Student's Individual Study #3

Discipline: Linux for DevOps

Student: Suleimenov Ayan

Topic: Installing packages

Target: Prepare environment for your programs by installing required packages.

Tasks

1. Define packages that are needed for the project, including libraries installable through package managers.
2. Install required packages on the system.
3. Setup firewall rules for secure network access.
4. Test installed packages using a smoke test.

Step-by-step report

1. Defining required packages

Для корректной работы backend-сервера на Go необходимо установить следующие пакеты и зависимости:

- **golang-go** — основной компилятор и инструменты Go.
- **git** — система контроля версий.
- **curl** — инструмент для HTTP-запросов.
- **make** — утилита автоматизации сборки.
- **ufw** — настройка и управление брандмауэром.
- **gin-gonic/gin** — HTTP-фреймворк для Go.
- **gorm.io/gorm** — ORM-библиотека для работы с базами данных.
- **gorm.io/driver/sqlite** — драйвер SQLite для GORM.
- **github.com/joho/godotenv** — библиотека для загрузки переменных окружения.

2. Installation process

Для установки всех пакетов и подготовки окружения был создан исполняемый скрипт `install_env.sh`. Он выполняет следующие действия:

1. Обновляет систему.
2. Устанавливает базовые утилиты и Golang.
3. Создаёт рабочую директорию и инициализирует модуль Go.
4. Устанавливает внешние зависимости через `go get`.
5. Настраивает UFW для разрешения SSH, HTTP и PostgreSQL.

3. Firewall configuration

Для защиты системы настроен брандмауэр с помощью UFW:

```
sudo ufw --force enable
sudo ufw allow ssh
sudo ufw allow 8080/tcp
sudo ufw allow 5432/tcp
sudo ufw status
```

После выполнения этих команд система разрешает доступ по SSH, HTTP-порту 8080 (для Gin) и 5432 (для PostgreSQL).

4. Smoke test

Для проверки успешной установки и работы библиотек был создан тестовый сервер на Go:

```
package main

import (
    "fmt"
    "github.com/gin-gonic/gin"
    "gorm.io/driver/sqlite"
    "gorm.io/gorm"
)

func main() {
    fmt.Println("Running smoke test...")

    db, err := gorm.Open(sqlite.Open("test.db"), &gorm.Config{})
    if err != nil {
        panic(" Database connection failed")
    }

    sqlDB, err := db.DB()
    if err != nil {
        panic(" Failed to get generic database object")
    }
    defer sqlDB.Close()

    fmt.Println(" GORM connected successfully")

    r := gin.Default()
    r.GET("/", func(c *gin.Context) {
        c.JSON(200, gin.H{"message": "Smoke test passed!"})
    })

    fmt.Println(" Gin server running at http://localhost:8080")
    r.Run(":8080")
}
```

При запуске сервера на `http://localhost:8080` возвращается JSON-ответ:

```
{"message": "Smoke test passed!"}
```

Conclusions

В ходе работы были установлены и протестированы все необходимые пакеты для backend-разработки на Go. Создан автоматизированный скрипт, который подготавливает окружение, настраивает брандмауэр и выполняет smoke-тест, подтверждающий корректность установки. Полученное решение является повторяемым, исполняемым и полностью соответствует требованиям индивидуального задания.

Repository

Исходные файлы и скрипт размещены в репозитории: `https://github.com/ssuleimenovv/linux`$_{f}or_{d}evops/tree/main/sis3$