ``socat``, извлекает из вывода ``socat`` номера созданных портов с помощью ``cut``, и сохраняет результат в файле с помощью перенаправления вывода.

3.3.2. Описание команд и операторов:

- команда ``socat``
 - ``socat`` создает виртуальные порты и позволяет имитировать различные типы соединений;
 - две опции ``-d`` включают режим отладки и выводят подробную информацию о работе ``socat``;
 - ``pty,raw,echo=0 pty,raw,echo=0`` – параметры типов создаваемых соединений;
 - оператор ``2>&1`` перенаправляет стандартный поток ошибок (`stderr`2``) в стандартный поток вывода (`stdout`1``);
 - оператор ``|`` перенаправляет вывод ``socat`` в стандартный ввод ``stdin`` следующей команды.
- команда ``sed``

³ https://drive.google.com/open?id=10hNQJ6okR21sU-7yAQsfOl7pQcZ2549o&usp=drive_fs

- утилита редактирования текста `sed` используется для удаления строк из вывода `socat`;
- опция `-u` указывает, что программа должна работать в некешированном (в немедленном, не буферизованном) режиме;
- опция `3d` удаляет третью строку из вывода `socat`.
- команда `stdbuf`
 - `stdbuf` с опцией `-oL` обеспечивает некешированный (немедленный) вывод стандартного потока `stdout` и стандартного ввода `stdin` для следующей команды.
- команда `cut`
 - `cut` извлекает части строк из стандартного ввода `stdin` и выводит их в стандартный вывод `stdout`;
 - опция `-d " "` указывает, что разделителем для извлеченных частей является пробел `" "`;
 - опция `-f 7` указывает, что извлеченная часть должна быть седьмой в строке.
- оператор `>` перенаправляет вывод `cut` в файл `socat.log`.
- оператор `&` запускает команду в фоновом режиме, таким образом `socat` будет выполняться в отдельном процессе, а не блокировать выполнение скрипта.
- команда `sleep` приостанавливает выполнение скрипта на 1 секунду.
- команда `cat` выводит содержимое файла `socat.log`.

3.3.3. Проверка работоспособности скрипта:

Скрипту были даны права на выполнение с последующим его запуском. Результат выполнения скрипта записан в лог-файл `socat.log`⁴.

3.4. Результат [[ссылка на содержание](#)].

- В результате вышеописанных действий была выполнена следующая работа:
- создан виртуальный последовательный порт и осуществлена работа с ним с использованием SSH-туннелей и программы `socat`;
 - создан скрипт для отладки и настройки виртуальных интерфейсов.

Приложение: иллюстративный материал на 7 л., в 1 экз.

О.В. Соколов

⁴ https://drive.google.com/open?id=10oGPzlaErXhU6CzB5WgiPzNGWWrFuRq5&usp=drive_fs

ИЛЛЮСТРАТИВНЫЙ МАТЕРИАЛ

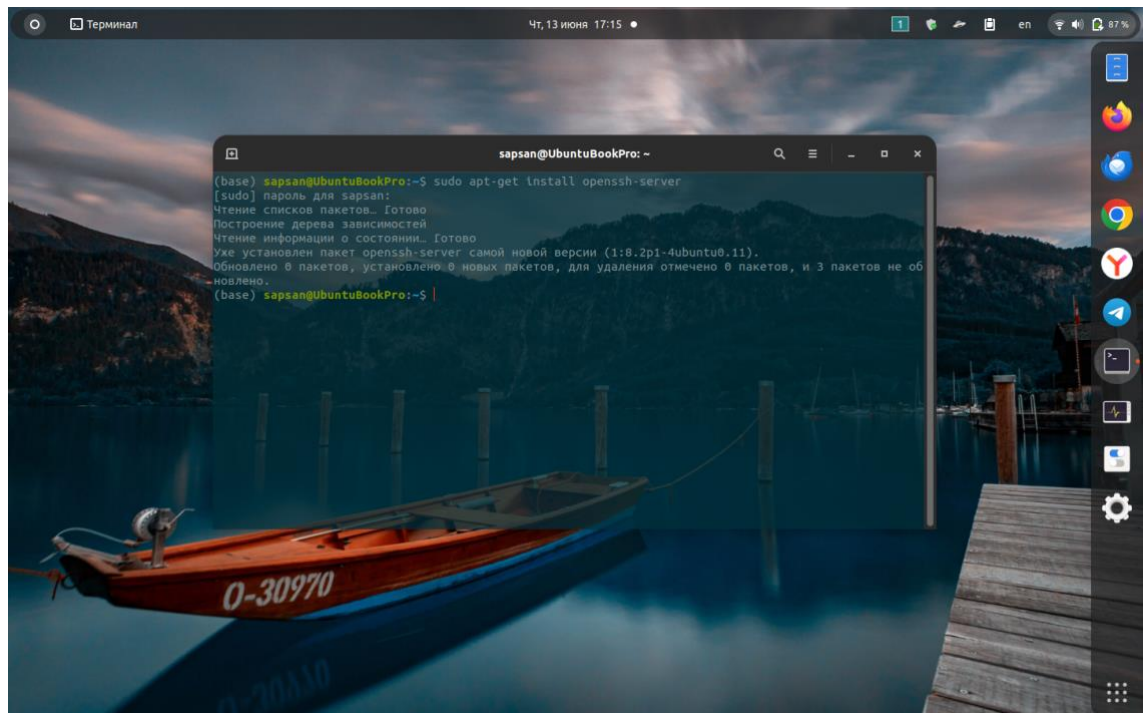


Иллюстрация 1. Установка `openssh-server` на хост-компьютер [[обратная ссылка](#)].

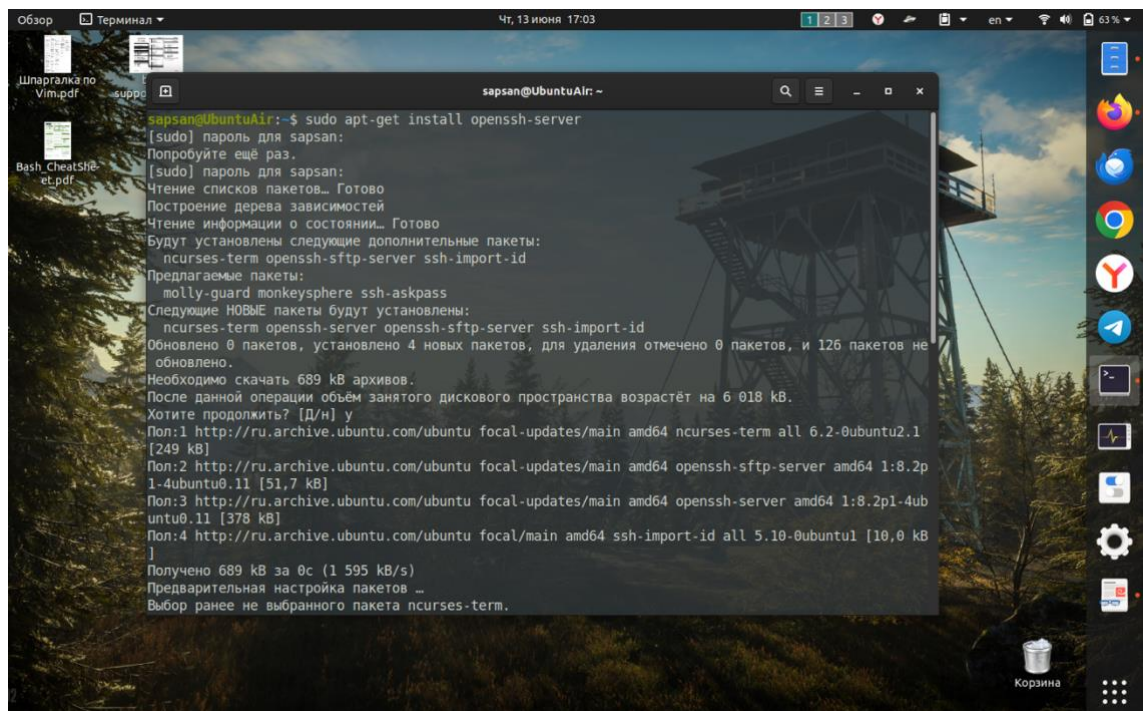


Иллюстрация 2. Установка `openssh-server` на клиент-компьютер [[обратная ссылка](#)].

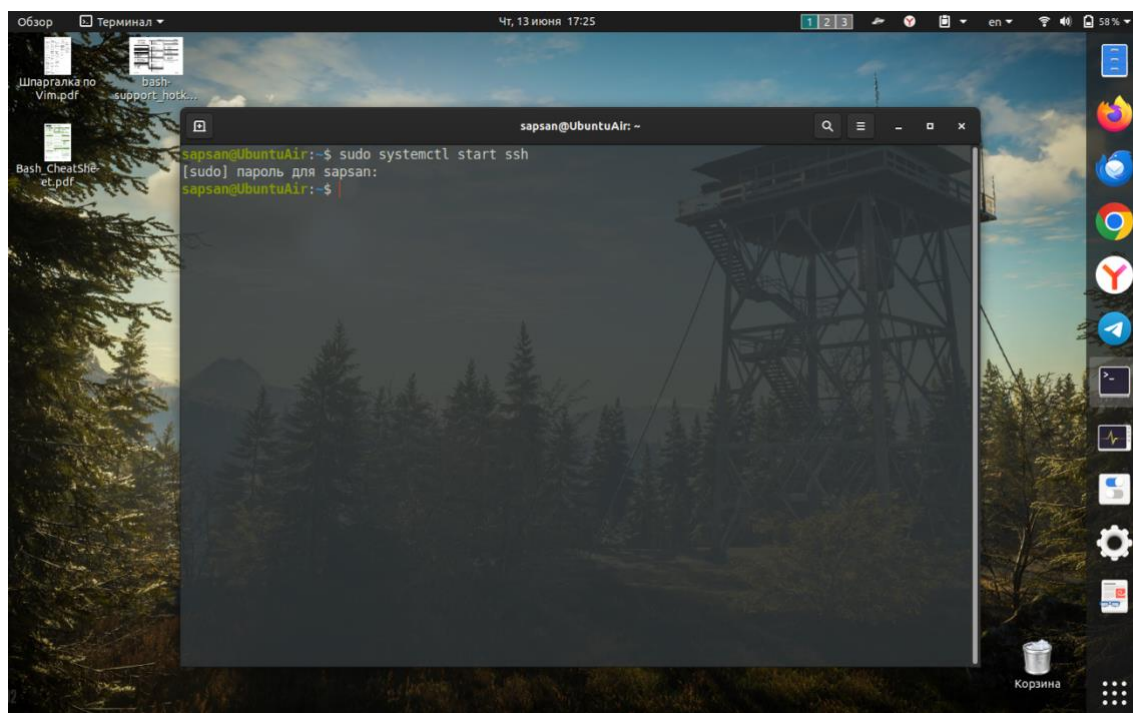


Иллюстрация 3. Настройка возможности подключения по протоколу SSH к клиент-компьютеру [[обратная ссылка](#)].

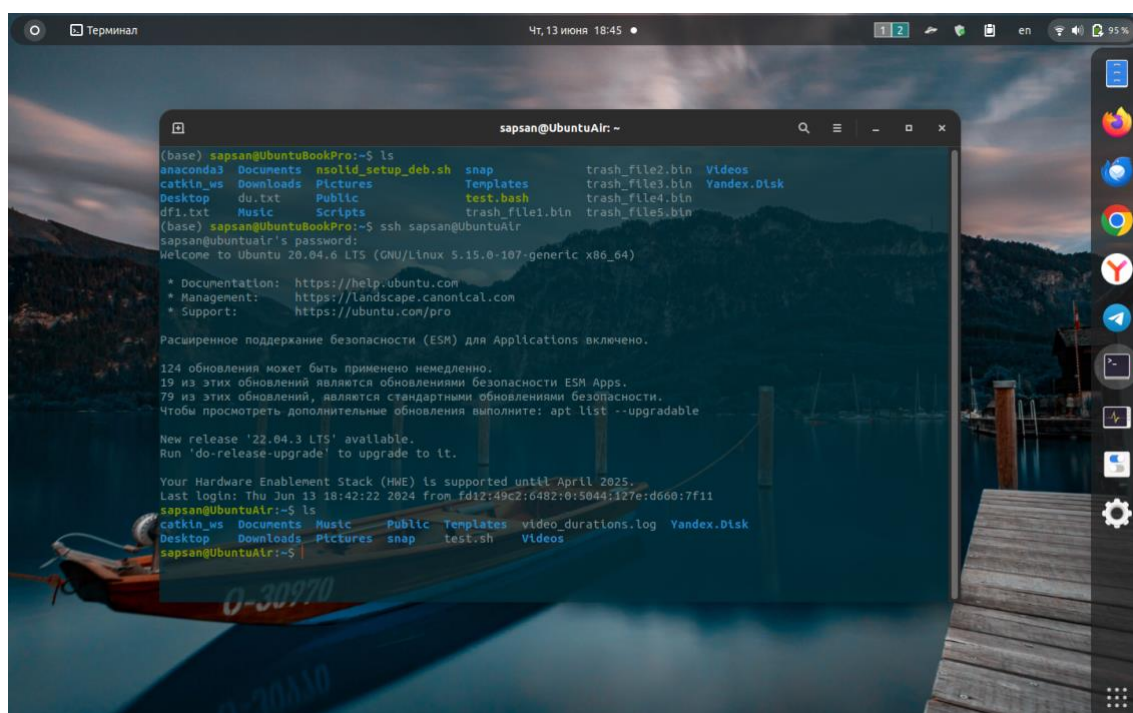


Иллюстрация 4. Подключение хоста к клиенту по SSH с иллюстрацией содержимого корневого каталога каждого компьютера [[обратная ссылка](#)].

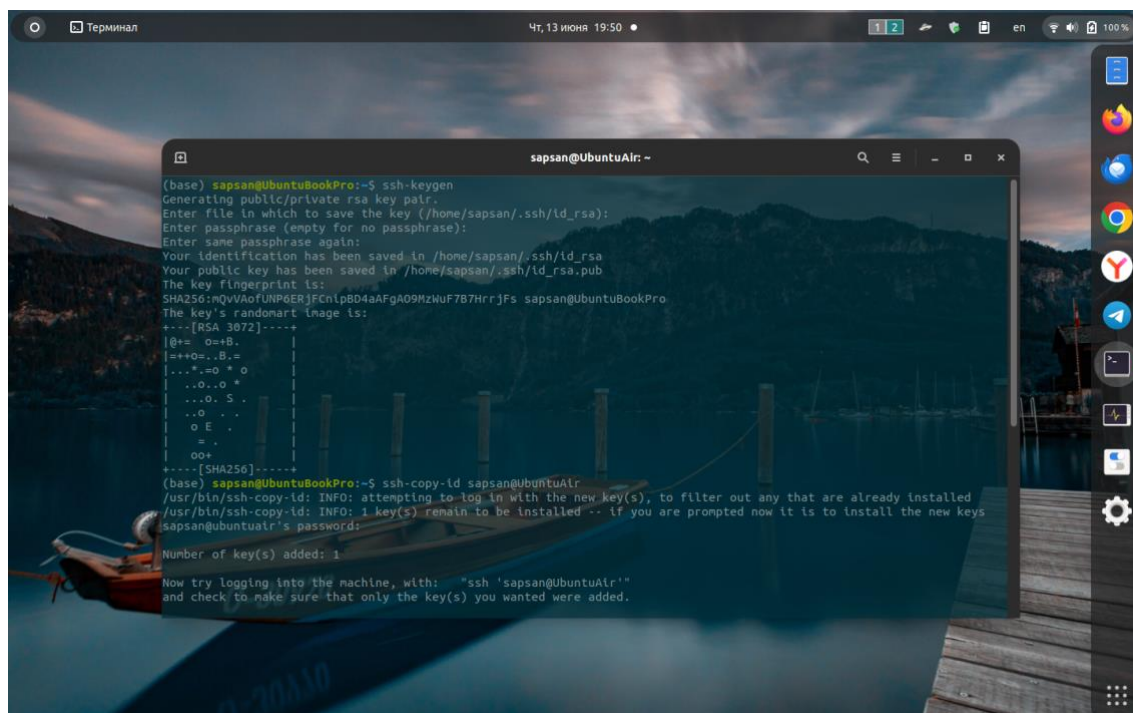


Иллюстрация 5. Создание SSH-ключей и копирование открытого ключа на клиент [[обратная ссылка](#)].

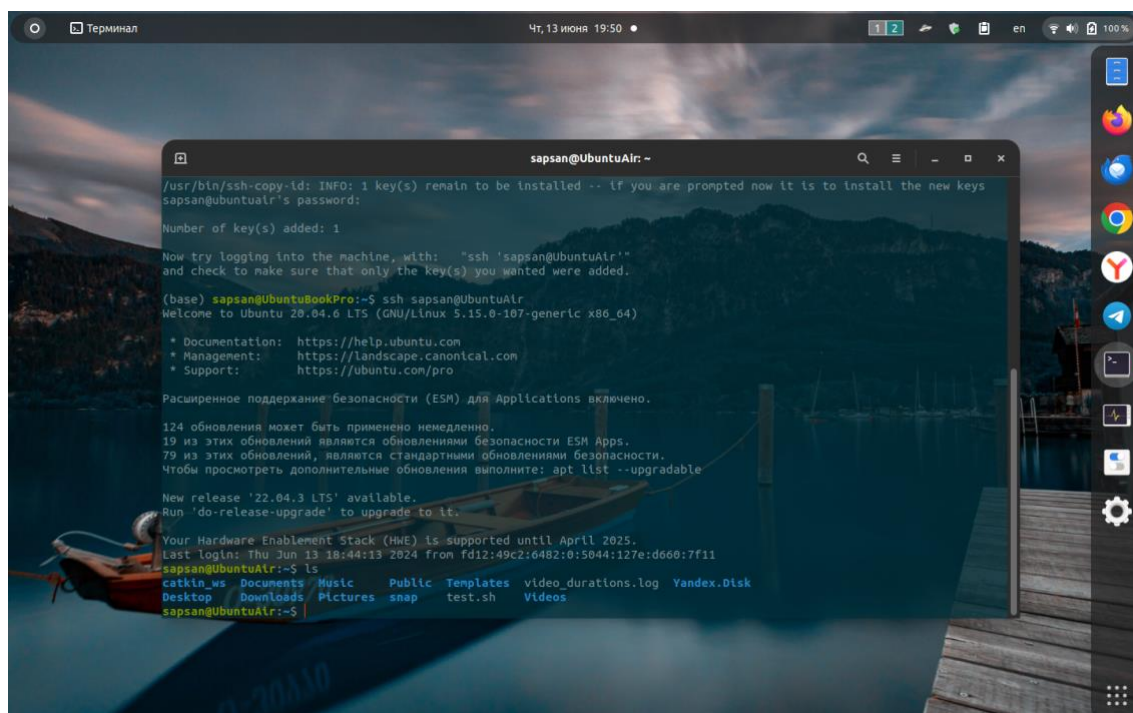


Иллюстрация 6. Подключение к клиенту с помощью открытого ключа с иллюстрацией содержимого его корневого каталога [[обратная ссылка](#)].

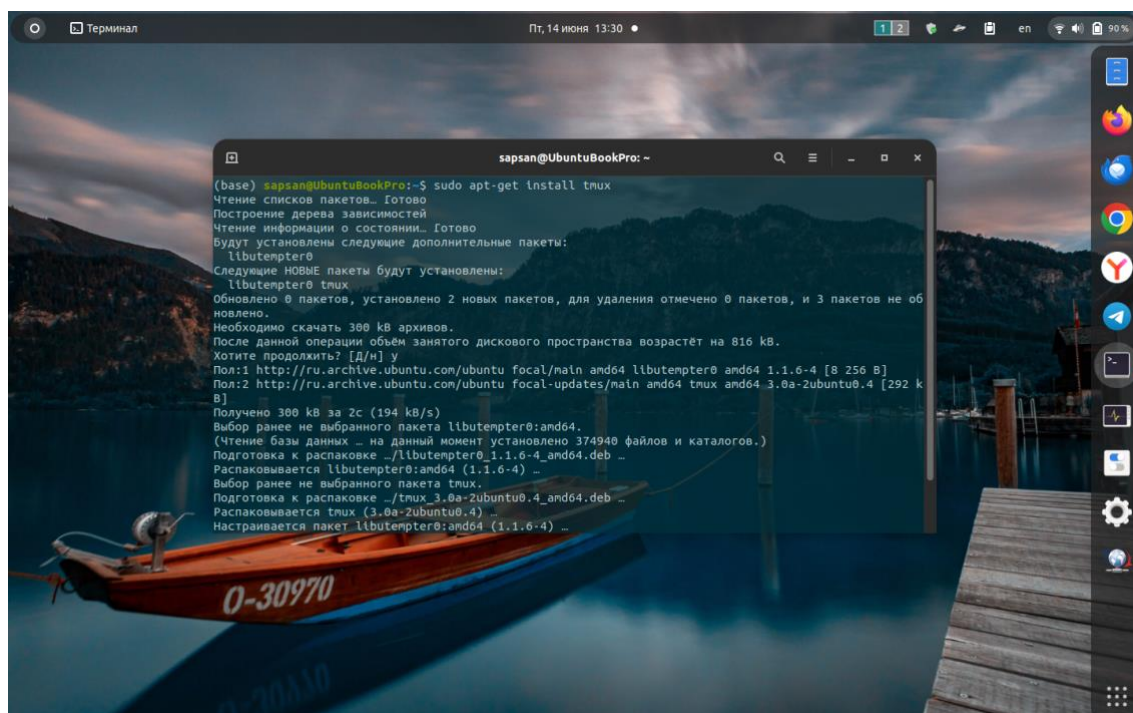


Иллюстрация 7. Установка `tmux` на хост- и клиент- компьютеры [[обратная ссылка](#)].

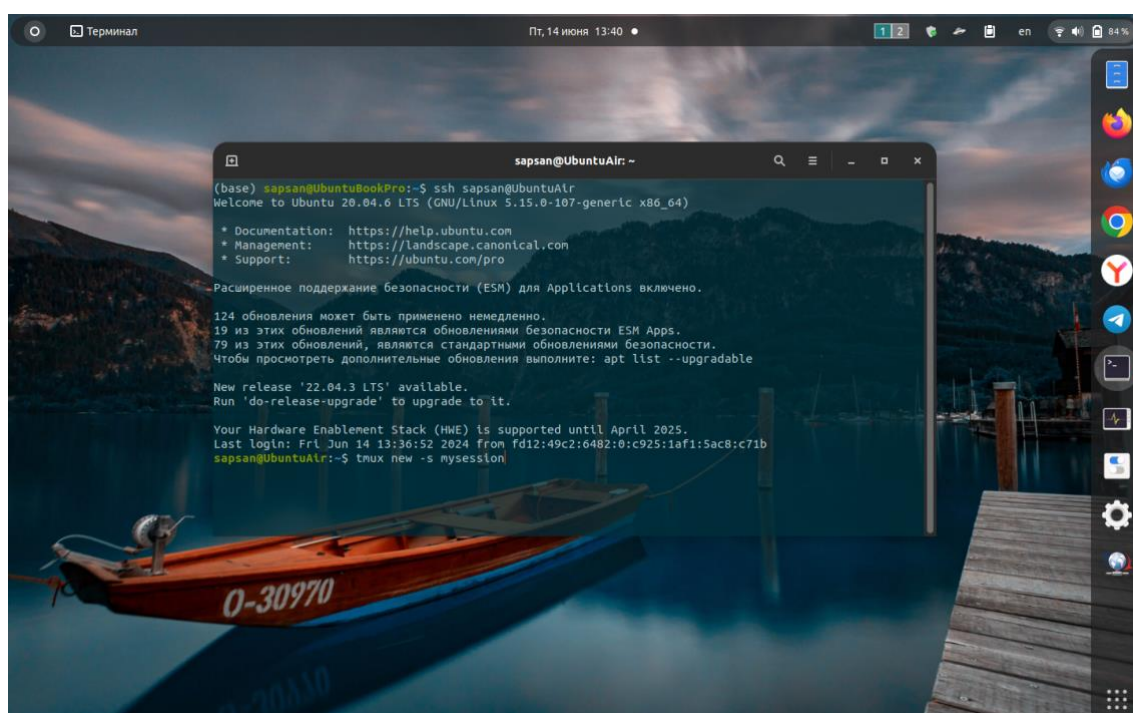


Иллюстрация 8. Запуск новой сессии `tmux` на клиенте, с использованием SSH-соединения [[обратная ссылка](#)].

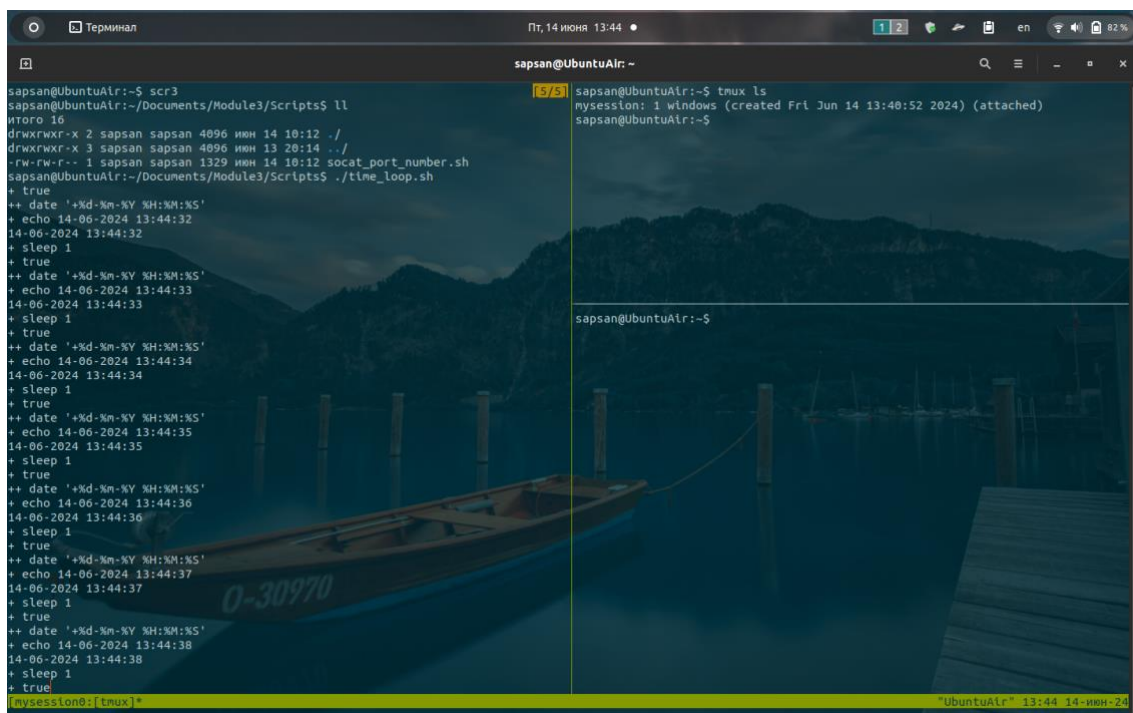


Иллюстрация 9. Запуск в отдельной панели скрипта `time_loop.sh`, с бесконечным циклом, выводящим текущее время и дату каждую секунду [[обратная ссылка](#)].

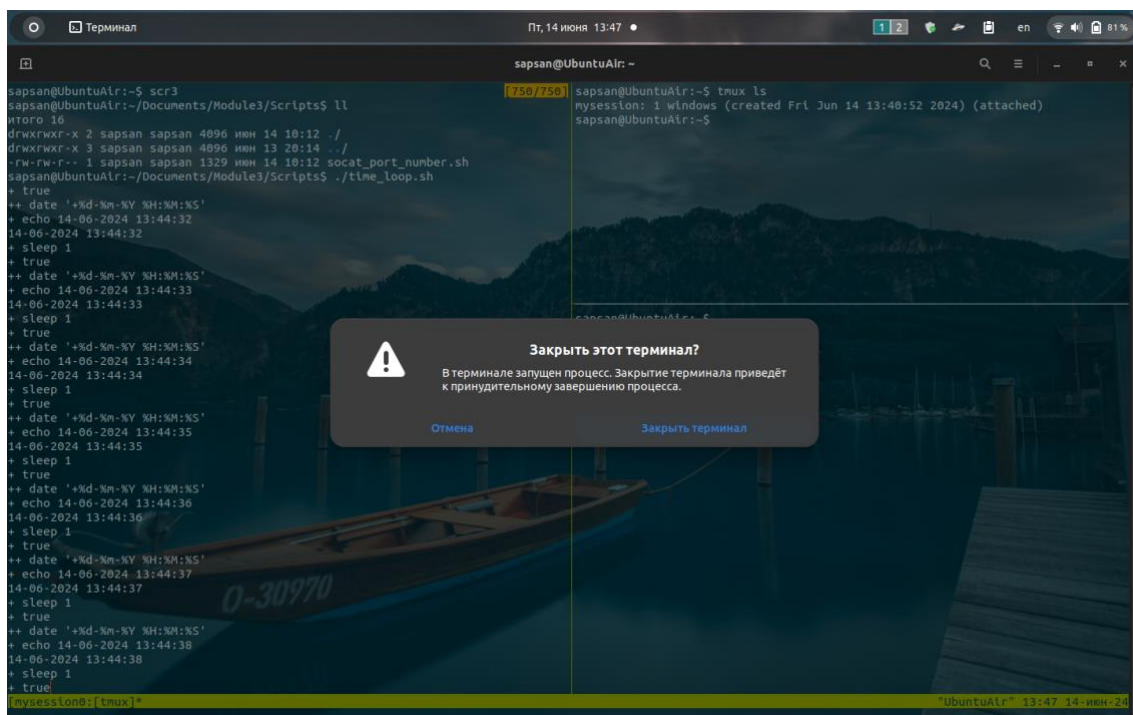


Иллюстрация 10. Имитация разрыва SSH-соединения путем закрытия окна терминала [[обратная ссылка](#)].

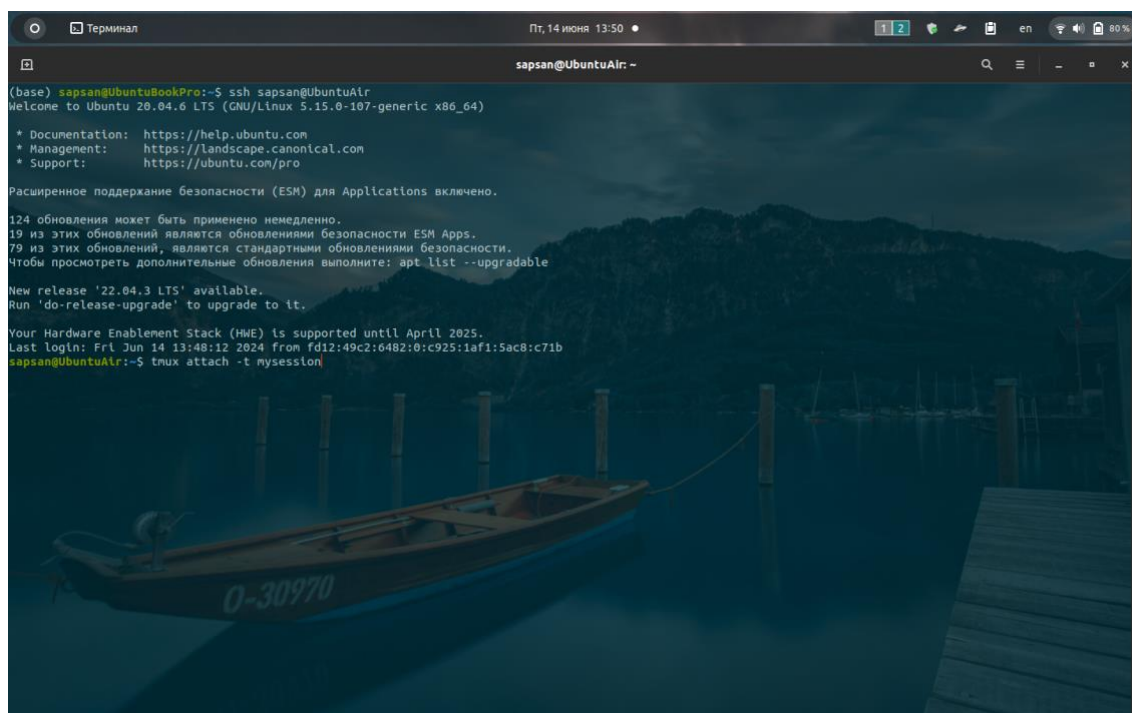


Иллюстрация 11. Повторное подключение к ранее закрытой сессии [[обратная ссылка](#)].

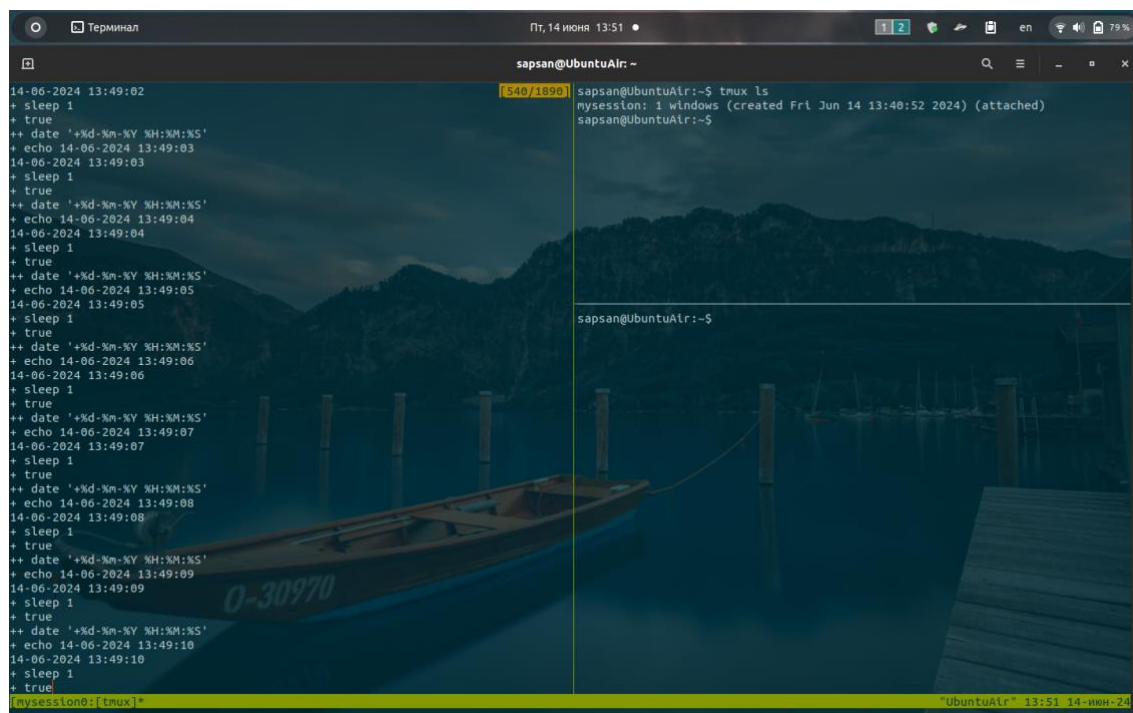


Иллюстрация 12. Продолжение работы запущенного скрипта после разрыва и восстановления SSH-соединения [[обратная ссылка](#)].

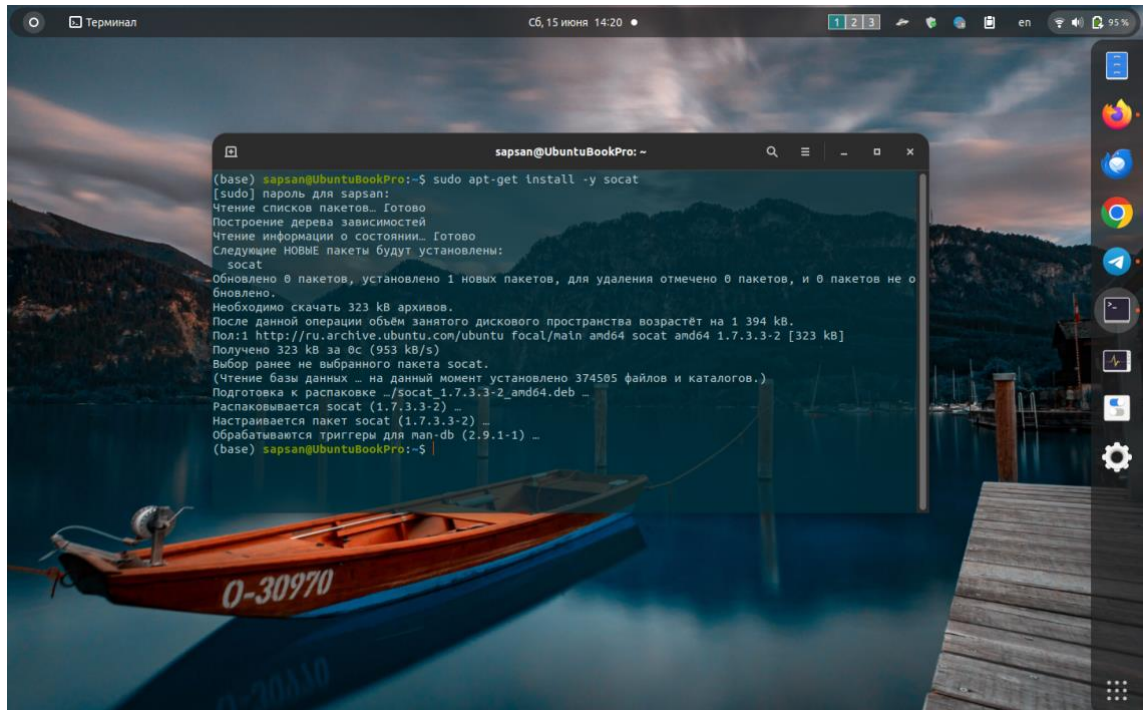


Иллюстрация 13. Установка `socat` на хост- и клиент- компьютеры [[обратная ссылка](#)].

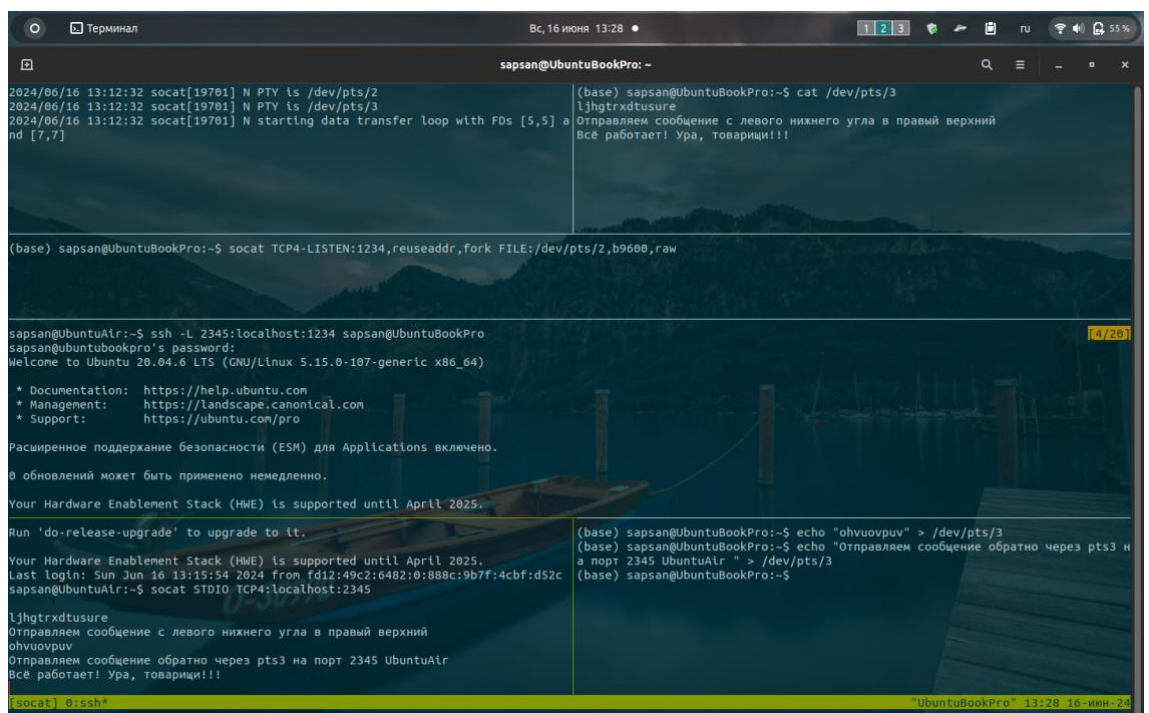


Иллюстрация 14. Создание виртуального последовательного порта и работа с ним с использованием SSH-туннелей и программы `socat` на хост- и клиент- компьютерах [[обратная ссылка](#)].