МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ КЫРГЫЗСКОЙ РЕСПУБЛИКИ

КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМ. И. РАЗЗАКОВА

Кафедра «Программное обеспечение компьютерных систем»

Методические указания к выполнению лабораторных работ по дисциплине «Коммуникационные средства автоматизированных систем» для студентов по направлению: 590100 "Информационная безопасность" профиль «Безопасность автоматизированных систем»

Бишкек - 2015

«Рассмотрено»
На заседании кафедры
«ПОКС»
Прот. №17 от 30.04.2015

«Одобрено»
Методическим советом
ФИТ
Прот.№12 от 12.05.2015

УДК 004.7

Составитель: Стамкулова Г.К., Вагнер А.А.

Методические указания к лабораторным работам по дисциплине «Коммуникационные средства автоматизированных систем» //КГТУ им. И. Раззакова; Сост.: Стамкулова Г.К., Вагнер А.А. / - Б.: ИЦ «Текник», 2015. - 30 с.

Представлены задания и подробные методические указания к выполнению лабораторных работ по автоматизации информационных процессов и процессов управления объектами различного назначения. Предназначено для студентов направления 590100 «Информационная безопасность» профиль «Безопасность автоматизированных систем»

Рисунков:18

Библиогр. 4 названий.

Рецензент: к.т.н., доцент кафедры ПОКС Мусина И.Р.

Корректор ***Эркинбек к. Ж.***
Редактор ***Турдукулова А.К.***
Тех.редактор ***Кочоров А.Д***

Подписано к печати 11.08.2015 г. Формат бумаги 60x84 $^{1}/_{16}$.
Бумага офс. Печать офс. Объем 3 п.л. Тираж 30 экз. Заказ 361. Цена 51,3с.

Бишкек, ул. Сухомлинова, 20. ИЦ “Текник” КГТУ им. И.Раззакова, т.: 54-29-43
e-mail: beknur@mail.ru

Оглавление

Ведение

Автоматизация в общем смысле заключается в применении технических средств, экономико-математических методов и систем управления, освобождающих человека частично или полностью от непосредственного участия в процессах получения, преобразования, передачи и использования энергии, материалов или информации.

В настоящем методическом указании рассмотрена понятия в аспекте автоматизации информационных процессов и процессов управления объектами различного назначения.

Современные информационные технологии представляют широкий набор способов реализации АСОИУ (автоматизированная система обработки информации и управления) в общем смысле представляет собой некоторую систему обработки данных, основанную на использовании ЭВМ и связанную с управлением теми или иными объектами (предприятиями, организациями, технологическими процессами).

Методическое указание дает студентам понятие АСОИУ рассматривается с точки зрения совокупности взаимодействующих компонентов разного уровня, обеспечивающих соответствующую функциональность.

В работе студенты имеют возможность получить знания о функции АС — это совокупность действий системы, направленных на достижение определенной частной цели (подцели) управления. Каждая система выполняет одну целевую функцию и ноль или более вспомогательных функций (подфункций).

Приведены различные способы классификации автоматизированных систем обработки информации и управления АСОИУ.

Лабораторная работа №1
Обзор программного обеспечения используемого в лабораторных работах

Цель:

Изучить программное обеспечение, которое применяется для выполнения данных ЛР.

В данных лабораторных работах будут использоваться следующие программные продукты:

VirtualBox 4.13.X — виртуальная машина, это программное обеспечение эмулирующее ПК. На данном программном обеспечении легко обучиться работе на разных ОС а так же их установке и настройке. В данных ЛР будет использоваться ВМ для Linux. Работа на версии для windows аналагична, так как VirtualBox крос-платформенное ПО.

https://www.virtualbox.org/

ОС GNU/Linux Debian 7 wheezy — операционная система основанная на ядре Linux. Этот дистрибутив является свободным программным обеспечением, отличается тем, что в состав этого дистрибутива входят наиболее старые версии ПО но и более стабильные. Расширения пакетов .deb

https://www.debian.org/index.ru.html

ОС OpenSUSE 12.1 — дистрибутив основанный на ядре Linux. Разрабатывается американской компанией Novel. Отличительной чертой является мэнеджер настройки и управления пакетами YAST.

https://ru.opensuse.org/%D0%94%D0%BE%D0%B1%D1%80%D0%BE_%D0%BF%D0%BE%D0%B6%D0%B0%D0%BB%D0%BE%D0%B2%D0%B0%D1%82%D1%8C_%D0%BD%D0%B0_openSUSE.org

Apache2 — самый популярный веб сервер, более 50% интернет ресурсов работают на этом сервере.

http://www.apache.ru/

Php5 — интерпретатор языка программирования php. Этот язык наиболее подходит для обучения программированию так как достаточно легок в освоении.

http://php.net/

MySQL — СУБД от компании ORACLE. Наиболее подходит для работы с php.

http://www.mysql.com/

GlassFish — сервер приложений для языка Java реализующий EJB технологию.

https://glassfish.java.net/downloads/v2.1.1-final.html

OpenSSH — сервер протокола SSH.
http://www.openssh.com/

MySQLWorkbench — среда для разработки и администрирования сервера MySQL
http://www.mysql.com/products/workbench/

Putty — ssh — клиент для Windows

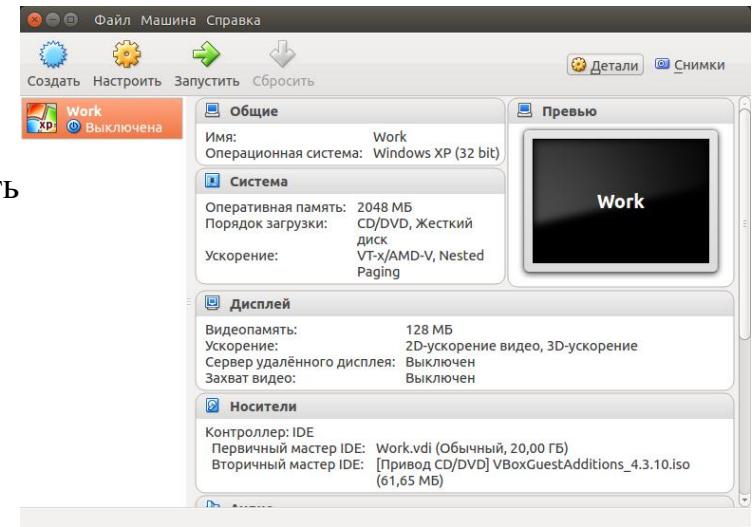
WinSCP — scp клиент для Windows

Контрольные вопросы
Что такое Операционная система Debian
Что такое Linux
Чем отличаются Debian от OpenSUSE
Зачем нужен Apache
Что делает VirtualBox
Что такое php
Что такое MySQL
Для чего нужна MySQL Workbench
Что такое Open SSH Server Open SSH Client
Что такое GlassFish

Лабораторная работа №2
Установка *nix подобной системы на примере Ubuntu 14.04

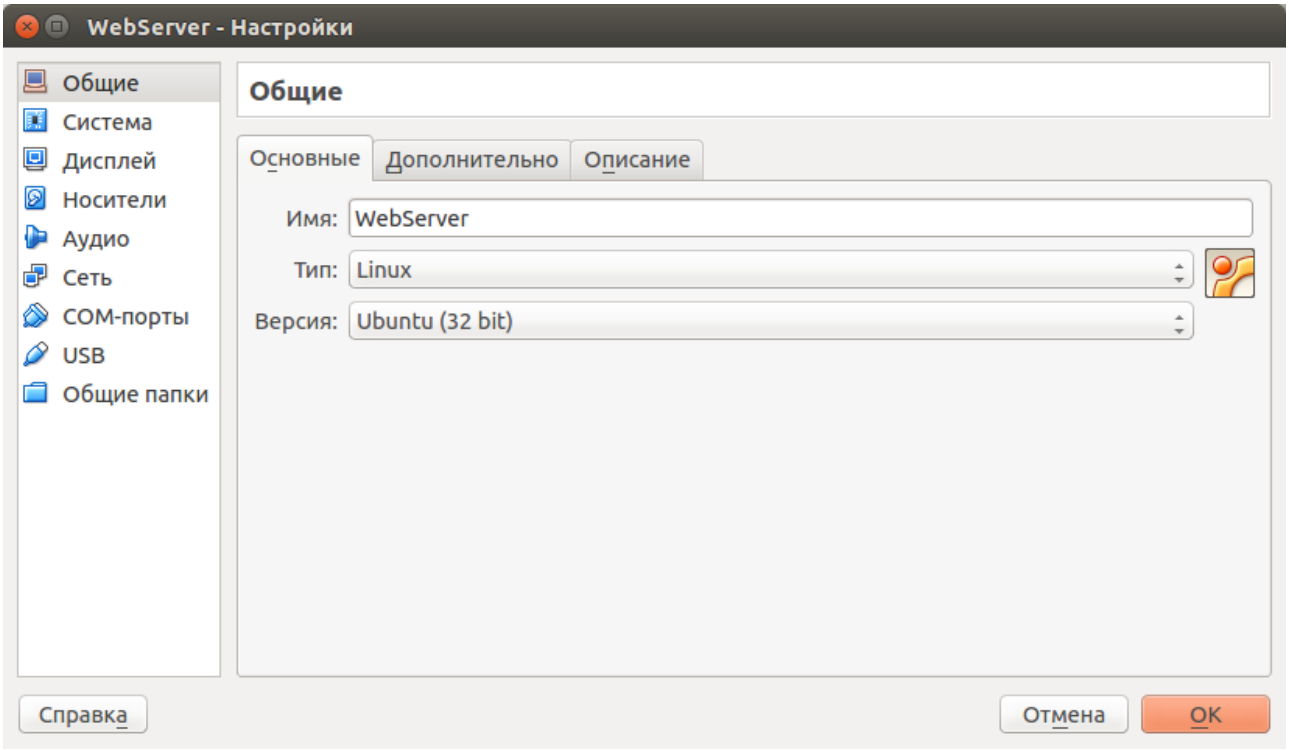
Ниже проилюстрированно как установить на виртуальную машину ОС Ubuntu 14.04

1 Запустите VirtualBox



нажмите на кнопку создать

заполните поля



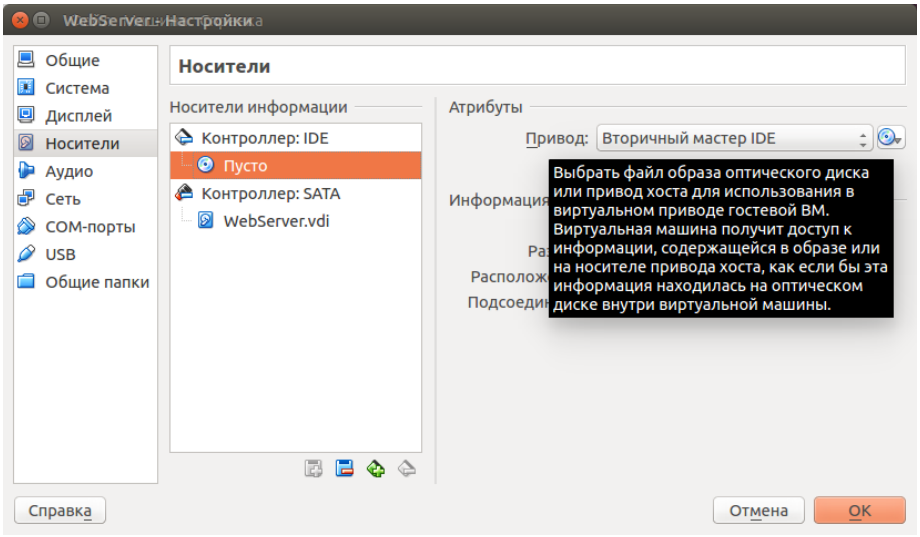
и нажмите Далее

укажите размер ОЗУ доступной ВМ (исходя из реального объема ОЗУ)

далее создайте жесткий диск
тип VDI, формат хранения Динамический, размер от 10 Гб
после этого нажмите на кнопку создать

Далее зайдите в настройки, Выделите созданную виртуальную машину и нажмите на кнопку настроить. Установите ОЗУ 1024 МБ, видео память 128 МБ.

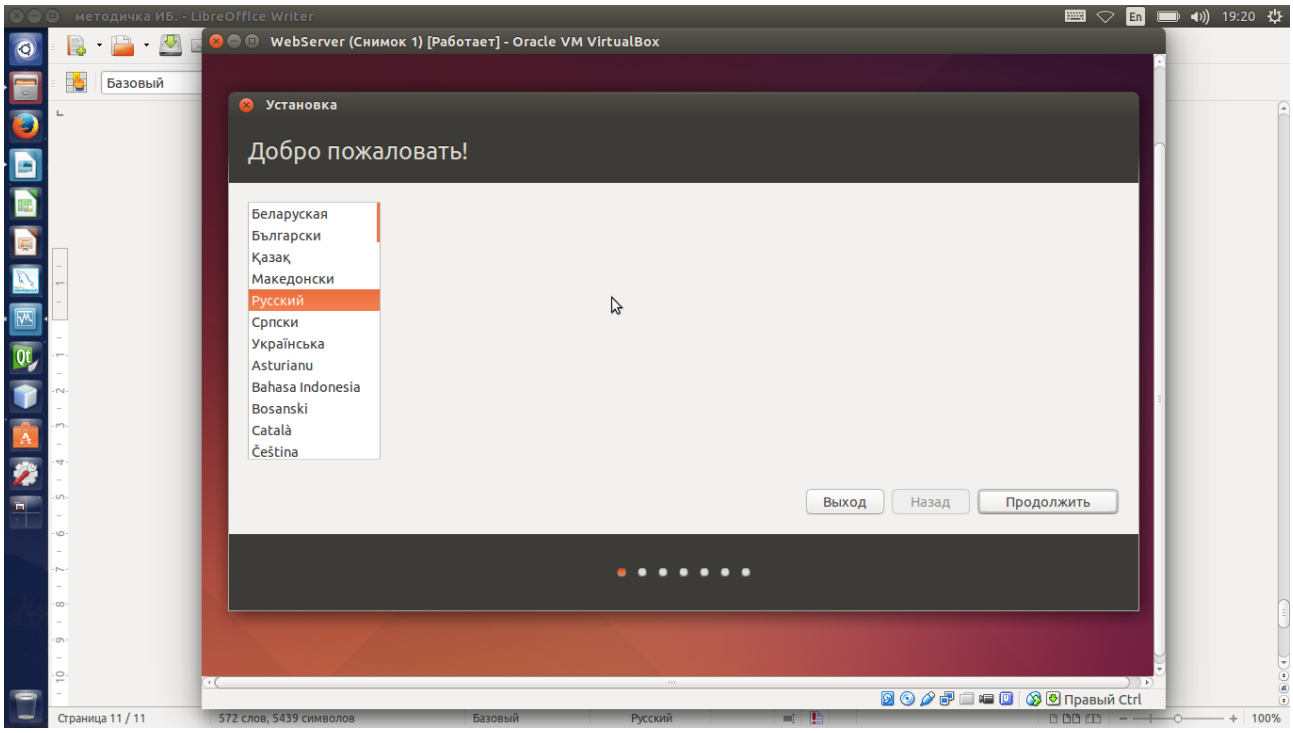
Выберите образ диска Ubuntu

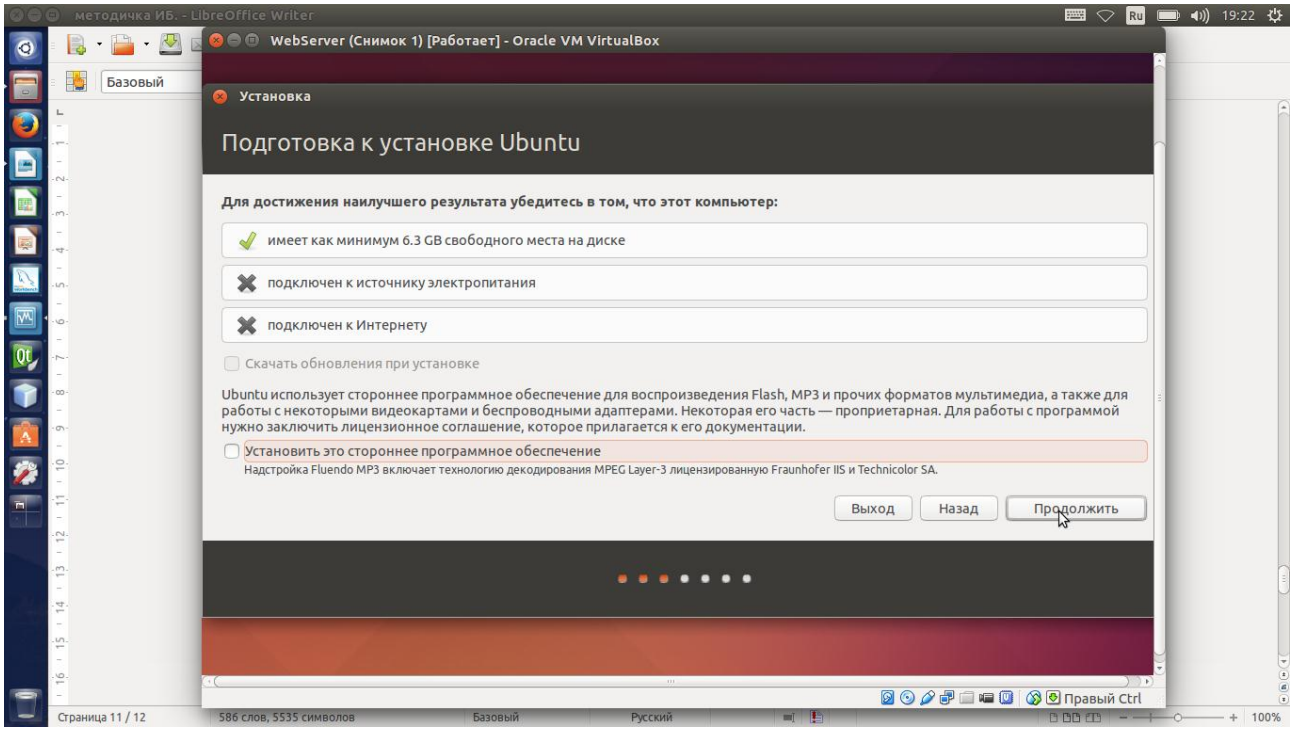


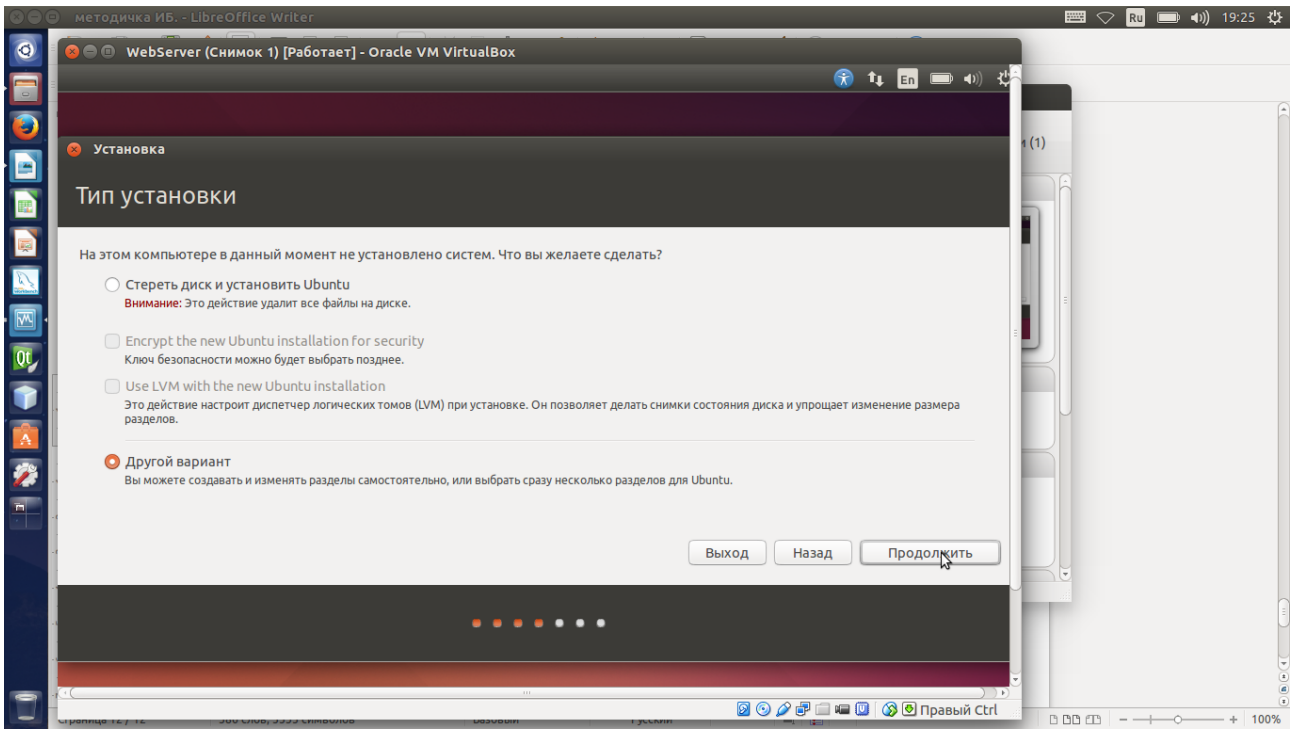
Далее нажмите кнопку запустить.

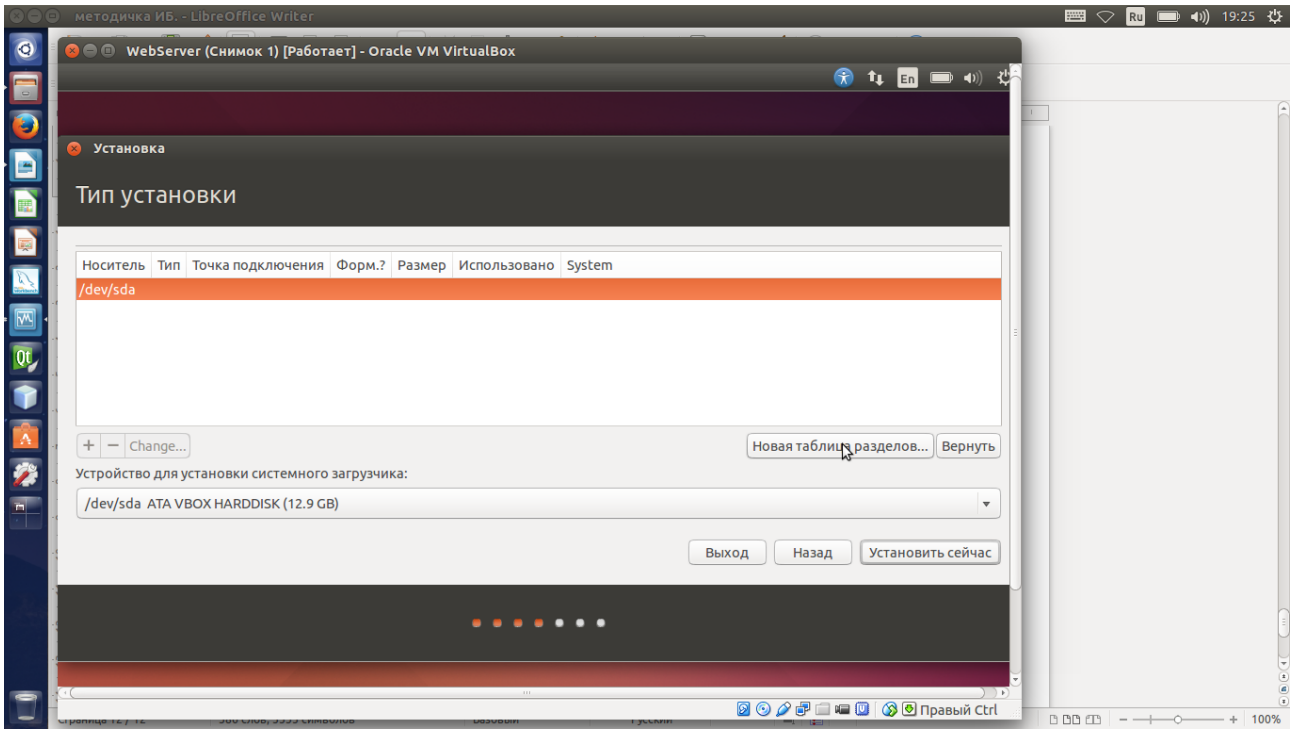
После выберите язык установки и выберите пункт меню Установить Ubuntu

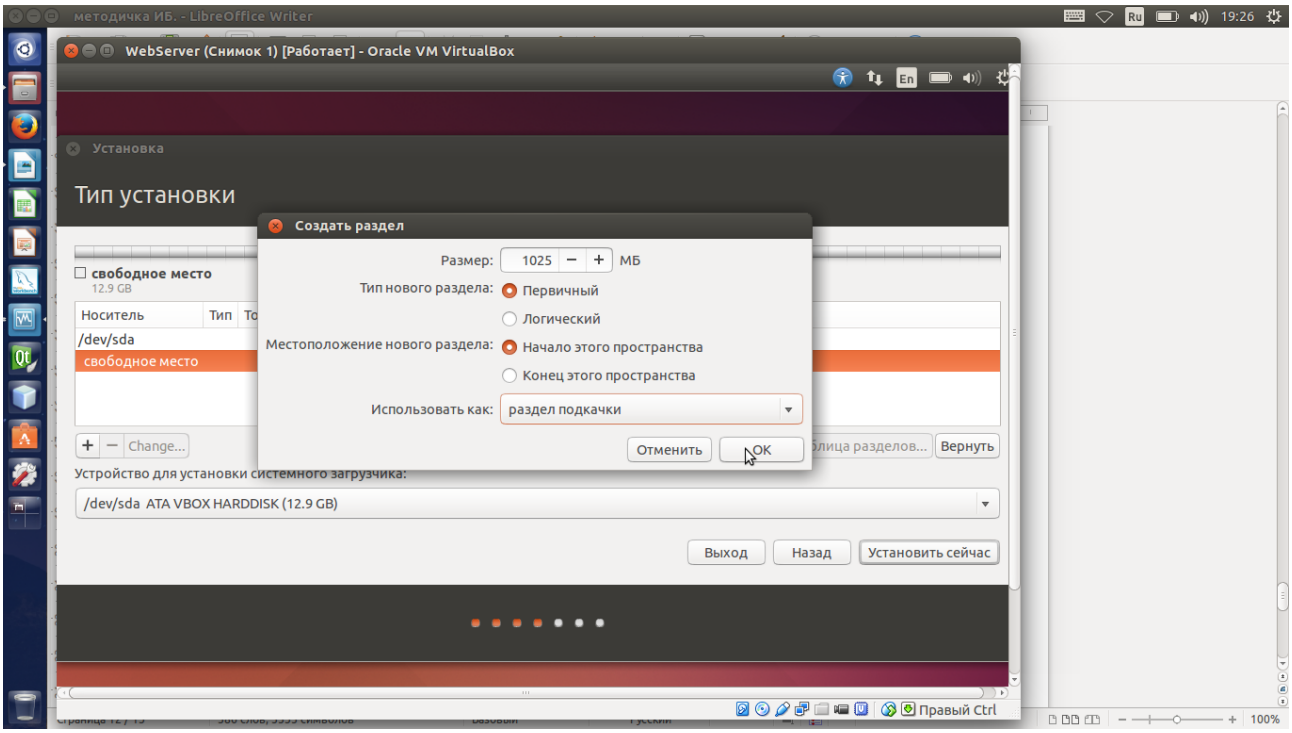
далее выберите язык системы

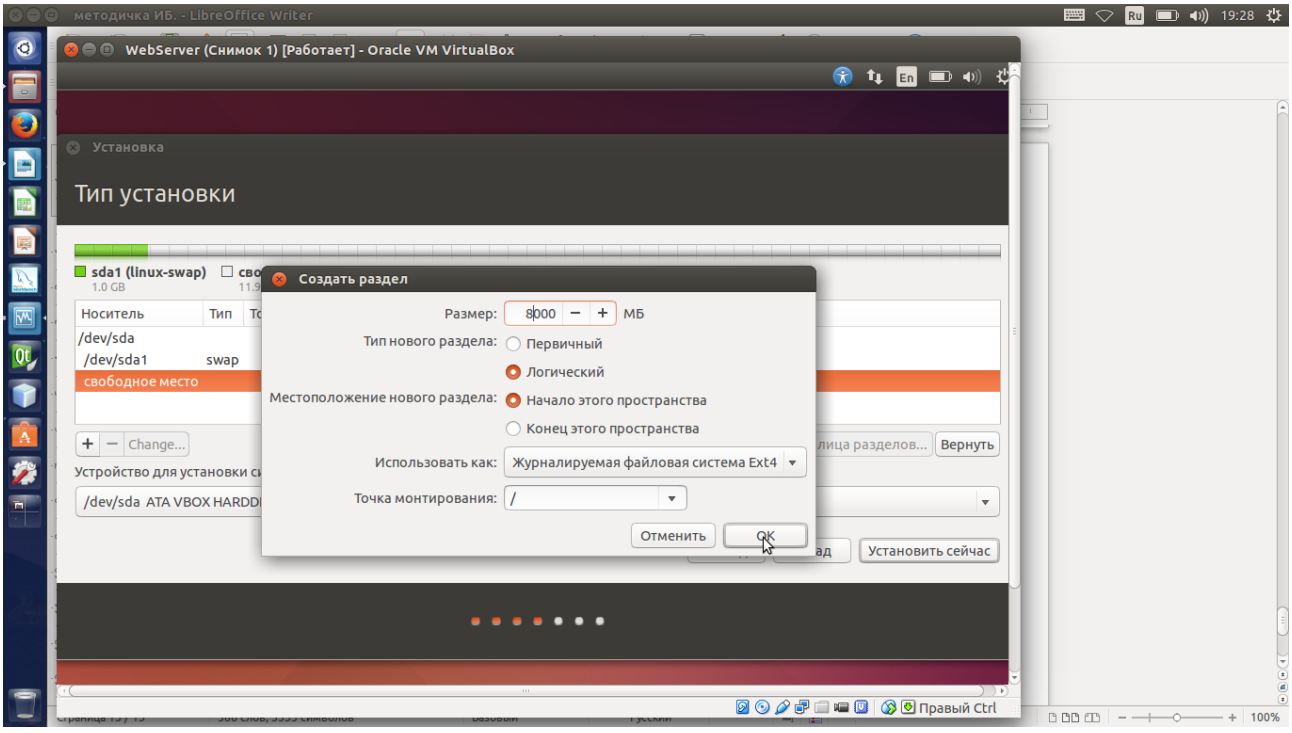


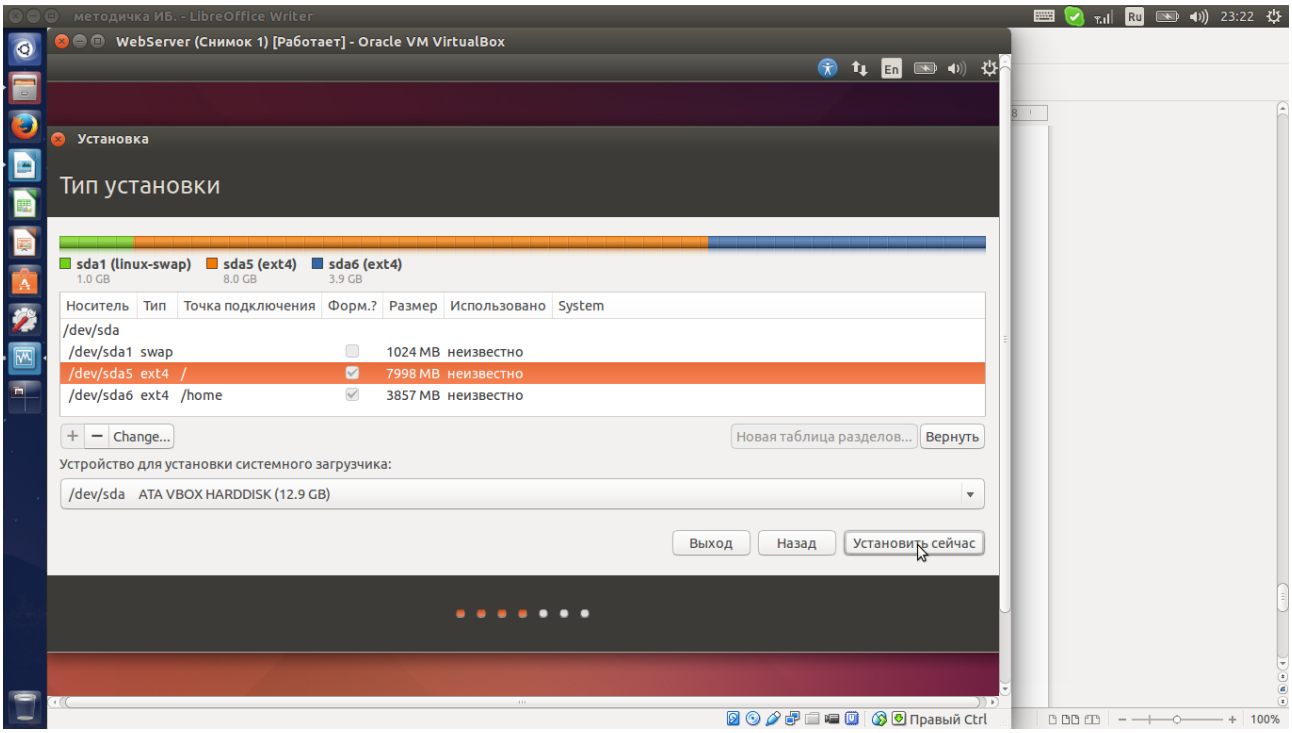



методичка ИБ. - LibreOffice Writer
WebServer (Снимок 1) [Работает] - Oracle VM VirtualBox
Установка
Тип установки
На этом компьютере в данный момент не установлено систем. Что вы желаете сделать?
Стереть диск и установить Ubuntu
Внимание: Это действие удалит все файлы на диске.
Encrypt the new Ubuntu installation for security
Ключ безопасности можно будет выбрать позднее.
Use LVM with the new Ubuntu installation
Это действие настроит диспетчер логических томов (LVM) при установке. Он позволяет делать снимки состояния диска и упрощает изменение размера разделов.
Другой вариант
Вы можете создавать и изменять разделы самостоятельно, или выбрать сразу несколько разделов для Ubuntu.
Выход
Назад
Продолжить
Правый Ctrl
100%

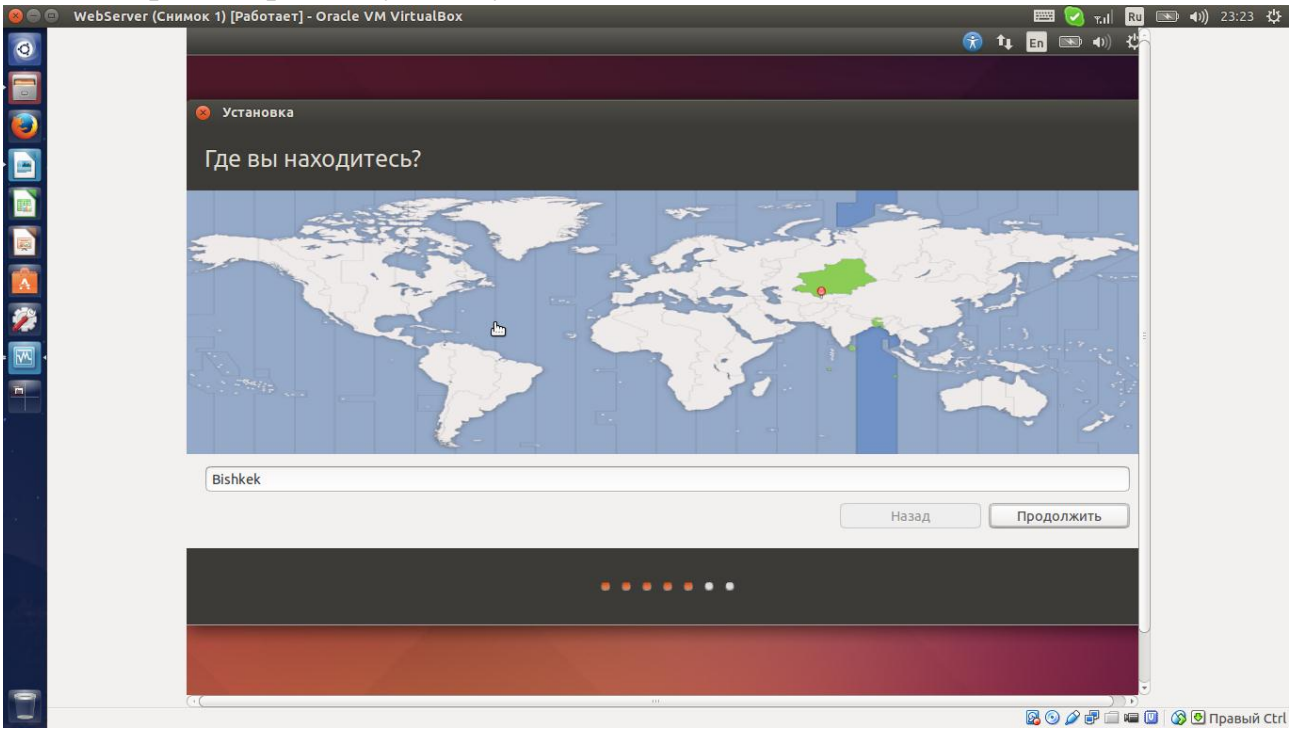

методичка ИБ. - LibreOffice Writer
WebServer (Снимок 1) [Работает] - Oracle VM VirtualBox
Установка
Тип установки
Носитель
Тип
Точка подключения
Форм.?
Размер
Использовано
System
/dev/sda
Change...
Новая таблица разделов...
Вернуть
Устройство для установки системного загрузчика:
/dev/sda ATA VBOX HARDDISK (12.9 GB)
Выход
Назад
Установить сейчас
Правый Ctrl
100%


методичка ИБ. - LibreOffice Writer
19:26
WebServer (Снимок 1) [Работает] - Oracle VM VirtualBox
Установка
Тип установки
Создать раздел
Размер: 1025 МБ
Тип нового раздела: Первичный
Логический
Местоположение нового раздела: Начало этого пространства
Конец этого пространства
Использовать как: раздел подкачки
Отменить
OK
свободное место
12.9 GB
Носитель
Тип
/dev/sda
свободное место
Change...
Вернуть
Устройство для установки системного загрузчика:
/dev/sda ATA VBOX HARDDISK (12.9 GB)
Выход
Назад
Установить сейчас
Правый Ctrl
100%

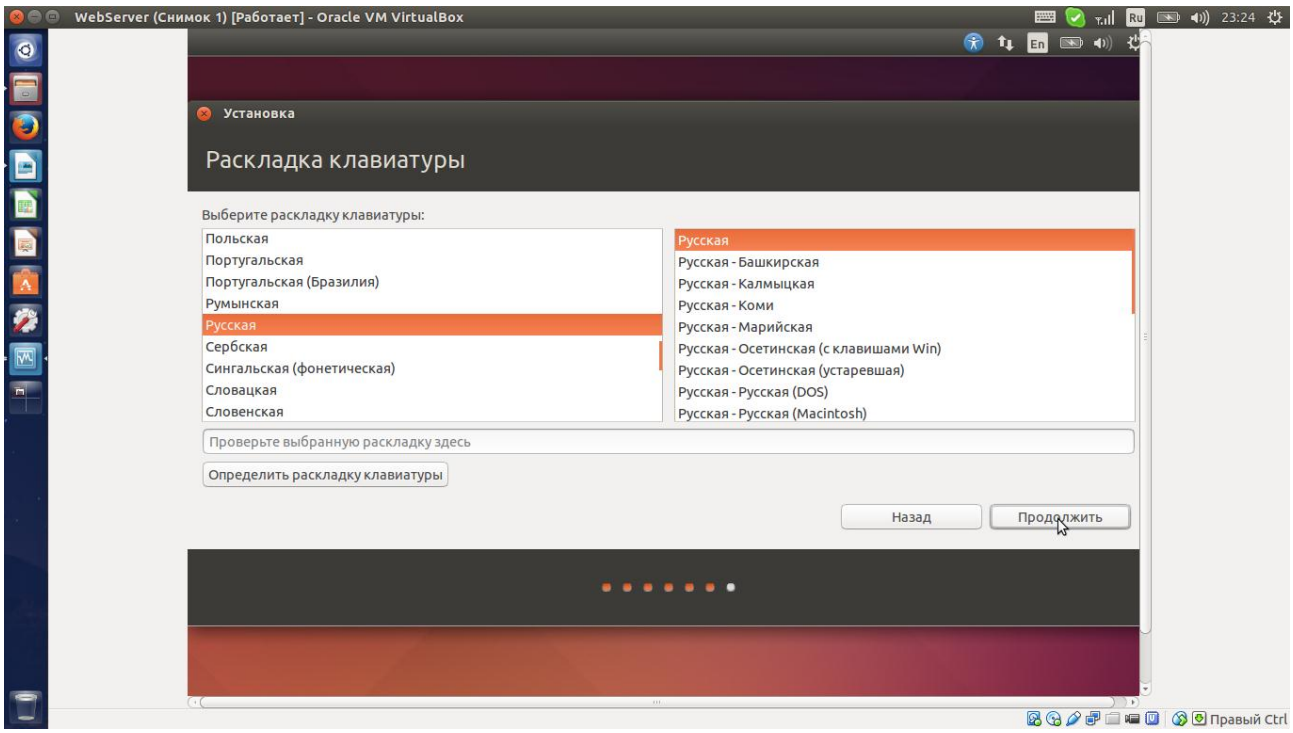

методичка ИБ. - LibreOffice Writer
19:28
WebServer (Снимок 1) [Работает] - Oracle VM VirtualBox
Установка
Тип установки
sda1 (linux-swap)
1.0 GB
Создать раздел
Размер: 8000 МБ
Тип нового раздела: Первичный
Логический
Местоположение нового раздела: Начало этого пространства
Конец этого пространства
Использовать как: Журналируемая файловая система Ext4
Точка монтирования: /
Отменить
OK
Носитель
Тип
/dev/sda
/dev/sda1 swap
свободное место
Change...
Вернуть
Установить сейчас
Правый Ctrl
100%



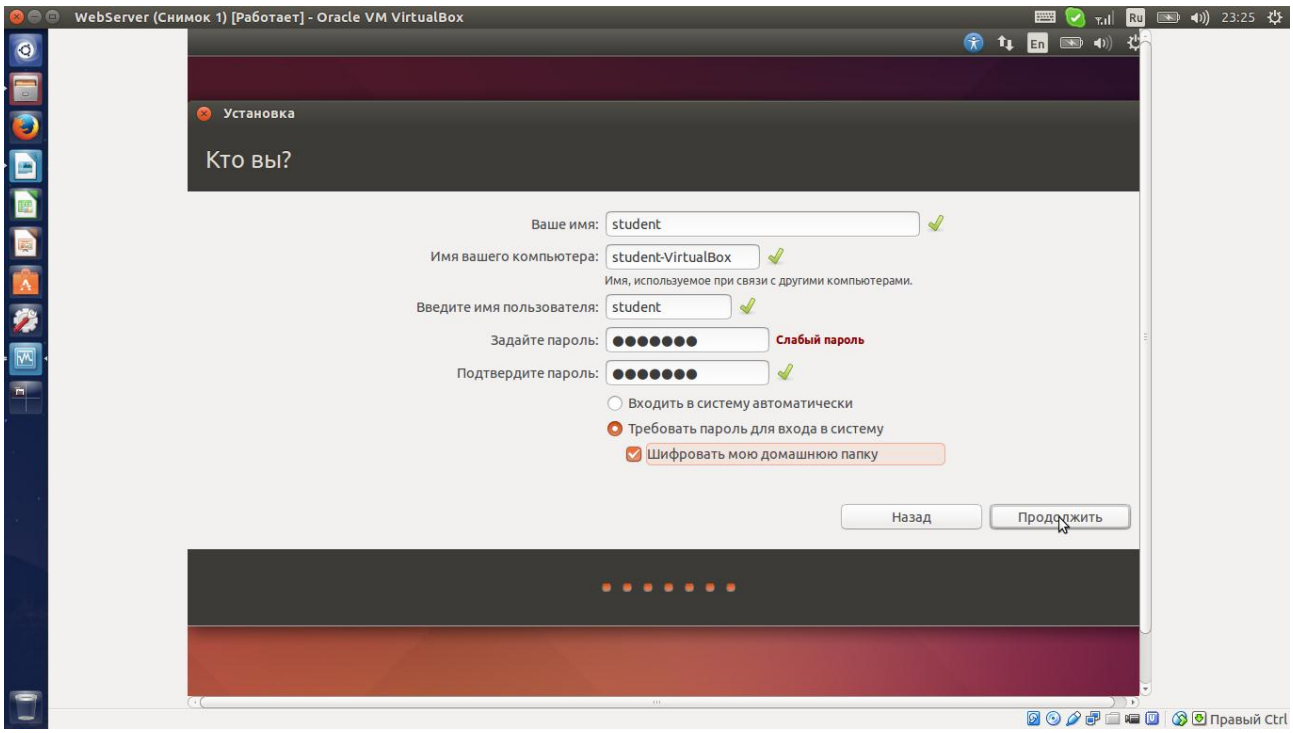
Далее настройте временную зону



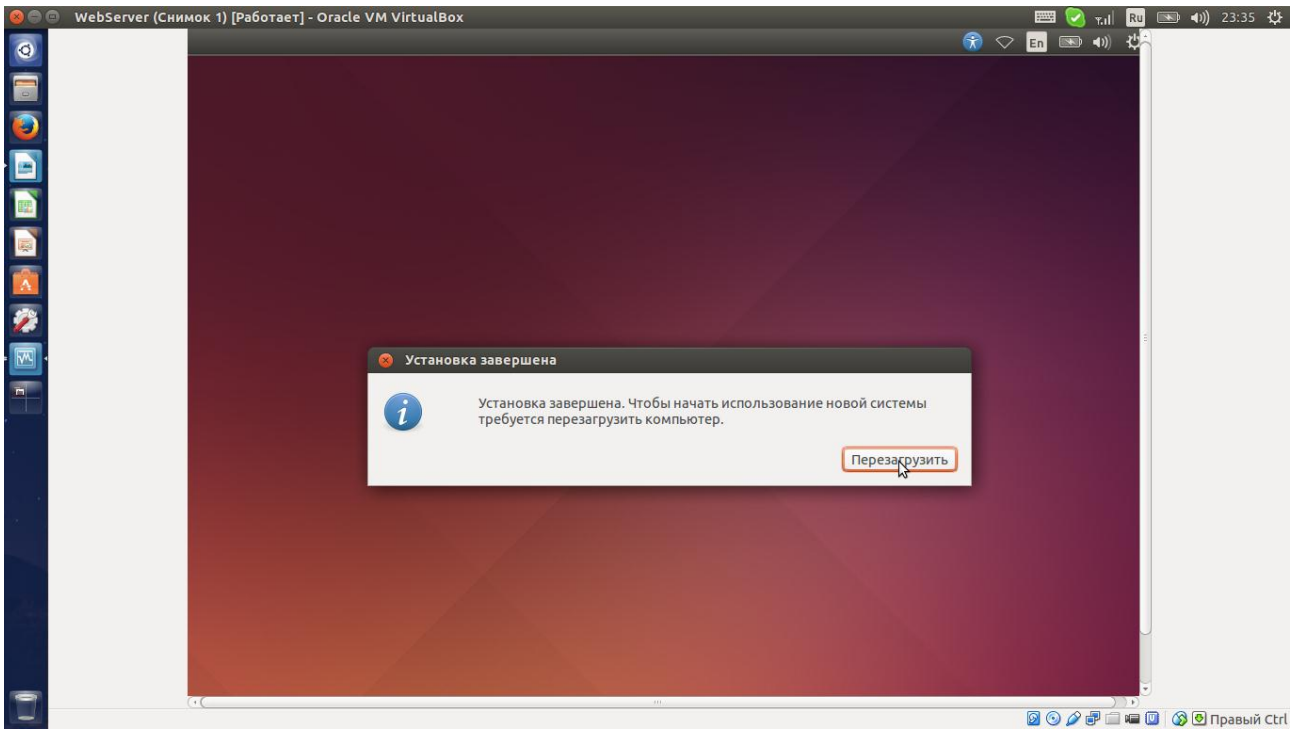
Раскладку клавиатуры



далее введите имя пользователя и пароль



После этого начнется установка системы, во время установки вы сможете почитать общие сведения о системе. **Примичание:** Если вы указали в настройках виртуальной машины использовать сеть, то во время установки скачиваются последние обновления системы.



Все готово система установилась, в следующей лабораторной приступим к основной ее настройке.

Задания для самостоятельной работы:
Установить самостоятельно на компьютер OpenSUSE.

Лабораторная работа №3
Установка и настройка веб сервера apache СУБД MySQL интерпритатора php

Цель: Освоить установку программного обеспечения в среде Linux и осуществить базовую настройку системы и ПО.

Ubuntu использует репозитории программных пакетов для установки ПО. Репозиторий это веб ресурс который предоставляет пакеты ПО которые можно скачать по протоколу http.

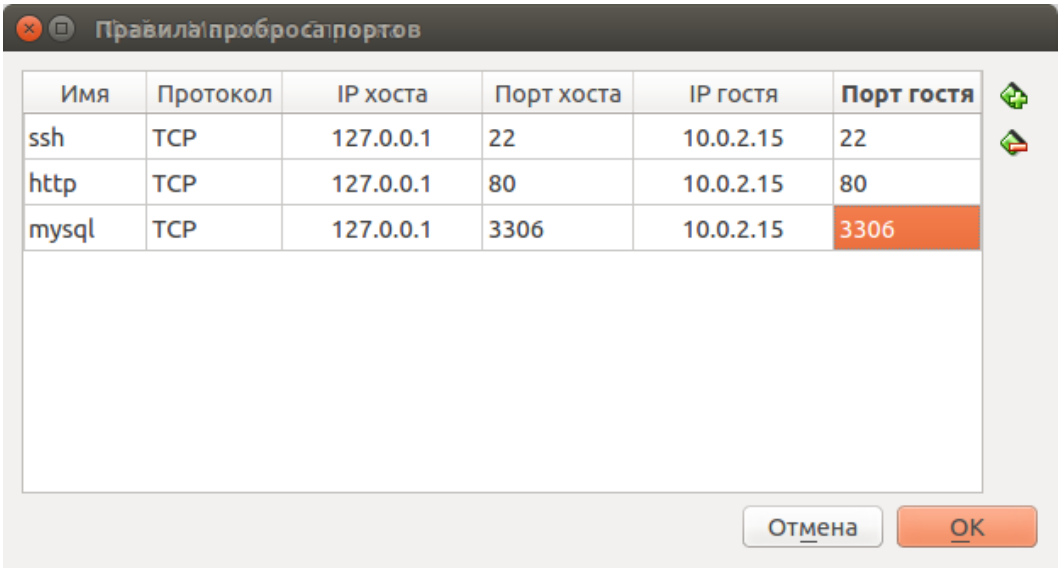
Для того чтобы начать устанавливать пакеты необходимо в настройках ВМ включить сеть если она была выключена.

Далее запустите ВМ и войдите в систему используя пароль который вы задали при установке.

Нажмите Ctrl+Alt+T запустится терминал в нем впишите
sudo apt-get update
и свой пароль и нажмите Enter. Эта команда скачает списки доступных пакетов из репозитория. **Примичание** При вводе пароля символы не отображаются.

Для удобства можете ввести команду clear после завершения обновления, она очистит область терминала.

Далее введите команду
sudo apt-get install openssh-server
после установки закройте терминал и выключите ВМ.
Теперь перейдем к настройке ВМ.
Перейдите в настройки ВМ пункт Сеть. Нажмите на кнопку Проброс портов.
и заполните таблицу правил как показано на рисунке

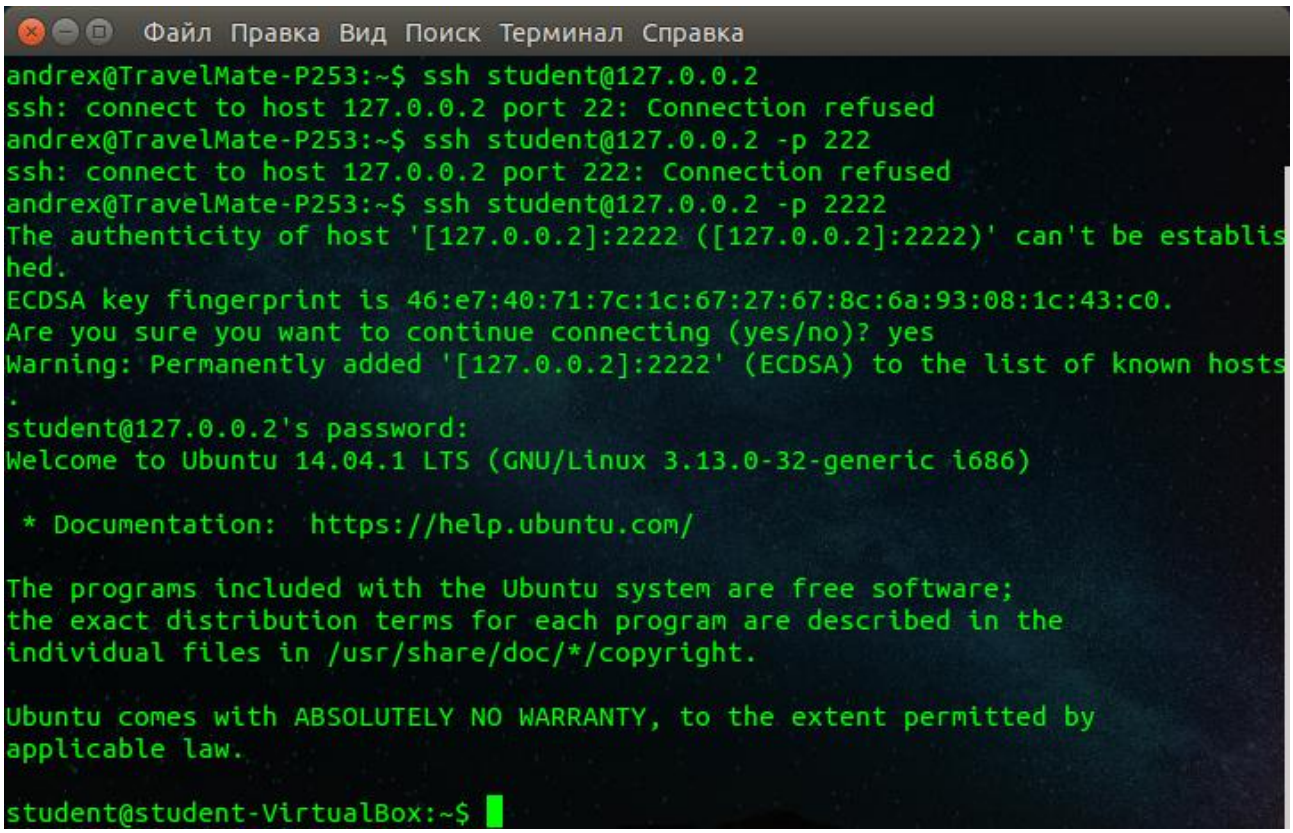


Это таблица переадресации запросов, т. е. Если вы обратитесь на порт 22 по IP 127.0.0.1 то запрос будет переадресован на IP 10.0.2.15 на котором работает ВМ.

Примечание Следует понимать что виртуальная машина это удаленный сервер, т. е. Вы не имеете прямого доступа к ней.

Если вы используете Windows то установите Putty и WinSCP.
Putty это ssh клиент. Я использую Linux по этому подключусь через терминал.

Включите ВМ и всерните ее после полной загрузки.



Теперь мы имеем доступ к машине по ssh

Приступим к установке ПО.

Для установки Веб сервера apache в Putty введите
sudo apt-get install apache2

sudo apt-get install mysql-client
sudo apt-get install mysql-server пароль оставить пустым

После установки сервера доступ к нему будет открыт только с 127,0,0,1 для того чтобы разрешить к нему подключаться с других IP нужно в консоле набрать команду sudo nano /etc/mysql/my.cnf и отредактировать файл bind-adress = * сохранить и перезапустить сервер теперь к серверу можно подключиться с любого IP

sudo apt-get install php5 установит php

sudo apt-get install apache2 установит apache2

Контрольные вопросы
Что такое apache
Что такое php
что такое Mysql
Зачем нужен bind-address в mysql

Лабораторная работа №4
Установка и настройка Oracle JAVA 6 и сервера приложений GlassFish 2.x

Для установки Java 6 выполните в терминале

Сперва проверяем какая версия Java у нас установлена.Для этого выполним команду в терминале: **java -version** Если будет **openjdk,** то нужно удалить этот пакет из нашей системы: **sudo apt-get remove openjdk*** Теперь переходим к установке.

В этом репозитории находятся пакеты Oracle Java SE, включают в себя JDK, JRE и плагин Java для браузера (то есть полный боевой комплект).

Приступим к установке:

1. Для начала добавляем репозиторий в систему, откройте терминал и выполните следующую команду:

sudo add-apt-repository ppa:webupd8team/java 2. Затем обновляем информацию о пакетах:

sudo apt-get update

Хочу заметить, что предпочтительней устанавливать Oracle Java 7. На сегодняшний день - это последняя стабильная версия.

Для установки Oracle Java 7 выполните команду в терминале:

sudo apt-get install oracle-java7-installer

Чтобы установить последнюю версию Oracle Java 6, выполните команду в терминале:

sudo apt-get install oracle-java6-installer

Чтобы установить последнюю версию Oracle Java 8, выполните команду в терминале:

sudo apt-get install oracle-java8-installer

Внимание!

Во время установки, потребуется принять лицензионные соглашения от компании Oracle, только после этого начнется установка.

Чтобы убедиться, что Oracle Java 7 установлена, выполняем знакомую команду в терминале: **java -version** Должно быть: **java version "1.7.0_25"**

Java(TM) SE Runtime Environment (build 1.7.0_25-b15)

Java HotSpot(TM) Server VM (build 23.25-b01, mixed mode)

Для других версий аналогично, только вместо 1.7, будет 1.8 или 1.6.

А также, проверяем версию компилятора Javac, для этого выполняем команду: javac -version Должно быть: **javac 1.7.0_25**

Внимание! Версии javac и java должны совпадать!

Что делать, если вы хотите несколько версий Java в системе?

Для этого есть команды. Можете спокойно устанавливать несколько

версий.

Чтобы пользоваться нужной вам версией по умолчанию, Вам нужно выполнить следующую команду в терминале:

Например, для установки Oracle Java 7 по умолчанию:

sudo update-java-alternatives -s java-7-oracle

Для других версий аналогично, просто меняем цифру версии.

Также, можно удалить остальные версии Java, оставив одну.

Следующая команда удаляет все остальные версии Oracle Java из системы, кроме 7-ой версии:

sudo apt-get install oracle-java7-set-default

Удаление пакета Java Oracle.

Если Вы больше не хотите пользоваться Java от Oracle, и захотите установить openjdk, например, воспользуйтесь следующей инструкцией.

Чтобы удалить Oracle Java 7 из системы, выполните в терминале команду

sudo apt-get remove oracle-java7-installer

С остальными версиями аналогично, меняем цифру.

В UBUNTU можно поставить из репозитория
введя команду
sudo apt-get install glassfish-javaee
Эта команда установит версию 2.1.1 сервера

2 вариант установить из пакета
для этого переходим по ссылке
https://glassfish.java.net/downloads/v2.1.1-final.html
и скачиваем необходимый jar файл, сохраняем его на компьютер.
Затем
в терминале
% java -Xmx256m -jar filename.jar
% cd glassfish

% chmod -R +x lib/ant/bin
% lib/ant/bin/ant -f setup.xml

для запуска сервера в терминале используется команда
cd %DIRRECTORY/domains/domain1/bin/
./startserv

./stopserv для остановки сервера
логин admin пароль по умолчанию adminadmin и changeit

Контрольные вопросы
Что такое JAVA
Зачем нужен Glassfish
Что такое Glassfish

Лабораторная работа №5
Настройка сервера openssh

В основном, SSH реализован в виде двух приложений — SSH-сервера и SSH-клиента1) В Ubuntu используется свободная реализация клиента и сервера SSH

При подключении клиент проходит процедуру авторизации у сервера и между ними устанавливается зашифрованное соединение. OpenSSH сервер может работать как с протоколом ssh1, так и с протоколом ssh2. В настоящее время протокол ssh1 считается небезопасным, поэтому его использование крайне не рекомендуется.

Установить OpenSSH можно из терминала командой:

sudo apt-get install ssh

В метапакете ssh содержится как клиент, так и сервер, но при этом, скорее всего, будет установлен только сервер, т. к. клиент уже есть в Ubuntu по умолчанию.

Настройка сервера

При установке SSH-сервер автоматически прописывается в автозагрузку. Управлять его запуском, остановкой или перезапуском можно с помощью команд:

sudo service ssh stop|start|restart

Основной файл конфигурации SSH-сервера — файл /etc/ssh/sshd_config, доступный для чтения или редактирования только суперпользователю. После каждого изменения этого файла необходимо перезапустить ssh-сервер для применения таких изменений.

Пример файла конфигурации сервера можно найти в интернете или в приложении к данной лабораторной работе Conf ssh server.doc

Сам по себе, неправильно настроенный SSH-сервер — огромная уязвимость в безопасности системы, т. к. у возможного злоумышленника есть возможность получить практически неограниченный доступ к системе. Помимо этого, у sshd есть много дополнительных полезных опций, которые желательно включить для повышения удобства работы и безопасности3).

Port, ListenAddress и AddressFamily

Эти три параметра определяют, на каких портах и адресах ваш сервер будет ждать входящие соединения. Во-первых, имеет смысл по возможности ограничить семейство обрабатываемых адресов реально используемыми, т. е.

если вы используете только IPv4 — отключите IPv6, и наоборот. Сделать это можно при помощи параметра AddressFamily, например (для разрешения IPv4 и запрета IPv6):

AddressFamily inet

Во-вторых, желательно сменить стандартный порт (22) на котором слушает sshd. Это связано с тем, что многочисленные сетевые сканеры постоянно пытаются соединиться с 22-м портом и как минимум получить доступ путем перебора логинов/паролей из своей базы. Даже если у вас и отключена парольная аутентификация — эти попытки сильно засоряют журналы и (в большом количестве) могут негативно повлиять на скорость работы ssh сервера. Если же вы по какой либо причине не желаете изменить стандартный порт вы можете использовать как различные внешние утилиты для борьбы брутфорсерами, например fail2ban, так и встроенные, такие как MaxStartups.

Задать порт можно как абсолютным значением для всех интерфейсов при помощи директивы Port, так и конкретным значением для каждого интерфейса, при помощи директивы ListenAddress. Например:

Port 2002

или

ListenAddress 192.168.0.1:2003
ListenAddress 192.168.1.1:2004

Запрещение удаленного доступа для суперпользователя

По умолчанию root-доступ разрешен. Это означает, что клиент при подключении в качестве пользователя может указать root, и во многих случаях получить контроль над системой. При условии, что по умолчанию в Ubuntu пользователь, добавленный при установке системы имеет возможность решать все административные задачи через sudo, создавать возможность root доступа к системе как минимум странно. Рекомендуется отключить эту опцию совсем, или применять ее только в режиме forced-commands-only. Отключить root-доступ можно так:

PermitRootLogin no

Парольная аутентификация

Разрешенная по умолчанию парольная аутентификация является практически самым примитивным способом авторизации в sshd. С одной стороны это упрощает конфигурацию и подключение новых пользователей

(пользователю достаточно знать свой системный логин/пароль), с другой стороны пароль всегда можно подобрать, а пользователи часто пренебрегают созданием сложных и длинных паролей. Специальные боты постоянно сканируют доступные из интернета ssh сервера и пытаются авторизоваться на них путем перебора логинов/паролей из своей базы. Настоятельно не рекомендуется использовать парольную аутентификацию. Отключить ее можно так:

PasswordAuthentication no

Если по каким либо причинам вам все таки хочется использовать парольную аутентификацию — позаботьтесь о том, чтобы никто не мог авторизоваться с пустым паролем. Для этого задайте директиву PermitEmptyPasswords:

PermitEmptyPasswords no

Протоколы SSH1 и SSH2

Как уже было сказано, sshd может работать с протоколами SSH1 и SSH2. При этом использование небезопасного SSH1 крайне не рекомендуется. Заставить sshd работать только с протоколом SSH2 можно так:

Protocol 2

Аутентификация на основе SSH2 RSA-ключей

Наиболее предпочтительным способом авторизации является аутентификация на основе SSH2 RSA-ключей. При таком способе пользователь генерирует на своей стороне пару ключей, из которой один ключ является секретным, а другой публичным. Публичный ключ копируется на сервер и служит для проверки идентичности пользователя. Более подробно про создание пары ключей и способы размещения их на сервере см. в описании SSH-клиента. Включить аутентификацию по публичному ключу можно так:

PubkeyAuthentication yes

Сервер должен знать, где ему следует искать публичный ключ пользователя. Для этого применяется специальный файл authorized_keys. Синтаксис его может быть следующим:

Коментарии записываются только с новой строки
общий вид записей в файле authorized_keys
[опции] тип_ключа(ssh-rsa или ssh-dss) очень_длинная_строка_непонятная_простому_человеку [логин@хост]

ssh-rsa AAAAB3Nza...LiPk== user@example.net

from="*.sales.example.net,!pc.sales.example.net" ssh-rsa AAAAB2...19Q== john@example.net

command="dump /home",no-pty,no-port-forwarding ssh-dss AAAAC3...51R== example.net

permitopen="192.0.2.1:80",permitopen="192.0.2.2:25" ssh-dss AAAAB5...21S==

tunnel="0",command="sh /etc/netstart tun0" ssh-rsa AAAA...== jane@example.net

Можно указать как один общий файл с ключами, так и по файлу на каждого пользователя. Последний способ является более удобным и безопасным, т. к. можно во-первых указывать разные комбинации ключей для каждого пользователя4), а во-вторых ограничить доступ к публичному ключу пользователя. Задать файл с ключами можно при помощи директивы AuthorizedKeysFile:

AuthorizedKeysFile %h/.ssh/my_keys

для схемы пользователь — файл
или

AuthorizedKeysFile /etc/ssh/authorized_keys

для схемы с общим файлом. По умолчанию SSH-клиент ищет ключи в файле ~/.ssh/authorized_keys .

Дополнительные настройки
Пользователи и группы.

Если у вас на сервере «живет» много пользователей, а доступ через ssh вы хотите разрешить только нескольким из них - вы можете использовать директивы DenyUsers, AllowUsers, DenyGroups, и AllowGroups. Более подробно про эти директивы см. комментарии в примере sshd_config.

Опции определения состояния соединения

По умолчанию из способов определения состояния соединения включен только способ проверки TCP соединения — TCPKeepAlive, однако, sshd умеет определять состояния соединения и более удобными и безопасными способами. Подробнее см. соответствующий раздел в примере sshd_config.

SFTP

В sshd по умолчанию встроен SFTP-сервер. Протокол SFTP (SSH File Transfer Protocol) - SSH-протокол для передачи файлов. Он предназначен для

копирования и выполнения других операций с файлами поверх надёжного и безопасного соединения. Как правило, в качестве базового протокола, обеспечивающего соединение, и используется протокол SSH2. Для того чтобы включить поддержку SFTP добавьте в sshd_config строку

Subsystem sftp /usr/lib/openssh/sftp-server

По умолчанию поддержка SFTP включена.

Контрольные вопросы
что такое RSA
Что такое SSH
что такое openssh
Зачем нужен openSSH-server
Зачем нужен openSSH-client

Лабораторная работа №6
20 советов по безопасному использованию сервера OpenSSH

OpenSSH является реализацией протокола SSH. OpenSSH рекомендуется использовать для удаленного доступа в систему, для создания резервных копий, для дистанционной передачи файлов по протоколам scp или sftp и для многого другого. SSH идеально подходит для сохранения конфиденциальности и обеспечения целостности данных при обмене данными между двумя сетями или системами. Однако основным его преимуществом является аутентификация на сервере с использованием криптографии с открытым ключом. Время от времени возникают слухи о том, что есть эксплойт zero day для OpenSSH . Ниже перечислено, что вам нужно настроить с тем, чтобы повысить безопасность сервера OpenSSH.

Конфигурационные файлы первоначальной настройки и порт SSH

/etc/ssh/sshd_config - конфигурационный файл сервера OpenSSH

/etc/ssh/ssh_config - конфигурационный файл клиентской части OpenSSH

~/.ssh/ - конфигурационная директория пользователей ssh

~/.ssh/authorized_keys или ~/.ssh/authorized_keys2 - список открытых ключей (RSA или DSA), которыми можно пользоваться при входе в систему по регистрационным записям пользователей

/etc/nologin - если этот файл существует, то демон sshd не даст никому из пользователей доступ, кроме пользователя root

/etc/hosts.allow (доступ разрешен) и /etc/hosts.deny(доступ запрещен): здесь находятся списки управления доступом, которые должны использоваться tcp-wrapper-ами .

SSH порт, используемый по умолчанию: : TCP 22

Сессия SSH в действии

№ 1: Отключите сервер OpenSSH

На рабочих станциях и ноутбуках можно работать без использования сервера OpenSSH. Если вам не нужен удаленный доступ и возможности передачи файлов через SSH, отключите и удалите сервер SSHD. В Linux системах CentOS / RHEL / Fedora пользователь может отключить и удалить сервер openssh с помощью команды yum:

```
# chkconfig sshd off
# yum erase openssh-server
```

В Linux системах Debian / Ubuntu пользователь может его отключить и одновременно удалить с помощью команды apt-get:

```
# apt-get remove openssh-server
```

Вам может потребоваться изменить свой скрипт iptables с тем, чтобы удалить из него правило исключения для ssh. В CentOS / RHEL / Fedora отредактируйте файлы /etc/sysconfig/iptables и /etc/sysconfig/ip6tables. После того, как это сделаете, перезапустите сервис iptables:

```
# service iptables restart
# service ip6tables restart
```

№ 2: Используйте только протокол SSH 2

В протоколе SSH версии 1 (SSH-1) имеются проблемы, касающиеся атак вида "man-in-the-middle" ("человек посередине", т.е. когда атакующий может читать и видоизменять сообщения, которыми обмениваются корреспонденты, причем ни один из них не знает о наличии такого посредника – прим. пер.) и уязвимости, связанные с безопасностью. SSH-1 устарел и его использование следует любой ценой избегать. Откройте файл sshd_config и убедитесь в том, чтобы там присутствовала следующая строка:

```
Protocol 2
```

№ 3: Ограничьте доступ пользователей через SSH

Во всех системах по умолчанию пользователям разрешено иметь доступ через SSH с использованием пароля или открытого ключа. Иногда Вы создаете учетную запись пользователя UNIX / Linux для доступа по протоколу ftp или использования электронной почты. Однако тот же самый пользователь может войти в систему через ssh. Он будет иметь полный доступ к системным средствам – компиляторам и таким скриптовым языкам, как Perl, Python, с помощью которых можно открывать порты и делать множество других удивительных вещей. В одной из моих клиентских программ есть устаревший php скрипт и атакующий может с помощью этого скрипта создать в системе новую учетную запись. Однако у атакующего нет возможности попасть в систему через ssh, поскольку он не указан в записи AllowUsers (пользователи, которым разрешен доступ).

Для того, чтобы доступ в систему через SSH был разрешен только пользователям root, vivek и jerry, добавьте в файл sshd_config следующую запись:

AllowUsers root vivek jerry

Вы можете, наоборот, разрешить всем пользователям использовать доступ через SSH, но для некоторых из них можете запретить его с помощью следующей записи:

DenyUsers saroj anjali foo

Вы также можете сконфигурировать Linux PAM, который будет разрешать или запрещать доступ через сервер sshd. Вы можете задать список имен групп, которым разрешен или запрещен доступ через ssh.

№ 4: Сконфигурируйте время закрытия неработающей сесси

Пользователь может заходить на сервер через ssh, а Вы для того, чтобы удалять брошенные сессии ssh, можете установить таймаут idle (время бездействия). Откройте файл sshd_config и убедитесь в том, что сконфигурированы следующие параметры:

ClientAliveInterval 300
ClientAliveCountMax 0

Вы устанавливаете таймаут idle в секундах (300 секунд = 5 минутам). После того, как указанное время истечет, бездействующий пользователь будет "выброшен" из системы (читайте – отключен от системы). Подробности о том, как автоматически отключать пользователей BASH / TCSH / SSH, смотрите по следующей ссылке - how to automatically log BASH / TCSH / SSH users out.

№ 5: Запретите использование файлов .rhosts

Не разрешайте использовать файлы ~/.rhosts и ~/.shosts, имеющиеся у пользователя. Измените в файле sshd_config следующую настройку:

IgnoreRhosts yes

Сервер SSH может эмулировать поведение устаревшей команды rsh, так что запретите небезопасный доступ через RSH.

№ 6: Запретите кросс-хостинговую аутентификацию (Host-Based Authentication)

Для того, чтобы запретить кросс-хостинговую аутентификацию, впишите в файл sshd_config следующую опцию:

HostbasedAuthentication no

№ 7: Запретите доступ пользователя root через SSH

Нет необходимости получать по сети через ssh доступ с правами root. Обычные пользователи могут использовать команду su или sudo (рекомендуется) для получения доступа с правами уровня root. Также удостоверьтесь, что Вы получаете полную информацию о том, кто в системе запускал привилегированные команды с использованием sudo. Для того, чтобы запретить доступ пользователя root через SSH, измените в файле sshd_config следующую строку:

PermitRootLogin no

№ 8: Добавьте предупреждающий баннер

Задайте предупреждающий баннер, дополнив файл sshd_config следующей строкой:

Banner /etc/issue

Пример файла /etc/issue:

You are accessing a XYZ Government (XYZG) Information System (IS) that is provided for authorized use only.

By using this IS (which includes any device attached to this IS), you consent to the following conditions:

+ The XYZG routinely intercepts and monitors communications on this IS for purposes including, but not limited to,

penetration testing, COMSEC monitoring, network operations and defense, personnel misconduct (PM),

law enforcement (LE), and counterintelligence (CI) investigations.

+ At any time, the XYZG may inspect and seize data stored on this IS.

+ Communications using, or data stored on, this IS are not private, are subject to routine monitoring,

interception, and search, and may be disclosed or used for any XYZG authorized purpose.

+ This IS includes security measures (e.g., authentication and access controls) to protect XYZG interests--not

for your personal benefit or privacy.

+ Notwithstanding the above, using this IS does not constitute consent to PM, LE or CI investigative searching

or monitoring of the content of privileged communications, or work product, related to personal representation

or services by attorneys, psychotherapists, or clergy, and their assistants. Such communications and work

product are private and confidential. See User Agreement for details.

Выше приведен стандартный образец, обратитесь к юристу вашего предприятия для уточнения деталей пользовательского соглашения / официального уведомления.

№ 8: Сконфигурируйте в брандмауэре доступ SSH к порту 22

Вам необходимо осуществлять доступ к порту 22 через брандмауэр – для этого измените конфигурацию брандмауэра iptables или pf. Обычно сервер OpenSSH должен использоваться для доступа только из вашей локальной сети или с удаленных сайтов вашей распределенной сети.

Конфигурирование netfilter (Iptables)

Измените файл /etc/sysconfig/iptables (в Redhat и аналогичных системах). Для того, чтобы разрешить доступ только с адресов 192.168.1.0/24 и 202.54.1.5/29, введите следующее:

-A RH-Firewall-1-INPUT -s 192.168.1.0/24 -m state --state NEW -p tcp --dport 22 -j ACCEPT

-A RH-Firewall-1-INPUT -s 202.54.1.5/29 -m state --state NEW -p tcp --dport 22 -j ACCEPT

Если у вас двухстековый sshd с IPv6, отредактируйте файл edit /etc/sysconfig/ip6tables (в Redhat и аналогичных системах): введите следующее:

-A RH-Firewall-1-INPUT -s ipv6network::/ipv6mask -m tcp -p tcp --dport 22 -j ACCEPT

Замените ipv6network::/ipv6mask фактическим диапазоном IPv6 адресов.

Конфигурирование брандмауэра *BSD PF

Если Вы используете брандмауэр PF, то измените файл /etc/pf.conf следующим образом:

pass in on $ext_if inet proto tcp from {192.168.1.0/24, 202.54.1.5/29} to $ssh_server_ip port ssh flags S/SA synproxy state

№ 9: Измените порт SSH и укажите конкретные IP адреса

По умолчанию SSH слушает все интерфейсы и IP адреса, имеющиеся в системе. Ограничьте адреса доступа к ssh и измените порт ssh (по умолчанию скрипты, использующие метод грубой силы brute forcing, пытаются подключиться к порту 22) . Для того, чтобы ограничиться IP адресами 192.168.1.5 и 202.54.1.5 и выбрать порт 300, добавьте или измените следующие строки:

```
Port 300
ListenAddress 192.168.1.5
ListenAddress 202.54.1.5
```

Лучший подход - это использовать скрипты активных сценариев, такие как fail2ban или denyhosts (смотрите ниже).

№ 10: Используйте сильные пароли

Нельзя переоценить, насколько важно использовать сильные пользовательские пароли или пароли для ваших ключей. Атака методом грубой силы работает, если Вы пользуетесь паролями, которые подбираются по словарю. Вы можете заставить пользователей не применять пароли, подбираемые по словарю. Используйте специальное средство john the ripper tool для поиска используемых слабых паролей. Ниже приведен пример генератора случайных паролей (поместите в ваш файл ~/.bashrc):

```
genpasswd() {
        local l=$1
        [ "$l" == "" ] && l=20
        tr -dc A-Za-z0-9_ < /dev/urandom | head -c ${l} | xargs
}
```

Запустите генератор:

```
genpasswd 16
```

Вы получите следующий результат:

```
uw8CnDVMwC6vOKgW
```

№ 11: Используйте аутентификацию с применением открытого ключа

Используйте пару открытый / закрытый ключ с паролем, защищающим закрытый ключ. Используйте аутентификацию с использованием ключей RSA и DSA. Никогда не используйте доступ с ключом без пароля.

№ 12: Используйте аутентификацию с применением Keychain

Keychain – специальный скрипт bash, созданный для аутентификации по ключу, исключительно удобен и гибок. В нем применены различные способы обеспечения безопасности на основе ключей без пароля. Подробности настройки смотрите по ссылке keychain software.

№ 13: Разрешите пользователям доступ только к своим домашним директориям (Chroot SSHD)

По умолчанию пользователям разрешается просматривать директории сервера, такие как /etc/, /bin и другие. Вы можете защитить ssh с помощью имеющейся в ОС команды chroot, либо с помощью специального инструментария rssh. В реализациях OpenSSH 4.8p1 и 4.9p при ограничении пользовательского доступа только домашними директориями вам уже не потребуется применять такие инструменты, полученные от третьих лиц, как rssh или использовать сложную команду chroot(1). Смотрите этот пост, касающийся новой директивы ChrootDirectory, с помощью которой доступ пользователей ограничивается только их домашними директориями.

№ 14: Используйте TCP Wrapper-ы

TCP Wrapper является кросс-хостинговой сетевой системой ACL (листы контроля доступа), используемой для фильтрации доступа из сети в Интернет. OpenSSH поддерживает работу с TCP wrapper-ами. Для того, чтобы разрешить доступ через SSH только с адресов 192.168.1.2 172.16.23.12, просто измените ваш файл /etc/hosts.allow следующим образом:

```
sshd : 192.168.1.2 172.16.23.12
```

Смотрите ссылку FAQ about setting and using TCP wrappers, в которой описывается настройка и использование TCP Wrapper-ов в Linux / Mac OS X и в UNIX-подобных операционных системах.

№ 15: Запретите доступ с "пустыми" паролями

Вы должны явно запретить удаленный доступ с использованием пустых паролей – дополните файл sshd_config следующей строкой:

```
PermitEmptyPasswords no
```

№ 16: Противодействуйте программам взлома SSH (атака методом грубой силы - Brute Force Attack)

Метод грубой силы является методом взлома криптографической схемы с помощью перебора большого количества вариантов, осуществляемого одним компьютером или распределенной сетью компьютеров. Для того, чтобы защитить SSH от атак вида brute force, используйте следующие программы:

DenyHosts – инструментарий на базе языка Python, обеспечивающий безопасность серверов SSH. Предназначен для предотвращения атак на сервера SSH методом brute force. Он отслеживает в регистрационных журналах неудавшиеся попытки доступа и блокирует IP адреса, с которых делались эти попытки.

Этот FAQ объясняет, как настроить DenyHosts в RHEL / Fedora и CentOS Linux

Fail2ban – аналогичная программа, которая предотвращает атаки вида brute force через SSH

security/sshguard-pf – защищает хосты с помощью pf от атак вида brute force через ssh и через другие сервисы.

security/sshguard-ipfw – защиет хосты с помощью ipfw от атак вида brute force через ssh и через другие сервисы.

security/sshguard-ipfilter – защищает хосты с помощью ipfilter от атак вида brute force через ssh и через другие сервисы

security/sshblock – блокирует атакующие попытки доступа через SSH.

security/sshit – выявляет атаки вида brute force на SSH/FTP и блокирует IP адреса.

BlockHosts автоматически блокирует адреса IP хостов, откуда делаются попытки проникновения.

Blacklist автоматически блокируются IP адреса тех хостов, откуда делаются попытки проникновения методом brute force.

Brute Force Detection модульный скрипт- оболочка для анализа журналов и проверки неудачных попыток аутентификации. Анализ делается согласно системе правил, в которых с помощью регулярных выражений записываются конкретные особенности каждого уникального формата аутентификации, используемого в прикладных программах.

IPQ BDB filter может рассматриваться как облегченная версия fail2ban.

№ 17: Ограничьте подключения к порту 22

Как в netfilter, так и в pf, имеется возможность ограничить прием запросов, поступающих на порт 22.

Пример Iptables

В следующем примере будут отвергаться запросы на подключение в случае, если в течение 60 секунд делалось более 5 попыток подключения к порту 22:

#!/bin/bash inet_if=eth1 ssh_port=22 $IPT -I INPUT -p tcp --dport ${ssh_port} -i ${inet_if} -m state --state NEW -m recent --set $IPT -I INPUT -p tcp --dport ${ssh_port} -i ${inet_if} -m state --state NEW -m recent --update --seconds 60 --hitcount 5 -j DROP

Вызывайте скрипт, описанный выше, из ваших скриптов iptables. Еще одна возможность конфигурирования:

$IPT -A INPUT -i ${inet_if} -p tcp --dport ${ssh_port} -m state --state NEW -m limit --limit 3/min --limit-burst 3 -j ACCEPT $IPT -A INPUT -i ${inet_if} -p tcp --dport ${ssh_port} -m state --state ESTABLISHED -j ACCEPT $IPT -A OUTPUT -o ${inet_if} -p tcp --sport ${ssh_port} -m state --state ESTABLISHED -j ACCEPT # another one line example # $IPT -A INPUT -i ${inet_if} -m state --state NEW,ESTABLISHED,RELATED -p tcp --dport 22 -m limit --limit 5/minute --limit-burst 5-j ACCEPT

Подробности смотрите в страницах man page для iptables.
Пример *BSD PF

В следующем скрипте максимальное число подключений от одного источника ограничивается 20-ю подключениями, а количество подключений за 5 секундный интервал времени ограничивается 15-ю подключениями. Если кто-нибудь нарушит эти правила, то будет добавлен в таблицу abusive_ips и все его дальнейшие попытки подключения будут блокированы. Благодаря тому, что указано ключевое слово flush, для хоста, который нарушил указанные ограничения, будут сброшены все ранее запомненные настройки, нарушений в которых не обнаружено.

sshd_server_ip="202.54.1.5" table <abusive_ips> persist block in quick from <abusive_ips> pass in on $ext_if proto tcp to $sshd_server_ip port ssh flags S/SA keep state (max-src-conn 20, max-src-conn-rate 15/5, overload <abusive_ips> flush)

№ 18: Используйте метод Port Knocking

Port knocking – метод открытия портов на брандмауэре извне с помощью генерации заранее заданной последовательности попыток подключения к закрытым портам. Как только будет получена правильная последовательность попыток подключения, правила работы брандмауэра динамически модифицируются, что позволит хосту, который посылал запросы на подключение, подключиться через конкретный порт (порты). Пример использования метода port Knocking для ssh с использованием iptables:

$IPT -N stage1
$IPT -A stage1 -m recent --remove --name knock
$IPT -A stage1 -p tcp --dport 3456 -m recent --set --name knock2

$IPT -N stage2
$IPT -A stage2 -m recent --remove --name knock2
$IPT -A stage2 -p tcp --dport 2345 -m recent --set --name heaven

$IPT -N door
$IPT -A door -m recent --rcheck --seconds 5 --name knock2 -j stage2
$IPT -A door -m recent --rcheck --seconds 5 --name knock -j stage1
$IPT -A door -p tcp --dport 1234 -m recent --set --name knock

$IPT -A INPUT -m --state ESTABLISHED,RELATED -j ACCEPT

$IPT -A INPUT -p tcp --dport 22 -m recent --rcheck --seconds 5 --name heaven -j ACCEPT

$IPT -A INPUT -p tcp --syn -j doo

fwknop . – реализация, в которой объединяются метод port knocking и пассивное определение сигнатуры ОС

Многопортовая реализация метода port knocking только с помощью Netfilter/IPtables.

№ 19: Используйте анализатор журнала

Читайте свои журналы с помощью logwatch или logcheck. Эти инструменты облегчат вам анализ ваших журналов. С их помощью журналы будут просматриваться периодически и будут создаваться отчеты о том, что Вы пожелаете, и с той детализацией, с которой Вы пожелаете. Удостоверьтесь, что в файле sshd_config для LogLevel указано INFO или DEBUG:

LogLevel INFO

№ 20: Своевременно обновляйте OpenSSH и операционные системы

Рекомендуется, чтобы Вы использовали такие инструменты, как yum, apt-get, freebsd-update и другие, для добавления последних патчей безопасности и поддержания системы в обновленном состоянии.

Другие возможности

Для того, чтобы скрыть идентификацию версии openssh, вам нужно изменить исходный код и заново откомпилировать openssh. Убедитесь, что в sshd_config включены следующие опции:

Turn on privilege separation – Включите разделение привелегий

UsePrivilegeSeparation yes

Prevent the use of insecure home directory and key file permissions

Запретите использование небезопасных прав доступа к домашнему директорию и к файлам

StrictModes yes

Turn on reverse name checking – Включите обратную проверку имени

VerifyReverseMapping yes

Do you need port forwarding? - Вам нужно перенаправление портов?

AllowTcpForwarding no

X11Forwarding no

Specifies whether password authentication is allowed. The default is yes.

Укажите, допускается ли аутентификация по паролю. По умолчанию - допускается

PasswordAuthentication no

Проверьте изменения, сделанные в вашем файле sshd_config перед тем, как выполнить перезапуск / перезагрузку:

Усильте безопасность использования SSH за счет двухуровневой или трехуровневой (или более) аутентификации.

Лабораторная работа №7
Основный принципы клиент серверного ПО. Протоколы прикладного уровня. XML

Протокол прикладного уровня (англ. Application layer) — протокол верхнего (7-го) уровня сетевой модели OSI, обеспечивает взаимодействие сети и пользователя. Уровень разрешает приложениям пользователя иметь доступ к сетевым службам, таким как обработчик запросов к базам данных, доступ к файлам, пересылке электронной почты. Также отвечает за передачу служебной информации, предоставляет приложениям информацию об ошибках и формирует запросы к уровню представления. Пример: HTTP, POP3, SMTP.

HTTP (англ. HyperText Transfer Protocol — «протокол передачи гипертекста») — протокол прикладного уровня передачи данных (изначально — в виде гипертекстовых документов в формате HTML, в настоящий момент используется для передачи произвольных данных). Основой HTTP является технология «клиент-сервер», то есть предполагается существование потребителей (клиентов), которые инициируют соединение и посылают запрос, и поставщиков (серверов), которые ожидают соединения для получения запроса, производят необходимые действия и возвращают обратно сообщение с результатом.

HTTP в настоящее время повсеместно используется во Всемирной паутине для получения информации с веб-сайтов. В 2006 году в Северной Америке доля HTTP-трафика превысила долю P2P-сетей и составила 46 %, из которых почти половина — это передача потокового видео и звука[1].

HTTP используется также в качестве «транспорта» для других протоколов прикладного уровня, таких как SOAP, XML-RPC, WebDAV.

Основным объектом манипуляции в HTTP является ресурс, на который указывает URI (Universal Resource Identifier) в запросе клиента. Обычно такими ресурсами являются хранящиеся на сервере файлы, но ими могут быть логические объекты или что-то абстрактное. Особенностью протокола HTTP является возможность указать в запросе и ответе способ представления одного и того же ресурса по различным параметрам: формату, кодировке, языку и т. д. (В частности для этого используется HTTP-заголовок.) Именно благодаря

возможности указания способа кодирования сообщения клиент и сервер могут обмениваться двоичными данными, хотя данный протокол является текстовым.

HTTP — протокол прикладного уровня, аналогичными ему являются FTP и SMTP. Обмен сообщениями идёт по обыкновенной схеме «запрос-ответ». Для идентификации ресурсов HTTP использует глобальные URI. В отличие от многих других протоколов, HTTP не сохраняет своего состояния. Это означает отсутствие сохранения промежуточного состояния между парами «запрос-ответ». Компоненты, использующие HTTP, могут самостоятельно осуществлять сохранение информации о состоянии, связанной с последними запросами и ответами (например, «куки» на стороне клиента, «сессии» на стороне сервера). Браузер, посылающий запросы, может отслеживать задержки ответов. Сервер может хранить IP-адреса и заголовки запросов последних клиентов. Однако сам протокол не осведомлён о предыдущих запросах и ответах, в нём не предусмотрена внутренняя поддержка состояния, к нему не предъявляются такие требования.

Программное обеспечение

Всё программное обеспечение для работы с протоколом HTTP разделяется на три большие категории:

Серверы как основные поставщики услуг хранения и обработки информации (обработка запросов).
Клиенты — конечные потребители услуг сервера (отправка запроса).
Прокси для выполнения транспортных служб.

Для отличия конечных серверов от прокси в официальной документации используется термин «исходный сервер» (англ. origin server). Один и тот же программный продукт может одновременно выполнять функции клиента, сервера или посредника в зависимости от поставленных задач. В спецификациях протокола HTTP подробно описывается поведение для каждой из этих ролей.

Клиенты

Первоначально протокол HTTP разрабатывался для доступа к гипертекстовым документам Всемирной паутины. Поэтому основными реализациями клиентов являются браузеры (агенты пользователя). Для просмотра сохраненного содержимого сайтов на компьютере без соединения с Интернетом были придуманы офлайн-браузеры. При нестабильном соединении для загрузки больших файлов используются менеджеры закачек. Они позволяют в любое время докачать указанные файлы после потери соединения с веб-сервером. Виртуальные атласы, такие как Google Планета Земля и NASA World Wind, тоже используют HTTP.

Нередко протокол HTTP используется программами для скачивания обновлений.

Целый комплекс программ-роботов используется в поисковых системах Интернета. Среди них веб-пауки (краулеры), которые производят проход по гиперссылкам, составляют базу данных ресурсов серверов и сохраняют их содержимое для дальнейшего анализа.

Исходные серверы

См. также: Список веб-серверов

Основные реализации: Apache, Internet Information Services (IIS), nginx, Google Web Server, lighttpd.

Прокси-серверы

См. также: Список веб-серверов

Основные реализации: Squid, UserGate, Multiproxy, Naviscope, nginx.

Более подробно о HTTP можно прочитать в приложении http.doc

Язык XML - практическое введение

За неполный год своего официального существования язык XML привлек к себе уже достаточно много внимания со стороны разработчиков и пользователей Интернет. Сегодня количество приверженцев этой новой технологии возрастает также стремительно, как и число сообщений об очередных взятых ею преградах на пути к всеобщему признанию. Несмотря на то, что XML очень молод (международная организация W3C утвердила спецификацию "Extensible Markup Language(XML) 1.0" чуть меньше года назад - в начале февраля 1998 г) и отдельные компоненты этого языка находятся еще в стадии доработки, уже сегодня появляются новые языки, созданные на основе XML, возникают многочисленные Web-сервера, использующие эту технологию для организации хранящейся на них информации. Мир Интернет вокруг нас в очередной раз преображается, и мы можем стать участниками этого процесса уже сегодня

Целью данной статьи является попытка на конкретных примерах показать некоторые из возможностей XML, ответить на ряд часто возникающих при знакомстве с новым языком вопросов. Что же такое XML? В чем заключаются его преимущества перед привычным уже нам языком HTML? Можно ли использовать XML на своих Web-страничках уже сегодня? А если можно, то как?

В конце статьи приведены ссылки на другие ресурсы Интернет, с помощью которых Вы сможете также получить более полную информацию по конкретным интересующим Вас вопросам, связанным с применением XML и

незатронутых нами в этой статье. Полные спецификации XML и связанных с ним языков доступны на официальной странице W3C - www.w3.org

Для чего нужен новый язык разметки?

Когда осенью 1991 года Интернет впервые услышал позывные новой технологии, название которой легко уместилось в три буквы, почти никто не мог представить себе, что завоевания ее окажутся настолько глобальными. Сегодня для многих неискушенных пользователей слово Интернет прочно ассоциируется с WWW и с уст специалистов не сходит тема будущего информационных систем и влияния на это будущее всемирной сетевой паутины.

Популярность World Wide Web и неотъемлемой ее части, HTML, безусловно, стала причиной повышенного внимания к системам гипертекстовой разметки документов. Хотя понятие гипертекста было введено В.Бушем еще в 1945 году и, начиная с 60-х годов стали появляться первые приложения, использующие гипертекстовые данные, всплеск активности вокруг этой технологии начался лишь тогда, когда возникла реальная необходимость в механизме объединения множества информационных ресурсов, обеспечения возможности создания, просмотра нелинейного текста. И примером реализации этого механизма послужила паутина WWW.

Язык разметки документов - это набор специальных инструкций, называемых тэгами, предназначенных для формирования в документах какой-либо структуры и определения отношений между различными элементами этой структуры. Тэги языка, или, как их иногда называют, управляющие дескрипторы, в таких документах каким-то образом кодируются, выделяются относительно основного содержимого документа и служат в качестве инструкций для программы, производящей показ содержимого документа на стороне клиента. В самых первых системах для обозначения этих команд использовались символы “<” и “>”, внутри которых помещались названия инструкций и их параметры. Сейчас такой способ обозначения тэгов является стандартным.

Использование гипертекстовой разбивки текстового документа в современных информационных системах во многом связано с тем, что гипертекст позволяет создавать механизм нелинейного просмотра информации. В таких системах данные представляются не в виде непрерывного потока текстовой информации, а набором взаимосвязанных компонентов, переход по которым осуществляется при помощи гиперссылок.

Самый популярный на сегодняшний день язык гипертекстовой разметки – HTML, был создан специально для организации информации, распределенной в сети Интернет, и является одной из ключевых составляющих технологии

WWW. С использованием гипертекстовой модели документа способ представления разнообразных информационных ресурсов в сети стал более упорядочен, а пользователи получили удобный механизм поиска и просмотра нужной информации.

HTML [8]является упрощенной версией стандартного общего языка разметки - SGML (Standart Generalised Markup Language[10]), который был утвержден ISO в качестве стандарта еще в 80-х годах. Этот язык предназначен для создания других языков разметки, он определяет допустимый набор тэгов, их атрибуты и внутреннюю структуру документа. Контроль за правильностью использования дескрипторов осуществляется при помощи специального набора правил, называемых DTD- описаниями(более подробно о DTD мы поговорим чуть позже), которые используются программой клиента при разборе документа. Для каждого класса документов определяется свой набор правил, описывающих грамматику соответствующего языка разметки. С помощью SGML можно описывать структурированные данные, организовывать информацию, содержащуюся в документах, представлять эту информацию в некотором стандартизованном формате. Но в виду некоторой своей сложности, SGML использовался, в основном, для описания синтаксиса других языков(наиболее известным из которых является HTML), и немногие приложения работали с SGML- документами напрямую.

Гораздо более простой и удобный, чем SGML, язык HTML позволяет определять оформление элементов документа и имеет некий ограниченный набор инструкций - тэгов, при помощи которых осуществляется процесс разметки. Инструкции HTML, в первую очередь, предназначены для управления процессом вывода содержимого документа на экране программы-клиента и определяют этим самым способ представления документа, но не его структуру. В качестве элемента гипертекстовой базы данных, описываемой HTML, используется текстовый файл, который может легко передаваться по сети с использованием протокола HTTP. Эта особенность, а также то, что HTML является открытым стандартом и огромное количество пользователей имеет возможность применять возможности этого языка для оформления своих документов, безусловно, повлияли на рост популярности HTML и сделали его сегодня главным механизмом представления информации в Web

Однако современные приложения нуждаются не только в языке представления данных на экране клиента, но и в механизме, позволяющем определять структуру документа, описывать содержащиеся в нем элементы. HTML обладает несложным набором команд и вполне успешно справляется с задачей описания текстовой информации и отображением ее на экране программы просмотра- броузера. Однако сами отображаемые данные никак не связаны с теми тэгами, которые используются для форматирования, поэтому у программ-анализаторов нет возможности использовать тэги HTML для поиска нужных нам фрагментов документа. Т.е. встретив, например, такое описание

rose,

программа просмотра будет знать, каким цветом отобразить текст, содержащийся внутри тэгов и, вероятно, отобразит его правильно, но ей абсолютно безразлично, в каком месте документа встретился этот тэг, в какие другие тэги заключен текущий фрагмент, существуют ли вложенные в него фрагменты, правильно ли построены отношения между объектами. Такое "безразличие" к структуре документа приводит к тому, что поиск или анализ информации внутри него ничем не будет отличаться от работы со сплошным, не разбитым на элементы текстовым файлом. А это, как известно, не самый эффективный способ работы с информацией.

Другим существенным недостатком HTML можно назвать ограниченность набора его тэгов. DTD- правила для HTML определяют фиксированный набор дескрипторов и поэтому у разработчика нет возможности вводить собственные, специальные тэги. Хотя время от времени появляются новые расширения языка(на сегодняшний день последней версией HTML является HTML 4.0), но долгий путь их стандартизации, сопровождаемый постоянными разногласиями между основными производителями броузеров делают практически невозможной быструю адаптацию языка, его использование для отображения специализированной информации(например, мультимедийной, математических, химических формул и т.д.).

Подводя итог всему сказанному, можно утверждать, что HTML уже сегодня не удовлетворяет в полной мере требованиям, предъявляемым современными разработчиками к языкам подобного рода. И ему на смену был предложен новый язык гипертекстовой разметки, мощный, гибкий, и, одновременно с этим, удобный язык XML. В чем же заключается его достоинства?

XML (Extensible Markup Language[1]) - это язык разметки, описывающий целый класс объектов данных, называемых XML- документами. Этот язык используется в качестве средства для описания грамматики других языков и контроля за правильностью составления документов. Т.е. сам по себе XML не содержит никаких тэгов, предназначенных для разметки, он просто определяет порядок их создания. Таким образом, если, например, мы считаем, что для обозначения элемента rose в документе необходимо использовать тэг <flower>;, то XML позволяет свободно использовать определяемый нами тэг и мы можем включать в документ фрагменты, подобные следующему:

<flower>rose</flower>

Набор тэгов может быть легко расширен. Если, предположим, мы хотим также указать, что описание цветка должно по смыслу идти внутри описания

оранжереи, в которой он цветет, то просто задаем новые тэги и выбираем порядок их следования:

```
<conservatory>
<flower>rose</flower>
</conservatory>
```

Если мы хотим посадить туда еще несколько цветочков, то должны внести следующие изменения:

```
<conservatory>
<flower>rose</flower>
<flower>tulip</flower>
<flower>cactus</flower>
</conservatory>
```

Как видно, сам процесс создания XML документа очень прост и требует от нас лишь базовых знаний HTML и понимания тех задач, которые мы хотим выполнить, используя XML в качестве языка разметки. Таким образом, у разработчиков появляется уникальная возможность определять собственные команды, позволяющие им наиболее эффективно определять данные, содержащиеся в документе. Автор документа создает его структуру, строит необходимые связи между элементами, используя те команды, которые удовлетворяют его требованиям и добивается такого типа разметки, которое необходимо ему для выполнения операций просмотра, поиска, анализа документа.

Еще одним из очевидных достоинств XML является возможность использования его в качестве универсального языка запросов к хранилищам информации. Сегодня в глубинах W3C находится на рассмотрении рабочий вариант стандарта XML-QL(или XQL), который, возможно, в будущем составит серьезную конкуренцию SQL. Кроме того, XML-документы могут выступать в качестве уникального способа хранения данных, который включает в себя одновременно средства для разбора информации и представления ее на стороне клиента. В этой области одним из перспективных направлений является интеграция Java и XML - технологий, позволяющая использовать мощь обеих технологий при построении машинно-независимых приложений, использующих, кроме того, универсальный формат данных при обмене информации.

Более подробно о XML можно почитать в дополнении xml.doc

Контрольные вопросы

Что такое протокол

Зачем нужны протоколы

Какие протоколы используются при создании собственных протоколов

Лабораторная работа №8
Защита http протокола https

Настройка HTTPS в Apache

Веб-сервер Apache полностью поддерживает работу по HTTPS. Для того, чтобы активировать поддержку HTTPS на уже установленном Apache необходимо выполнить следующее.

Создание ключа и ssl-сертификата

Использование самоподписанных сертификатов хоть и защищает от пассивного прослушивания, тем не менее не гарантирует клиентам, что сервер является именно тем сервером, который им нужен. Преймуществом самоподписанных сертификатов является их бесплатность. Сертификат, подписанных компанией-сертификатором (Certificate authority) стоит денег.

Для создания ключа и сертификата вводим команду:

```
openssl req -new -x509 -days 30 -keyout server.key -out server.pem
```

На вопрос «Enter PEM pass phrase:» отвечаем паролем, подтверждаем и запоминаем. На все последующие вопросы отвечаем произвольно, можно просто щелкать по Enter соглашаясь с предложенными вариантами, только на вопрос «Common Name (eg, YOUR name) []:» отвечаем именем сайта, для которого создаем сертификат, например www.example.com.

После ответа на все вопросы в директории должны появиться два новых файла - server.pem и server.crt (ключ и сертификат, соответственно).

Чтобы использовать сгенерированный ключ нужно знать пароль введенный нами, и Apache будет спрашивать его у нас при загрузке, а к чему нам лишние вопросы от демонов? :) Поэтому снимаем пароль с ключа:

```
cp server.key{,.orig}
openssl rsa -in server.key.orig -out server.key
rm server.key.orig
```

Скопируем их в /etc/ssl и назначим файлу ключа права чтения только администратору:

```
sudo cp server.pem /etc/ssl/certs/
sudo cp server.key /etc/ssl/private/
sudo chmod 0600 /etc/ssl/private/server.key
```

Настройка Apache

Для начала необходимо активировать mod_ssl:

sudo a2enmod ssl

А затем включить настройки HTTPS сайта по умолчанию:

sudo a2ensite default-ssl

Теперь необходимо отредактировать файл с настройками HTTPS сайта по умолчанию, указав в нём пути к вашим сертификатам. Сам файл называется /etc/apache2/sites-enabled/default-ssl (или /etc/apache2/sites-enabled/default-ssl.conf).

В этом файле рекомендуется после директивы

SSLEngine on

добавить строчку

SSLProtocol all -SSLv2

дабы запретить использование устаревшего протокола SSLv2.

Дальше вам необходимо отредактировать параметры, ответственные за сертификаты.

```
# Публичный сертификат сервера
SSLCertificateFile    /etc/ssl/certs/server.pem
# Приватный ключ сервера
SSLCertificateKeyFile /etc/ssl/private/server.key
```

Теперь просто перезагрузите Apache:

sudo service apache2 restart

И если все параметры указаны верно, ваши сайты станут доступны по HTTPS.

Перенаправление HTTP запросов на HTTPS

Если вы хотите запретить использование HTTP, то самым разумным будет перенаправлять все HTTP запросы к страницам на их HTTPS адрес. Чтобы добиться этого можно, например, организовать перенаправление с помощью mod_rewrite. Для этого сначала необходимо его активировать:

sudo a2enmod rewrite
sudo service apache2 restart

Затем изменить файл /etc/apache2/sites-enabled/000-default, отвечающий за виртуальный хост по умолчанию для HTTP запросов. В этот файл надо добавить внутрь блока

<VirtualHost *:80>
...
</VirtualHost>

строчки:

RewriteEngine On
RewriteCond %{HTTPS} off
RewriteRule (.*) https://%{HTTP_HOST}%{REQUEST_URI}

При этом все настройки директорий можно удалить, поскольку по HTTP на ваши сайты всё равно будет не попасть.

Всё, теперь ещё раз перезапустите Apache и убедитесь, что при заходе по HTTP вы автоматически перенаправляетесь на HTTPS страницу.

Контрольные вопросы
Что такое ssl
зачем нужен файл .pem
Зачем нужен мод ssl
Зачем нужен файл .serv
Что защищает ssl

Лабораторная работа №9
Реализация клиент-серверного приложения

требуется написать клиент серверное приложение, которое бы использовало протокол для общения с клиентами, а так же провести мероприятия по защите сервера приложений.

регистрация в системе
URL http://127.0.0.1/reg.php
запрос
POST [username, login, password, phone, mail, dateofbidth, inn]
ответ
XML

<answer>
<result>true</true>
<info>Пользователь зарегистрирован</info>
</answer>

авторизация в системе
URL http://127.0.0.1/login.php
запрос
POST [login, password]
ответ
<answer>
<result>true</result>
<uid>123456789</uid>
<account>123456789</account>
<balance>100</balance>
<url>account.php</url>
</answer>

проверка реквизита
URL http://127.0.0.1/chech.php
запрос
POST[account]
ответ
<answer>
<result>true</result>
<info>Василий пупкин</info>
</answer>

поплнение баланса
URL http://127.0.0.1/billing.php
запрос
POST[account, sum]
ответ
<answer>
<result>true</result>
<info>баланс пополнен</info>
<balance>150</balance>
</answer>

Продемонстрировать работу приложения (сервера и клиента) на заранее подготовленных тестовых данных. Рассказать о методах защиты протокола.

Библиографический список

1. Смирнова Г.Н., Тельнов Ю. Ф. Проектирование экономических информационных систем / Московский государственный университет экономики, статистики и информатики, 2004.

2. Гвоздева Т.В., Баллод Б.А. Проектирование информационных систем / «Феникс», 2009.

3. Соловоьев И.В., Майоров А.А. Проектирование информационных систем / «Академический проект», 2009.

4. Голицына, О. Л. Информационные системы [Текст] : учебное пособие / О. Л.