

Введение

В наше времена оборот информации в бизнес сфере огромен, отцифровано практически всё. Большая часть информации передаётся по средствам компьютерных сетей, в частности глобальной – Интернет.

Технология Интернет приспособленна для передачи любого вида закодированной информации, чем пользуется в преимуществе большая часть бизнес отраслей для получения своей прибыли. Но во время развития Интернет появилась такая ниша как Взломщики, люди посягающие на не санкционированное получения доступа к ресурсам подключенных к Интернет. Вследствии начали стремительно развиваться технологии Криптографии , которые существующие еще со времён древнего Рима, и других методов борьбы с Взломщиками и не только.

Взломщикам могут быть интересны любые ресурсы: журналы Бухгалтерии, системы управления предприятием и т.д.

Подходя к теме о ведении бизнеса в сфере ЧОП многие подозрительные личности могут быть заинтересованн в получении запланированных маршрутов инкосации, адреса жительства сотрудников... Для пресечения перечисленных выше махинаций можно полностью отказаться от ведения своей деятельности в Интернет.

Если это не выход – тогда необходимо развертывание системы защиты, чем мы и займемся в пределах данной работы. Также самым современным решением будет использование Blockchain – децентрализованный метод хранения данных, но данный вариант не будет рассматриваться, так как всё сильно усложняет и, скорее всего, по просту не приемлем.

Решение будет выполненно в несколько шагов:

1. Анализ структуры ЧОП.
2. Анализ проходимых процессов в ЧОП.
3. Описание средств разработки.
4. Составление базы данных ЧОП.
5. Разработка архитектуры приложения для взаимодействия с ЧОП.
6. Реализация архитектурных решений
7. Написание исходных кодов.
8. Тестирование.

Теоритическая часть

Организация ЧОП

Охранное предприятие занимается охраной какой либо частной или государственной собственности.

ЧОП, чаще всего состоит из трех подразделений:

- Бухгалтерия
- Дирекция(она же администрация)
- Отдел кадров
- Отдел охраны
- Отдел инкосации
- Отдел вооружения



Если говорить о предоставляемых услугах более конкретно, то ЧОП для потребителя предлагает:

- Охрана объекта
- Охрана ценных бумаг или металов(инкосация)

Информационные потоки ЧОП



Процессы внутри ЧОП





Описание средстд разработки

В данном разделе будут рассмотрены средства разработки, используемые мной при создании ПС для ЧОП.

Утилиты для разработки баз данных

SQLite v3.35.5 stable

 SQLite — компактная встраиваемая СУБД. Исходный код библиотеки передан в общественное достояние. Данная СУБД работает в безсерверной конфигурации. Если сравнивать с другими СУБД, то в равных условиях запись SQLite осуществляет медленее на 20-30% чем другие СУБД, но чтение превосходит другие на 40-50%. SQLite не имеет привелегий, только систему авторизации, но это и не нужно в моем проекте, об этом будет сказано позже.

 *Примичание: Библиотека SQLite не будет использована в чистом виде, а в составе Qt v6.2.4*

SQLiteBrowser v3.12.2


Удобный FOSS браузер баз данных SQLite, использованный для отладки.

DBVisualizer v12.1.8

Проприетарная утилита для работы с разными СУБД, использован для генерации графа таблиц составленной базы данных ЧОП.

Библиотеки

Qt v6.2.4

 Qt - один из самых популярных и больших фреймворков c++ на рынке. Важная характеристика Qt - переносимость, т.к. я работаю на Linux.

Системы сборки и компиляторы

CMake v3.22.3

 Система сборки c++. Сборка проекта и передача более низкоуровневому средству.

Ninja c1.10.2

Еще одна система сборки, только уже более низкого уровня, чем CMake. Передача исходного кода на компиляцию.

clang v13.0.1

Компилятор семейства C. Без комментариев.

IDE

QtCreator v6.0.2

QtCreator logo IDE от компании The Qt Company, использованный только как средство отладки и создания скелетов форм пользовательского интерфейса.

VIM v8.2

VIM logo Моя любимая IDE, в своей основе так же прост как и каноничный "блокнот" в Windows, только с максимальной степенью кастомизации. Главное достоинство, по моему мнению, - это управление без использования мыши и возможность настройки управляющих комбинаций максимально удобно, что сокращает время на бесполезное перемещение рук по рабочему пространству.

Плагины

vim Plug 'https://github.com/xolox/vim-misc' " auto load Plug 'https://github.com/xolox/vim-session' " session manager Plug 'wakatime/vim-wakatime' " wakatime.com Plug 'SirVer/ultisnips' " snippets Plug

Разное

Plantuml v1.2021.16

Средство создания UML диаграм. Использовано для визуализации объектов и просецсов.

BASH v5.1.16

Bourne Again Shell - интерпритатор, использован для автоматизации некоторых процессов.

Примечание: Оглавления данной работы было автоматически сгенерированно данным bash-скриптом: `""bash cat "$1" > "$1".indexed index`

i=(-1 1) prevLen=0 while read -r line; do hash="\$(md5sum <<< "\$line" | cut -d ' ' -f 1)" printf "\n%s\n" "\$hash" "\$line" > tmp sed "/\$line/{ x r /home/xewii/Documents/TIT/ZXC/tmp }" "\$1".indexed > "\$1".indexed.tmp mv "\$1".indexed.tmp "\$1".indexed hdrLen=\$(awk -F# '{print NF-1}' <<< "\$line") hdrTxt=\$(echo "\${line//#/}") ((\$hdrLen > 1)) && for ((j=1; j<\$hdrLen*4; j++)); do printf ' '; done ((\$prevLen < \$hdrLen)) && i[\$hdrLen]=1 printf "%d.%s\n" \${i[\$hdrLen]} "\$hdrTxt" "\$hash" prevLen=\$hdrLen let i[\$hdrLen]++ done <<< "\$(grep --color=no -E "^#+\$" "\$1")" > index

mv "\$1".indexed tmp printf "# Содержание\n" > "\$1".indexed cat index >> "\$1".indexed cat tmp >> "\$1".indexed

rm tmp rm index

...

Отладка

Valgrind v3.18

Утилита профилирования и отладки программы, использовано в основном для обнаружения утечек памяти.

VCS

GIT v2.35.1

GIT - система контроля версий, сомо о себе говрит. Использовался в основном для перенесения кода между машинами и как средство дистрибьюции.

Разработки базы данных ЧОП

База данных ЧОП в пределах данной работы - головная сущность, вокруг которой будет строится весь функционал. База данных будет существовать под управлением SQLite.

Таблицы

В компанию, как известно, входит некоторое количество сотрудников, по этому, я создаю таблицу Users. Название выбрано таковым, потому что она будет содержать данные учетных записей сотрудников и клиентов ЧОП.

sql CREATE TABLE "Users"("id" INTEGER NOT NULL UNIQUE, "name" TEXT NOT NULL, "entryDate" TEXT NOT NULL, "role_id" INTEGER NOT NULL, "wapon_id" INTEGER, "email" TEXT UNIQUE, "login" TEXT NOT NULL UNIQUE, Таблица содержит данные для идентификации:

- login
- password
- salt

Пароль не храниться в открытов виде, а зашифрован с использованием динамической соли по алгоритму "Prefered salt algorithm", более подробно будет рассмотрен далее.

Как видно, таблица Users зависит от таблиц Roles и Wapons, собственно вот они:

sql CREATE TABLE "Roles"("id" INTEGER NOT NULL UNIQUE, "name" TEXT NOT NULL UNIQUE, "commands_id" INTEGER NOT NULL, "payMultiplier" DECIMAL(10, 3) NOT NULL, "payPeriod" INTEGER NOT NULL, FOREIGN KEY("com Роль определяет какие данные и соответственно команды можешь выполнять на сервере. Ссылается на таблицу roleComands - это SQL массив с ID команд, которые может выполнять пользователь с данной ролью, по этому я и отказался от других, более тяжелых СУБД, т.к. все необходимые действия делегируются на Сервер, что будет рассмотрено далее, от СУБД требуется только хранить данные и извлекать их. Связанная таблица roleCommands: sql CREATE TABLE "roleCommands"("role_id" INTEGER NOT NULL, "command_id" INTEGER NOT NULL, FOREIGN KEY("role_id") REFERENCES "Roles"("id")ON DELETE RESTRICT) Таблица Wapons: sql CREATE TABLE "Wapons"("id" INTEGER NOT NULL UNIQUE, "employee_id" INTEGER UNIQUE, "name" TEXT NOT NULL, "ammo" INTEGER NOT NULL, "price" DECIMAL(10, 3) NOT NULL, "ammoPrice" DECIMAL(10, 3) NOT NULL, Каждая запись в таблице Wapons - это единица зарегистрированного оружия, в полной мере описывающая необходимые характеристики для ЧОП.

Так как организация имеет свои расходы и доходы, нам нужно сохранять эти данные. Таблица Accounting:

Запись в таблице Contracts это сделка вида, описанного в связанной таблице objectType. Таблица objectType: sql CREATE TABLE "objectType" ("id" INTEGER NOT NULL UNIQUE, "name" TEXT NOT NULL UNIQUE, "price" DECIMAL(10, 3) NOT NULL, PRIMARY KEY("id")) Запись в данной таблице описывает объект контракта, где указывается базавая цена за период оплаты(payPeriod) исполнителя/исполнителей контракта(assignedEmployees_id). Для привязки нескольких сотрудников, была создана еще одна таблица AssignedEmployees, являющейся массивом. sql CREATE TABLE "AssignedEmployees" ("id" INTEGER NOT NULL, "employee_id" INTEGER NOT NULL, "guiltyPercent" DECIMAL(10, 3) NOT NULL, "usedAmmo" INTEGER, FOREIGN KEY("employee_id") REFERENCES "Users"("id") ON DELETE RESTRICT) Для создания контракта в данной таблице нужно только 2 поля - id, employee_id. Так так во время исполнения может произойти какой то ицидент, то была создана таблица Accidents: sql CREATE TABLE "Accidents" ("id" INTEGER NOT NULL UNIQUE, "name" TEXT NOT NULL, "contract_id" INTEGER NOT NULL, "usedAmmoCount" INTEGER, "damagePrice" DECIMAL(10, 3), "assignedEmployees_id" INTEGER, F Описание происшествя, произошедшие во время исполнения контракта. Если у нас есть контракты, описывающие некую деятельность с учетом выходных и рамок начала и окончания службы, то можно было бы ускорить вычисление рабочего времени по дням с помощью препроцессинга данных из записи Contracts. Таблицей, в которую сохраняются транслированные данные является - DutySchedule: sql CREATE TABLE "DutySchedule" ("employee_id" INTEGER NOT NULL, "day" INTEGER NOT NULL FOREIGN KEY("emploee_id") REFERENCES "Users"("id") ON DELETE RESTRICT) day - это 64х битная цифра со знаком в формате UNIX time(secs since epoch).



Разработка архитектуры приложения

Приложение для взаимодействия с моделью ЧОП, построенной ранее, должно предоставлять функционал для реализации всех описанных процессов взаимодействия с моделью ЧОП.

Основа

Основной функцией приложения, как понятно из темы курсовой работы, будет обеспечение безопасности данных. Исходя из этого в голову приходит идея организовать клиент-серверную архитектуру приложения, но это не главная причина почему выбрана такая архитектура. Основная причина - необходимость принимать заказы от клиентов, предоставить им функционал для удобного взаимодействия с персоналом ЧОП, но по большей части он будет взаимодействовать с клавиатурой. И для персонала ЧОП тоже будет намного удобнее и быстрее использовать унифицированные методы итерации с базой данных и самой организацией.

Информационные потоки ЧОП с учетом сервера



Обращение к базе данных

Система безопасности

Система идентификации и авторизации

Система команд

Клиент-Сервер

Протокол

Клиент

Сервер

Пользовательский интерфейс

Реализация приложения

Пользовательский интерфейс

PagesManager

NotifyManager

Заключение

Вот и сказочки конец, кто дочитал - тот молодец.

Список литиратуры

1. <https://wiki.qt.io> - документация Qt
2. <https://www.sqlite.org/> - документация SQLite