

鲜蔬配送管理系统的设计与开发

摘要

伴随着互联网+时代的到来,传统的鲜蔬配送平台已不能满足当今飞速发展行业的需要,物流体系不健全、配送信息不完善等现实问题,制约着鲜蔬行业发展的进程。为解决这些问题,设计开发了一个针对鲜蔬配送的信息化管理系统。

本文设计开发的鲜蔬配送管理系统是一个基于 Web 应用、系统架构选用方便操作的 B/S 架构的系统,该系统目的在于实现企业、供应商、客户三者间商品信息共享,同时为客户和企业提供一个网上配送一体化的交易平台。系统功能上包含九大模块,即登录注册、鲜蔬信息管理、进货及配送管理、订单及配送管理、供应商信息管理、库存管理、仓库管理及权限控制。三大业务流程为进货及配送、订单及配送和库存管理。进货及配送建立了从供应商采购鲜蔬再由专人配送至冷库保鲜的供应商链,该供应链环节由系统紧密监控,提供信息实时化处理。订单及配送建立了客户线上采购鲜蔬再由专人配送至客户处的直销链,该链环节由系统实现其高度信息化。库存管理建立了完整的鲜蔬出入库链,由系统视实际情况严格控制数据误差。系统其他业务主要基于基础信息管理,包含对员工、客户、供应商、仓库等基础类信息和采购单、订单、入库出库单等业务信息的管理。

本文实现的系统解决了现有配送模式下信息化不足的弊端,及鲜蔬不易保鲜造成的数据不对等,采用权限控制实现多角色用户对系统的使用,保证了系统数据的安全性、准确性。

本文介绍了一个具有商业应用价值的鲜蔬配送管理系统,通过该系统充分实现了对鲜蔬配送的信息化管理,极大地推进了鲜蔬行业的发展。

关键词: 鲜蔬配送, 信息化, Web 应用, B/S 结构。

DESIGN AND DEVELOPMENT OF FRESH VEGETABLE DISTRIBUTION MANAGEMENT SYSTEM

ABSTRACT

With the advent of the Internet era, the traditional fresh vegetable distribution platform has been unable to meet the needs of the rapid development of the industry today, logistics system is not perfect, distribution information is not perfect and other practical problems, restricting the development of fresh vegetable industry process. In order to solve these problems, an information management system for fresh vegetable distribution is designed and developed.

The fresh vegetable distribution management system designed and developed in this paper is based on Web application, and the system structure is based on the B / S architecture which is easy to operate. The purpose of the system is to realize the sharing of commodity information among enterprises, suppliers and customers. At the same time for customers and enterprises to provide an integrated online distribution trading platform. The system includes nine modules, that is, login registration, fresh vegetable information management, purchase and distribution management, order and distribution management, supplier information management, inventory management, warehouse management and authority control. The three major business processes are purchase and distribution, order and distribution and inventory management. Procurement and distribution established from suppliers The fresh vegetables are then distributed to the chain of suppliers in the cold storage. The supply chain is closely monitored by the system to provide real-time processing of information. The order and distribution set up the direct selling chain of purchasing fresh

vegetables online and then distributing it to the customer by special person. The chain is highly information based on the system. Inventory management established a complete fresh vegetable input and storage chain, and the system strictly controlled the data error according to the actual situation. The other business of the system is mainly based on basic information management, including the management of basic information of employees, customers, suppliers, warehouses, purchase orders, warehouse and other business information.

The system realized in this paper has solved the shortcoming of information deficiency in the current distribution mode, and the data inequality caused by the difficulty of fresh vegetable preservation. The power control is adopted to realize the use of the system by multi-role users, and the security of the system data is ensured. Accuracy.

In this paper, a management system of fresh vegetable distribution with commercial application value is introduced. Through this system, the information management of fresh vegetable distribution is fully realized, and the development of fresh vegetable industry is greatly promoted.

Key words: Fresh Vegetable Distribution, information, B/S structure, Web application.



目 录

| | |
|------------------------|----|
| 1 绪论..... | 1 |
| 1.1 课题研究背景..... | 1 |
| 1.2 课题研究现状..... | 1 |
| 1.3 课题研究内容..... | 2 |
| 1.4 课题论文结构..... | 2 |
| 1.5 本章小结..... | 3 |
| 2 相关技术介绍..... | 4 |
| 2.1 系统开发工具..... | 4 |
| 2.1.1 系统开发工具..... | 4 |
| 2.1.2 数据库及其工具..... | 4 |
| 2.2 关键技术介绍..... | 4 |
| 2.2.1 Spring Boot..... | 4 |
| 2.2.2 Spring MVC..... | 4 |
| 2.2.3 Mybatis..... | 5 |
| 2.2.4 Bootstrap..... | 5 |
| 2.2.5 Swagger..... | 5 |
| 2.2.6 Ajax..... | 5 |
| 2.3 本章小结..... | 6 |
| 3 系统需求分析..... | 7 |
| 3.1 可行性分析..... | 7 |
| 3.2 系统功能需求..... | 7 |
| 3.2.1 登录注册..... | 7 |
| 3.2.2 鲜蔬信息管理..... | 8 |
| 3.2.3 供应商信息管理..... | 8 |
| 3.2.4 员工信息管理..... | 9 |
| 3.2.5 进货及配送管理..... | 9 |
| 3.2.6 订单及配送管理..... | 10 |
| 3.2.7 库存管理..... | 10 |
| 3.2.8 仓库管理..... | 11 |
| 3.2.9 权限控制..... | 11 |
| 3.2 系统性能需求..... | 11 |
| 3.3 本章小结..... | 12 |
| 4 系统设计..... | 13 |



| | |
|-----------------------|----|
| 4.1 系统概要设计..... | 13 |
| 4.1.1 系统的功能结构..... | 13 |
| 4.1.2 系统的流程图..... | 14 |
| 4.2 系统数据库设计..... | 14 |
| 4.2.1 ER 图设计..... | 14 |
| 4.2.2 系统数据库表定义..... | 15 |
| 4.3 系统模块详细设计..... | 25 |
| 4.3.1 登录注册模块..... | 25 |
| 4.3.2 鲜蔬信息管理模块..... | 26 |
| 4.3.3 供应商信息管理模块..... | 26 |
| 4.3.4 员工信息管理模块..... | 27 |
| 4.3.5 进货及配送管理模块..... | 28 |
| 4.3.6 订单及配送管理模块..... | 28 |
| 4.3.7 库存管理模块..... | 29 |
| 4.3.8 仓库管理模块..... | 31 |
| 4.3.9 权限控制模块..... | 31 |
| 4.4 本章小结..... | 31 |
| 5 系统实现与测试..... | 32 |
| 5.1 系统数据库的实现..... | 32 |
| 5.1.1 建立数据库..... | 32 |
| 5.1.2 连接数据库..... | 32 |
| 5.2 系统功能模块的实现..... | 33 |
| 5.2.1 用户登陆注册模块..... | 33 |
| 5.2.2 鲜蔬信息管理模块..... | 36 |
| 5.2.3 供应商信息管理模块..... | 38 |
| 5.2.4 员工信息管理模块..... | 39 |
| 5.2.5 进货及配送管理模块..... | 41 |
| 5.2.6 订单及配送管理模块..... | 42 |
| 5.2.7 库存管理模块..... | 45 |
| 5.2.8 仓库管理模块..... | 48 |
| 5.2.9 权限控制模块..... | 49 |
| 5.3 系统功能测试..... | 50 |
| 5.3.1 对登录注册的测试..... | 50 |
| 5.3.2 对进货过程的测试..... | 51 |
| 5.3.3 对出货过程的测试..... | 52 |
| 5.3.4 对基础信息管理的测试..... | 53 |
| 5.3.5 对权限控制的测试..... | 54 |
| 5.4 系统性能测试..... | 54 |
| 5.5 本章小结..... | 55 |

| | |
|--------------|----|
| 6 总结与展望..... | 56 |
| 6.1 总结..... | 56 |
| 6.2 展望..... | 56 |
| 参考文献..... | 58 |
| 致谢..... | 59 |
| 文献翻译 原文..... | 60 |
| 文献翻译 译文..... | 88 |

1 绪论

本章主要是对系统开发的背景，现状以及主要内容进行分析，为系统的开发进行初步的调查，并简要阐述系统所要实现的基本功能。

1.1 课题研究背景

随着我国人民生活水平和经济发展水平的提高，恩格尔系数的降低，人们追求健康的理念意识正在逐渐提升^[4]，鲜蔬配送日益成为城市居民消费的客观需要，因此，鲜蔬配送信息化管理必不可少。

根据中国电子商务研究中心发布的中国网络零售市场数据检测报告来看，电商渗透率比想象中发展的快了许多，也意味着整个生鲜市场，尤其是电商市场正在逐步回暖，生鲜蔬菜配送再次热了起来。鲜蔬产品由其产品的特殊性（新鲜、保质期短、产品价格低、物流配送不便等特点），造成其物流配送具有极大地难度，因此一家专门支持鲜蔬产品网上交易、物流配送专业化、符合产业发展实际需求的电子商务服务平台是急需的。

鲜蔬配送管理系统是典型的信息管理系统，配送管理系统（Distribution Management System，简称 DMS）是指利用互联网技术和相关硬件设施，通过对信息一系列操作，实现相关信息数字化处理的系统，而鲜蔬配送管理系统是应用于鲜蔬公司对其从供应商进货，配送到仓库存贮货物，产品销售，进而将鲜蔬货物配送到客户这一系列过程的信息操作和管理。目的是为了更好的帮助公司运营鲜蔬配送产业，了解当前公司的运作情况，统计分析货物的流动状况，最后以直观的数据呈现给用户。大大的减少了手动进行这一系列信息操作的时间，解决低下的工作效率和不良的工作效果，并且保证了信息的正确性，加强了企业的对鲜蔬产品、人员信息等的管理力度。

1.2 课题研究现状

经过数十年的改革与发展，我国大中城市目前已基本具备了发展蔬菜配送业的经济环境和市场条件。城市鲜蔬配送业正呈现方兴未艾的趋势，但在信息化建设和管理方面还存在着一些制约其发展的问题。

O2O 供应链鲜蔬配送市场从配送客户分类来说主要分为中小餐厅食材配送，企事业单位食堂和酒店高端餐厅食材配送，垂直行业食材配送（如幼儿园食材配送，西餐厅食材配送），垂直品类的食材配送（冻货，调料，海鲜，肉类），生鲜超市食材配送。

国内 O2O 供应链鲜蔬配送典型代表是通过“互联网+资本”的方式重构中小餐厅配送的企业，给广大蔬菜配送行业的创业者们一个很好的启发，通过互联网这种方式来改造优化传统行业，其带来的效率是惊人的，可见其可复制的信息化手段和管理体系是成功与否的关键。

基于国内鲜蔬配送现状的研究，鲜蔬配送信息化管理体系架构应运而生。该配送体系可以概括为找到优质鲜蔬（供应商基地）、通过运输途径环节，把鲜蔬送至冷库保鲜，继而配送至客户（餐饮及团购等）。通过该体系，可以对无公害蔬菜从生产加到到配送的各个环节实施信息化管理，确保无公害鲜蔬安全、高效、无误地配送到客户手中。

1.3 课题研究内容

结合系统业务流程以及公司信息管理的需要，设计与实现了一个鲜蔬配送管理系统，主要包括建立了完整性和一致性的数据库。另通过权限控制保证了数据的安全性。本系统主要包括的功能如下：

- （1）基本信息录入和管理：对相关员工的基本信息进行操作，建立系统内部信息网络。以及供应商链和客户链相关信息查看与操作。
- （2）采购进货配送管理：系统对鲜蔬采购继而配送到公司仓库过程有信息化实时监控，保证数据的可靠性。
- （3）客户线上交易生成订单：客户网上选购鲜蔬生成订单推送给后台。
- （4）订单出货配送管理：系统对鲜蔬订单配送至客户过程信息化实时监控。
- （5）库存货物信息管理：系统对冷库中鲜蔬存储量进行实时操作与监控，保证鲜蔬的新鲜度，务求配送至客户处的鲜蔬绿色、无害、健康。

1.4 课题论文结构

本文采用面向对象的分析方法，大致结构如下：

第一章绪论阐述了鲜蔬配送管理系统选题的背景、研究现状，从而引出课题

的研究内容；

第二章相关技术阐述了本系统开发的技术和工具；

第三章系统概述对需求进行了详细分析并划分功能模块；

第四章系统设计详细阐述了系统的概要设计、数据库设计和模块详细设计；

第五章系统实现阐述了本系统具体实现效果和代码；

第六章总结和展望阐述了本文总结和对系统的展望；

文末是本文参考文献和致谢。

1.5 本章小结

本章主要介绍了选题背景，关于鲜蔬配送行业的现状研究以及内容，鲜蔬配送管理系统是为鲜蔬配送公司的一个信息管理系统。通过现在的 O2O 模式实现鲜蔬配送与互联网的完美结合，从而简化鲜蔬交易的流程，保证鲜蔬配送业务相关数据信息的统一性、安全性和可靠性。另还对本文的论文结构进行了简单介绍。

2 相关技术介绍

本章节主要介绍系统开发过程中所用到的相关工具和技术。

2.1 系统开发工具

2.1.1 系统开发工具

使用 Spring Initializr 快速创建 Spring Boot 项目，通过 IntelliJ IDEA 进行系统功能代码的实现。应用部署服务器采用 Spring Boot 内嵌 Tomcat。通过 Maven 管理项目的依赖、编译、文档等信息。框架采用：Spring MVC、Spring Boot、Bootstrap 和 Mybatis。项目使用 Swagger 构建 RESTful API 实现前后端分离开发。

2.1.2 数据库及其工具

系统使用开源关系型数据库管理系统 MySQL 对数据库表进行维护。数据库管理和开发工具：Navicat for MySQL，它通过使用数据表样式的网格查看及一系列数据编辑工具来添加、修改和删除记录，方便编辑数据。Navicat 提供了有效管理数据所需的工具，并确保能顺利进行。

2.2 关键技术介绍

2.2.1 Spring Boot

Spring Boot 是一套全新的框架，以约定大于配置的核心思想，默认进行了很多设置，多数 Spring Boot 应用只需要很少的 Spring 配置^[9]。

使用 Spring Boot 可以轻松的创建独立运行的程序，独立的服务组件，是实现分布式架构、微服务架构的利器。一键继承开发部署到发布的整个工作流程，使开发者更加关注实现业务。

Spring Boot 特性有：方便对外输出各种形式的服务，比如：REST API、Web、Streaming；支持关系数据库和非关系型数据库；支持运行期内嵌容器，如：Tomcat、Jetty；通过提供的 starter 实现自动管理依赖和自带应用监控。

2.2.2 Spring MVC

Spring MVC 是 Spring 提供的一个强大的 Java web 框架。Spring MVC 借助注解提供几乎 POJO 的开发模式，使得控制器的开发和测试更加简单便捷。控制

器不直接处理请求而是委托给 Spring 上下文的其它 bean，这些 bean 通过 Spring 的依赖注入被注入到控制器中。

Spring MVC 主要由 DispatcherServlet (开发 Web 应用)、处理器 (控制器)、处理器映射、视图、视图解析器组成。其两个核心分别是：

处理器映射：选择使用哪一个控制器来处理当前请求

视图解析器：选择结果如何渲染

2.2.3 Mybatis

MyBatis 是一款支持定制化 SQL、存储过程以及高级映射的持久层框架。

Mybatis 功能架构包括：API 接口层、数据处理层和基础支撑层。

其优点包括：通过提供 DAL 层，将业务逻辑和数据访问逻辑分离，解除了 SQL 与程序代码的耦合；提供 xml 标签，支持编写动态 SQL；提供对象关系映射标签支持对象关系组建维护等。

2.2.4 Bootstrap

Bootstrap 是一个开发响应式和移动优先的 Web 应用的 HTML、CSS、JavaScript 的流行框架。Bootstrap 实现了一套代码在不同的设备上自适应显示想要的视图功能，还提供了大量美观的 HTML 元素组件和 jQuery 插件。

2.2.5 Swagger

Swagger 是一个规范和完整的框架，用于生成、描述、调用和可视化 RESTful 风格的 Web 服务。使客户端和文件系统服务器以同样的速度更新。文件的方法、参数和模型紧密集成到服务器端的代码，允许 API 来保持同步。

它可以轻松的整合到 Spring Boot 中，并与 Spring MVC 配合生成具有互动性的 RESTful API 文档。它既可以减少我们创建文档的工作量，同时说明内容又整合入实现代码中，让维护文档和修改代码整合为一体，可以让我们在修改代码逻辑的同时方便的修改文档说明。另外 Swagger 也提供了的页面测试功能来调试每个 RESTful API。

用 Swagger 文件生成互动的 API 文档是最精简的，它展示了资源、参数、请求、响应。但是它不会提供你的 API 如何工作的其他任何一个细节。

2.2.6 Ajax

AjAx (Asynchronous JavaScript and XML) 是一种用户创建快速动态网页的

技术。通过在后台与服务器进行少量的数据交换就可以实现网页异步更新。通俗地讲, Ajax 是 JS 调用异步通讯组件并使用格式化的数据来更新 web 页面上的内容或操作过程^[9]。

2.3 本章小结

本章阐述了系统开发过程用到的开发工具和关键技术。开发工具包括功能代码开发相关工具和数据库及其应用开发工具。关键技术几乎涵括了系统开发技术, 为读者介绍了它们是什么及各自的优缺点分析。

3 系统需求分析

本章节主要介绍对系统多方面的分析。

3.1 可行性分析

本系统采用 Integrated Development Environment (IDE) 作为开发平台，以 Sql Server 作为管理数据的数据库工具，开发语言选用 JAVA，使用 Maven 项目管理工具，开发采用 Spring MVC、Spring Boot 等框架，采用 RESTful API 架构。

综上所述，我们认为鲜蔬配送管理系统从技术实现的角度上来讲是可行的。

3.2 系统功能需求

鲜蔬配送管理系统可以划分为五个角色和九个功能模块：

角色：

- ❖ 客户
- ❖ 采购员
- ❖ 配送司机
- ❖ 仓库管理员
- ❖ 办公室

模块：

- ❖ 登录注册
- ❖ 鲜蔬信息管理
- ❖ 供应商信息管理
- ❖ 员工信息管理
- ❖ 进货及配送管理
- ❖ 订单及配送管理
- ❖ 库存管理
- ❖ 仓库管理
- ❖ 权限控制

3.2.1 登录注册

(1) 用户登录

系统不同用户角色均通过登录进入到系统，系统根据用户角色实现权限控制，不同角色进入到系统不同功能模块。用户登录系统时需要进行用户账号密码的验证，出于安全性考虑，防止外来攻击者进入系统造成数据破坏。系统自动识别用户权限，根据登录用户角色不同进入系统不同操作界面。

用户输入用户名和密码后，系统对用户输入的账户信息进行验证，通过验证之后才能进入系统的不同主界面；否则提示用户名或密码错误，要求用户重新输入登录信息。

（2）用户注册

由于系统是适用于一体化鲜蔬直营配送的管理系统，且客户为长期合作中小型餐饮对象，故用户注册模块只向客户角色开放，系统剩余其他角色由系统生成。

3.2.2 鲜蔬信息管理

鲜蔬信息管理主要是系统“办公室”角色进行操作，并由系统权限控制进行判断用户是否可以操作该模块内容。

该模块，主要是对鲜蔬产品的增删改操作，其中包括：鲜蔬基本信息管理，鲜蔬种类管理。

（1）蔬菜基本信息管理对应页面包括：对蔬菜数据库编号，名称，价格，描述，创建时间，状态等鲜蔬基本信息的修改管理；对鲜蔬上架下架销售的管理；添加新鲜蔬产品并验证该蔬菜是否已经存在；根据鲜蔬各基本信息精准查询和模糊查询。

（2）鲜蔬种类管理对应页面包括：查看鲜蔬种类数据库编号，种类名称及创建时间等基本信息；添加新种类鲜蔬并验证是否已存在；根据鲜蔬种类编号等基本信息进行精准查询。

3.2.3 供应商信息管理

供应商信息管理主要是由系统“办公室”和“采购员”角色进行操作，主要是对供应商相关信息的增删改查，动态地更新供应商与公司合作内容及现状。由采购员前往洽谈合作，正式签订供货合同。

该模块，主要包括：“办公室”角色下的查看和增加新供应商并完善其基本信息，“采购员”角色下的查看和添加合作供应商信息。

（1）“办公室”角色下对应页面功能包括：查看当前所有供应商名称，地址，

电话,合作时间等信息并可进行精准查询;更改供应商是否与公司合作状态;添加新合作供应商及其提供鲜蔬产品;查看及删除单个供应商所提供鲜蔬产品。

(2)“采购员”角色下对应页面功能包括:查看合作供应商编号、地址等基本信息且可根据单个基本信息进行精准查询;查看单个供应商合作鲜蔬产品并进行添加和删除操作

3.2.4 员工信息管理

员工信息管理主要是系统“办公室”角色下对系统内部各角色工作人员基本信息的管理和各用户角色下对用户本身基本信息的管理。

该模块,主要包括:用户角色对用户自身信息的查看和修改,“办公室”角色下分别对内部工作人员的增加、查询和删除。

(1)用户角色下对应页面功能包括:用户登录系统后进入个人主页查看个人基本信息并进行相应修改,个人信息包括姓名、年龄等基础信息,也包括登录系统用户名密码,岗位角色等不可修改内容,以只可读形式呈现给用户。

(2)“办公室”角色下对应页面功能包括:该用户角色下按部门显示员工信息,人员部门有:客户部、供应商部、采购部、库管部和物流部。其中,除客户部外,剩余部门人员均有“办公室”角色操作添加并完善相关信息。该角色下拥有查看,添加,查询和删除各部门人员信息等权限,其中,查询为精准查询且可按照人员不同信息进行批量查询。

3.2.5 进货及配送管理

进货及配送管理主要对应系统流程中的“进货”和“进货配送”。该流程下由系统“采购员”和“配送司机”角色进行操作。

该模块,主要包括:采购员角色下的采购鲜蔬生成进货单,查看历史进货单。配送司机角色下的配送当前进货单至仓库及查看历史进货单。

(1)生成进货单对应页面功能包括:采购员前往供应商处进行鲜蔬进货,浏览该供应商信息并填写进货单,进货单生成后采购员查看当前配送司机工作状态,安排空闲司机进行进货配送。进货单主要信息包括:进货供应商、配送司机、配送仓库、进货总价和总重。

(2)查看历史进货单对应页面功能包括:采购员查看个人所负责历史进货单及配送情况;配送司机查看个人所负责历史进货单。

(3) 配送进货单对应页面功能包括：采购员安排司机进货配送后，在该司机系统页面下显示司机当前需配送进货单，司机点击确认根据进货单信息进行进货配送。

3.2.6 订单及配送管理

订单及配送管理主要对应系统流程中“订单的生成”及“订单配送”环节。该流程环节由系统角色“客户”和“配送司机”权限下操作。

该模块，主要包括：“客户”权限下的浏览在售鲜蔬并购买产品生成订单，查看客户历史订单。“配送司机”权限下的查看待配送订单及历史配送订单。

(1) 生成订单对应功能页面包括：客户登录系统后查看公司当前在售鲜蔬产品信息，在线下单所需鲜蔬，生成订单推送给后台。

(2) 查看历史订单对应功能页面包括：客户查看个人账户历史订单及订单详情。配送司机查看个人负责配送的历史订单信息。

(3) 配送订单对应功能页面包括：配送司机查看当前待配送订单，根据待配送订单信息至仓库配货并配送至客户处。

3.2.7 库存管理

库存管理主要对应的是系统流程中的“进货鲜蔬入库”和“订单鲜蔬出库”环节。由系统“库存管理员”角色权限下操作。主要是对公司进货和根据订单出货的鲜蔬产品的库存管理，动态的管理和记录仓库中每种鲜蔬产品的入库，出库情况，并能统计和校对现有及历史的库存信息。

该模块，主要包括：库存管理员根据进货单和订单生成相应入库单和出库单，对入库单、出库单进行查询和统计历史入库单、出库单操作，仓库现有鲜蔬产品数量的查询，统计和管理操作。

(1) 仓库鲜蔬产品管理对应功能包括：库存管理员对本仓库所含鲜蔬产品的查询，更改数量，删除以及查看等。

(2) 入库单生成对应功能包括：配送司机配送进货鲜蔬抵达指定仓库时，库存管理员查看司机所携带进货单信息安排鲜蔬入库并生成入库单。其中，由于鲜蔬产品的特殊性（易破损、保鲜性低等），库存管理员根据实际抵达仓库鲜蔬明细填写入库单信息及明细，如每种鲜蔬产品实际重量，保证数据准确性。

(3) 出库单生成对应功能包括：配送司机携带订单信息至仓库进行鲜蔬配

货，库存管理员根据司机所携带订单查询该订单明细，并根据订单明细进行鲜蔬配货。由于鲜蔬配送过程中会有损耗，为保证送达至客户手中的鲜蔬足重足鲜，库存管理员根据个人经验出库每种鲜蔬重量。

(4) 历史出库、入库单对应功能包括：库存管理员对本仓库历史出库、入库单的查询等操作，包括其明细。

3.2.8 仓库管理

仓库管理主要是系统“办公室”角色权限下的操作，主要对公司现有及历史仓库的统计和管理。

该模块，主要包括：办公室对仓库的增加和删除，对各仓库现有鲜蔬产品数量等的实时统计。

(1) 对仓库的管理对应包括：办公室权限下对现有仓库的精准查询；增加公司新开仓库；删除公司废弃仓库。

(2) 对各仓库现有菜品的管理包括：办公室权限下按仓库查看该仓库现有鲜蔬情况，并进行实时监测。

3.2.9 权限控制

权限控制主要是出于系统安全性和明确公司内部员工级别的考虑。系统根据角色不同赋予其不同权限，而不同权限下对系统的操作也不尽相同。根据用户权限严格区分用户可操作功能，防止外来恶意攻击或用户恶意操作。当用户试图通过不当操作使用其权限外系统的功能时，均提示“权限不够”。

该模块，主要包括系统六大角色：客户、供应商、采购员、配送司机、库存管理员和办公室。角色权限根据用户角色需要赋予。

3.2 系统性能需求

(1) 安全性

由于涉及个人信息资料等，所以对系统安全性有较高的要求。设置不同用户角色权限来控制用户对系统不同功能的操作；对每一个系统用户密码进行哈希加密保证系统用户安全性；对数据操作进行严格控制，增强后台数据录入安全性等保证系统的安全。

(2) 数据精确度

本系统对数据查询分为数据精确查询和数据模糊查询。前台输入数据多采用

下拉列表选择等方式提交数据，保证数据库数据高度统一。

（3）容错性

系统对前台输入数据采用前端验证控件检验数据准确性。

3.3 本章小结

本章主要对论文课题进行了需求分析，包括技术层面的可行性分析和系统实际应用方面的系统需求，根据业务流程将系统划分为九个模块和五个角色，并针对各个模块进行了基础的模块功能分析。肯定了系统的可行之处，后文会以本章为基础，详细分析介绍系统。

4 系统设计

本章主要陈述了系统概要设计、数据库设计和各模块的详细设计。

4.1 系统概要设计

4.1.1 系统的功能结构

根据第三章系统的需求分析，系统的主要功能模块如下图所示：

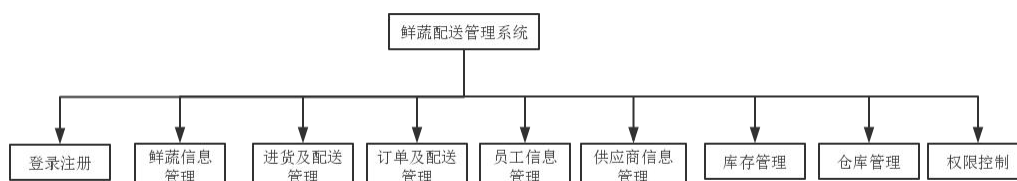


图 4-1 系统功能模块图

整个系统被划分成九个子模块，每个模块的大致功能如下：

- (1) 登录注册模块，各系统用户角色登录系统，客户角色注册系统用户等功能。
- (2) 鲜蔬信息管理模块，包括鲜蔬新增，鲜蔬信息修改，鲜蔬上架及下架，鲜蔬查询，价格管理等功能。
- (3) 供应商信息管理模块，包括供应商的新增删除，管理供应商提供鲜蔬品种，供应商的查询，供应商信息修改等功能。
- (4) 进货及配送管理模块，包括采购员进货，安排进货配送，进货鲜蔬入库，进货单的增删改查等功能。
- (6) 订单及配送管理模块，包括客户下单，订单配货及配送，订单相关信息的查询等。
- (7) 库存管理模块，分入库和出库过程，包括库存统计校对，入、出库单的查询等相功能。
- (8) 仓库管理模块，包括仓库的新增、删除、查询和修改，仓库详情的修改等功能。

4.1.2 系统的流程图

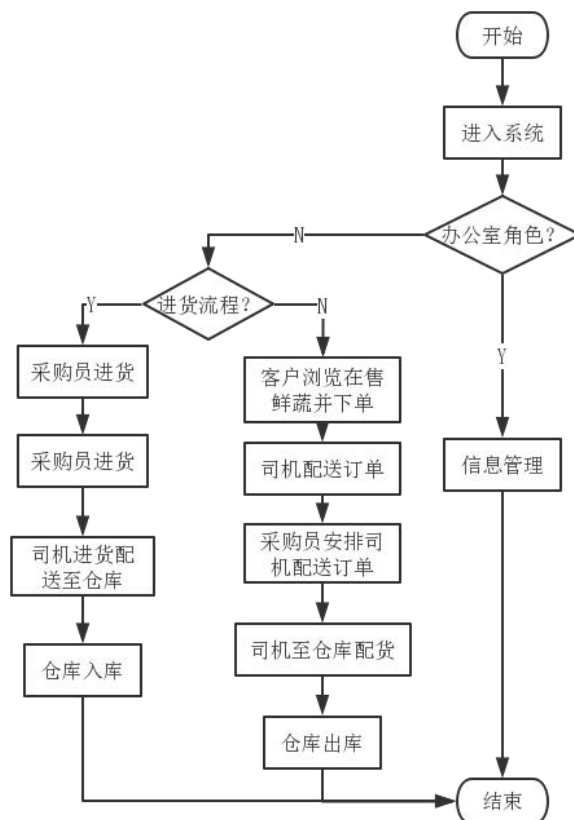


图 4-2 系统流程图

4.2 系统数据库设计

4.2.1 ER 图设计

为了可以简洁清晰地说明整个系统中实体之间的复杂联系,现采用ER图(实体-联系图)来展示它们之间的概念结构。一个好的系统全局ER图,能够准确、全面地反映用户的功能需求,立足于整个系统,对系统中的主要对象及其之间的操作流程进行简述。

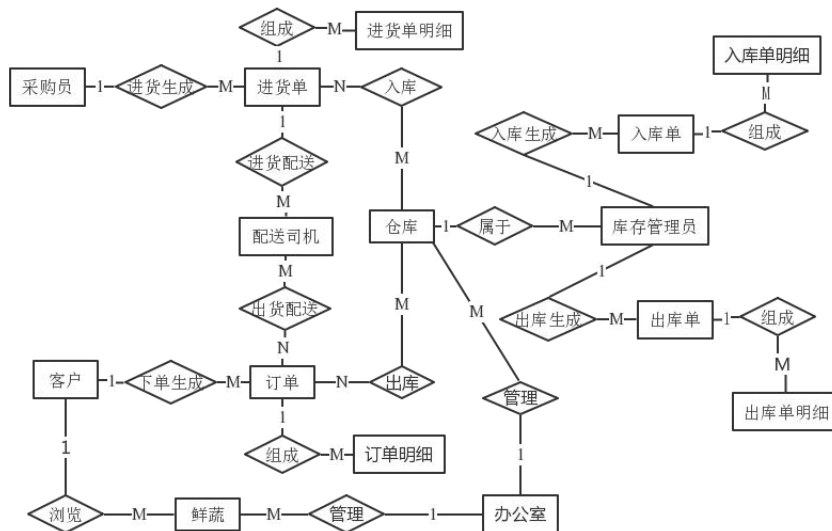


图 4-3 系统 ER 图

系统主要流程如下：

进货流程：采购员采购鲜蔬生成进货单，配送司机将采购员采购的鲜蔬运送至仓库储存保鲜，库存管理员根据进货单安排鲜蔬进库并生成对应进库单。

出货流程：客户在线浏览公司在售鲜蔬产品，根据本身需要下单生成订单，配送司机携带订单至仓库配货继而配送至客户手中。

办公室管理鲜蔬、仓库等：办公室权限下对鲜蔬价格，仓库存货进行统计和更改。

4.2.2 系统数据库表定义

系统一共包括 17 张表，下面对数据库中各张表的设计及其关联方式作详细说明。

4.2.2.1 sys_user 系统用户表

主要用于记录系统用户，其中主要储存了用户登录用户角色、用户名和密码等主要信息。为满足系统角色需要与其他表进行关联，表中还存储了多张表的外键。

表 4-1 系统用户表(sys_user)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|----------|--------|---------|-----|-----|----|------|
| id | 系统用户编号 | bigint | 20 | 0 | √ | √ |
| role_id | 角色编号 | bigint | 20 | 0 | | √ |
| dept_id | 部门编号 | bigint | 20 | 0 | | √ |
| username | 用户登录名 | varchar | 20 | 0 | | √ |
| password | 账号密码 | varchar | 100 | 0 | | |

| | | | | | | |
|-----------------------|--------|---------|-----|---|--|--|
| name | 用户姓名 | varchar | 255 | 0 | | |
| gender | 性别 | varchar | 255 | 0 | | |
| age | 年龄 | int | 11 | 0 | | |
| email | 电子邮件 | varchar | 255 | 0 | | |
| tel | 电话 | varchar | 255 | 0 | | |
| state | 状态 | varchar | 255 | 0 | | |
| location | 地址 | varchar | 255 | 0 | | |
| store_id | 仓库编号 | bigint | 20 | 0 | | |
| buyer_id | 采购员编号 | bigint | 20 | 0 | | |
| is_expired | 是否过期 | tinyint | 1 | 0 | | |
| is_enabled | 是否锁定 | tinyint | 1 | 0 | | |
| is_enabled | 是否允许 | tinyint | 1 | 0 | | |
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |

部分字段说明：

用户登录名（username）：用户身份在数据库中除系统用户编号外唯一标识，不可重复，非空；

账号密码（password）：将用户登录密码加密以乱码形式存储在数据库表中；

角色编号（role_id）：关联角色表的外键，非空；

部门编号（dept_id）：关联部门表的外键，非空；

仓库编号（store_id）：关联仓库表的外键，库存管理员独有，可空；

采购员编号（buyer_id）：关联系统用户表，供应商角色独有，可空。

4.2.2.2 sys_role 系统角色表

主要用于记录用户所属角色的相关信息，实现权限控制。

表 4-2 系统角色表(sys_role)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|-----------|--------|---------|-----|-----|----|------|
| id | 系统角色编号 | bigint | 20 | 0 | √ | √ |
| role | 角色 | bigint | 20 | 0 | | √ |
| role_name | 角色名称 | varchar | 20 | 0 | | |
| note | 备注 | varchar | 255 | 0 | | |

| | | | | | | |
|-----------------------|--------|---------|----|---|--|--|
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |

4.2.2.3 sys_auth 系统权限表

主要用于存放系统各种权限信息。

表 4-3 系统权限表(sys_auth)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|-----------------------|--------|---------|-----|-----|----|------|
| id | 系统权限编号 | bigint | 20 | 0 | √ | √ |
| auth | 权限 | bigint | 20 | 0 | | √ |
| auth_name | 权限名称 | varchar | 20 | 0 | | |
| note | 备注 | varchar | 255 | 0 | | |
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |

4.2.2.4 sys_role_auth 角色权限对应表

用于存放角色与权限对应关系，角色与权限为多对多关系,表中无主键。

表 4-4 系统角色权限对应表(sys_role_auth)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|-----------------------|--------|---------|----|-----|----|------|
| role_id | 角色编号 | bigint | 20 | 0 | | √ |
| auth_id | 权限编号 | bigint | 20 | 0 | | √ |
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |

4.2.2.5 type 鲜蔬种类表

主要用于记录鲜蔬产品种类的基本信息。

表 4-5 鲜蔬种类表(type)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|-----------------------|--------|---------|-----|-----|----|------|
| id | 鲜蔬编号 | bigint | 20 | 0 | √ | √ |
| type | 鲜蔬种类 | varchar | 255 | 0 | | |
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |

4.2.2.6 veges 鲜蔬信息表

主要用于记录公司鲜蔬产品信息。

表 4-6 鲜蔬信息表(veges)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|-----------------------|--------|---------|-----|-----|----|------|
| id | 鲜蔬编号 | bigint | 20 | 0 | √ | √ |
| type_id | 鲜蔬种类编号 | bigint | 20 | 0 | | √ |
| name | 鲜蔬名称 | varchar | 255 | 0 | | |
| price | 鲜蔬价格 | double | 10 | 2 | | |
| des | 描述 | varchar | 255 | 0 | | |
| state | 状态 | varchar | 255 | 0 | | |
| pth | 图片路径 | varchar | 255 | 0 | | |
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |

部分字段说明：

鲜蔬编号（id）：鲜蔬信息在数据库中唯一标识；

鲜蔬种类编号（type_id）：关联鲜蔬种类表的外键；

状态（state）：此字段用于判断该鲜蔬是否在售；

图片路径（pth）：此字段存放鲜蔬图片的服务器路径，用于读取图片。

4.2.2.7 storehouse 仓库表

主要用于记录公司仓库的基本信息。

表 4-7 仓库表(storehouse)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|-----------------------|--------|---------|-----|-----|----|------|
| id | 仓库编号 | bigint | 20 | 0 | √ | √ |
| name | 仓库名称 | varchar | 255 | 0 | | √ |
| tel | 电话 | varchar | 255 | 0 | | |
| address | 仓库地址 | varchar | 255 | 0 | | |
| state | 状态 | varchar | 255 | 0 | | |
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |

部分字段说明：

仓库编号（id）：仓库在数据库中唯一标识；

状态（state）：此字段用于判断该仓库是否使用中，字段值有：使用中，废弃。

4.2.2.8 storeitem 仓库明细表

主要用于记录公司各个仓库中鲜蔬明细情况，仓库与明细为一对多关系。

表 4-8 仓库明细表(storeitem)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|-----------------------|--------|---------|-----|-----|----|------|
| id | 仓库明细编号 | bigint | 20 | 0 | √ | √ |
| store_id | 仓库编号 | bigint | 20 | 0 | | √ |
| veges_id | 鲜蔬编号 | bigint | 20 | 0 | | |
| name | 鲜蔬名称 | varchar | 255 | 0 | | |
| number | 鲜蔬存储量 | double | 255 | 0 | | |
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |

部分字段说明：

仓库明细编号（id）：仓库明细信息在数据库中唯一标识；

仓库编号（store_id）：关联仓库表的外键，一对多关系，非空；

鲜蔬编号（veges_id）：关联鲜蔬信息表的外键。

4.2.2.9 supplier_veges 供应商鲜蔬对应表

主要用于记录不同供应商所提供鲜蔬明细表，为多对多关系。

表 4-9 供应商鲜蔬对应表(supplier_veges)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|-----------------------|--------|---------|----|-----|----|------|
| supplier_id | 供应商编号 | bigint | 20 | 0 | | |
| veges_id | 鲜蔬编号 | bigint | 20 | 0 | | |
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |

部分字段说明：

供应商编号（id）：关联系统用户表的外键，与鲜蔬编号为多对多关系；

鲜蔬编号（veges_id）：关联鲜蔬信息表外键，与供应商编号为多对多关系。

4.2.2.10 import 进货单信息表

主要用于记录系统采购鲜蔬所生成的进货单基本信息。

表 4-10 进货单信息表(import)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|-----------------------|---------|---------|-----|-----|----|------|
| id | 进货单编号 | bigint | 20 | 0 | √ | √ |
| buyer_id | 采购员编号 | bigint | 20 | 0 | | √ |
| supplier_id | 供应商编号 | bigint | 20 | 0 | | √ |
| driver_id | 配送司机编号 | bigint | 20 | 0 | | √ |
| totalprice | 总价 | double | 20 | 2 | | √ |
| totalweight | 总重 | double | 20 | 2 | | √ |
| store_id | 仓库编号 | bigint | 20 | 0 | | √ |
| keeper_id | 库存管理员编号 | bigint | 20 | 0 | | |
| state | 状态 | varchar | 255 | 0 | | √ |
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |

| | | | | | | |
|----------------|-------|---------|----|---|--|--|
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |
|----------------|-------|---------|----|---|--|--|

部分字段说明：

进货单编号（id）：进货单信息在数据库中唯一标识；

采购员编号（buyer_id）：关联系统用户信息表（采购员）的外键，非空；

供应商编号（supplier_id）：关联系统用户信息表（供应商）的外键，非空；

配送司机编号（driver_id）：关联用户表（配送司机）的外键，非空；

仓库编号（store_id）：关联仓库表的外键，非空；

库存管理员编号（keeper_id）：关联系统用户表（库存管理员角色）的外键。

4.2.2.11 importitem 进货单明细表

主要用于记录进货单明细，与进货单表为多对一关系。

表 4-11 进货单明细表(importitem)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|-----------------------|---------|---------|-----|-----|----|------|
| id | 进货单明细编号 | bigint | 20 | 0 | √ | √ |
| import_id | 进货单编号 | bigint | 20 | 0 | | √ |
| veges_id | 鲜蔬编号 | bigint | 20 | 0 | | √ |
| name | 鲜蔬名称 | varchar | 255 | 0 | | |
| price | 进货价格 | double | 20 | 2 | | √ |
| number | 数量 | double | 20 | 2 | | √ |
| total | 总价 | double | 20 | 2 | | |
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |

部分字段说明：

进货单明细编号（id）：进货单单项明细在数据库中唯一标识；

进货单编号（import_id）：关联进货单表的外键，非空，多对一关系；

鲜蔬编号（veges_id）：关联鲜蔬表的外键，非空；

进货价格（price）：该鲜蔬进货单价，单位/kg，两位小数。

4.2.2.12 instore 入库单表

主要用于记录司机配送至仓库中的进货鲜蔬基本信息。比如：总重、总价等。

表 4-12 入库单表(instore)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|-----|-------|--------|----|-----|----|------|
| id | 入库单编号 | bigint | 20 | 0 | √ | √ |

| | | | | | | |
|-----------------------|---------|---------|-----|---|--|---|
| import_id | 进货单编号 | bigint | 20 | 0 | | √ |
| store_id | 仓库编号 | bigint | 20 | 0 | | √ |
| keeper_id | 库存管理员编号 | bigint | 20 | 0 | | |
| driver_id | 配送司机编号 | bigint | 20 | 0 | | √ |
| totalprice | 总价格 | double | 20 | 2 | | |
| totalweight | 总重量 | double | 20 | 2 | | |
| state | 状态 | varchar | 255 | 0 | | |
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |

部分字段说明：

入库单编号（id）：入库单基本信息在数据库中唯一标识；

进货单编号（import_id）：关联进货单表的外键，非空，一对一关系；

仓库编号（store_id）：该字段表明进货鲜蔬所进仓库，仓库表外键，非空；

配送司机编号（driver_id）：该字段表明该进货鲜蔬由哪位司机进行配送，系统用户表的外键，非空；

库存管理员编号（keeper_id）：该字段表明该笔进货由哪位库管员负责入库。

4.2.2.13 instoreitem 入库单明细表

主要用于记录每张入库单的详细明细信息，与入库单表为多对一关系。

表 4-13 入库单明细表(instoreitem)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|-----------------------|---------|---------|----|-----|----|------|
| id | 入库单明细编号 | bigint | 20 | 0 | √ | √ |
| instore_id | 入库单编号 | bigint | 20 | 0 | | √ |
| veges_id | 鲜蔬编号 | bigint | 20 | 0 | | |
| name | 鲜蔬名称 | varchar | 20 | 0 | | √ |
| number | 数量 | double | 20 | 2 | | √ |
| price | 单价 | double | 20 | 2 | | √ |
| total | 总价 | double | 20 | 2 | | |
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |

| | | | | | | |
|----------------|-------|---------|----|---|--|--|
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |
|----------------|-------|---------|----|---|--|--|

部分字段说明：

入库单明细编号（id）：入库单明细基本信息在数据库中唯一标识；

入库单编号（instore_id）：关联入库单表的外键，非空，多对一关系；

4.2.2.14 ord 订单表

主要用于记录订单。

表 4-14 订单表(ord)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|-----------------------|--------|---------|-----|-----|----|------|
| id | 订单编号 | bigint | 20 | 0 | √ | √ |
| cust_id | 客户编号 | bigint | 20 | 0 | | |
| driver_id | 配送司机编号 | bigint | 20 | 0 | | |
| tel | 电话 | varchar | 255 | 0 | | |
| address | 配送地址 | varchar | 255 | 0 | | |
| msg | 备注 | varchar | 255 | 0 | | |
| state | 状态 | varchar | 255 | 0 | | |
| totalprice | 总价格 | double | 20 | 2 | | |
| totalweight | 总重量 | double | 20 | 2 | | |
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |

部分字段说明：

订单编号（id）：订单信息在数据库中唯一标识；

客户编号（cust_id）：关联用户表的外键，非空，一对一关系；

配送司机编号（driver_id）：关联系统用户表的外键，非空。

4.2.2.15 orderitem 订单明细表

主要用于记录订单的明细。

表 4-15 订单明细表(orderitem)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|----------|--------|---------|-----|-----|----|------|
| id | 订单明细编号 | bigint | 20 | 0 | √ | √ |
| order_id | 订单编号 | bigint | 20 | 0 | | |
| veges_id | 鲜蔬编号 | bigint | 20 | 0 | | |
| name | 鲜蔬名称 | varchar | 255 | 0 | | |
| price | 售价 | varchar | 255 | 0 | | |
| number | 数量 | varchar | 255 | 0 | | |
| total | 总售价 | double | 20 | 2 | | |

| | | | | | | |
|-----------------------|--------|---------|----|---|--|--|
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |

部分字段说明：

订单明细编号（id）：订单明细信息在数据库中唯一标识；

订单编号（order_id）：关联订单表的外键，非空，多对一关系。

4.2.2.16 outstore 出库单表

主要用于记录出库单的概要信息。

表 4-16 出库单表(outstore)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|-----------------------|---------|---------|-----|-----|----|------|
| id | 出库单编号 | bigint | 20 | 0 | √ | √ |
| order_id | 订单编号 | bigint | 20 | 0 | | √ |
| driver_id | 配送司机编号 | bigint | 20 | 0 | | √ |
| store_id | 仓库编号 | bigint | 20 | 0 | | √ |
| keeper_id | 库存管理员编号 | bigint | 20 | 0 | | √ |
| totalprice | 总价 | double | 20 | 2 | | |
| totalweight | 总重 | double | 20 | 2 | | |
| state | 状态 | varchar | 255 | 0 | | |
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |

部分字段说明：

出库单编号（id）：出库单信息在数据库中唯一标识；

订单编号（order_id）：关联订单表的外键，非空，一对一关系。

4.2.2.17 outstoreitem 出库单明细表

主要用于记录出库单明细的信息。

表 4-17 出库单明细表(outstoreitem)

| 字段名 | 字段说明 | 数据类型 | 长度 | 小数点 | 主键 | 不可为空 |
|-------------|---------|--------|----|-----|----|------|
| id | 出库单明细编号 | bigint | 20 | 0 | √ | √ |
| outstore_id | 出库单编号 | bigint | 20 | 0 | | |

| | | | | | | |
|-----------------------|--------|---------|-----|---|--|--|
| veges_id | 鲜蔬编号 | bigint | 20 | 0 | | |
| name | 鲜蔬名称 | varchar | 255 | 0 | | |
| price | 售价 | double | 20 | 2 | | |
| number | 数量 | double | 20 | 2 | | |
| total | 总价 | double | 20 | 2 | | |
| object_version_number | 版本 | bigint | 20 | 0 | | |
| creation_date | 创建日期 | date | 0 | 0 | | |
| created_by | 创建者 | varchar | 20 | 0 | | |
| last_update_date | 最后更新日期 | date | 0 | 0 | | |
| last_update_by | 最后更新者 | varchar | 20 | 0 | | |

部分字段说明：

出库单明细编号（id）：出库单明细在数据库中唯一标识；

出库单编号（order_id）：关联出库单表的外键，非空，多对一关系。

4.3 系统模块详细设计

经过对系统进行初步的分功能模块，现针对各模块功能需求进行设计。

4.3.1 登录注册模块

（1）用户登录

用户登陆时输入账号信息，并在前台做校验，检验是否输入正确格式，接着提交后台验证账号是否正确，若正确，则根据用户角色进入不同角色页面；若不正确，则提示用户账号错误，需重新输入。

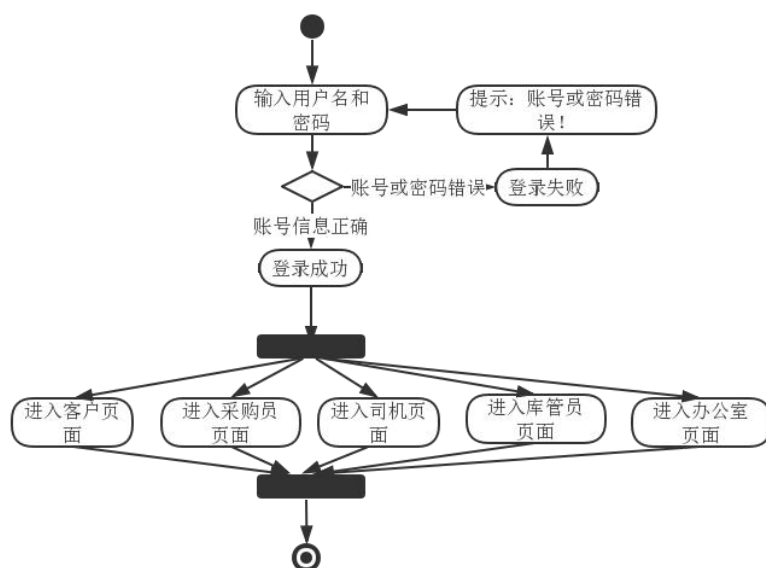


图 4-3 用户登录活动图

(1) 用户注册

只有用户角色中的“客户”使用此模块功能。且系统中各个用户的用户名唯一。

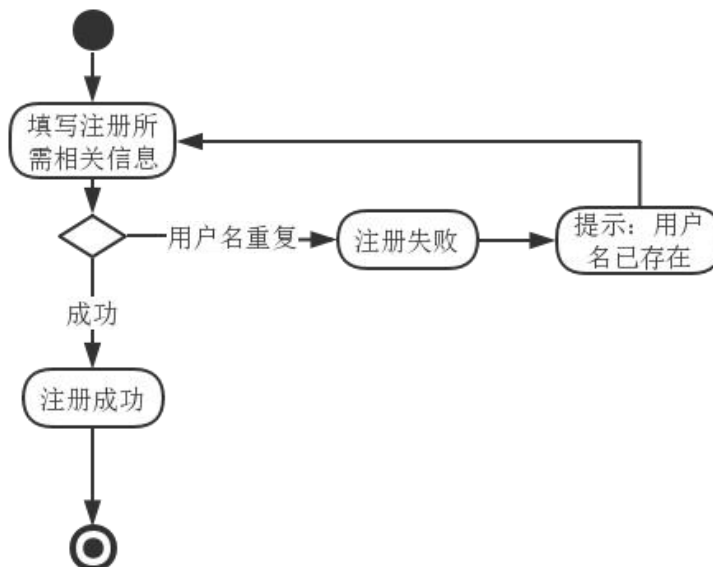


图 4-4 用户注册活动图

4.3.2 鲜蔬信息管理模块

该模块的用例图如下图所示，清晰简洁的展示了办公室角色在本模块下对鲜蔬的一系列操作。

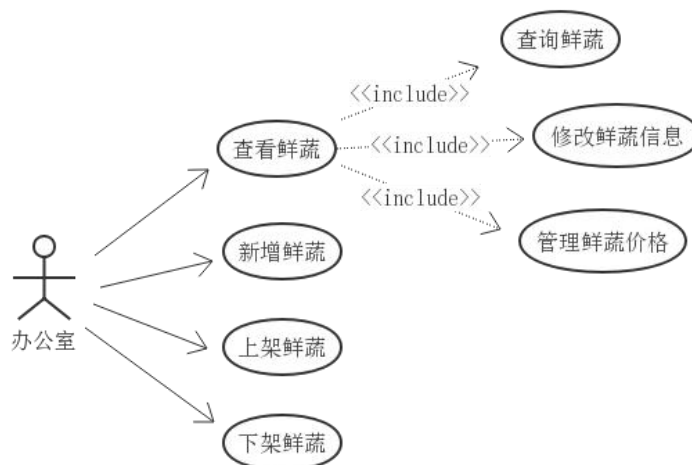


图 4-5 鲜蔬信息管理用例图

4.3.3 供应商管理模块

(1) 新增供应商：由系统办公室角色填写供应商的名称、地址、电话等基本信息，并填写采购员编号，采购员是与此供应商进行洽谈合作的员工。

(2) 管理供应商鲜蔬信息：展示供应商信息页面里，每一个供应商对应其

详情信息，点击查看详情，详情页面展示该供应商所提供鲜蔬种类。用户可以选择删除供应商下某种鲜蔬，也可为供应商添加新鲜蔬。

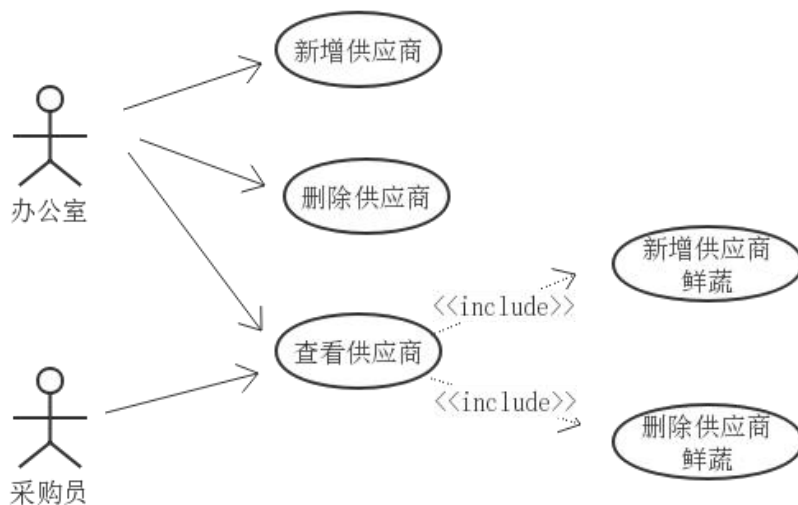


图 4-6 供应商管理用例图

4.3.4 员工信息管理模块

主要是用户对个人信息的修改查看及办公室对所有员工信息的查看和增加。

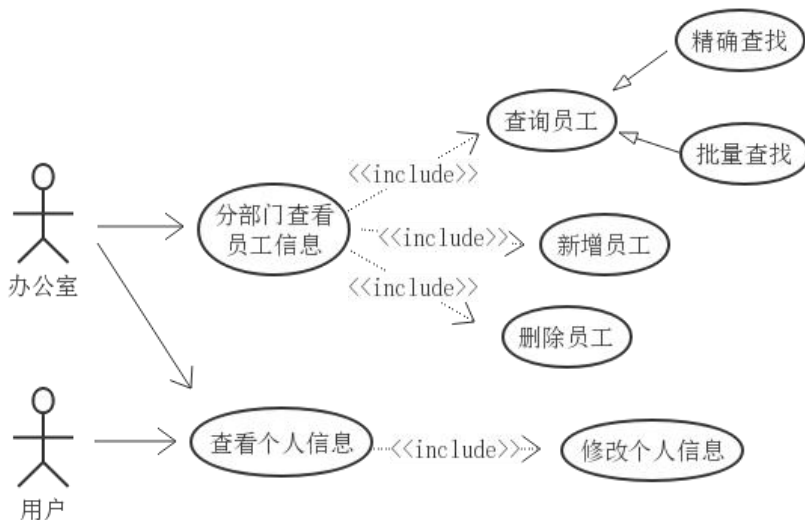


图 4-7 员工信息管理用例图

(1) 个人信息管理：

系统用户均可查看个人相关信息，包括但不限于姓名，电话，电子邮件，年龄等。并且可以修改这些基本信息，但不可修改权限以外的信息，如：部门，用户名，入职时间等。

(2) 部门员工信息管理：

当公司新增加合作客户或者新招收工作人员时，办公室角色可以根据部门向系统数据库添加用户。在不同部门页面按部门添加员工，为他们指定用户名密码等信息，系统根据部门页面上自动为新添加用户绑定部门角色等信息。

公司可按照部门查看当前部门内员工信息，并进行删除员工操作，删除后的员工将失去登录系统的账户信息，离职员工原账户信息保留，但不可用。

公司还可根据员工信息属性列表批量查询符合该条件的所有员工，也可精准查找某员工。

4.3.5 进货及配送管理模块

如下图所示，是一个完整的进货及其配送的流程，涉及三个角色。

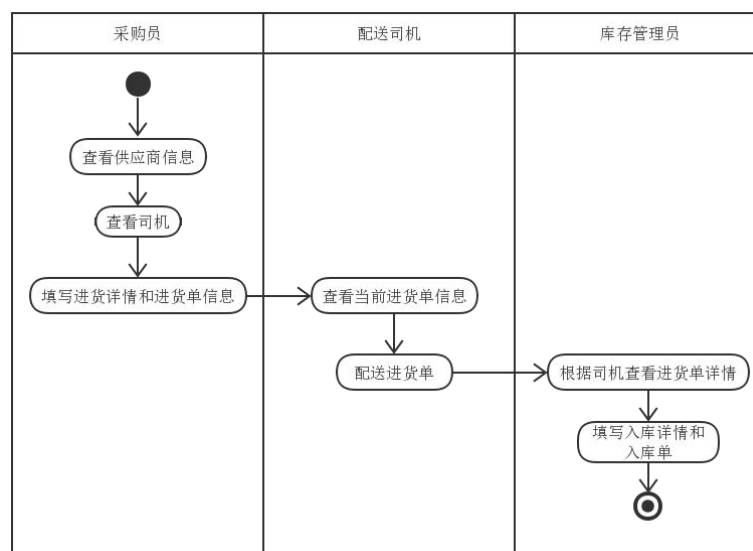


图 4-8 进货及配送管理活动图

(1) 进货及配送：首先是采购员被上级派往采购鲜蔬，采购员根据所在供应商精准查找到该供应商，将供应商相关信息填写在进货单内；接着采购员查看当前各司机的状态，安排空闲状态的司机进行进货配送；司机查看当前被安排配送的进货单信息，根据进货单信息进行指定仓库的配送；货物送达指定仓库后，由库存管理员负责将该批鲜蔬入库，填写入库单详情，系统自动更改该仓库库存信息。

(2) 进货单相关：采购员可对进货单进行新增、查找和查看详情操作；司机可对自己负责配送过的进货单进行查询操作，包括精确查找和批量查找。

4.3.6 订单及配送管理模块

如下图所示，是一个完整的订单及其配送的流程，涉及三个角色。

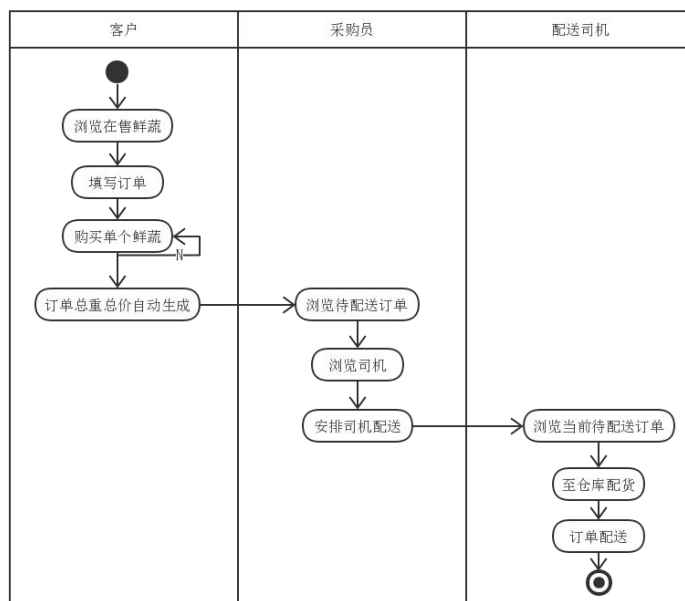


图 4-9 订单及配送管理活动图

(1) 订单及配送：客户登录系统在线浏览公司正在销售的鲜蔬，根据本身需要购买对应鲜蔬生成订单；客户新生成的订单推送给采购员端，采购员查看目前司机状态选择司机进行订单配送；司机端查看自己被安排的待配送订单，至仓库配货后进行订单配送。

(2) 订单信息相关：客户端查看自己的历史订单，包括根据各信息精准查找和批量查找订单，对当前订单进行收货操作；办公室端查看和查询所有订单信息及其详情；司机端查看和查询个人负责配送的历史订单，并可进行司机端的确认送到操作。

4.3.7 库存管理模块

(1) 入库模块：

司机携带鲜蔬配送至，库存管理员根据司机携带的进货单编号查询该进货单的进货明细，并根据明细依次检查每批鲜蔬实际重量与新鲜度，并根据实际送达的鲜蔬明细情况填写入库单及入库明细详情。

库存管理员可查看本仓库的入库单及其详情，并根据入库信息进行批量或精准查询。

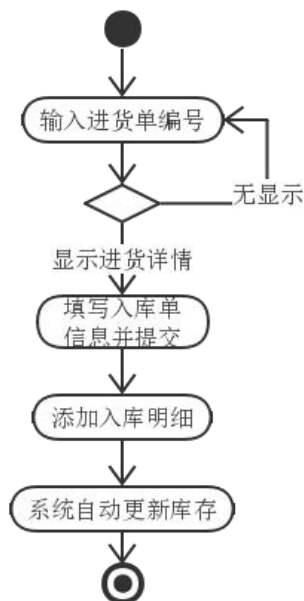


图 4-10 入库活动图

(3) 出库模块:

司机携带所打印的订单至仓库配货, 库存管理员根据司机携带的订单编号查询订单明细后, 根据个人经验判断为该订单配货并填写出库单和明细; 司机将库存管理员配好的鲜蔬运送至客户处。

库存管理员可查看所属本仓库的出库单信息及其明细, 并进行根据信息批量查询等操作。

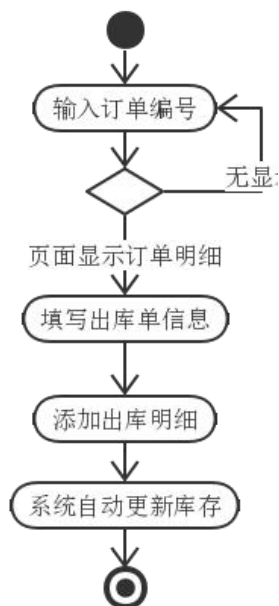


图 4-11 出库活动图

4.3.8 仓库管理模块

本模块大致功能如下用例图所示：

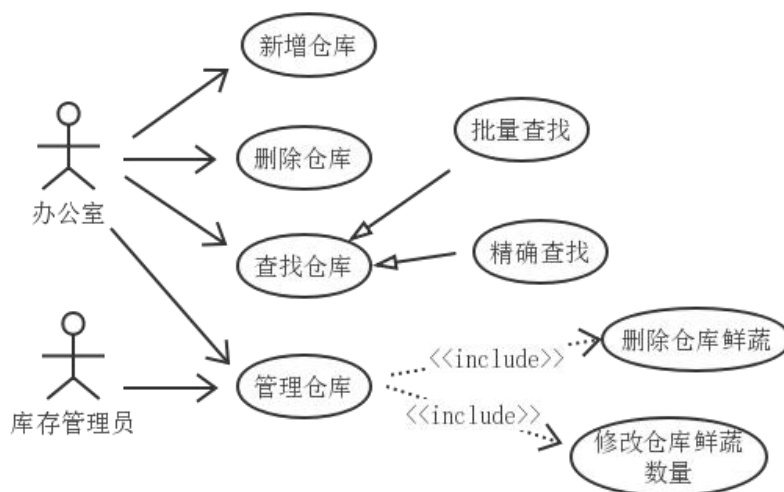


图 4-12 仓库管理用例图

(1) 库存管理员相关：库管员通过对本仓库的实时监测和统计，及时更新系统数据库中本仓库鲜蔬的数量情况，或删除本仓库某种鲜蔬产品。

(2) 办公室相关：当公司为满足生意需要新开仓库时，办公室通过本系统新增一个仓库，并为仓库安排工作人员，动态地记录该仓库的数据变化；当某个仓库不再满足公司需要，办公室可将该仓库从系统删除，但并不删除数据，其状态变化为“废弃”，表明该仓库已不再投入使用。

4.3.9 权限控制模块

本模块根据需求分析需要将系统用户分为不同角色，在对系统进行操作时，保证数据一致性，可靠性。

4.4 本章小结

本章主要介绍了鲜蔬配送管理系统开发相关设计，其中包括数据库设计、概要设计和各功能模块的详细设计。从不同设计层面通过 UML 建模（用例图、活动图），流程图等展示本系统的开发。

5 系统实现与测试

本章主要是阐述本系统各模块及数据库具体实现方法,包括实际运行界面效果展示和关键代码展示。

5.1 系统数据库的实现

5.1.1 建立数据库

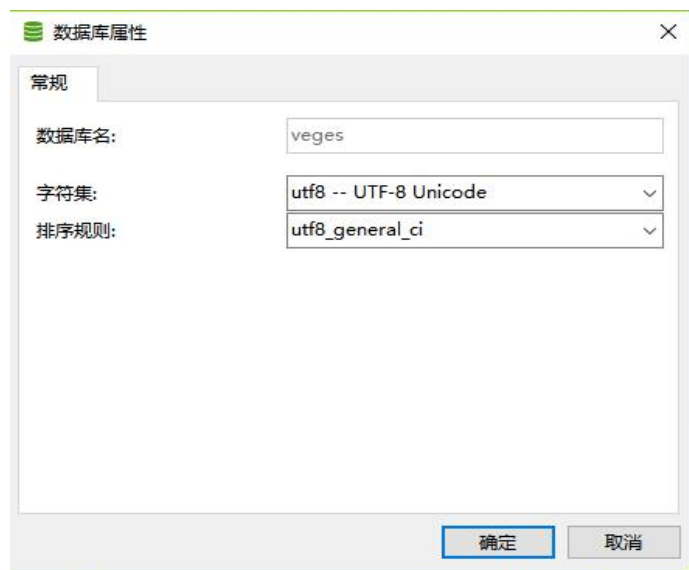


图 5-1 新建数据库

5.1.2 连接数据库

本系统开发采用 Mybatis, 其独有的特性可以实现系统多种功能。在 pom.xml 文件里依赖注入, 注入代码如下所示:

```
<dependency>
  <groupId>org.mybatis.spring.boot</groupId>
  <artifactId>mybatis-spring-boot-starter</artifactId>
  <version>1.3.1</version>
</dependency>
```

```
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <scope>runtime</scope>
</dependency>
```

实现系统与该数据库的链接在 application.yml 实现，代码如下：

```
# spring
spring:
  datasource:
    driver-class-name: com.mysql.jdbc.Driver
    url: jdbc:mysql://localhost:3306/veges
    username: root
    password: 111111
    dbcp2:
      initial-size: 1
      min-idle: 1
      max-total: 20
  http:
    multipart:
      max-file-size: 20MB
      max-request-size: 20MB
  jpa:
    hibernate:
      ddl-auto: update
      show-sql: true
  jackson:
    serialization: true

# mybatis
mybatis:
  type-aliases-package: dhu.sun.vege.entity
  mapper-locations: classpath:mapper/*.xml
```

5.2 系统功能模块的实现

5.2.1 用户登陆注册模块

(1) 用户登录

设计两个文本框分别接受用户输入的用户名和密码，点“登录”按钮，触发 js 方法，前台获取该两个文本框的内容，通过 ajax 传到后端对应接口，并在后端将传输过来的参数进行数据库校验是否存在且正确。



图 5-2 登录界面

当用户输入的信息正确时，根据用户输入账号的角色权限实现不同主页面跳转，具体实现 Ajax 代码如下：

```
success: function (data) {  
    //控制权限  
    if (data.token != null) {  
        localStorage.setItem("token", data.token);  
        localStorage.setItem("baseinfo", JSON.stringify(data));  
    }
```

```
if(data.role.role=="customer")  
    $(window).attr('location', 'customer/index.html');  
if(data.role.role=="driver")  
    $(window).attr('location', 'Driver/index.html');  
if(data.role.role=="buyer")  
    $(window).attr('location', 'index.html');  
if(data.role.role=="keeper")  
    $(window).attr('location', 'Keeper/index.html');  
if(data.role.role=="office")  
    $(window).attr('location', 'office/index.html');}
```

跳转至主界面，以办公室为例：

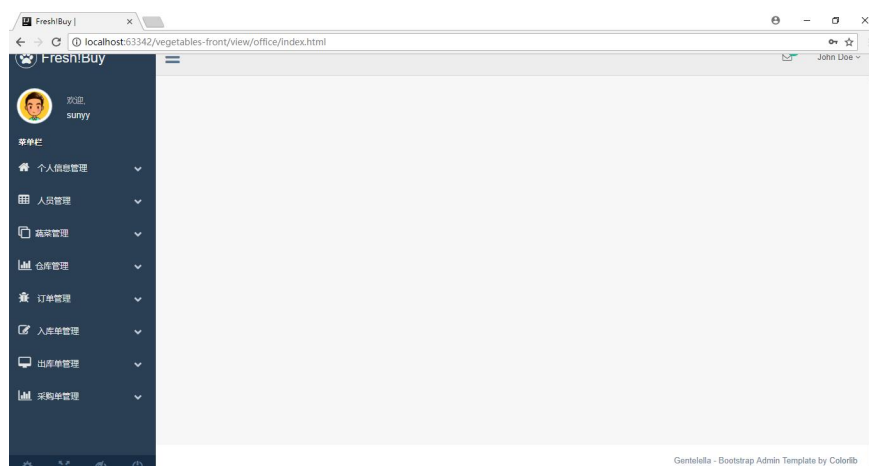


图 5-3 主页面

(1) 用户注册

三个必填项：用户名、邮件及密码，点击“提交”触发 js 方法将所填传到后台进行注册，后台代码对传过来的实体进行数据库校验，是否存在用户名重复，如不存在则成功注册并跳转到登录页面，否则提示用户“用户名已存在”。



图 5-4 注册页面

后台对前台传输过来的实体对象进行操作，首先检验是否用户名重复，继而将用户输入的密码进行加密后添加到数据库的操作，

```
String username = user.getUsername();
//用户名不允许重复
if (userMapper.selectUserByUsername(username) != null) {
    return null;
}
//对密码进行强哈希编码
BCryptPasswordEncoder encoder = new BCryptPasswordEncoder();
user.setPassword(encoder.encode(user.getPassword()));
```

5.2.2 鲜蔬信息管理模块

(1) 鲜蔬种类管理

a. 查看种类：点击右侧导航栏下种类管理，跳转到 `type.html`，通过 Datatable 控件以表格的形式显示数据库中所有的种类信息。



图 5-5 种类信息页面

b. 添加种类：点击左下角的添加，弹出 modal 框，输入新添加的种类名称，点击“添加”按钮，触发前台方法。



图 5-6 添加鲜蔬种类

c. 查询种类：前台控件实现根据表头元素精确或批量查找。



图 5-7 查询种类

d. 附加功能：可将任何表格数据以 word、excel 或 pdf 格式下载、打印。

(2) 鲜蔬管理

a. 查看鲜蔬：在 `veges.html` 显示数据库中所有鲜蔬详细信息。

| Copy | CSV | Print | Search: | | | | | |
|------|-----|-------|---------|----|----|------|---------------------|---|
| 编号 | 种类 | 名称 | 价格 | 描述 | 状态 | 创建时间 | 最后修改时间 | 操作 |
| 1 | 1 | 生菜 | 11 | | | | 2018-04-08 22:41:03 | 修改菜品 上架 下架 |
| 2 | 1 | 白菜 | 2 | | | | 2018-04-08 22:40:15 | 修改菜品 上架 下架 |
| 3 | 2 | 土豆 | 2 | | 下架 | | 2018-04-08 22:40:44 | 修改菜品 上架 下架 |
| 4 | 1 | 豆芽 | 3 | | | | 2018-04-08 22:40:48 | 修改菜品 上架 下架 |
| 5 | 1 | 海带 | 7 | | | | 2018-04-08 22:41:01 | 修改菜品 上架 下架 |
| 6 | 2 | 地瓜 | 3 | | | | 2018-04-08 22:40:54 | 修改菜品 上架 下架 |
| 7 | 1 | 娃娃菜 | 4 | | | | 2018-04-08 22:41:09 | 修改菜品 上架 下架 |

图 5-8 查看所有鲜蔬

b. 修改鲜蔬信息：在 `veges.html` 鲜蔬表格里某一项最后一栏点击修改菜品，跳转到 `changeveges.html` 内，修改鲜蔬信息，菜品编号与状态为不可修改内容。

修改蔬菜信息

[基本信息](#)
[修改](#)

菜品编号

1

状态

菜品名称

生菜

种类

a

价格

11

描述

取消

重置

提交

图 5-9 修改鲜蔬

c. 添加鲜蔬：输入鲜蔬信息及其图片，提交到后台。后台将图片上传到服务器上并将图片服务器地址写入鲜蔬数据库属性“pth”。

```
File file = new File(AppConst.basePath + newFileName);
File parent = file.getParentFile();
if (!parent.exists())
    parent.mkdirs();
try {
    upload.transferTo(file.getAbsolutePath());
}
```



添加菜品

请输入菜品

菜品名称:

种类*:

价格*:

描述*:

上传菜品图片

上传图片: 未选择任何文件

图 5-10 添加鲜蔬

d. 上架（下架）鲜蔬：点击表格操作栏按钮上架（下架），触发事件把鲜蔬 id 传到后台，后台将该鲜蔬的状态改为上架（下架）。

5.2.3 供应商信息管理模块

a. 查看所有供应商信息：选择查看其详情跳转到 viewsuppveges.html



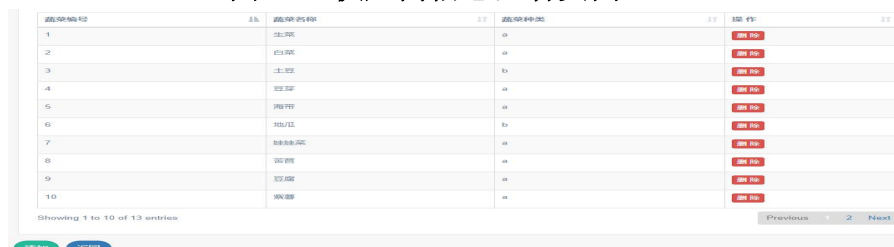
| 编号 | 头像 | 用户名 | 名称 | 电话 | 地址 | 采购员 | 状态 | 操作 |
|----|----|-----------|-----------|----|-------------|-----|-----|---------------------------------------|
| 7 | | sy | sy | a | 11100 | 9 | 未合作 | 详情 删除 |
| 14 | | sunyunyun | sunyunyun | | 15026651761 | 2 | 合作中 | 详情 删除 |
| 15 | | sw | sw | aa | 1111 | 2 | 合作中 | 详情 删除 |

Showing 1 to 3 of 3 entries

[Previous](#) [Next](#)

[添加](#)

图 5-11 供应商信息查看页面



| 蔬菜编号 | 蔬菜名称 | 蔬菜种类 | 操作 |
|------|------|------|--------------------|
| 1 | 生菜 | 叶菜 | 删除 |
| 2 | 白菜 | 叶菜 | 删除 |
| 3 | 土豆 | 根菜 | 删除 |
| 4 | 豆芽 | 豆类 | 删除 |
| 5 | 海带 | 藻类 | 删除 |
| 6 | 地瓜 | 根菜 | 删除 |
| 7 | 辣椒 | 茄果 | 删除 |
| 8 | 茄子 | 茄果 | 删除 |
| 9 | 豆角 | 豆类 | 删除 |
| 10 | 洋葱 | 根菜 | 删除 |

Showing 1 to 10 of 13 entries

[Previous](#) [Next](#)

[添加](#) [删除](#)

图 5-12 供应商鲜蔬详情

该页面可进行删除或添加某鲜蔬操作，点击添加按钮弹出 modal 框，但新添加鲜蔬不可重复，实现代码如下：

```
var bool=false;
for(var j=0;j<Veges.length;j++){
    if($("##vegename").val()==Veges[j].name) {
        vegesId=Veges[j].id; bool=true;}}

```

b. 添加新供应商：点击左下角“添加”按钮，填写新供应商信息，其中需要填写采购员编号，该文本框输入限定 number 类型。



添加供应商

请输入供应商信息

| | |
|-------------|------------------------------------|
| 供应商 用户名 | <input type="text" value="用户名"/> |
| 供应商 名称 * | <input type="text" value="名称"/> |
| 电话 * | <input type="text" value="电话"/> |
| Email * | <input type="text" value="Email"/> |
| 地址 * | <input type="text" value="地址"/> |
| 采购员 * | <input type="text" value="采购员编号"/> |

图 5-13 添加供应商

5.2.4 员工信息管理模块

(1) 个人信息管理

a. 查看修改个人信息：右侧导航栏个人信息管理下查看 `personalinfo.html`，其中只读内容不可修改。



修改个人信息

修改信息

| | |
|---|--|
| <input type="text" value="exo"/> | <input type="text" value="男"/> |
| <input type="text" value="389015586@qq.com"/> | <input type="text" value="12345"/> |
| 年龄 | <input type="text" value="29"/> |
| 员工编号 | <input type="text" value="19"/> |
| 用户名 | <input type="text" value="exo"/> |
| 密码 | <input type="password" value=""/> |
| 部门 | <input type="text" value="Read-Only Input"/> |
| 岗位 | <input type="text" value="buyer"/> |

图 5-14 个人信息页面

(2) 员工信息管理

a. 查看部门所有员工：以采购部门采购员为例，右侧导航栏“员工管理”下各部门“采购部门”查看。可对采购员进行删除操作，删除该员工后，修改该员工账号密码为“out”且其状态为“离职”。实现代码如下：

b. 其中根据不同角色删除时，根据角色 `role` 设置其状态分别问：“黑名单”、

“未合作”及“离职”。

```
User u=userMapper.selectByPrimaryKey(id);
if(u.getRoleId()==1){
    u.setState("黑名单");
    u.setPassword("out");
    u.setLastUpdateDate(new Date());
    userMapper.updateByPrimaryKey(u); }
else if(u.getRoleId()==2){
    u.setState("未合作");
    u.setLastUpdateDate(new Date());
    userMapper.updateByPrimaryKey(u); }
else{
    u.setState("离职");
    u.setPassword("out");
    u.setLastUpdateDate(new Date());
    userMapper.updateByPrimaryKey(u); }
```

采购员信息 个人

Copy CSV Print Search:

| 编号 | 头像 | 用户名 | 姓名 | 性别 | 年龄 | 电话 | Email | 入职时间 | 状态 | 操作 |
|----|----|------|-------|----|----|--------|------------------|---------------------|----|--------------------|
| 2 | | user | Tony | | 0 | 1234 | | | 离职 | 删除 |
| 3 | | test | Scott | | 0 | 123455 | | | y | 删除 |
| 9 | | a | a | 女 | 12 | 10086 | 389015568@qq.com | | 空闲 | 删除 |
| 19 | | exo | exo | 男 | 29 | 12345 | 389015586@qq.com | 2018-04-08 11:00:20 | 空闲 | 删除 |

Showing 1 to 4 of 4 entries Previous 1 Next

[添加](#)

图 5-15 部门员工信息

c. 添加员工：不同部门页面添加员工根据所在页面前台赋值其 role 属性。

添加采购员

请输入采购员信息

用户名

密码*

姓名*

电话*

Email*

性别*

年龄*

[添加](#) [取消](#)

图 5-16 添加员工

d. 查询员工：按属性查找。

5.2.5 进货及配送管理模块

(1) 进货及配送

a. 采购员进货：填写进货单必填信息后，点击提交按钮，此时后台接收到该进货单实体对象信息，并将其状态设置为“进货未完成”，接着采购员店家右上角“添加”按钮，添加进货明细，此时明细里只读文本框—进货单编号自动生成，采购员从蔬菜文本框下拉选项填写鲜蔬品种，并自动校验该鲜蔬是否正确输入，每添加一项明细即显示在右侧小表格内，方便采购员实时监测是否正确进货。当所有明细填写完毕后，点击右侧“添加完成”按钮，触发事件，后台将该进货单状态改为“进货已完成”，并推送给司机端。

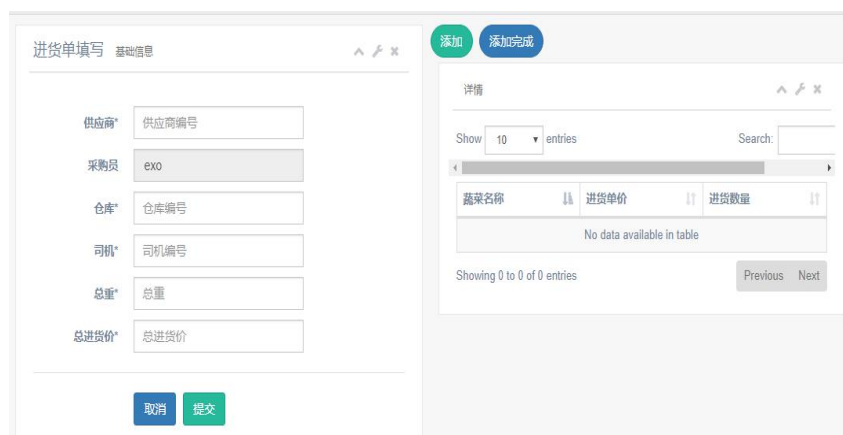


图 5-17 进货页面

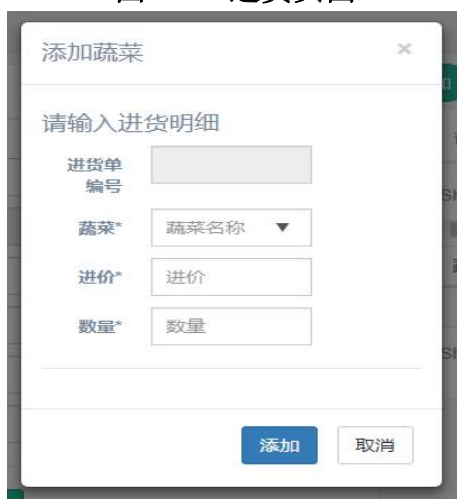


图 5-18 进货单明细填写

b. 司机配送进货鲜蔬：司机端接收到当前进货单信息，点击查看 `currentImpo.html` 页面，以配送单的形式显示给司机，司机点击右上角“确认配送”

按钮，触发事件，该进货单状态修改为“配送中”，司机可将该配送单打印，方便数据交接。



当前进货单

确认配送 配送完成

进货单明细

时间: 2018-04-17 14:20:17

进货单

供应商: 谭杰
a
13818525341
1269003816@qq.com

仓库: a
aaa
15026631681

进货单编号: 11

采购员: a

进货单状态: 进货已完成

Show: 10 entries

Search:

| 编号 | 蔬菜名称 | 单价 | 重量 | 总价 |
|----|------|----|----|----|
| 14 | 白菜 | 1 | 11 | 11 |

Showing 1 to 1 of 1 entries

Previous Next

进货单:

配送: d1

总重: 11

总价: 122

图 5-19 进货配送单

(3) 进货单

- a. 查看历史进货单: 查看所有的历史进货单及其详情，并可进行属性查询。

Copy CSV Print

Search:

| 编号 | 供应商 | 仓库 | 采购员 | 司机 | 总进货价 | 总重 | 创建日期 | 最后修改日期 | 状态 | 操作 |
|----|-----------|----|-----|--------|------|--------|---------------------|---------------------|-----------|----------------------|
| 1 | syy | a | a | ab | 200 | 100.02 | | 2018-04-07 13:20:06 | 配送完成, 已入库 | 查看详情 |
| 2 | syy | a | a | s | 100 | 220 | 2018-03-30 18:24:30 | | ok | 查看详情 |
| 3 | syy | a | a | Tony | 21 | 222 | 2018-03-31 14:04:23 | | dd | 查看详情 |
| 4 | syy | a | a | 谭杰 | 22 | 11 | 2018-03-31 14:06:49 | | 进货未完成 | 查看详情 |
| 5 | syy | a | a | 谭杰 | 123 | 1234 | 2018-03-31 14:16:16 | 2018-04-07 13:58:18 | 配送完成, 已入库 | 查看详情 |
| 6 | syy | a | ab | a | 11 | 12 | 2018-03-31 14:17:18 | 2018-04-07 13:59:39 | 配送完成, 已入库 | 查看详情 |
| 7 | syy | a | a | a | 125 | 1234 | 2018-04-01 11:29:26 | | 进货未完成 | 查看详情 |
| 8 | syy | a | a | a | 145 | 123 | 2018-04-01 11:32:27 | | 进货未完成 | 查看详情 |
| 9 | syy | a | a | sunyun | 111 | 12 | 2018-04-02 16:36:07 | | 进货已完成 | 查看详情 |
| 10 | sunyunyun | a | a | sunyun | 123 | 123 | 2018-04-17 | | 进货已完成 | 查看详情 |

图 5-20 历史进货单

5.2.6 订单及配送管理模块

(1) 订单及配送: 客户通过浏览公司正在销售的鲜蔬选购菜品，去点击下单页面，填写订单电话地址等基本信息，然后点击右上角的“添加”按钮，添加订单明细，其中明细中鲜蔬种类下拉框选择并自动显示当前售价等信息，客户可在当前页面查看该鲜蔬售价核对信息，不必跳转。添加完后，点击右上角添加完成更改该订单状态为“待配送”。采购员端接收到客户新生成的订单，查看当前待配送订单信息，并根据实际情况为每一笔订单安排司机进行配送。司机端接收到当

前待配送订单，点击待配送订单右侧“详情”按钮，将订单明细以配送单的形式打印出，携带订单至仓库配货。

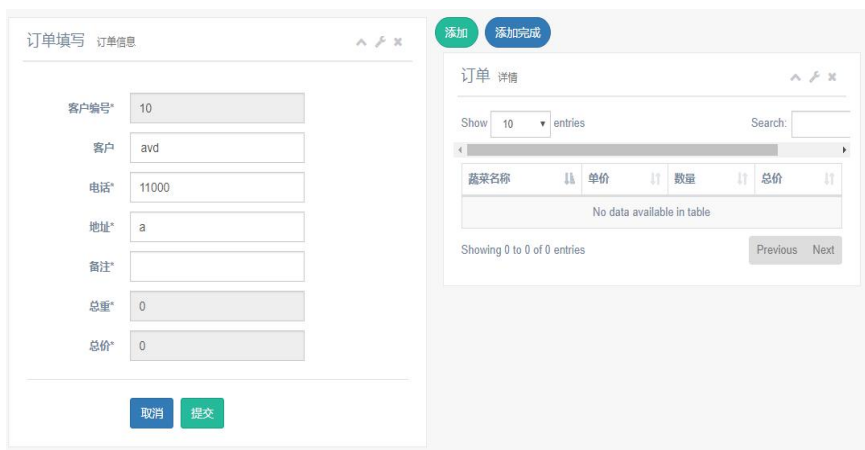


图 5-21 订单生成页面



图 5-22 添加订单明细

当前订单 客户

| 编号 | 客户 | 电话 | 地址 | 总进价 | 总重 | 备注 | 创建日期 | 最后更新日期 | 状态 | 操作 |
|----|-----|----|-----|-----|------|----|------|---------------------|-----|--------------------|
| 2 | avd | 11 | 上海市 | 11 | 1234 | | | 2018-04-20 21:22:21 | 待配送 | 详情 |

Showing 1 to 1 of 1 entries

图 5-23 司机查看当前订单

出货订单.

客户: avd
上海市
11

订单编号: 2

订单日期:

Show: 10 entries

Search:

| 编号 | 蔬菜名称 | 单价 | 重量 | 总价 |
|----|------|----|----|----|
| 1 | 豆腐 | 1 | 2 | 11 |

Showing 1 to 1 of 1 entries

Previous Next

订单:

配送: d1

总重: 11

总价: 1234

打印

图 5-24 司机查看当前订单明细

为订单安排司机

请输入订单和司机编号

订单编号

司机编号*

司机姓名*

性别*

确定 取消

图 5-25 安排订单配送司机

订单明细下拉框自动显示鲜蔬售价具体代码实现如下:

```
function check(vegesname) {
    var veges=JSON.parse(localStorage.getItem("veges"));
    for(var i=0;i<veges.length;i++){
        if(vegesname==veges[i].veges.name)
        {
            $("#price").val(veges[i].veges.price);
        }
    }
}
```

安排司机下拉框选定司机自动显示司机信息同理。

(2) 订单信息

a. 查看历史订单: 办公室查看所有订单并进行详情查看; 客户查看个人历史订单及详情; 司机查看配送历史订单详情。

历史订单 客户

| 编号 | 客户 | 司机 | 电话 | 地址 | 总进货价 | 总重 | 备注 | 创建日期 | 最后修改日期 | 状态 | 操作 |
|----|-----|----|----|-----|------|------|----|------|---------------------|--------|----------------------|
| 1 | 谭杰 | d1 | | | 1232 | 111 | | | 2018-04-18 14:40:40 | 司机确认送达 | 查看详情 |
| 2 | avd | d1 | 11 | 上海市 | 11 | 1234 | | | 2018-04-20 21:22:21 | 待配送 | 查看详情 |

Showing 1 to 2 of 2 entries

Previous 1 Next

图 5-26 历史订单页面

b. 订单确认收货：司机配送订单至客户处，司机端可进行确认送达操作，订单状态为“司机确认送达”；客户端可对订单进行查收，确认收货后订单状态为“客户确认收货”，以此区分保证配送链数据完整性。

5.2.7 库存管理模块

(1) 入库：库存管理员在页面右上角输入司机配送的进货单编号，下方表格动态显示该进货单明细，库存管理员对照该进货单明细核对入库信息，并填写入库单，入库单基本信息对应进货单基本信息，其中入库单总重由库存管理员根据实际总重手动填写，点击“添加”按钮，生成入库单，此时，入库单状态“入库未完成”，点击右上角“添加”，弹出 modal 框添加入库明细，其中入库明细鲜蔬文本框下拉框选择，自动显示该明细重量，进货价等信息。添加完毕后，点击右侧“添加完成”按钮，完成入库，入库单状态为“入库完成”。

进货单

请输入司机的进货单编号 搜索

| 编号 | 进货单编号 | 蔬菜名称 | 进价 | 数量 | 总进价 |
|----|-------|------|----|----|-----|
| 14 | 11 | 白菜 | 1 | 11 | 11 |

Showing 1 to 1 of 1 entries

Previous 1 Next

生成入库单

图 5-27 入库单生成页面

动态显示代码如下：

```
function check(id) {
    var importItem=JSON.parse(localStorage.getItem("importitem"));
    for(var i=0;i<importItem.length;i++){
        if(id==importItem[i].id)
        {
            $("#vegesname").val(importItem[i].name);
            $("#price").val(importItem[i].price);
            $("#number").val(importItem[i].number); } } }
```



图 5-28 添加入库明细

(2) 出库：库存管理员在右上角输入司机携带的订单编号，下方表格显示该订单明细信息，库存管理员根据该订单明细生成出库单，点击“生成出库单”，跳转到 `makeoutstore.html`，填写出库单信息并添加出库明细，点击“提交”，生成出库单且状态为“出库未完成”，点击“添加”添加明细信息，最后点击“添加完成”成功生成出库单。



图 5-29 查看当前待配货订单

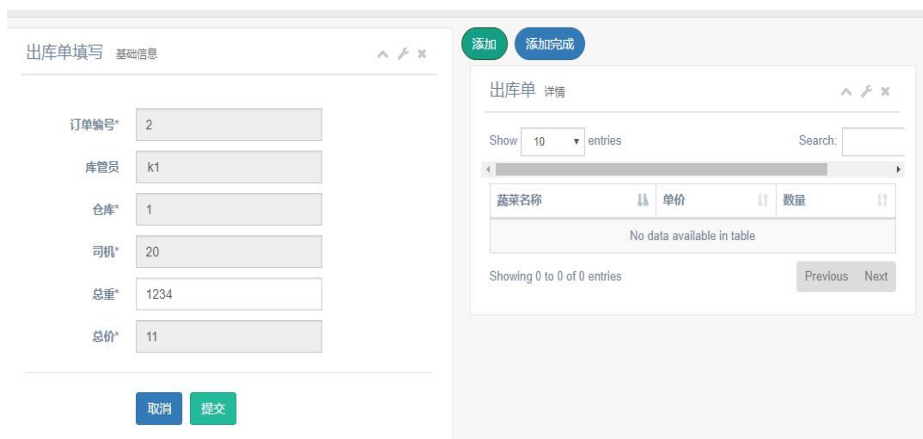


图 5-30 生成出库单

添加出库单明细时，下拉选择框显示该订单明细，代码实现如下：

```
function showitem() {
    var token = localStorage.getItem("token");
    var baseInfo = JSON.parse(localStorage.getItem("baseinfo"));
    var orderItem = JSON.parse(localStorage.getItem("orderitem"));
    for(var i=0;i<orderItem.length;i++){
        $("#items").append("<option>" + orderItem[i].id + "</option>");
    }
}
```

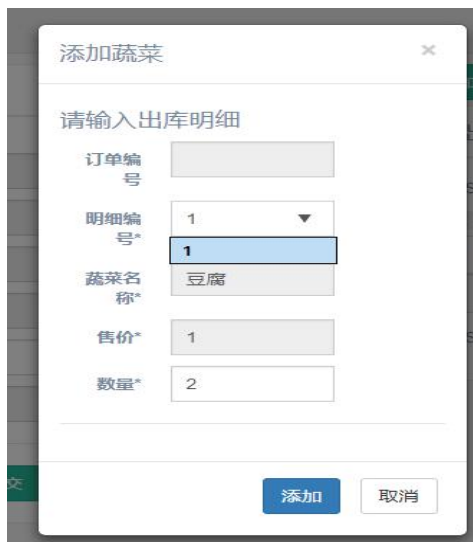


图 5-31 添加出库明细


(3) 对本仓库鲜蔬信息的管理：库存管理员实时监测本仓库内鲜蔬新鲜度等情况，根据腐烂程度及时更新仓库鲜蔬菜品剩余数量。当某鲜蔬数量改变时，在表格上方搜索框内输入该鲜蔬名称或编号查询到鲜蔬，点击右侧“更新”按钮，弹出 modal 框更新鲜蔬数量。若要删除某鲜蔬，在表格内输入鲜蔬名称或其他属性查找到该鲜蔬点击右侧“删除”按钮。

更新请搜索!

| Copy | CSV | Print | Search: |
|------|------|-------|---------------------|
| 编号 | 蔬菜名称 | 数量 | 最后修改日期 |
| 3 | 豆芽 | 110 | 2018-04-12 17:23:26 |
| 5 | 白菜 | 345 | 2018-04-07 13:58:18 |
| 12 | 豆腐 | 323 | 2018-04-07 15:33:29 |

Showing 1 to 3 of 3 entries
Previous 1 Next

图 5-32 仓库鲜蔬信息查看页面



更新蔬菜

蔬菜: 白菜

数量*: 345

更新 取消

图 5-33 更新仓库中鲜蔬

(4) 历史出（入）库单：库管员查看本仓库历史出（入）库单信息及其详细信息，并可做查询操作；办公室则可查看所有仓库历史出（入）库单信息，并根据仓库、负责库存管理员、配送司机等属性批量查询。

历史入库单 全部

Copy CSV Print Search:

| 编号 | 进货单编号 | 供应商 | 司机 | 采购员 | 库管员 | 总进货价 | 总重 | 创建日期 | 状态 | 操作 |
|----|-------|-----|----|-----|-----|------|-----|---------------------|-------|------|
| 1 | 1 | syy | ab | a | 谭杰 | 111 | 123 | | 入库完成 | 查看详情 |
| 3 | 5 | syy | a | a | a | 11 | 12 | 2018-04-06 19:52:49 | 入库完成 | 查看详情 |
| 4 | 6 | syy | a | ab | a | 11 | 12 | 2018-04-07 13:40:29 | 入库未完成 | 查看详情 |
| 5 | 6 | syy | a | ab | a | 11 | 12 | 2018-04-07 13:49:14 | 入库未完成 | 查看详情 |
| 6 | 6 | syy | a | ab | a | 11 | 12 | 2018-04-07 13:59:25 | 入库完成 | 查看详情 |
| 7 | 6 | syy | a | ab | d1 | 11 | 12 | 2018-04-17 14:24:50 | 出库未完成 | 查看详情 |

Showing 1 to 6 of 6 entries Previous 1 Next

图 5-34 历史入库单

历史出库单 全部

Copy CSV Print Search:

| 编号 | 订单编号 | 库管员 | 司机 | 总进货价 | 总重 | 创建日期 | 状态 | 操作 |
|----|------|------|----|------|------|---------------------|------|------|
| 1 | 1 | Tony | 谭杰 | 1 | 1 | | 入库完成 | 查看详情 |
| 2 | 2 | d1 | d1 | 11 | 1234 | 2018-04-18 16:53:08 | 入库完成 | 查看详情 |

Showing 1 to 2 of 2 entries Previous 1 Next

图 5-35 历史出库单

5.2.8 仓库管理模块

a. 办公室点击右侧导航栏仓库管理，跳转到 `storehouse.html`，显示当前数据库中所有的仓库及其信息，点击右侧“详情”按钮，跳转到 `storeveges.html` 显示该仓库目前的鲜蔬存储情况。点击右侧“删除”按钮，将仓库状态修改为“废弃”，代

表该仓库不在被投入使用。

仓库信息 单个

Copy CSV Print Search:

| 编号 | 名称 | 电话 | 地址 | 创建时间 | 状态 | 操作 |
|----|----|-------------|----------|---------------------|-----|---------------------------------------|
| 1 | a | 15026631681 | aaa | | 废弃 | 详情 删除 |
| 2 | s1 | 15026643578 | shanghai | 2018-04-08 13:58:55 | 废弃 | 详情 删除 |
| 3 | s2 | 10086 | beijing | 2018-04-08 13:59:33 | 使用中 | 详情 删除 |

Showing 1 to 3 of 3 entries Previous Next

添加

图 5-36 仓库信息页面

b. 添加仓库：点击坐下当“添加”按钮，弹出 modal 框，输入新增仓库信息。

添加仓库

请输入仓库信息

名称*

电话*

地址*

图 5-37 添加仓库

5.2.9 权限控制模块

在用户登录后根据用户的 role 属性将表 sys_role、sys_auth 通过表 sys_role_auth 关联，并将用户所拥有 auth 以数组形式存放在用户登陆后的 Token 里，前台操作时，根据 Token 里的 auth 通过 JWT 判断该用户是否拥有此接口的权限，如果有，则显示该功能，否则提示用户“权限不够”。

前台权限验证相关代码：

```
beforeSend: function (request) {
    request.setRequestHeader("Authorization", "Bearer " +
        localStorage.getItem("token"));
},
```

后台权限验证相关代码：

```
@PreAuthorize("hasAnyAuthority('office')")
```



```

@Component
public class JwtAuthenticationTokenFilter extends OncePerRequestFilter
{
    @Autowired
    private UserDetailsService userDetailsService;
    @Value("${jwt.header}")
    private String header;
    @Value("${jwt.tokenHead}")
    private String tokenHead;
    @Autowired
    private JwtTokenUtil jwtTokenUtil;
    @Override
    protected void doFilterInternal(HttpServletRequest request,
        HttpServletResponse response, FilterChain chain) throws
        ServletException, IOException {
        String username;
        String token = jwtTokenUtil.getTokenFromRequest(request);
        //如果 token 合法
        if ((username = jwtTokenUtil.getUsernameFromToken(token)) != null)
        {
            logger.info("Check token start, username [" + username + "]);
            //security 中没有认证
            if (SecurityContextHolder.getContext().getAuthentication() == null)
            {
                UserDetails userDetails =
                this.userDetailsService.loadUserByUsername(username);
                if (jwtTokenUtil.validateToken(token, userDetails)) {
                    UsernamePasswordAuthenticationToken authentication = new
                    UsernamePasswordAuthenticationToken( userDetails, null,
                    userDetails.getAuthorities());
                    authentication.setDetails(new
                    WebAuthenticationDetailsSource().buildDetails( request));
                    logger.info("Check token pass, username [" +
                    username + "], setting security context");
                    SecurityContextHolder.getContext().setAuthentication(authentication);
                    logger.info("Check token all pass!"); } } }
                chain.doFilter(request, response); } }
    
```

5.3 系统功能测试

5.3.1 对登录注册的测试

登录注册应具备的功能如下：

- (1) 登录账户信息的验证，并根据用户角色跳转不同页面。
- (2) 注册新账户用户名不可重复，且注册后账户角色为“客户”。

测试过程：

- (1) 测试名称：登陆注册
- (2) 测试前提：无。
- (3) 测试过程：使用不同角色账户进入系统 Fresh!Buy 登录/注册页面，填写账户信息。
- (4) 输出标准：系统成功登录进入“采购员”、“客户”、“库存管理员”、“配送司机”及“办公室”的主页面。数据库 sys_user 表中存在新注册的用户，且其角色为“客户”。
- (5) 测试结果：和输出标准相同，如图 5-38 所示。

| id | role_id | dept_id | username | password | name | gender | age | email |
|----|---------|---------|----------|---|-------|--------|--------|-------------------|
| 1 | 1 | 1 | root | root | 谭杰 | (Null) | (Null) | 1269003816@qq.com |
| 2 | 3 | 3 | user | user | Tony | (Null) | 0 | (Null) |
| 3 | 3 | 0 | test | test | Scott | (Null) | (Null) | (Null) |
| 4 | 0 | 0 | aaa | aaa | AAA | (Null) | (Null) | (Null) |
| 7 | 2 | 2 | syy | \$2a\$10\$0Qx0juvvSR0wjszgoJL6ae7xK4oZXr8Hv syy | | (Null) | 0 | 389015586@qq.com |
| 9 | 3 | 3 | a | \$2a\$10\$/2HGdXqkxLUegRAJxQaZz.g84XtfzBN: a | 女 | | 12 | 389015568@qq.com |
| 10 | 1 | 1 | 客户 | \$2a\$10\$UEq1OIDzricJiuxvqjORseAim04FH7Wro avd | 男 | | 0 | 389015586@qq.com |

图 5-38 登录注册测试

5.3.2 对进货过程的测试

进货过程应具备的功能如下：

- (1) 采购员能够生成新采购单。
 - (2) 配送司机端可接收到当前待配送采购单。
 - (3) 库存管理员端可根据采购单明细生成入库单，并由系统自动更新库存。
- 测试过程：
- (1) 测试名称：进货过程
 - (2) 测试前提：采购员，相关配送司机及库存管理员登录系统。
 - (3) 测试过程：以采购员身份登录系统，浏览合作供应商信息及配送司机目前状态，根据需要填写进货单及其明细；以指定司机账户登录系统，查看当前待配送进货单页面并打印；以指定仓库附属库存管理员账户登录系统，在当前进货单页面下输入新生成进货单编号，查看该进货单明细并生成入库单明细及基本信息。
 - (4) 输出标准：司机端显示该采购员新生成的进货单明细，库存管理员根据该进货单编号可查看到其明细，并在生成入库单页面显示该进货单明细信息，

且系统自动更新该仓库库存信息。

(5) 测试结果：与输出标准相同，如图 5-39 所示。

| id | buyer_id | supplier_id | driver_id | totalprice | totalweight | store_id | keeper_id | state | object_version_number | creation_date |
|----|----------|-------------|-----------|------------|-------------|----------|-----------|-------|-----------------------|---------------------|
| 1 | 9 | 7 | 12 | 200.00 | 100.02 | 1 | 1 | 配送完成 | (Null) | (Null) |
| 2 | 9 | 7 | 11 | 100.00 | 220.00 | 1 | (Null) | ok | (Null) | 2018-03-30 18:24:30 |
| 3 | 9 | 7 | 2 | 21.00 | 222.00 | 1 | (Null) | dd | (Null) | 2018-03-31 14:04:23 |
| 4 | 9 | 7 | 1 | 22.00 | 11.00 | 1 | (Null) | 进货未完 | (Null) | 2018-03-31 14:06:49 |
| 5 | 9 | 7 | 1 | 123.00 | 1234.00 | 1 | 9 | 配送完成 | (Null) | 2018-03-31 14:16:16 |
| 6 | 12 | 7 | 9 | 11.00 | 12.00 | 1 | 9 | 配送完成 | (Null) | 2018-03-31 14:17:18 |

(a)

| id | import_id | veges_id | name | price | number | total | object_version_number | creation_date |
|----|-----------|----------|------|-------|--------|-------|-----------------------|---------------------|
| 1 | 6 | 2 | 白菜 | 1.1 | 122 | 134.2 | (Null) | 2018-04-01 11:25:58 |
| 2 | 6 | 2 | 白菜 | 2 | 3 | 6 | (Null) | 2018-04-01 11:29:39 |
| 3 | 6 | 7 | 娃娃菜 | 2 | 23 | 46 | (Null) | 2018-04-01 11:29:47 |
| 4 | 6 | 4 | 豆芽 | 34 | 13 | 442 | (Null) | 2018-04-01 11:30:25 |

(b)

| id | import_id | store_id | keeper_id | driver_id | totalprice | totalweight | state | object_version_number | creation_date |
|----|-----------|----------|-----------|-----------|------------|-------------|-------|-----------------------|---------------------|
| 1 | 1 | 1 | 1 | 12 | 111 | 123 | 入库完成 | (Null) | (Null) |
| 3 | 5 | 1 | 9 | 9 | 11 | 12 | 入库完成 | (Null) | 2018-04-06 19:52:20 |
| 4 | 6 | 1 | 9 | 9 | 11 | 12 | 入库未完 | (Null) | 2018-04-07 13:40:11 |

(c)

图 5-39 进货过程测试

5.3.3 对出货过程的测试

出货过程应具备的功能如下：

- (1) 客户下单生成待配送订单
- (2) 采购员端查看当前新订单并安排司机配送
- (3) 司机端查看待配送订单进行配送
- (4) 库存管理员端根据新订单编号生成出库单，并由系统自动更新库存。

测试过程：

- (1) 测试名称：出货过程
- (2) 测试前提：客户、采购员、司机、库存管理员均登录系统。

(3) 测试过程：以客户账户登录系统，浏览在售鲜蔬信息，添加鲜蔬生成订单；以采购员身份登录系统，查看当前客户新生成待配送的订单，为每一笔订单安排司机进行配送；以司机身份登录系统，查看名下待配送订单并打印；以库存管理员身份登录系统，输入司机该订单编号查看订单明细并生成出库单。

(4) 输出标准：客户端显示在售鲜蔬并生成订单；采购员端显示待配送订单并安排司机；司机端显示待配送订单；采购员端输入订单编号显示订单明细，在生成出库单页面自动显示订单明细，并可修改明细重量；系统自动更新数据库中该仓库的库存信息。

(5) 测试结果：与输出标准相同，如图 5-40 所示：

| id | cust_id | driver_id | tel | address | msg | state | totalprice | totalweight | object_version_number | creation_date |
|----|---------|-----------|-----------|---------|--------|--------|------------|-------------|-----------------------|-------------------|
| 1 | 1 | 1 | 20 (Null) | (Null) | (Null) | 司机确认送达 | 1232 | 111 | (Null) | (Null) |
| 2 | 10 | 10 | 20 11 | 上海市 | (Null) | 待配送 | 11 | 1234 | (Null) | (Null) |
| 3 | 10 | 10 | 20 11000 | a | 要快点到 | 订单未完成 | 12 | 12 | (Null) | 2018-04-23 10:52: |

(a)

| id | order_id | veges_id | name | price | number | total | object_version_number | creation_date |
|----|----------|----------|------|-------|--------|-------|-----------------------|-------------------|
| 1 | 2 | 9 | 豆腐 | 1 | 2 | 11 | (Null) | (Null) |
| 2 | 3 | 1 | 生菜 | 11 | 1 | 11 | (Null) | 2018-04-23 10:53: |
| 3 | 3 | 4 | 豆芽 | 3 | 2 | 6 | (Null) | 2018-04-23 10:53: |

(b)

| id | driver_id | order_id | store_id | keeper_id | totalprice | totalweight | state | object_version_number | creation_date |
|----|-----------|----------|----------|-----------|------------|-------------|--------|-----------------------|-------------------|
| 1 | 1 | 1 | 1 | 1 | 2 | 1 | 1 入库完成 | (Null) | (Null) |
| 2 | 20 | 3 | 1 | 20 | 11 | 1234 | 1 入库完成 | (Null) | 2018-04-18 16:53: |

(c)

图 5-40 出货过程测试

5.3.4 对基础信息管理的测试

基础信息管理应具备的功能如下：

- (1) 对员工信息的基本管理，如增删查。
- (2) 对系统业务流程产生的数据，如进货单，入库单，出库单，订单等的相关查询。
- (3) 对仓库信息的基本维护。
- (4) 对供应商及其所供鲜蔬信息的维护。
- (5) 对鲜蔬信息的管理。

测试过程：

- (1) 测试名称：基础信息管理
- (2) 测试前提：登录系统，且数据库中含有相关数据。
- (3) 测试过程：使用办公室角色登录系统，查看并按部门查询当前员工信息，并分别为每个部门增加一个员工，删除一个员工；查看并查询进货单、入库出库单及订单的基本信息及详情；查看现有使用仓库基本信息及其库存明细，并进行管理；查看当前合作供应商并对其详情进行维护；查看当前鲜蔬，并对鲜蔬产品修改价格等信息。

(4) 输出标准：员工信息显示给办公室角色的用户，且增加员工后数据库中存在该员工信息；进货单、入库出库单与数据库中相同；增加仓库在数据库中存在，且库存信息与修改的相同；所修改鲜蔬价格与数据库中该鲜蔬信息相同。

(5) 测试结果：与输出标准相同。

5.3.5 对权限控制的测试

权限控制应具备的功能如下：

(1) 各角色权限下不可操作该角色不包含权限下的功能。

测试过程：

(1) 测试名称：权限控制

(2) 测试前提：系统未登录

(3) 测试过程：以采购员角色登录系统，在浏览器下输入该角色不拥有页面，对该页面惊醒功能操作。其他角色同理。

(4) 输出标准：页面弹出框提示“权限不够”，该用户无法操作。

(5) 测试结果：与输出标准相同。

5.4 系统性能测试

a. 对安全性的测试

安全性测试应具备的功能：数据库表中用户密码储存方式为乱码，便于保密。

测试过程：登录系统注册新用户，以及用办公室角色登录系统增加新员工。

输出标准：数据库表 `sys_user` 中该新增用户密码为乱码。

测试结果：与输出标准相同。

b. 对数据精确度的测试

数据精确度应具备的功能：数据类型涉及到价格重量相关需可以保留两位小数。

测试过程：输入鲜蔬产品价格小数保留两位。

输出标准：数据库中该鲜蔬价格小数为输入的两位。

测试结果：与输出标准相同。

c. 对容错性的测试

容错性应具备的功能：使用系统输入各类型数据型，当输入数据类型与要求输入数据类型不符合时，系统提示用户并修改。

测试过程：输入鲜蔬价格信息为字符串。

输出标准：系统前台页面提示用户输入数据类型错误，要求用户重新输入。

测试结果：与输出标准相同。

5.5 本章小结

本章从数据库的建立和连接以及各大功能模块的代码和界面截图详细展示了系统是如何实现的，并对系统进行了多方面的测试，测试数据为数据库中表的相关数据，经过测试，系统符合预期需求分析。

6 总结与展望

本章主要是对系统从设计到实现的整个阶段的总结和对系统的展望。对系统已经实现的部分做出了概括,同时针对系统的一些不足和需要继续完善的部分进行了阐述。

6.1 总结

随着生鲜市场升级的加速以及批发市场的外迁,高端餐厅,企事业单位食堂和酒店采购的透明互联网化已然开始,通过互联网来改造优化传统行业带来的效率是惊人的,这就为传统电商行业模式带来了挑战。在人力成本日渐走高的影响下,对信息化的重视程度也越来越高。本文在传统行业模式下提出了创新,针对鲜蔬特性设计并实现了一套鲜蔬配送管理系统。该系统实现了完整的鲜蔬产品供应链、客户、配送和库存管理体系,将员工管理和合作客户高度信息化集成,实现配送公司专业化运作。

该系统采用当下较为流行的开发模式,充分考虑了第三方技术集成难度问题,使用 Spring Boot 创建了一个独立运行、准生产级别的 Spring 框架的项目。鲜蔬配送管理系统呈现出一个功能强大、完整的信息管理平台,各类数据一览无余,今日订单、今日金额、新增用户、库存情况等。最新数据实时更新让配送公司时刻掌握最新运营状态。通过权限控制实现一套系统多角色用户,对应不同功能体系,方便灵活且安全性极高。系统功能特点在于:可按照供应商、采购员等多维度生成采购单;库存和订单汇总联动,备货精准,进销存管理(入库、出库精确报损报溢等);配送任务实时下达,司机提前规划;多仓库库存盘点,入库、出库数据等系统记录,随时导出;处理订单、配送订单等诸多权限可分配不同员工。

6.2 展望

当前系统已实现基本的配送运营模式,但还有进一步提升的空间。

(1) 对支持多重经营模式系统还未能实现,如:到点自取、按时达、立即送、周期购、预售等多重销售模式。

(2) 物流配送方面实现根据送货位置分布、订单量自动规划线路；支持扫码核对配送单，配送准确达 100%。

(3) 分析各供应商报价信息，多种供应商报价一体化展示及自动选择合适报价者省去审查麻烦。

这些更完善的功能促使我在以后的学习和工作中汲取更多知识，多方位学习专业知识，充实自己。

参考文献

- [1] Gutierrez, F. Spring with Spring Boot[J].Pro Spring Boot, 2016: 89-105.
- [2] Nash,D.&E,Gardner.Population Maven[J].Health Data Manag, 2015, 23 (8): 36.
- [3] Kwong,D.Get Coding!:Learn HTML,CSS,and JavaScript and Build a Website,APP,and Game[j].School Library Journal, 2017, 63 (7) : 106.
- [4] 周望德. 城市绿色蔬配送管理浅析[J]. 商, 2014(6):209-209.
- [5] 汪云飞, Java EE 开发的颠覆者: Spring Boot 实战[M].北京: 电子工业出版社, 2016.
- [6] 王云, 郭外萍, 陈承欢等. Web 项目中的 SQL 注入问题研究与防范方法. 计算机工程与设计, 2010; 31(5) : 976-978.
- [7] 杨家炜.基于 Spring Boot 的 web 设计与实现[J].轻工科技, 2016, (7) : 86-89.
- [8] 王永和,张劲松,邓安明,周智勋.Spring Boot 研究和应用[J].信息信. 2016(10).
- [9] 屈武江. 基于 Ajax 技术的 ASP.NET 数据分页. 计算机系统应用, 2013; 09(129) : 154-159.
- [10] 耿祥义, 张跃平.Java 2 实用教程[M].北京: 清华大学出版社, 2012-8.
- [11] 埃克尔著,陈昊鹏等译. Java 编程思想[M]. 机械工业出版社. 2005-5.
- [12] 张孝祥. 深入 Java Web 开发内幕——核心基础[M]. 北京: 电子工业出版社.2006.
- [13] 吴吉义, 平玲娣. Web2.0 主流应用技术——AJAX 性能分析. 计算机工程与设计, 2013; 09(129) : 154-159.

致谢

毕业设计已然接近尾声，回顾这数月的忙碌，有迷茫有奋斗，更多地还是感激和留恋。

毕业设计有幸在吴国文老师门下受教，吴老师是一位和蔼亲切，博学多识的教授。从一开始的选题到开题再到中期检查，吴老师为我指明了方向，并严格要求，极大地锻炼了我的独立自主能力，在此，衷心感谢吴老师。

同时也要感谢大学四年的专业课老师，是你们细心的传授了我专业知识。感谢大学四年的同学，是你们给了我一段美好又难忘的大学回忆。

最后感谢我的母校东华大学和计算机科学与技术学院。

文献翻译 原文

Vulnerability aware graphs for RFID protocol security benchmarking

ABSTRACT

Security and privacy issues in Radi Frequency Identification(RFID)systems mainly result from limited storage and computation resources of RFID tags and unpredictable communication environment.Although many security protocols for RFID system have been proposed, most of them have various flaws. We propose a random graph-based methodology enabling automated benchmarking of RFID security.First, we formalize the capability of adversaries by a set of atomic actions. Second, Vulnerability Aware Graphs (VAGs) were developed to elaborate the interactions between adversaries and RFID systems, which are used to discover the potential attacks of adversaries via some paths on the graphs. The quantitative analysis on VAGs can predict the probability that the adversary leverages the potential flaws to perform attacks. Moreover, a joint entropy-based method is provided to measure the indistinguishability of RFID tags under passive attacks. Analysis and simulation were conducted to show the validity and effectiveness of VAGs.

KEYWORDS

RFID Security protocol Vulnerability aware graphs Benchmarking

1.Introduction

Radio frequency identification (RFID) is an emerging technology that has posed great benefit to many applications such as logistics [1], manufacture [2], retailing [3], transportation [4], anti-counterfeiting [5], etc. When attached to

objects or persons, RFID tags can be used for identification and verification. It is important to prevent RFID systems from leaking confidential information of their users, and provide security guarantee for each participant in the system [6]. To this end, security protocols have been extensively studied to provide secure guarantee for data delivery between readers and tags, thwarting attackers.

RFID tags usually have tough constraints on the storage and computing resources (due to the cost consideration), which implies that the security protocols applied to RFID systems are expected to be lightweight, and many conventional en-ryption schemes used for wired systems, for example asymmetric key based ciphers, are not suitable for RFID systems. Therefore, RFID oriented security protocols tend to have much more flaws than those used for wired environments or other wireless applications. Worse yet, the common environments wherein RFID tag are deployed are open and unpredictable. An adversary can launch sophisticated attacks to penetrate the protocols, raising serious threats to RFID systems.

For modeling the security of RFID systems, some researchers have proposed either formal definitions of the privacy and security [7–10], or formalization of adversaries [11]. However, designing and analyzing RFID security protocols are still challenging since existing models are high-leveled and coarse-grained. With previous models, it is difficult to detect exact flaws in RFID protocols or attacks based on those flaws. Moreover, the security of RFID protocols as well as the impact of attacks cannot be quantitatively analyzed. In fact, the security analysis of RFID protocols is still on a pedestrian level. Thus, developing automated analysis and benchmarking measures for security protocols are extremely important to enhance the security of RFID systems. In this paper, we propose a random graph-based method coordinated with a fine-grained adversary model enabling automated analysis and benchmarking of RFID security protocols. Our contribution mainly comes from five aspects.

First, we propose an adversary model with a more flexible, fine-grained,

and realistic structure. We formalize the capability of an adversary by a set of atomic actions. Any attack launched by an adversary can be accurately represented via some of the atomic actions. Unlike previous models, atomic actions defined by our model are more accurate in simulating adversaries in real world.

Second, for the purpose of expressing how an adversary interacts with a tag or a reader separately, we do not generate a single state transition graph to show interactions between tag and reader. Instead, we propose a novel random graph-based method, termed VAGs (Vulnerability Aware Graphs), which uses a pair of graphs, Tag Graph (TG) and Reader Graph (RG), to reflect the interactions between the adversary and tag, the adversary and reader, respectively. VAGs enable the detection of the potential flaws in security protocols and the analysis of the adversary's attack patterns. By utilizing VAGs, an attack can be mapped to a harmful path on either TG or RG. We develop a set of rules to define the harmful paths. Each path refers to a flaw of the protocol. Each edge on a harmful path denotes an action taken by the adversary. Thus, by finding the harmful paths, we can derive the attack patterns.

Third, an adversary can hardly utilize a flaw which is extremely imperceptible. It is reasonable to calculate the probability that the adversary can detect and make use of the certain flaw, i.e., the probability he or she can walk along the path representing the flaw. We further analyze the probability an adversary can exploit a flaw and launch an attack quantitatively. Forth, since the passive adversaries, which do not disturb the communication between tags and readers, can only pose a threat to the indistinguishability of tags, the VAGs of a passive adversary are reduced to a pair of graphs indicating the interactions between a pair of reader and tag. We propose another joint entropy-based method to measure the indistinguishability of tags under passive attacks. Finally, we achieve the whole benchmarking methodology for RFID security protocols.

The rest of the paper is organized as follows. In Section 2, we discuss the

related work. We introduce the preliminaries in Section 3 and present the attack model in Section 4. We formalize the problem in Section 5 and detail our VAGs methodology in Section 6. Section 7 discusses the measuring of indistinguishability of tags under passive attacks. Section 8 shows the case study. We conclude our work in Section 9.

1. Related work

In this section, we will briefly introduce the previous researches closely related to our work, including RFID security models, graph-based methods for network security analysis, RFID benchmarking and security evaluation of RFID protocols. Table 1 shows the differences of existing work and the proposed one from several aspects, including proposing security models, conducting security benchmarking, automatic protocol analyzing, quantitative analyzing and type of attacks defending.

2.1 RFID security models

Some researchers are working on modeling the security or privacy of RFID systems [7–11]. A. Juels et al. [9] define the strong privacy of RFID system used for basic analysis of RFID systems. Whether a protocol is strong privacy can be analyzed manually. However, how to automatically analyze protocols with the definition is not mentioned. X. Zhang et al. [7] describe some security requirements, including privacy of tag data, privacy of ownership, integrity of tag data, and availability of tag identity in RFID system. It then formalizes the definitions of these requirements. I. Damgard et al. [10] model the behaviors of adversaries by specifying the actions she can take, but the adversary model is coarse-grained and inflexible for characterizing realistic adversaries. G. Avoine [11] also introduces an adversary model suitable for RFID environments. The formalization of an adversary can only be used to analyze the protocols in terms of traceability. S. Vaudenay [8] proposes a model and definition for anonymous identification of RFID system based on [9]. It considers the situation that tags

within an RFID system hold dependent keys, and discusses the security–efficiency tradeoff.

2.2 Graph-based methods for network security analysis

Both scenario graph [12] and attack graph [13] (which is a specific kind of scenario graph) are developed to evaluate the security of networked systems. There are a lot of work both on analyzing network vulnerabilities by using attack graph and generating attack graph [13–16]. Attack graph can show the relationship of vulnerabilities on different hosts in network. It takes into account the interrelationship of vulnerabilities activated by the interactions between hosts, since

Table 1

The differences between existing work and the proposed VAGs.

| Achievements | Security models | Security benchmarking | Automatically protocol analysis | Quantitative analysis | Attacks (active or passive) |
|--------------|-----------------|-----------------------|---------------------------------|-----------------------|-----------------------------|
| [7–11] | yes | no | no | no | both |
| [17,18] | no | no | no | yes | none |
| [27] | no | yes | no | yes | passive |
| [28,29] | no | yes | no | no | both |
| [30] | yes | yes | no | no | both |
| VAGs | yes | yes | yes | yes | both |

several vulnerabilities on different hosts can be utilized together to achieve certain attack objects. However original attack graph cannot sufficiently express the relationships that may be utilized by adversaries in RFID systems. In RFID systems, the adversary can not only utilize the relationship between the secrets of tags in the system (actually, in many RFID system, tags

are independent for the consideration of strong privacy [8], and our work is focused on such RFID systems), More important is the adversary can utilize the relationship between several executions of the protocol on the same tag which cannot be described by an attack graph.

2.3 RFID benchmarking

There are also some existing archives on RFID benchmarking [17,18], but they mainly focus on the performance and reliability of MAC or physical layer of tags. They aim to evaluate the impacts of surrounding objects, readers' power, the distance between tags and readers, or the physical orientation of tags on the performance of tags, such as read rate.

2.4 Security evaluation of RFID protocols

M. Alizadeh et al. analyzed the security (in terms of confusion and diffusion) of several lightweight encryption algorithms used in RFID applications. The authors only concern passive attacks on confidentiality of tags [27]. Some researchers performed the security and privacy analyses on lightweight protocols proposed recently and discuss their advantages and security issues [28,29]. Changshe Ma et al. proved that *ind-privacy* is weaker than *unp-privacy*. The necessary and sufficient condition on RFID tags to achieve *unp-privacy* is determined [30]. A pseudo-random function family is the minimal requirement on an RFID tag's computational power for enforcing strong RFID system privacy.

3. Preliminaries

3.1 RFID system

RFID tag can be divided into two types: *passive* and *active* tag. The main difference between them is that the passive tag does not have internal power, while the active tag does [19]. The passive tag makes use of RF waves emitted from the reader as the power supply. It thereby cannot afford complex

microprocessors that require relatively high power to work. In order to reduce the cost of production, the storage capacity of passive tag is also extremely limited (current passive tag commonly has couples of kilobytes as the storage). Due to the above hardware constraints, security risks on passive tags are extremely high, and security protocols of passive tags tend to have more flaws than that of active tags. In this paper, we focus on the security issues on passive tags. Note that our method can be compatibly applied to the protocols of active tags.

In RFID systems, the reader is a device used to interrogate RFID tags. For passive tags, the reader is also responsible for activating the tag's microchip. The reader does not store sensitive information about the tags or systems. After obtaining responses of tags, it simply transmits the result to backend server for further processing. Generally, the channel between a reader and backend server is considered safe. Tags and readers communicate via two open unsecure wireless channels, i.e., reader-to-tag (forward channel) and tag-to-reader (backward channel) communications. Communication distance of forward channel is larger than that of backward channel, since the reader's transmission power is much higher than that of tags. Correspondingly, the eavesdropping distance of adversaries on the forward channel is also much larger than that on the backward channel. The back-end server processes the information related to tags in the system and is powerful in terms of computational resource. Hence, it is considered as a secure and trusted entity, which cannot be compromised by attackers. In this paper, we consider the 'reader' as a single unit combining the RFID reader and backend server.

The communication between the passive tag and reader adopts the challenge-response mechanism. The reader initiate a challenge (interrogation). After receiving the challenge, the tag computes a result, i.e. response, and sends it back to the reader. The security protocol executed between the reader and tags is also based on the challenge-response mechanism. A successful execution

of protocol called a session. In this paper, we denote m a message delivered between a tag and a reader, s_i is the i th session of a RFID security protocol, and tag j is the j th tag in the system. The automatic challenge–response communication mechanism used by RFID tags and readers leaves a security flaw, with which the attacks can easily gain unauthorized access to RFID data. Therefore, RFID security and privacy are increasingly important.

3.2 Security issues of RFID system

A tag indicates the type of an object to which it is attached, or even uniquely identifies the object. Therefore, the data a tag carries on is highly related to the confidential information of the object's owner. Concerns of an adversary aiming at breaking down an RFID system include *privacy*, *security*, and *functionality* of it. We formally define the concerns as follows.

Definition 1 (Privacy). The privacy problem of an RFID system comes from two aspects: one is data leakage of RFID-tagged belongings, and the second is behavioral tracking or personal identification by tracing tag IDs.

In privacy-guaranteed RFID system, anonymity^A and untraceability should be considered. Anonymity indicates that a tag's responses should be unlinkable to the ID of the tag. Untraceability requires that cannot distinguish a particular response of targeted tag from those of other tags (otherwise can trace the tag). Indeed, anonymity and untraceability can be satisfied by *indistinguishability* [20], hence we focus on the *indistinguishability* in the paper.

In order to ensure indistinguishability, it is necessary that transmitted tag's information should not be the same or cor-relate with each other from the perspective of inappropriate parties. The two most significant threats on indistinguishability are tracking and hotlisting.

Definition 2 (Security). Security is focused on the tag impersonation^A problem in RFID systems, meaning that is not able to cheat a reader, i.e., to make the reader to accept responses from a faked *tag*. (This definition is consistent with the

definition of security in [10].)

Security refers to the certainty that the transmitted tag's information is not tampered with during or after challenge–response processes. It implies that the data will not be modified or destroyed by unauthorized parties. The *security* of the RFID challenge–response processes can be compromised through relay, replay, message reconstruction, data modification/in-section attacks.

Definition 3 (*Functionality*). *Functionality* denotes that all the functions provided by the RFID systems can be correctly conducted. Towards an RFID system that has a functionality guarantee, the adversary cannot disturb its functions, for example, making a legitimate reader to reject a legitimate tag, or vice versa.

Functionality requires that the information is available when it is needed. In order for an RFID system to demonstrate *functionality*, it must have properly functioning computing systems, security controls and communication channels. Denial of Service (DoS) attacks are one of the most challenging threats against *functionality* since they can be easily deployed while they are hard to defend against. This type of attacks includes passive degrading the RF signal, and active jamming or disrupting communications. De-synchronous attack is another challenging threat which causes reader reject legitimate tags. Notice that the above issues we concerns are similar to the classical CIA (Confidentiality, Integrity, and Availability) requirements, where *privacy* to confidentiality, *security* to integrity, and *functionality* to availability.

4. Attack modeling

In this section, we discuss the behaviors of adversaries, and construct several atomic actions on which any attack can be built. We then model how an adversary can intervene in challenge–response processes between a tag and a reader.

An adversary A is an entity that attempts to thwart the *security* and *privacy*, or disable the system *functionality*. We assume that the adversary knows the security scheme used by the RFID system, i.e., the specification of security protocol. A does not know the secret shared between the tag and reader (backend server), unless he or she compromises the tag. A can compromise a tag via some kinds of physical attacks, but these physical attacks are destructive, which means the tag become useless after the compromising. Due to the fact that the wireless communication channels between the tag and reader are open, A is able to acquire partial or full control over the messages transmitted on both forward and backward channels, which is dependent on the capabilities of.

A can launch various attacks, such as eavesdropping which is the interception of communication between a legitimate reader and tag, rogue scanning which involves a malicious reader that could access a tag without its owner's permit, tracking, hijack, physical attacks, spoofing (cloning, swapping, relay, replay etc.), and DoS attacks.

Although the purposes and patterns of those attacks are diverse, the behavior of each attack can be resolved into a finite set of atomic actions. In other words, any attack comprises of a successive series of atomic actions. We consider those actions as oracles (each oracle can be considered as a theoretical black box, which is able to solve certain decision problems) that can be accessed by A .

Definition 4 (*The atomic actions*). A set of oracles that an adversary A can access:

SendToReader (m, s_i): this oracle formalizes the model of sending message m to the reader in session s_i of A the protocol through backward channel.

SendToTag (m, tag_j, s_i): this oracle formalizes the model of sending message m to tag_j in session s_i of the protocol through forward

channel.

ReceiveFromTag (m, tag_j, s_i): this oracle formalizes the model of receiving message m from tag_j in session s_i of the protocol through backward channel.

ReceiveFromReader (m, s_i): this oracle formalizes the model of

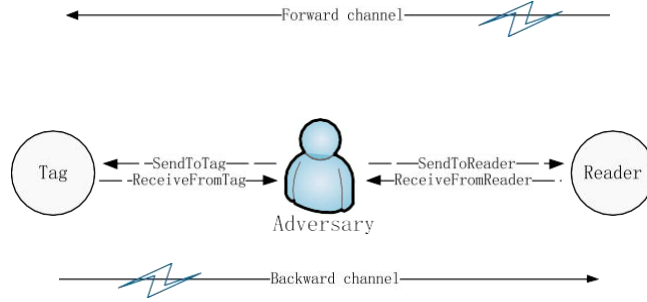


Fig. 1. The adversary serves as a part of RFID system.

receiving message m from the reader in session s_i of the protocol through forward channel.

- **Corrupt** (tag_j): this oracle formalizes the A model of compromising tag_j . can obtain the current secret of tag_j , and make the tag disabled, and hence all the atomic actions corresponding to tag_j cannot be used any more.

SideChannel (tag_j, s_i): this oracle formalizes the model of A observing the result of tag_j in session s_i of the protocol.

The capability of A is a subset of the oracles defined in Definition 4. Now we elaborate the capabilities of A and how interferes with RFID systems in our model. We consider serving as a relaying forwarder similar to a router in a conventional IP network.

By accessing the **SendTo*** and **ReceiveFrom*** oracles in Definition 4, can control all the communications between the tag and reader (as shown

in Fig. 1) and decide whether a message is relayed faithfully, distorted, or dropped (i.e. intercepted).

A accesses **SendToReader** and **SendToTag** oracles to send a message that complies with the specification of the protocol to the reader and tag, respectively. **ReceiveFromReader** and **ReceiveFromTag** represent that receives the messages transmitted on the two channels. Note **ReceiveFrom*** implies that not only can eavesdrop on the messages transmitted on channels, but also A fully controls those messages. For example, intercepting a message means that accesses **ReceiveFrom** while ignoring **SendTo***.

By accessing **Corrupt**, A can obtain the current secret stored in the tag, and compute the corresponding information using compromised secret and historical observations.

- **SideChannel** allows A to obtain the result of protocol execution of a given session, i.e., accepting or rejecting a tag.

Particularly, the oracles above can model not only active but also passive adversaries, who only listen on communication between tags and readers, but do not modify the message stream in any way. For a passive adversary, the actions of he or she should follow the following principles:

- The adversary cannot access the **Corrupt** oracle.
- In session s_i , **ReveiveFromReader** (m, s_i) and **SendToTag** $(m, tag\ j, s_i)$ should always access in pairs. Furthermore, **ReveiveFromReader** (m, s_i) always follows with **SendToTag** $(m, tag\ j, s_i)$ (similarly, **ReveiveFromTag** $(m, tag\ j, s_i)$ always follows with **SendToReader** (m, s_i)), which means that the adversary faithfully relay message m .

Above oracle designs pose several advantages to our model, such as the flexibility, fine-grained structure, and accurate definitions for real adversary's

capabilities. For example, different from existing adversary models, our model resolves the interaction between A and reader (or tag) into oracles **ReceiveFromReader** (or **ReceiveFromTag**) and **SendToReader** (or **SendToTag**) instead of treating the entire interaction as a single oracle. Similarly, we should also split the **ReceiveFrom**^{*} oracle used in previous approaches into two more fine-grained oracles, **ReceiveFromReader** and **ReceiveFromTag**. Hence, an adversary who can access oracle **ReceiveFromReader** may not be able to access oracle **ReceiveFromTag**, which is more close to reality. It is because in real RFID systems, the eavesdropping range is relatively larger than the scanning range. The eavesdropping range of tag-to-reader and eavesdropping range of reader-to-tag are also different (the former one is smaller than the latter one). On the other hand, the reader and tag will not always response after receiving a message, which is dependent on the certain protocol. To reflect those incomplete responses, we also need a more flexible definition for the actions of A .

5. Problem formulation

In this section, we first describe the state transition model that represents the interactions between the reader and A , the tag and A , respectively. The couple of state transition model describe how A intervenes in the communications between A the tag and reader. It also describes how state transitions of the tag or reader proceed under the actions launched by A . Then we introduce the formal definition of VAGs, which is a random graph-based methodology used for automated analysis and benchmarking of RFID security protocols.

5.1 Formalizations of state transition model

We model both the state transitions of the tag and reader forced by A 's actions using an uniform transition system $M = (S, I, R, E)$ where:

- a) $\mathcal{S} = s_1, s_2, \dots, s_n$: a set of states, s.t. \mathcal{S} .
- b) I : the initial state
- c) $R \subseteq \mathcal{S} \times \mathcal{S}$: a set of transitions
- d) $E \subseteq \mathcal{S}$: a set of end states.

State: a *state* of a reader (or tag) is defined by its *Internal State* and *External Output*. Specifically, the state of a tag and reader can be written as shown below:

$$s^t = [s^t \cdot ins, s^t \cdot resp]^{\Sigma}$$

Internal State (Ins): an *Internal State* includes a set of variables, which are secrets either only stored in a tag (or reader) or shared with its counterparts. The secrets can be a counter, session ID, key, etc. Note that although there are different kinds of information stored in tags or readers, for a particular protocol, only those related to state transitions should be concerned.

External Output: the *External Output* of a reader is its challenge sent to a tag (*Chlg*), while that of a tag is its response to a challenge (*Resp*).

- $s^t \cdot resp$ (*Resp*): *Resp* denotes a response returned by a tag at state s^t . If A accesses **SendToTag**, the tag at state s^t may transfer to a new state s_j by calculating a new *Resp* as $s_j \cdot resp$, while update $s_i \cdot ins$ to $s_j \cdot ins$. A can retrieve $s_j \cdot resp$ by accessing **ReceiveFromTag**.
- $s^r \cdot chlg$ (*Chlg*): *Chlg* denotes a challenge generated by a reader at s^r . *Chlg* can be the first message launched by the reader in a session proactively. For a protocol involving several challenge–response interactions, *Chlg* can also be generated according to the message m in **SendToReader** that A accesses. Hence, as the reaction of **SendToReader**, the reader at **ReceiveFromReader**.

- **The initial state** (Ini): Ini represents a special state with the original internal condition before the interaction.
- $R(s_i, s_j)$ represents a transition from state s_i to s_j . In our model, $R_T(s^t, s^t)$ exists if A can take either a **SendToTag** or **Corrupt** action to trigger the state transition of the tag from s^t to s^t . $R_R(s^r, s^r)$ exists if A can take a **SendToReader** action to trigger the state transition of the reader from s^r to s^r . Since M_T and M_R reflect the interactions between the tag and reader, any tag's response representing a state in tag's model must correspond to at least one transition of the reader's model. On the other hand, any reader's challenge representing a state in reader's model must correspond to at least one transition of the tag's model.
- E_T represents the end state of a tag caused by a **Corrupt** action. A tag in E_T cannot perform functions any more. Because A can never compromise a reader, the reader does not have such states.

Definition 5 (*Size of state space*). The size of a model $M = (S, I, R, E)$ is defined as $|S|$, i.e. the size of its state space.

5.2 Vulnerability aware graphs

Although M_T and M_R describe how each component of RFID system responses and updates its states under the actions of attackers, they are still inconvenient to express the attacking procedure. Moreover, it is difficult to find all the potential attacks with M_T and M_R only. Hence, we develop a random graph-based methodology, termed as Vulnerability Aware Graphs (VAGs). VAGs explicitly depicts the relevance of states in M_T and M_R from three aspects: the causal relationship of a pair of challenge and response within one session, relevance of several sessions which may be utilized by A , interactions between

tag and reader with the existence of A .

There are many non-equivalent ways to define random graphs. In this paper, we adopt the definition used in [21] that the formation of a random graph is a random process, which has advantages on describing the generation of VAGs. Moreover we extend the standard model of the random graphs by allowing slings and directed edges.

VAGs contains a pair of directed random graph, one reflects the interaction between tag and the adversary called *Tag Graph* (TG), while another reflects the interaction between reader and the adversary called *Reader Graph* (RG). The two graphs are constructed according to their state transition models. The vertices in TG and RG denote the states in M_T and M_R , respectively. Edges in the graphs represent state transitions caused by **SendTo*** or **Corrupt** actions taken by A . Each possible edge occurs in a random graph with a certain probability. Each graph has a starting vertex without predecessor, representing the initial state I .

Specifically, for $M_T = (S_T, I_T, R_T, E_T)$, each $s^t \in S_T$ corresponds to a vertex on TG. $R_T(s^t, s^t)$ corresponds to a directed edge which starts from vertices s^t and ends at s^t . $e_t \in E_T$ is a vertex without an edge beginning from it. For corresponds to a directed edge which starts.

M_T and M_R can be constructed as discrete time step models to create TG and RG, respectively. The formation of VAGs as follows: each graph starts with a starting vertex (I). At each time step, all the possible edges starting from the existing vertex set are added, and then all the corresponding vertices at the end of the edges that have not been created yet are generated as their successors. These vertices then are labeled with *Internal State*, *External Output*, and added into existing vertices set. The iterations continue until no new edges can be added. Each directed edge between two vertices stands for an action taken by A to make a state transition. The

probability that an edge occurs in the graph equals to the probability that A can take the corresponding action.

Definition 6 (Path). A path π in VAGs indicating the transition system $M = (S, I, R, E)$ is either an infinite or finite sequence of state transitions $R(s_i, s_{i+1}) \in R$, which connects a sequence of states $\{s_0, \dots, s_i, \dots, s_k\}$, $s_i \in S$, $0 \leq i \leq k$.

VAGs aim to automatically analyze and benchmark RFID security protocols, to detect the flaw of the protocols and point out potential attacks. Any interactions series between a tag (or reader) and can be viewed as a path on TG (RG).

We draw the VAGs of the well-known OSK protocol [20] as an example. The OSK protocol is shown in Fig. 2. Both $H(\cdot)$ and $F(\cdot)$ are hash functions. Initially, tag has initial information K , and backend server stores all the (ID, K) pairs, which are different for each tag in a system. For the first time a tag receives a challenge m from a reader, it calculates $F(K)$, where K is its current secret. Then it sends $RespF(K)$ to the reader, and updates its secre $K H(K)$. A tag repeats above operations each time it gets a challenge.

For example, after i session, the reader sends the $(i+1)$ -th challenge. After receiving m , the tag computes $F(\cdot)$ using its current secret, which is $H^i(K)$, $(H^i(K) H(H(\dots H_i(K))))$. Then the tag sends $Resp$, i.e., $F(H^i(K))$, to the reader, and updates its secret to $H^{i+1}(K)$. The reader receives $Resp$ from the tag, and calculates $F(H^i(K))$ for each (ID, K) pair. If $F(H^i(K)) = Resp$, the tag will be accepted. Note that OSK proposes a fixed upper bound n of the number of time steps over which tags are operated. After the n -th interrogation, the tag yields random output, which cannot be accepted by reader.

6. Vulnerability aware graphs

Indistinguishability is merely related to TG, because it only concerns the

relationship between tag's responses and adversary's challenges. On TG, it can be observed that *indistinguishability* of an RFID security scheme depends on whether there is a distinguishable path on TG. A distinguishable path exists if relations between challenges and responses on the path can be recognized as a characteristic by A . Here we just give several popular characteristics on the graph that might be exploited by adversary. In practice, more characteristics can be found using our model.

- **Self-loop:** a self-loop on TG indicates that there is an $R_T(s^t, s^t)$. It implies A can take some actions to make the tag return the same response.
- **Loop:** a loop on TG indicates that there is a series of successive $R_T(s^t, s^t)$ that begins and ends with the same state. It implies A can take some actions to make the tag return the same response periodically.

Definition 7 (*Distinguishable path*). A path on TG where the External Outputs (Resps) on it show some certain characteristics, for example self-loop or loop.

If there exists a distinguishable path on a TG, may be able to walk along the path by performing a series of attack actions, and hence can observe the characteristics. Then can distinguish the tag from other tags based on the observation. Although the characteristics can be multifarious, it must be noticed that if there exists such characteristics on TG, the only thing that should do is to find the path holding the characteristics. Since different challenges will lead to different responses represented by different paths on TG, to observe the characteristic of a path, should decide the challenge(message m in **SendToTag**) in each interaction to the tag, so that the responses of the tag are in accordance with those responses at each tag's state along the path. Hence, we get the following conclusion:

Proposition 1. The ability that can distinguish a tag from others is defined as the probability that he or she can choose the correct challenges along the distinguishable paths on TG.

6.2 Security

Security means that is not able to cheat a reader, or make the reader to accept responses from fictitious tags. If the reader accepts a legitimate (uncompromised) tag, but the tag does not have a matching interaction with the reader, it means that breaks down the security of the RFID system. A matching interaction (successful execution of the protocol) means that the reader and tag exchange challenges and responses well interleaved and faithfully (but may be with some time delay) over the execution of a session [10]. We make the following definitions to describe that if a tag and reader are under a proper execution of the protocol.

Definition 8 (*Half-matching states*). Suppose state s_t on TG and s_r on RG, if $s_t \text{ resp}$ is related to a transition starting from s_r on RG, **OR** the $s_r \text{ chlg}$ is related to a transition pointing to s_t on tag graph, these two states are half-matching states, and denoted as $s_t \text{ resp}$, s_r or $s_r \text{ chlg}$, s_t . We say there exists a half-matching between s_r and s_t if at least one of the above relation is true.

Definition 9 (*Matching states*). Suppose state s_t on TG satisfies $s_r \text{ chlg}$, s_t , **AND** s_r on RG satisfies $s_t \text{ resp}$, s_r . The two states are defined as matching states. We say there exists a matching between s_t and s_r . The reader and tag on matching states imply that the interaction between them is legal to the protocol.

Definition 10 (*Mismatching states*). Suppose state s_t on TG and s_r on RG. If $s_t \text{ resp}$ is **NOT** related to any transition pointing to s_r on RG and $s_r \text{ chlg}$ is **NOT** related to any transition pointing to s_t on TG, s_t and s_r are named mismatching states. If a tag and a reader are on such states, they should not be in the same session of the protocol. We say that s_t and s_r are mismatching.

7. Measuring indistinguishability of tags under passive attacks

Since the passive adversaries only eavesdrop on the forward and backward channels between tags and readers, which does not disturb the

communication between tags and readers, they can only pose a threat to the *indistinguishability* of tags. In this section, we represent how to measure the *indistinguishability* of RFID tags under passive attacks.

Although passive adversaries can be formalized well by using the atomic actions in Definition 4, and passive attacks can certainly be represented by VAGs, passive attackers, in essence, do not take any action to force tags or readers make responses to counterfeit challenges. It implies the VAGs will degenerate to a pair of graphs indicating the interactions between a pair of reader and tag. Thus, we propose another convenient method to measure the *indistinguishability* of tags under passive attacks.

From the perspective of passive adversaries, a tag is considered as a black-box in the sense that it only knows the challenge–responses pairs, but knows nothing about the internal state of it. For any challenge–response mechanism, we first measure the *indeterminacy* (uncertainty) of the challenges and responses, and the *mutual dependence* of the challenge– response pairs respectively as follows.

Definition 11 (*Indeterminacy of challenges*). The indeterminacy of challenges C of a tag is the entropy of it, denoted by $H(C)$ as follows

$$H(C) = - \sum_{c \in \partial} p(c) \log p(c),$$

where ∂ is the domain of the challenges, c is one of the challenges, $p(c)$ is the probability c occurs. $H(C)$ can be used to measure the feasibility that an adversary guesses a challenge. By pretending a legal challenge, the adversary can get a particular response.

Definition 12 (*Indeterminacy of responses*). The indeterminacy of a response R is the entropy of it, denoted by $H(R)$ as follows.

$$H(R) = - \sum_{r \in \varphi} p(r) \log p(r),$$

where φ is the domain of the response, r is one of the responses, $p(r)$ is the

probability r occurs. $H(R)$ can be used to measure the feasibility that an adversary guesses a response. By pretending a legal response, the adversary can cheat the reader.

Definition 13 (*Mutual dependence of a challenge–response pair*). The mutual dependence of a challenge–response pair (C, R) is the mutual information between them, denoted by $I(C, R)$ as follows

$$I(C, R) = - \sum_{c, r} p(c, r) \log p(c, r),$$

where $p(c, r)$ is the joint probability distribution of c and r . $I(C, R)$ is used to measure how much a certain challenge tells us about the corresponding response. i.e., how much the entropy of a response is reduced if we know a certain challenge?

Then given a pair of tag and reader, we measure the indistinguishability of the tag as follows:

Definition 14 (*Indistinguishability*). We use the joint entropy to measure the indistinguishability a challenge–response pair (C, R) , denoted by $H(C, R)$, as follows

$$H(C, R) = - \sum_{c \in \mathcal{C}} \sum_{r \in \mathcal{R}} p(c, r) \log p(c, r).$$

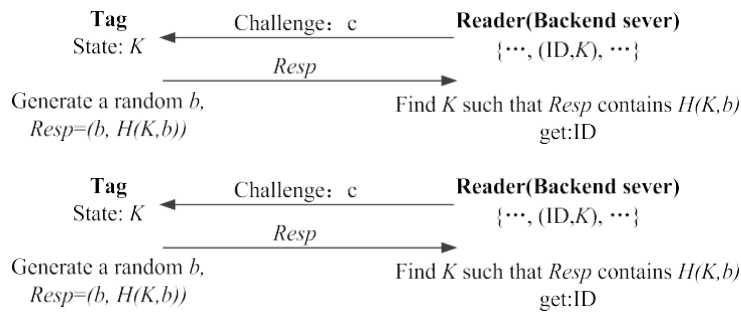


Fig. 5. WRSE's randomized hash-lock protocol.

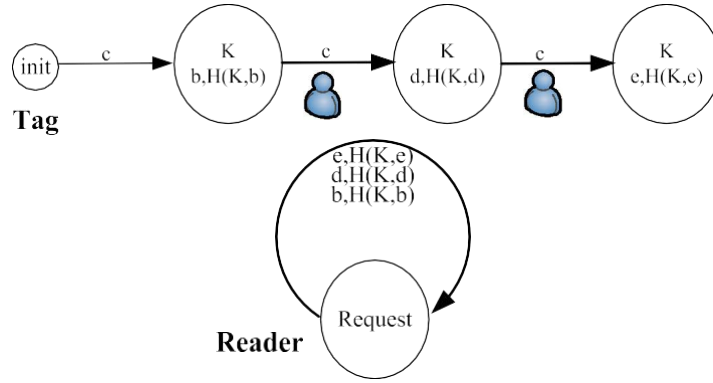


Fig. 6. VAGs of WRSE.

We use the joint entropy to measure how much entropy is contained in RFID systems which use challenge–response mechanism. C and R describe related events, the total entropy of the systems should be as follows:

$$H(C, R) = H(C) + H(R) - I(C, R).$$

By measuring the challenge–response pairs of a certain tag, we can evaluate the indistinguishability of such tag. In any RFID system, for a tag and reader pair, a large $H(C, R)$ implies high indistinguishability of the tag.

8. Benchmarking case study

In this section, we first apply the benchmarking methodology to WRSE’s randomized hash-lock scheme , and a variant of OSK protocol [8,22] (VOSK), to show the correctness and effectiveness of our method. We generate the VAGs of WRSE and VOSK and present our analysis. The security parameter k is related to the size of state space. For a practical protocol used in RFID systems, k should be sufficiently large to guarantee that an adversary cannot exhaust all the state space via brute force attacks. The generation of VAGs will suffer the state explosion problem as k increases. In this paper, we do not aim to develop a method to resolve the state explosion issue, instead we can use the existing abstraction techniques for model checking [23,24] to generate an abstract model that contains a smaller set of states, while preserving

safety properties of original graph. We can also use on-the-fly techniques [25,26] to avoid generating the paths that may occur with extremely small probabilities.

8.1 WRSE's randomized hash-lock protocol

We illustrate WRSE's randomized hash-lock (WRSE [31]) in Fig. 5. $H(\cdot)$ is a hash function. Each tag has secret information K , and backend server stores all the (ID, K) pairs, which are different for each tag in a system. Reader sends a constant challenge c . After receiving c , tag chooses a random k -bit *nonce*, and calculates $H(K, \text{nonce})$. Then it sends *Resp* as $(\text{nonce}, H(K, \text{nonce}))$ to reader. Reader searches for K among all the (ID, K) pairs such that r_0 contains $H(K, \text{nonce})$. Reader accepts the tag if such a K exists and rejects otherwise.

For providing an intuitive impression of WRSE, we first give a VAGs of WRSE with quite small state space in Fig. 6. We set that the nonce is chosen form Xb, d, e . We depict all the states and transitions of WRSE under the setting of above parameters.

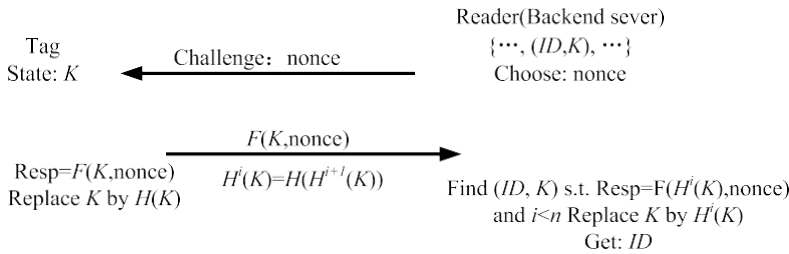


Fig. 7. A variant of OSK protocol

8.2A variant of OSK protocol

We illustrate VOSK in Fig. 7, which has two differences compared with original OSK. First, the reader sends the challenge with a nonce (a random α -bit string). Second, after the backend server accepts a tag, the reader updates current K .

For providing an intuitive impression of VAGs, we first give a VAGs of VOSK with quite small state space in Fig. 8. We set n as 4, and the nonce is chosen form Xa, b, c . We depict all the states and transitions of VOSK under the setting of above parameters.

On TG of Fig. 8, we use bold arrow lines to represent the paths that can find. gets the s_1 Chlg on RG by accessing **RFR**, then accesses **STT** (s_1 Chlg) to query tag repeatedly. As a result, can find the path $Ini s_4 s_2 s_9 s_{10}$ and launch a DoS attack to disrupt the *functionality* of the system. Moreover, can access **SideChannel** to learn that the tag is rejected by reader, which is also a distinguishable characteristic for tracing.

We design an algorithm to generate VAGs and calculate the probability that each path is found by . Fig. 9 shows the TG of VOSK with 50 states generated by our algorithm. The path with red solid line indicates a path located by to launch a DoS attack.

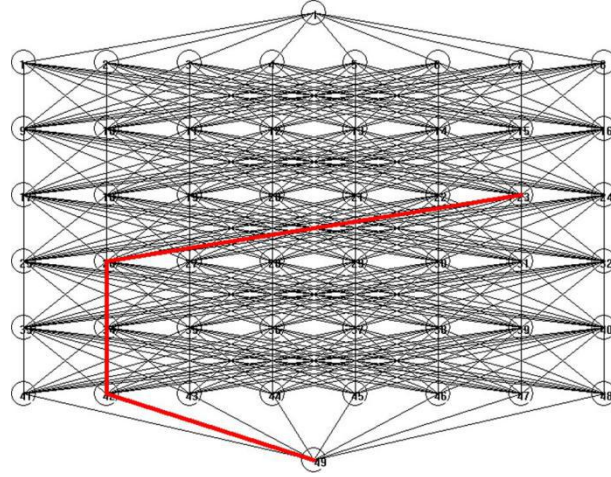


Fig. 10. VAG of VOSK generated automatically.

In Fig. 10, the probability for the adversary to locate each path is calculated, and we compare the number of paths with different probabilities when changing the size of state space and the amount of knowledge about the system that already has. In Fig. 8, R is related to the length of nonce (α), C denotes the value of n , A indicates the number of actions has taken. It can be observed that in VOSK protocol, after one attack action, the probability that adversary can launch an attack has greatly increased. It also can be seen that under the same state space, a larger n will result in higher security in VOSK protocol.

9. Conclusion

In this paper, we propose a random graph-based methodology to facilitate the benchmarking and quantitative evaluation for RFID security protocols. The methodology includes a fine-grained and accurate model for expressing adversary's capabilities, so that the attacks launched by adversaries can be resolved into a set of atomic actions. We formalize how the adversary intervenes in the RFID system by using the state transition models for both the tag and reader. Based on these models, we propose a random graph-based method, VAGs, to reflect the interactions between the adversary and tag, adversary and reader, respectively. By investigating the characteristic that a harmful path should hold, we use VAGs to exploit the flaws in RFID security protocols and to analyze the attack patterns of adversary. Moreover, we discuss the probability an adversary can successfully find the flaws and launch an attack to the RFID system.

References

- [1] R. Oh, J. Park, A development of active monitoring system for intelligent RFID logistics processing environment, in: International Conference on Advanced Language Processing and Web Information Technology, 2008, pp. 358–361.
- [2] Y. Lee, F. Cheng, Y. Leung, Exploring the impact of RFID on supply chain dynamics, in: Conference on Winter Simulation, 2004, pp. 1145–1152.
- [3] M. Bertolini, G. Ferretti, G. Vignali, A. Volpi, Reducing out of stock, shrinkage and overstock through RFID in the fresh food supply chain: evidence from an Italian retail pilot, *Int. J. RF Technol., Res. Appl.* 4 (2013) 107–125.
- [4] H. Hsu, H. Liao, A mobile RFID-based tour system with instant microblogging, *J. Comput. Syst. Sci.* 77 (4) (2011) 720–727.
- [5] T. Staake, F. Thiesse, E. Fleisch, Extending the EPC network: the potential of RFID in anti-counterfeiting, in: ACM Symposium on Applied Computing,

- [6] J. Jang-Jaccard, S. Nepal, A survey of emerging threats in cybersecurity, *J. Comput. Syst. Sci.* 80 (5) (2014) 973–993.
- [7] X. Zhang, B. King, Modeling RFID security, in: *Conference on Information Security and Cryptology*, in: LNCS, vol. 3822, 2005, pp. 75–90.
- [8] S. Vaudenay, On privacy models for RFID, in: *ASIACRYPT*, in: LNCS, vol. 4833, 2007, pp. 68–87.
- [9] A. Juels, S. Weis, Defining strong privacy for RFID, in: *Pervasive Computing and Communications Workshops*, 2007, pp. 342–347.
- [10] I. Damgard, M. Pedersen, RFID security: tradeoffs between security and efficiency, in: *CT-RSA*, 2008.
- [11] G. Avoine, Adversarial model for radio frequency identification, *IACR E-print*, vol. 49, 2005.
- [12] S. Jha, J. Wing, Survivability analysis of networked systems, in: *International Conference on Software Engineering*, 2001, pp. 307–317.
- [13] O. Sheyner, Scenario graphs and attack graphs, Thesis, Carnegie Mellon University, 2004.
- [14] P. Ammann, D. Wijesekera, S. Kaushik, Scalable, graph-based network vulnerability analysis, in: *ACM Conference on Computer and Communications Security*, CCS, 2002, pp. 217–224.
- [15] X. Ou, W. Boyer, M. McQueen, A scalable approach to attack graph generation, in: *ACM Conference on Computer and Communications Security*, CCS, 2006, pp. 336–345.
- [16] O. Sheyner, J. Haines, S. Jha, R. Lippmann, J. Wing, Automated generation and analysis of attack graphs, in: *IEEE Symposium on Security and Privacy*, S&P, 2002, pp. 273–284.
- [17] S. Cheung, W. Chan, P. Lee, L. Ni, P. Ng, A combinatorial methodology for RFID benchmarking, in: *RFID Academic Convocation*, 2006, pp. 1–6.
- [18] M. Buettner, D. Wetherall, An empirical study of UHF RFID performance, in: *ACM International Conference on Mobile Computing and Networking*,

- Mobicom, 2008, pp. 223–234.
- [19] A. Juels, RFID security and privacy: a research survey, *IEEE J. Sel. Areas Commun.* 24 (2006) 381–394.
- [20] M. Ohkubo, K. Suzuki, S. Kinoshita, Cryptographic approach to “privacy-friendly” tags, in: *RFID Privacy Workshop*, 2003.
- [21] P. Erdos, A. Renyi, On the evolution of random graphs, *Publ. Math. Inst. Hung. Acad. Sci* 5 (1960) 17–61.
- [22] G. Avoine, E. Dysli, P. Oechslin, Reducing time complexity in RFID systems, in: *Selected Areas in Cryptography, SAC*, 2006, pp. 291–306.
- [23] A. Gupta, Learning abstractions for model checking, Thesis, Carnegie Mellon University, 2002.
- [24] M. Lohrey, S. Maneth, M. Schmidt-Schauß, Parameter reduction and automata evaluation for grammar-compressed trees, *J. Comput. Syst. Sci.* 78 (5) (2012) 1651–1669 (Special Issue: Cloud Computing).
- [25] R. Gerth, D. Peled, M. Vardi, P. Wolper, Simple on-the-fly automatic verification of linear temporal logic, in: *International Symposium on Protocol Specification, Testing and Verification, PSTV*, 1995, pp. 3–18.
- [26] A. Bouajjani, S. Tripakis, S. Yovine, On-the-fly symbolic model checking for real-time systems, in: *Real-Time Systems Symposium, RTSS*, 1997.
- [27] M. Alizadeh, M. Salleh, M. Zamani, J. Shayan, S. Karamizadeh, Security and performance evaluation of lightweight cryptographic algorithms in RFID, in: *Recent Researches in Communications and Computers*, 2012.
- [28] E. Vahedi, R.K. Ward, I.F. Blake, Security analysis and complexity comparison of some recent lightweight RFID protocols, in: *Computational Intelligence in Security for Information Systems, CISIS 2011*, in: *LNCS*, vol. 6694, 2011, pp. 92–99.
- [29] B. Niu, H. Li, X. Zhu, C. Lv, Security analysis of some recent authentication protocols for RFID, in: *International Conference on Computational Intelligence and Security, CIS*, 2011, pp. 665–669.

- [30] C. Ni, Y. Li, R. Deng, T. Li, RFID privacy: relation between two notions, minimal condition, and efficient construction, in: ACM conference on Computer and Communications Security, CCS, 2009, pp. 54–65.
- [31] S. Weis, S. Sarma, R. Rivest, D. Engels, Security and privacy aspects of low-cost radio frequency identification systems, in: International Conference on Security in Pervasive Computing, SPC, in: LNCS, vol. 2802, 2003, pp. 454–469.

文献翻译 译文

RFID 协议安全基准测试的漏洞感知图

摘要

RFID 系统的安全和隐私问题主要源于 RFID 标签有限的存储和计算资源以及不可预计的通信环境。尽管已提出了很多关于 RFID 系统的安全协议，但大多数是有问题的。我们提出一个随机图论方法启用自动基准测试 RFID 安全。首先，我们将攻击者的攻击能力拆分为一组原子操作。然后，使用 VAGs 来加强攻击者和 RFID 系统的联系，其目的是通过一些方法发现潜在攻击者。关于 VAGs 的定量分析可以预测出攻击者利用潜在缺陷攻击的概率。此外，使用基于联合熵的方法可以测量 RFID 在被动攻击下的不可分辨性。通过分析和模拟来说明 VAGs 的正确性和有效性。

关键词：RFID 安全协议 脆弱感知图 基准测试

1、简介：

RFID 是一种新兴的技术，被用于多种应用，例如：物流业，制造业，零售业，运输业以及防伪等。当攻击对象或者人时，RFID 标签可以被用来识别和认证。防止 RFID 系统泄漏用户的隐私信息，确保系统中的参与者的安全性就显得尤为重要。为此，安全协议被广泛研究以确保数据在阅读器和标签之间传输时的安全性。

RFID 标签通常在存储和计算资源上有严格的限制（考虑到成本），这就意味着运用在 RFID 系统上的安全协议需要轻量，许多常用的加密方法被用于有线系统，例如：基于非对称的密钥，不适合 RFID 系统。因此，面向 RFID 的安全协议比那些在有线环境或者其他无线环境中的应用存在更多的缺陷。更糟糕的是，RFID 标签的普遍部署环境是开放、不可预知的。一个攻击者可以部署复杂攻击来识破这个协议，提高了 RFID 系统的威胁性。

为了建立 RFID 系统的安全模型，一些学者要么提出关于隐私和安全的形式

化的定义，要么是关于攻击者的形式化定义。然而，由于现存在的模型是高层和粗粒度的，设计和分析 RFID 安全协议仍面临着巨大的挑战。根据先前的模型，很难探测到 RFID 协议确切的缺陷或者基于这些缺陷的攻击。此外，RFID 协议的安全和攻击的影响不能被定量分析。事实上，RFID 协议的安全分析仍只停留在初级水平上。因此，为安全协议开发自动化分析和标记管理方法对于提高 RFID 系统的安全性是相当重要的。在这篇文章中，我们提出了随机图论的方法协同细粒度的攻击者模型来自动分析标记 RFID 安全协议。我们的工作主要来源于五个方面。

第一点，我们提出了一个更为灵活的攻击者模型，细粒度，现实结构。我们形式化攻击者的一组原子操作。通过一些原子操作，攻击者的任何攻击可以被准确的表示。与之前的模型有所不同，我们的模型所定义的原子操作可以更加精准的仿真现实世界中的攻击者。

第二点，为了描述一个攻击者是如何与一个标签或单独一个阅读器产生联系的，我们并没有生成一个单独的状态传输图来表示标签和阅读器之间的关系。我们提出了一个新颖的随机图论方法，叫做 VAGs（脆弱性感知图），这个方法用到了一对图：标签图（TG）和阅读器图（RG），分别用来表示攻击者与标签以及攻击者和阅读器之间的联系。VAGs 探测安全协议的潜在缺陷和分析对手的攻击模式。通过利用 VAGs，一个攻击可以在标签图或者阅读器，图中被映射成一条有害路径。我们设定一组规则定义有害路径。每一条路径代表一个有缺陷的协议。每一条有害路径代表攻击者的一个动作。因此，通过找出有害路径，我们可以派生出攻击模式。

第三，攻击者几乎不能利用极其微小的缺陷。可以计算出攻击者能探测和使用缺陷的可能性，即，他或她能沿着代表缺陷的路径的概率，我们进一步分析了攻击者可以利用流量和定量攻击的概率。

第四，被动的攻击者不会干扰标签和阅读器之间的通信，他仅仅只给标签的不可辨别性带来了威胁，一个被动的攻击者的 VAGs 被缩减为一对图像，说明了一对阅读器和标签之间的联系。我们提出另一个基于联合熵的方法来测量被动攻击下，标签的不可辨别性。

最后，我们实现了 RFID 安全协议的整体基准测试方法。

本文其他部分的组织如下：在第二部分我们讨论了相关工作，第三部分介绍了预备知识，第四部分是攻击模型，第五部分形式化定义了这个问题，VAGs 方法在第六部分详细进行了介绍。第七部分讨论了在被动攻击下测量标签的不可区分性。第八部分展示了案例研究，第九部分结束我们的工作。

2、相关工作

在本节中，我们将简要介绍以前与我们工作密切相关的研究，包括 RFID 安全模型，基于图的网络安全分析方法，RFID 基准测试和 RFID 协议的安全评估。表 1 从提出安全模型，安全基准测试，自动协议分析，定量分析和攻击防范类型等几个方面说明了现有工作与提出的工作的不同之处。

2.1 RFID 安全模型

一些研究人员正致力于建模 RFID 系统的安全或隐私[7-11]。A. Juels 等人 [9] 定义了用于 RFID 系统基本分析的 RFID 系统的强大的私密性。协议是否是强大的私密性可以手动分析。但是，该定义没有提及如何自动分析协议。X.Zhang 等人 [7]描述了一些安全要求，包括标签数据的隐私性，所有权的隐私性，标签数据的完整性以及 RFID 系统中标签身份的可用性。然后正式确定这些要求的定义。I. Damgard 等人 [10]通过指定她可以采取的行动来模拟对手的行为，但是敌手模型对于表征现实对手是粗粒度和不灵活的。G. Avoine [11]也引入了适用于 RFID 环境的手对手模型。对手的形式化只能用于分析协议的可追溯性。S.Vaudenay [8]基于文献[9]提出了 RFID 系统匿名识别的模型和定义。它考虑了 RFID 系统中的标签持有相关密钥的情况，并讨论了安全与效率的权衡。

2.2 基于图形的网络安全分析方法

开发了场景图[12]和攻击图[13]（这是一种特定类型的情景图）来评估网络系统的安全性。利用攻击图和生成攻击图分析网络漏洞已有很多工作[13-16]。攻击图可以显示网络中不同主机上的漏洞之间的关系。它考虑到了由主机之间的交互激活的漏洞之间的相互关系，因为不同主机上的多个漏洞可以一起用来实现某些攻击对象。但是原始攻击图不能充分表达敌手在 RFID 系统中可能使用的关系。在 RFID 系统中，攻击者不仅可以利用系统中标签秘密之间的关系（实际上，在许多 RFID 系统中，对考虑强隐私来说标签是独立的[8]，我们的工作主要集中在

这样的 RFID 系统），更重要的是攻击者可以利用在同一标签上的几次协议执行之间的关系，而这些关系不能用攻击图来描述。

2.3 RFID 基准测试

还有一些关于 RFID 基准测试的现有档案[17,18]，但它们主要关注 MAC 或物理层标签的性能和可靠性。它们旨在评估周围物体的影响，阅读器的力量，标签与阅读器之间的距离，或标签在标签性能方面的物理取向，如读取速率。

2.4 RFID 协议的安全性评估

M. Alizadeh 等人 分析了 RFID 应用中使用的几种轻量级加密算法的安全性（就混淆和扩散而言）。作者只关注对保密标签的被动攻击[27]。一些研究人员对最近提出的轻量级协议进行了安全和隐私分析，并讨论了它们的优点和安全问题[28,29]。Changshe Ma 等人证明 ind 隐私弱于 unpr 隐私。RFID 标签的必要和充分条件是实现 unpr 隐私性[30]。伪随机函数族是对 RFID 标签的计算能力的最低要求，用于执行强大的 RFID 系统隐私。

表 1 现有工作与协议 VAGs 之间的差异。

| 成就 | 安全模型 | 安全基准测试 | 自动协议分析 | 定量分析 | 攻击（被动或主动） |
|---------|------|--------|--------|------|-----------|
| [7-11] | 是 | 否 | 否 | 否 | 两者 |
| [17,18] | 否 | 否 | 否 | 是 | 无 |
| [27] | 否 | 是 | 否 | 是 | 主动 |
| [28,29] | 否 | 是 | 否 | 否 | 两者 |
| [30] | 是 | 是 | 否 | 否 | 两者 |
| VAGs | 是 | 是 | 是 | 是 | 两者 |

3、预备知识

3.1 RFID 系统

RFID 标签分为两个类型：被动和主动标签，他们之间主要的区别是被动标签没有内部能量，但主动标签有，被动标签使用阅读器的射频波作为能量的提供。它因此无法承受复杂的需要相对高功率工作的微处理器。为了降低生产成本，被动标签的存储能力也受到了限制（目前的被动标签一般有千字节的存储）。由于

以上硬件的限制,被动标签存在的安全风险是相当高的,被动标签的安全协议比主动标签也存在更多的缺陷。在这篇文章中,我们关注被动标签的安全问题,我们的方法可以协调的应用主动标签的协议。

在 RFID 系统中,阅读器是用来询问 RFID 标签的一个设备。对于被动标签,阅读器负责激活标签的微芯片。阅读器没有存储关于标签或者系统的敏感信息。在获得标签的响应后,它简单的传输结果到后台服务器用于进一步加工。一般的,阅读器和后台服务器的信道认为是安全的。标签和阅读器的通信时通过两个开放的未加密的无线信道,即阅读器-标签(正向通道)和标签-阅读器(反向信道)通信。正向通道的通信距离比反向通道大的多,因为阅读器的传输能量比那些标签要大的多。相应的,攻击者在正向信道的窃听距离也比反向信道大的多。后端服务器处理的系统中处理信息,与标签相关,同时也有强大的计算资源。因此,它被认为是一个安全可信的实体,无法被攻击者盗用。这篇文章中,我们把阅读器视为 RFID 阅读器和后台服务器相结合的一个单元。

被动标签和阅读器之间的通信采用问答机制。阅读器先发起询问,在收到回应后,标签计算出结果,即响应,然后发回给阅读器。在阅读器和标签中安全协议的执行也是基于问答机制的。一个成功的协议的执行叫做会话。在这篇文章中,我们将 m 定义为标签和阅读器的消息传输, s_i 是 RFID 安全协议的第 i 个会话,标签 j 是系统中的第 j 个标签。RFID 标签和阅读器的自动问答机制一个安全隐患,可以被攻击者轻易获取未经授权的 RFID 数据。因此,RFID 安全和隐私越来越重要了。

3.2 RFID 系统的安全问题

一个标签显示了被连接的对象类型,甚至唯一的标识该对象。因此,标签携带的信息与对象所有者的机密信息高度相关。考虑一个攻击者目的在于打破一个 RFID 系统包括隐私,安全和功能。我们的形式化定义如下。

定义 1 (隐私): RFID 系统的隐私问题来源于两个方面,一个是带 RFID 标签的数据泄漏,另一个是通过跟踪标签 ID 用户行为跟踪或者身份识别。

在确保隐私的 RFID 系统中,需要考虑匿名和不可追踪性。匿名指一个标签的响应与 ID 和标签没有联系。不可追踪性需要 A 不能将一个目标标签相应于其他标签区分开(否则 A 可以追踪这个标签)。事实上,匿名性和不可跟踪性满

是不可区分性，因此这篇文章中我们关注不可区分性。

为了确保不可区分性，传输标签的信息应该互不相同或者相关，从不合适的当事人的角度来说，这是有必要的。不可区分性的两个最重要的威胁就是跟踪和活动表。

定义 2（安全）：安全关注与 RFID 系统的标签仿造问题，意思是 A 不能欺骗阅读器，即，让阅读器接收伪造标签的响应。（这个定义是与[10]中的安全定义一致的）。

安全性值得是确保标签传输的信息没有在响应式进程中或者之后被篡改。它表明数据没有被非法修改或者销毁。RFID 响应式进程的安全可以通过转播，重播，信息重建，数据修改/插入攻击达到折衷。

定义 3（功能性）：功能性指所有 RFID 系统提供的所有功能可以被正确地执行。对于 RFID 系统来说，它有功能保证，攻击者不能干扰它的功能，例如，让一个合法的读卡机拒绝合法标签，反之亦然。

功能性要求当需要时信息可用。RFID 系统为了展示功能性，它必须妥善的运作计算机系统，安全控制和通信信道。拒绝服务攻击是最威胁功能性的方式之一，由于他们可以被轻易的部署，却很难抵御。攻击的类型包括被动衰减的 RF 信号，和主动干扰和中断通信。De-synchronous 攻击时另一个具有挑战的威胁，它会造成阅读器拒绝合法标签。

关注以上问题，我们考虑的与传统的 CIA(保密性，完整性和可用性)，需求类似，隐私性对应保密性，安全性对应于完整性，功能性对应可用性。

4、攻击模型

在这部分，我们讨论攻击者的行为，构建几个攻击者可能创建的原子操作。然后我们可以对攻击者在应答式进程中，如何在标签和阅读器中进行干预建立模型。

攻击者 A 是一个实体，它尝试阻碍安全性和隐私性，或禁用系统功能。我们假设攻击者知道 RFID 系统采用的安全方案，即安全协议的规范。A 不知道标签和阅读器（后台服务器）之间的秘密共享，除非他或她破解了标签。A 可以通过一些物理攻击破解标签，但是这些物理攻击时有破坏性的，这就意味着标签在破

解后已经失效了。由于标签和阅读器的无线通信是开放的，A 可以获取部分或完整的消息，可以从正向信道或是反向信道，这依赖于 A 的能力。

A 可以发出各种攻击，例如窃取拦截合法阅读器和标签直接的通信，流氓扫描包括一个恶意的阅读器，该阅读器可以在没有主人允许的情况下读取标签，跟踪，截取，物理攻击，欺骗（复制，交换，转播，重播等），还有 DOS 攻击。

尽管这些攻击的目的和形式多样，每个攻击的行为可以被解析为一个由原子操作构成的有限集。换句话说，任何攻击都有一系列的原子操作构成。我们可以把那些操作看作一些预言（每个预言可以看作是黑盒理论，这可以解决某些决策问题），A 可以访问。

定义 4(原子操作):一组攻击者可以访问的操作。

• $\text{SendToReader}(m, s_i)$: 这个操作形式化定义了一个模型，在一个通过反向信道协议的会话 s_i 中，A 发送消息 m 到阅读器。

• $\text{SendToTag}(m, \text{tag}_i, s_i)$: 这个操作形式化定义一个模型，通过正向信道的协议的会话中，A 发送消息 m 到标签 j 。

• $\text{ReceiveFromTag}(m, \text{tag}_i, s_i)$: 这个操作形式化定义一个模型，在反向信道的协议的会话中，A 收到来自 tag_i 的消息 m 。

• $\text{ReceiveFromReader}(m, s_i)$: 这个操作定义一个模型，在正向信道协议的会话中，A 收到来自阅读器的消息 m 。

• $\text{Corrupt}(\text{tag}_j)$: 这个操作形式化定义一个模型，A compromising tag_j A 能获取 tag_j 当前的秘密，让 tag 失效，提高所有的原子操作，与标签 j 一致的都不能再用了。

• $\text{SideChannel}(\text{tag}_j, s_i)$: 这个操作定义了一个模型，在协议的会话 s_i 中，A 发现 tag_j 的结果。

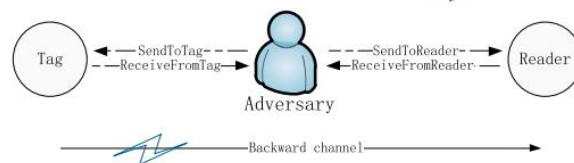


Fig. 1. The adversary serves as a part of RFID system.

图 1.对手是 RFID 系统的一部分。

A 的性能是定义 4 中定义的操作的一个子集。现在我们阐述 A 的性能以及在我们的模型中 A 是如何干扰 RFID 的。我们把 A 看作是类似于传统的网络中的路由器的中继转发。

- 通过访问定义 4 中的 **SendTo***和 **ReceiveFrom***操作，A 能控制所有存在于标签和阅读器（如图一）的通信，决定信息是否是可信，虚假或丢弃（即被拦截）的转播。

- SendToReader** 和 **SendToTag** 操作的通道用来发送消息，该消息遵从阅读器和标签的协议，**ReceiveFromReader** 和 **ReceiveFromTag** 代表 A 可以两个信道的传输收到消息。**ReceiveFrom*** 表示，A 不仅能窃听在信道传输的消息，也能完全控制这些消息。例如，拦截一条消息代表 A 用 **ReceiveFrom** 的通道，而忽略 **SendTo***。

- 通过访问 **Corrupt**，A 可以获取标签中当前的秘密，并通过泄露的秘密和历史信息的观察计算出相应的信息。

- SideChannel** 允许 A 在给定的会话的获取协议的结果，即接受或者拒绝一个标签。

尤其，上述这些操作不仅可以建立主动也可建立被动攻击者模型，它只监听标签和阅读器之间的通信，并没有以任何方式修改消息流。对于一个被动攻击者来说，他(她)的动作应该遵循以下规则：

- 攻击者不能 access **Corrupt** 动作

- 在会话 si 中，**ReveiveFromReader** (m, si) 和 **SendToTag** ($m, tag\ j, si$)需要成对出现。此外，**ReveiveFromReader** (m, si)也是遵循 **SendToTag** ($m, tag\ j, si$)（类似

的, $\text{ReveiveFromTag}(m, \text{tag } j, si)$ 也伴随着 $\text{SendToReader}(m, si)$), 这意味着攻击者如实的传递消息 m 。

以上的 oracle 设计展示了我们模型的优点, 例如灵活性, 细粒度结构, 对真实攻击者能力准确的定义。例如, 与现存的攻击者模型不同的是, 我们的模型解决了把 A 和阅读器 (或者标签) 的联系变成操作 ReceiveFromReader (或者 ReceiveFromTag)和 SendToReader (或者 SendToTag), 以代替将整个联系看作一个单一的动作。类似的, 我们应该把前面用到的 ReceiveFrom* 动作拆分为两个更细粒度的动作, ReceiveFromReader 和 ReceiveFromTag 。因此, 能访问 ReceiveFromReader 动作的攻击者也许不能访问 ReceiveFromTag , 这与现实情况更为接近。这是因为在现实的 RFID 系统中, 窃听范围比扫描范围相对大一些。Tag-to-reader 的窃听范围与 reader-to-tag 的也不相同 (前者比后者小)。从另一方面, 阅读器与标签也不总是在收到消息后响应, 这主要有协议决定。为了反应那些不完全的响应, 我们也需要对动作 A 进行更灵活的定义。

5、问题的制定

在本节中, 我们首先描述状态转换模型, 分别表示阅读器和 A, 标签和 A 之间的交互。这对状态转换模型描述了 A 如何介入标签和阅读器之间的通信。它还描述了标签或阅读器的状态转换如何在 A 发起的操作下进行。然后我们介绍 VAGs 的正式定义, 这是一种随机图形方法, 用于 RFID 安全协议的自动分析和基准测试。

5.1 状态转换模型的形式化

我们使用统一的转换系统 $M = (S, I, R, E)$ 对由 A 的动作强制形成的标签和阅读器的状态转换进行建模, 其中:

- a) $S = \{s_1, s_2, \dots, s_n\}$: 一组状态, $S \neq \emptyset$
- b) I : 初始状态
- c) $R \subseteq S \times S$: 一组转换
- d) $E \subseteq S$: 一组最终状态

我们将标签和阅读器视为彼此的对等体。特别的， $M_T = (S_T, I_T, R_T, E_T)$ 是具有状态集合 $S_T = \{s_1^t, s_2^t, \dots, s_n^t\}$ 的标签状态转换模型。 $M_R = \{S_R, I_R, R_R, \Phi\}$ 是状态集合 $S_R = \{s_1^r, s_2^r, \dots, s_n^r\}$ 的阅读器状态转换模型。接下来，我们专注于 M_T 和 M_R 的细节。

- 状态：阅读器（或标签）的状态由其内部状态和外部输出定义。具体来说，标签和阅读器的状态可以写成如下所示：

$$s_i^r = (s_i^r \cdot ins, s_i^r \cdot chlg)$$

$$s_i^t = (s_i^t \cdot ins, s_i^t \cdot resp)$$

- 内部状态 (Ins): 内部状态包括一组变量，这些变量是只存储在标签（或阅读器）中或与其对应方共享的秘密。秘密可以是计数器，会话 ID，密钥等。注意，尽管在标签或阅读器中存储有不同类型的信息，但对于特定的协议，只有与状态转换相关的信息应该被关注。

- 外部输出：阅读器的外部输出是发送给标签 (Chlg) 的挑战，而标签的外部输出是对挑战 (Resp) 的响应。

- $s_i^t \cdot resp$ (Resp): Resp 表示 s_i^t 状态下的标签返回的响应。如果 A 访问 SendToTag，那么状态 s_i^t 的标签可以通过计算新的 Resp 为 $s_j^t \cdot resp$ 而转移到新的状态 s_j^t ，同时将 $s_i^t \cdot ins$ 更新为 $s_j^t \cdot ins$ 。A 可以通过访问 ReceiveFromTag 来检索 $s_j^t \cdot resp$ 。

- $s_i^r \cdot chlg$ (Chlg): Chlg 表示一个由阅读器在 s_i^r 生成的挑战。Chlg 可以成为读者在会议中主动发起的第一条消息。对于涉及多个挑战 - 响应交互的协议，Chlg 也可以根据 A 访问的 SendToReader 中的消息 m 生成。因此，作为 SendToReader 的反应，通过准备 $s_j^r \cdot chlg$ 和将 $s_i^r \cdot ins$ 更新为 $s_j^r \cdot ins$ ， s_i^r 上的阅读器转移到新的状态 s_j^r 。A 通过访问 ReceiveFromReader 接管 Chlg。

- 初始状态 (Ini): 在交互之前，Ini 代表了一个具有原始内部条件的特殊

状态。

- $R(s_i, s_j)$ 表示从状态 s_i 到 s_j 的一个转换。在我们的模型中，如果 A 可以采取 SendToTag 或 Corrupt 动作触发标签从 s_i 到 s_j 的状态转换，则存在 $R_T(s_i, s_j)$ 。由于 M_T 和 M_R 反映了标签和阅读器之间的交互作用，因此代表标签模型中状态的任何标签响应都必须对应于阅读器模型的至少一次转换。另一方面，代表阅读器模型中状态的阅读器挑战必须对应于标签模型的至少一次转换。

- E_T 代表由一次 Corrupt 行为引起的标签的结束状态。 E_T 中的标签不能再执行功能。因为 A 永远不会损害阅读器，阅读器不会有这样的状态。

定义 5（状态空间的大小）。模型 $M = (S, I, R, E)$ 的大小定义为 $|S|$ ，即其状态空间的大小。

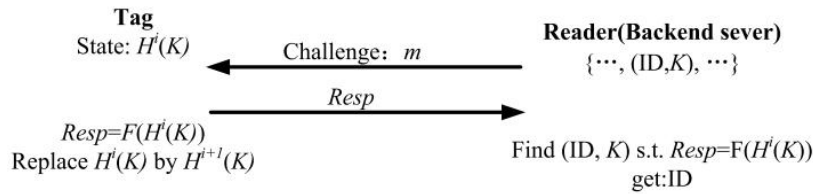


Fig. 2. OSK protocol.

图 2.OSK 协议

5.2 脆弱性感知图

尽管 M_T 和 M_R 描述了 RFID 系统的每个组件如何在攻击者的行为下响应和更新其状态，但他们仍然不便表达攻击程序。而且，仅用 M_T 和 M_R 很难找到所有潜在的攻击。因此，我们开发了一个随机基于图的方法，称为脆弱性感知图（VAGs）。VAGs 从三个方面明确描述了状态在 M_T 和 M_R 中的相关性：一个会话内一对挑战 and 响应的因果关系，A 可能使用的几个会话的相关性，标签和阅读器之间的交互在 A 的存在下。

有很多非等价的方法来定义随机图。在本文中，我们采用[21]中使用的定义：随机图的形成是一个随机过程，它在描述 VAGs 的生成方面具有优势。此外，我们通过允许吊索和有向边来扩展随机图的标准模型。

VAGs 包含一对有向随机图，一个反映标签与对手之间的交互作用，称为 Tag Graph (TG)，另一个反映读者与被称为 Reader Graph (RG) 的对手之间的交互。这两个图是根据它们的状态转换模型构建的。TG 和 RG 中的顶点分别表示 M_T 和 M_R 中的状态。图中的边表示由 SendTo *或 A 由采取 Crroupt 动作引起的状态转换。每个可能的边都以一定概率出现在随机图中。每个图都有一个没有前导的起始顶点，代表初始状态 I 。

具体而言，对于 $M_T = (S_T, I_T, R_T, E_T)$ ，每个 $s_i^t \in S_T$ 对应于 TG 上的一个顶点。 $R_T(s_i^t, s_j^t)$ 对应于从顶点 s_i^t 开始并在 s_j^t 结束的有向边。 $e_t \in E_T$ 是一个没有从它开始的边的顶点。对于 $M_R = \{S_R, I_R, R_R, \Phi\}$ ，每个 $s_i^r \in S_R$ 对应于 RG 上的顶点， $R_R(s_i^r, s_j^r)$ 对应于从顶点 s_i^r 开始到 s_j^r 结束的有向边。

M_T 和 M_R 可以构建为离散时间步长模型以分别创建 TG 和 RG。VAGs 的形成如下：每个图从一个起始顶点 (I) 开始。在每个时间步中，添加从现有顶点集开始的所有可能边，然后生成尚未创建的边的末端所有对应顶点作为它们的后继。然后这些顶点标记为<内部状态，外部输出>，并添加到现有顶点集中。迭代继续，直到不能添加新的边缘。两个顶点之间的每个有向边表示 A 为 A 进行状态转换所采取的动作。图中出现边的概率等于 A 可以采取相应动作的概率。

定义 6（路径）。表示过渡系统 $M = (S, I, R, E)$ 的 VAGs 中的路径 π 是有限或无限的转换状态序列 $R(s_i, s_{i+1}) \in R$ ，该序列连接了一系列状态 $\{s_0, \dots, s_i, \dots, s_k\}, s_i \in S, 0 \leq i \leq k$ 。

VAGs 的目标是自动分析和测试 RFID 安全协议，检测协议的缺陷并指出潜在的攻击。标签（或阅读器）和 A 之间的任何交互系列可以被视为 TG (RG) 上的路径。

我们以著名的 OSK 协议[20]的 VAGs 作为例子。OSK 协议如图 2 所示。 $H(\cdot)$ 和 $F(\cdot)$ 都是散列函数。最初,标签具有初始信息 K ,后端服务器存储所有 (ID, K) 对,对于系统中的每个标签都是不同的。标签第一次收到来自阅读器的挑战 m ,

它计算 $F(K)$ ，其中 K 是它当前的秘密。然后它将 $RespF(K)$ 发送给阅读器，并更新其秘密 $K = H(K)$ 。每次遇到挑战时，标签都会重复上述操作。

例如，在 I 会话之后，阅读器发送第 $(i+1)$ 个挑战。在接收到 m 之后，标签使用其当前秘密 $H^i(K)$ （即 $H^i(K) = H(H(\dots H_i(K)))$ ）计算 $F(\cdot)$ 。然后，标签发送 $Resp$ ，即 $F(H^i(K))$ 到读取器，并将其秘密更新为 $H^{i+1}(K)$ 。阅读器从标签接收 $Resp$ ，并为每个 (ID, K) 对计算 $F(H^i(K))$ 。如果 $F(H^i(K)) = RespF(H^i(K)) = Resp$ ，标签将被接受。请注意，OSK 提出了标签操作时间步数的固定上限 n 。在第 n 次询问之后，标签产生随机输出，这是阅读器不能接受的。

6、脆弱性感知图

我们分别考虑被动和主动的对手。对于积极的敌手 A ，我们从不可区分性，安全性和功能性角度对 RFID 安全协议的安全属性进行基准测试。从图表的角度来看，构成攻击的 A 的行为可以映射到 TG 或 RG 上的有害路径。与传统的攻击图或模型不同，我们将安全属性的违规视为某些显示有害特征的路径，而不是检查路径是否会达到某些不安全状态。

在本节中，我们将介绍我们的基准测试方法，该方法可检测 RFID 协议的潜在缺陷并定义 VAGs 上的有害路径。我们还讨论如何量化 A 可以利用这个缺陷发起攻击的概率。对于被动攻击者来说，它只能对标签的不可分辨性构成威胁。我们在下一节讨论被动攻击的基准测试方法。

6.1 不可分辨性

不可区分性仅与 TG 有关，因为它只关注标签的反应与对手的挑战之间的关系。在 TG 上，可以观察到 RFID 安全方案的不可区分性取决于 TG 上是否存在可区分的路径。如果路径上的挑战 and 响应之间的关系可以被 A 认为是一个特征，那么存在一条可区分的路径。在这里，我们只给出一些可能被攻击者利用的流行特征。实际上，使用我们的模型可以找到更多的特征。

- 自循环：TG 上的自循环表明存在 $R_T(s'_i, s'_i)$ 。这意味着 A 可以采取一些行动使标签返回相同的响应。

- 循环：TG 上的循环表示存在一系列连续的 $R_T(s'_i, s'_j)$ 。以相同的状态开始和结束。这意味着 A 可以采取一些行动来使标签定期返回相同的响应。

定义 7（区分路径）。TG 上的路径，其上的外部输出（Resps）显示一些特定的特征，例如自循环或循环。

如果在 TG 上存在可区分的路径，则 A 可以通过执行一系列攻击行为来沿着路径行走，并因此可以观察特征。然后 A 可以根据观察结果区分标签和其他标签。虽然特征可能是多方面的，但必须注意的是，如果在 TG 上存在这样的特征，那么 A 应该做的唯一事情就是找到具有特征的路径。由于不同的挑战会导致 TG 上不同路径所表示的不同响应，为了观察路径的特征， A 应该在与标签的每次交互中决定挑战（SendToTag 中的消息 m ），以便标签的响应在根据路径上每个标签状态的响应。因此，我们得出以下结论：

命题 1. A 可以将标签与他人区分的能力被定义为他或她可以在 TG 上沿着可区分路径选择正确挑战的概率。

6.2 安全

安全意味着 A 不能欺骗阅读器，或者让阅读器接受来自虚构标签的回应。如果读者接受合法（不妥协）的标签，但标签与阅读器没有匹配的交互，则意味着 A 破坏了 RFID 系统的安全性。匹配的交互（协议的成功执行）意味着阅读器和标签交换的挑战和响应交织良好，忠实（但可能有一些时间延迟）在会话执行过程中[10]。我们提出以下定义来描述如果标签和阅读器处于正确执行协议的状态。

定义 8（半匹配状态）。假设状态 s_t 在 TG 上且状态 s_r 在 RG 上，如果 $s_t \cdot resp$ 与从 RG 上的 s_r 开始的转换有关，或者 $s_r \cdot chlg$ 与指向 s_t 标记图上的转换有关，则这两个状态是半匹配状态，并表示为 $\langle s_t \cdot resp, s_r \rangle$ 或 $\langle s_t \cdot chlg, s_r \rangle$ 。如果至少有一个上述关系是真的，我们说 s_t 和 s_r 之间存在一个半匹配。

定义 9（匹配状态）.假设 TG 上的状态 s_t 满足 $\langle s_r \cdot \text{chlg}, s_t \rangle$ ，并且 RG 上的 s_r 满足 $\langle s_t \cdot \text{resp}, s_r \rangle$ 。这两个状态被定义为匹配状态。我们说 s_t 和 s_r 之间存在匹配。匹配状态的阅读器和标签意味着它们之间的交互对协议是合法的。

定义 10（不匹配状态）.假设 TG 上的状态 s_t 和 RG 上的 s_r 。如果 $s_t \cdot \text{resp}$ 与指向 s_r 上的任何转换不相关， $s_r \cdot \text{chlg}$ 与 TG 上指向 s_t 的任何转换都不相关，则 s_t 和 s_r 被命名为不匹配状态。如果标签和阅读器处于这种状态，它们不应该在协议的同一个会话中。我们说 s_t 和 s_r 是不匹配的。

在 A 发起攻击之前，一对阅读器和标签处于特定的匹配状态。 A 使读者通过伪造合法标签的响应来接受虚构标签。协议的成功执行可以被映射到在 VAGs 上具有 $\langle s_t \cdot \text{resp}, s_r \rangle$ ， $\langle s_r \cdot \text{chlg}, s_t \rangle$ 的几对匹配状态。一般来说，有两种方式来伪造回应。首先， A 记录由合法标签生成的响应，并在以后直接重播。其次，根据她对系统的了解，猜测一条与真实信息相同的信息。为了获得足够的信息来应对挑战， A 应找到一条包含 TG 所需信息 A 的路径。

命题 2.给定一个匹配状态 $\langle s_t, s_r \rangle$ ，如果 A 可以从状态 s_t 找到与 s_r 有半匹配的 TG 上的一些路径，即 $\langle s_t \cdot \text{resp}, s_r \rangle$ 到状态 s_t ，则 A 可能伪造 s_t 的响应，并使假冒的回复被阅读器接受。

6.3 功能性

功能意味着协议执行的正确性。由于阅读器功能强大，并且放置在某个安全的地方，因此 A 无法将其分解。然而， A 会干扰 RFID 系统的功能，例如，使读者拒绝合法标签或阻止标签识别合法挑战。这意味着 A 能够在 TG 和 RG 上执行一系列操作，以便标签无法识别读者，反之亦然，在他们的当前状态。

命题 3.攻击从一对匹配状态开始。如果 A 可以在 TG 或 RG 上找到路径，以使路径的最终状态为不匹配状态，则在最终状态下功能会崩溃。

有两种方法故意使标签对阅读器无法识别。一种是在 TG 上从当前半匹配状态 $\langle s_t \cdot \text{resp}, s_r \rangle$ 找到一些路径到另一个与 s_r 不匹配的状态 s_t 。另一种方法是在 RG 上找到从 $\langle s_t \cdot \text{resp}, s_r \rangle$ 到状态 s_r 的一些路径，这是与 $s_t \cdot \text{resp}$ 不匹配的状态。然后，

A 尝试采取沿路径上的有向边表示的一系列动作。如果标签和阅读器之间没有交互发生, 阅读器的状态不会改变, 而标签的状态通过 A 被驱动到路径的目标状态。

A 标签拒绝阅读器识别的策略与上述两种方法类似。不同之处在于, 在当前状态 $\langle s_r \cdot \text{chlg}, s_t \rangle$ 时, A 应找到一个路径, 其结束状态与 RG 上的 $s_r \cdot \text{chlg}$ 不匹配或与 TG 上的状态 s_t 不匹配。

6.4 量

如上所述, 隐私, 安全和功能方面的缺陷都可以被视为有害的途径。因此, 在我们的定量分析中, 我们只考虑所有有害路径的统计数据, 没有通过缺陷类型来区分路径。但是, 我们的测量也可以用来分析由某种路径表示的某种缺陷。我们定义了三个用于评估 RFID 系统安全协议的安全因素:

- 缺陷数量 (fn): 由 VAGs 上的有害路径的数量指示。
- 缺陷的不可感知性 (fi): 缺陷可以被 A 利用的概率。
- 缺陷率 (for): 在 VAGs 中有害路径上的状态总数。

必须强调的是, 所有这三个因素一起决定了协议的安全性。应全面考虑每个因素对方案安全性的影响。由于所有有害路径都可以位于 VAG 上, 因此我们可以很容易计算出 fn 和 fr 。在下面, 我们讨论如何计算路径的 fi :

A 所利用的缺陷意味着 A 能够以高概率沿着代表缺陷的有害路径行走。 A 可以通过沿着 VAG 上的有害路径执行一系列操作来阻碍协议的安全性的概率是:

$$C_{\text{path}:s_i \rightarrow s_j} = \prod_{s_i \rightarrow s_p \in s_i \rightarrow s_j} P_r(R_{\text{VAGs}}(s_i, s_p)) \quad (1)$$

$\text{Pr}(R_{\text{VAGs}}(s_i, s_p))$ 是 A 驱动 VAGs 上从 s_i 到 s_p 的状态转换的概率。它由两个条件概率确定: 可以从 s_i 过渡到 s_p 的行動的概率, 以及 A 可以执行行動的概率。 A 可以执行状态转换的行動的概率定义为 A 可以选择相关的正确消息 m 的概率。

$$\text{Pr}(R_{\text{VAGs}}(s_i, s_p)) = \sum_m \text{Pr}(s_i \rightarrow s_p | \text{STT}(m)) \times \text{Pr}(m | \text{RFT}, \text{RFR}) \quad (2)$$

在 A 使用消息 m 访问 STT 的条件下, $\text{Pr}(s_i \rightarrow s_p | \text{STT}(m))$ 是从 s_i 到 s_p 的状态转换概率。它可以从图表中获得一个关于从 s_i 开始的与 m 有关的转换的数量。

$\text{Pr}(m | \text{RFT}, \text{RFR})$ 是 A 选择正确的 m 并根据从以前的交互中接收到的信息使

用它来访问 Oracle SendToTag 的概率。A 可以通过访问 ReadFromTag (RFT) 和 ReadFromReader (RFR) 来获取信息，以窃听来自阅读器的挑战和来自标签的响应。请注意，A 获得的消息数量取决于对手的能力。在从标签和阅读器获得消息之后，A 执行该信息并有可能猜出所需的消息，这取决于先前在特定协议下由标签和阅读器发送的消息的相关性。如果消息之间有很强的相关性，则 A 很有可能猜出所需的消息。

我们给出一个例子来说明如何计算 $\Pr(R_{VAGs}(s_i, s_p))$ ，如图 4 所示。x, y 和 z 是 STT 中可能引发状态转换的三条消息。假设他们是独立的，A 不知道他们，所以 A 猜测每个消息的概率为 1/3。从状态 s_1 开始，x 可引起 $R_{VAGs}(s_1, s_2)$ 或 $R_{VAGs}(s_1, s_4)$ ，y 可引起 $R_{VAGs}(s_1, s_2)$ 或 $R_{VAGs}(s_1, s_4)$ ，并且 z 可引起 $R_{VAGs}(s_1, s_3)$ 或 $R_{VAGs}(s_1, s_4)$ 。我们计算 A 可以在状态 s_1 触发某个状态转换的概率如下：

$$\Pr(R_{VAGs}(s_1, s_2)) = \sum_m \Pr(s_1 \rightarrow s_2 | STT(m)) \times \Pr(m) = \frac{1}{3} \times \frac{1}{2} + \frac{1}{3} \times \frac{1}{2} = \frac{1}{3}$$

$$\Pr(R_{VAGs}(s_1, s_3)) = \sum_m \Pr(s_1 \rightarrow s_3 | STT(m)) \times \Pr(m) = \frac{1}{3} \times \frac{1}{2} = \frac{1}{6}$$

$$\Pr(R_{VAGs}(s_1, s_4)) = \sum_m \Pr(s_1 \rightarrow s_4 | STT(m)) \times \Pr(m) = \frac{1}{3} \times \frac{1}{2} + \frac{1}{3} \times \frac{1}{2} + \frac{1}{3} \times \frac{1}{2} = \frac{1}{2}$$

7、在被动攻击中测量标签的不可区分性

由于被动攻击者只是窃听在标签和阅读器间的正向信道和反向信道，这并不会干扰标签和阅读器之间的通信，这只会给标签的不可区分性带来干扰，在这部分，我们阐述了在被动攻击下，如何测量 RFID 标签的不可区分性。

尽管被动攻击可以通过使用定义 4 中的原子操作，被很好的形式化定义，被动攻击可以被 VAGs 准确表示，被动攻击者，本质上，并不采取任何行动迫使标签和阅读器应对假冒挑战。这意味着 VAGs 将退化成一对图，表示一对阅读器和标签之间的联系。因此，我们提出了另一个方便的方法来测量在被动攻击下标签的不可区分性。

从被动攻击者的角度来看，标签就被看作一个黑盒子，它只知道

challenge-responses 对，而并不知道它的内部状态。对任何 challenge-response 机制，我们首先测量了询问和响应的不确定性，和问答对的各自的相互依赖如下。

定义 11（不确定的挑战）：标签中，询问 C 的不确定性是用熵标记，记作 $H(C)$ ，

$$\text{如下： } H(C) = -\sum_{c \in \mathcal{C}} p(c) \log p(c)$$

这里 \mathcal{C} 是询问的定义域， c 是其中一个询问， $p(c)$ 是 c 出现的可能性。 $H(C)$ 可以被用来测量攻击者猜测询问的可能性。通过伪造一个合法的询问，攻击者可以获得一个特定的响应。

定义 12（响应的不确定性）：响应 R 的不确定性用熵表示，记作 $H(R)$ ，如下：

$$H(R) = -\sum_{r \in \mathcal{R}} p(r) \log p(r)$$

\mathcal{R} 是响应的定义域， r 是其中一个响应， $p(r)$ 代表 r 出现的概率。 $H(R)$ 可以被用来测量攻击者猜测到这个响应的可能性。通过假装一个合法的相应，攻击者可以假冒阅读器。

定义 13(应答对的相互依赖)：应答对 (C, R) 的相互依赖是他们之间相互传递的消息。用 $I(C, R)$ 表示，如下：

$$\sum_{r \in \mathcal{R}} \sum_{c \in \mathcal{C}} p(c, r) \frac{\log p(c, r)}{p(c)p(r)}$$

$p(c, r)$ 是 c 和 r 的联合概率分布。 $I(C, R)$ 用于测量 how much a certain challenge tells us about the corresponding response。即，如果我们知道了一个询问，我们能减少多少响应的熵？

于是给了一组标签和阅读器，我们测量标签的不可区分性如下：

定义 14（不可区分性）：我们使用联合熵测量问答对 (C, R) 的不可区分性，记作 $H(C, R)$ ，如下：

$$H(C, R) = -\sum_{c \in \mathcal{C}} \sum_{r \in \mathcal{R}} p(c, r) \log p(c, r)$$

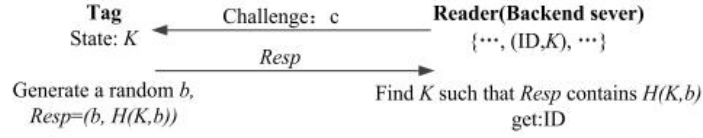


Fig. 5. WRSE's randomized hash-lock protocol.

图 5.WRSE 的随机散列锁协议

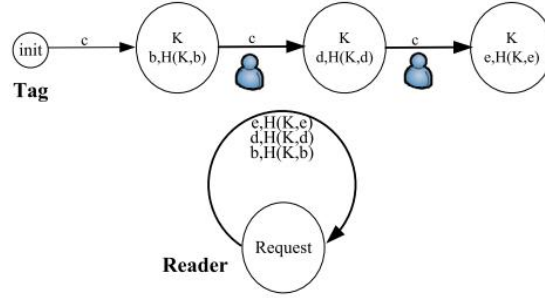


Fig. 6. VAGs of WRSE.

图 6.WRSE 的 VAGs

我们使用联合熵测量在 RFID 系统中熵含多少，这利用了应答机制。C 和 R 描述相关事件，系统的总熵如下：

$$H(C, R) = H(C) + H(R) - I(C, R)$$

通过测量一个特定标签的应答对，我们可以评估该标签的不可区分性。在任何 RFID 系统中，对于一个标签和阅读器对， $H(C, R)$ 越大，代表标签的不可区分性越大。

8、基准案例研究

在本节中，我们首先将基准方法应用于 WRSE 的随机哈希锁定方案[]和 OSK 协议的变体[8,22]（VOSK），以显示我们方法的正确性和有效性。我们生成 WRSE 和 VOSK 的 VAGs 并呈现我们的分析。安全参数 k 与状态空间的大小有关。对于 RFID 系统中使用的实际协议， k 应该足够大以保证对手不能通过暴力攻击耗尽所有状态空间。随着 k 的增加，VAGs 的产生将遭遇状态爆炸问题。在本文中，我们的目标不是开发解决状态爆炸问题的方法，而是我们可以使用现有的抽象技术进行模型检查[23,24]，以生成包含一组较小状态的抽象模型，同时保留原始图的安全属性。我们还可以使用即时技术[25,26]来避免以极小的概率产



生可能发生的路径。

8.1 WRSE 的随机散列锁协议

我们在图 5 中说明了 WRSE 的随机散列锁 (WRSE [31]). $H(\cdot)$ 是一个散列函数。每个标签具有秘密信息 K ，后端服务器存储所有 (ID, K) 对，对于系统中的每个标签都是不同的。阅读器发送一个持续的挑战 c 。在接收到 c 后，标签选择一个随机的 k 位随机数，并计算 $H(K, \text{随机数})$ 。然后它将 Resp 作为 $(\text{nonce}, H(K, \text{nonce}))$ 发送给阅读器。阅读器在所有 (ID, K) 对中搜索 K ，使得 r_0 包含 $H(K, \text{nonce})$ 。如果存在这样的 K 并且拒绝，则阅读器接受标签。

为了给 WRSE 提供一个直观的印象，我们首先给出一个 WRSE 的 VAGs，其状态空间非常小，如图 6 所示。我们设置 nonce 从 $X = \{b, d, e\}$ 中选择。在上述参数的设置下，我们描述了 WRSE 的所有状态和转换。

在图 6 的 TG 上，我们使用粗体箭头线来表示 A 可以找到的路径。 A 通过访问 RFR 获取 RG 上的挑战 c ，然后访问 STT (c) 重复查询标签。因此， A 可以强制标签将其状态从 s_1 发送到 s_2 和 s_3 ，并发起欺骗攻击来破坏系统的安全；也就是说， A 可以访问 STR 来欺骗阅读器接受假标签。

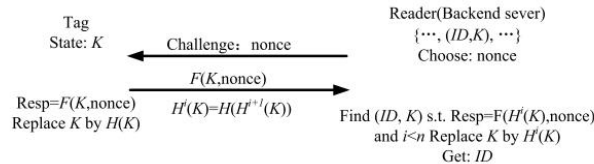


Fig. 7. A variant of OSK protocol.

图 7. OSK 协议的变体

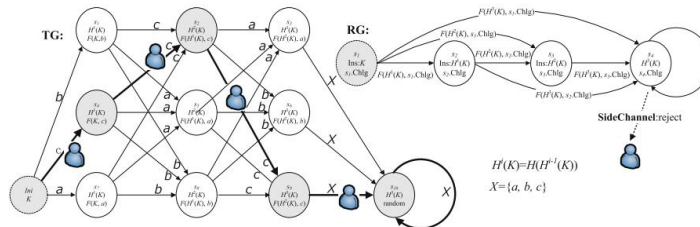


Fig. 8. VAGs of VOSK.

图 8.VOSK 的 VAGs

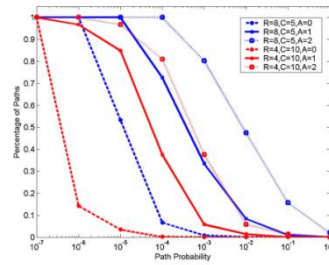


Fig. 9. A variant of OSK protocol.

图 9.OSK 协议的变体

8.2 OSK 协议的变体

我们举例说明图 7 中的 VOSK，它与原始的 OSK 有两个不同。首先，阅读器用随机数（一个随机的 α 位串）发送挑战。其次，在后端服务器接受标签之后，阅读器更新当前 K。

为了给 VAGs 提供一个直观的印象，我们首先在图 8 中给出一个具有相当小的状态空间的 VOS 的 VAGs。我们将 n 设置为 4，并且随机数从 $X = \{a, b, c\}$ 中选择。在上述参数的设置下，我们描述了 VOSK 的所有状态和转换。

在图 8 的 TG 上，我们使用粗体箭头线来表示 A 可以找到的路径。 A 通过访问 RFR 获取 RG 上的 $s_1 \cdot \text{Chlg}$ ，然后访问 STT ($s_1 \cdot \text{Chlg}$) 来重复查询标签。因此， A 可以找到 $\text{Ini} \rightarrow s_4 \rightarrow s_2 \rightarrow s_9 \rightarrow s_{10}$ 的路径，并发起 DoS 攻击来破坏系统的功能。此外， A 可以访问 SideChannel 以了解标签被阅读器拒绝，这也是跟踪的一个可区分特征。

我们设计一个算法来生成 VAGs 并计算每个路径被 A 找到的概率。图 9 显示了由我们的算法生成的具有 50 个状态的 VOSK 的 TG。带有红色实线的路径表示由 A 定位的路径发起 DoS 攻击。

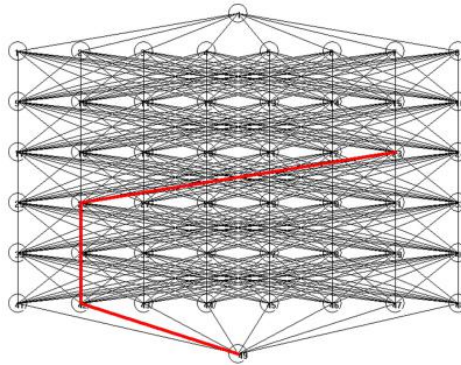


Fig. 10. VAG of VOSK generated automatically.

图 10.自动生成 VOSK 的 VAGs

在图 10 中, 计算对手定位每条路径的概率, 并且在改变状态空间的大小和 A 已有系统的知识数量时, 比较不同概率的路径数量。在图 8 中, R 与随机数 (α) 的长度有关, C 表示 n 的值, A 表示 A 采取的动作次数。可以看出, 在 VOSK 协议中, 一次攻击后, 攻击者发起攻击的概率大大增加。还可以看出, 在相同的状态空间下, 较大的 n 会导致 VOSK 协议的安全性较高。

9、总结

在本文中, 我们提出了一种基于随机图的方法来促进 RFID 安全协议的基准测试和定量评估。该方法包括一个用于表达攻击者能力的细粒度和准确模型, 以便攻击者发起的攻击可以被解析为一组原子动作。我们通过对标签和阅读器使用状态转换模型来形式化对手如何干预 RFID 系统。基于这些模型, 我们提出了一种基于随机图的方法 VAG, 分别反映对手与标签, 敌手和读者之间的相互作用。通过研究有害路径应该具备的特点, 我们使用 VAG 来利用 RFID 安全协议中的缺陷并分析攻击者的攻击模式。此外, 我们讨论对手可以成功发现缺陷并向 RFID 系统发起攻击的可能性。

