Предварительная версия текста

**Вступление**

Добрый день

**Слайд 2. Актуальность, цель работы.**

Всё более широкое применение начинают приобретать автономные роботы. Например, можно уже сейчас увидеть на дорогах роботов-доставщиков, производящих доставку еды и не только. Кроме доставки разных товаров существует большое количество ещё не автоматизированных задач, которые можно делегировать роботам.

Целью нашей работы является разработка автономной системы управления роботами для их навигации в закрытых помещениях. Разрабатываемая система должна обладать достаточной гибкостью для настройки под конкретную задачу и обеспечивать хорошую защиту от взлома.

**Слайд 3. Описание системы. КТС.**

Для достижения поставленной цели спроектирована информационная система, состоящая из камер видеонаблюдения, сервера и мобильных роботов, подключенных к серверу по Wi-Fi. Сервер обрабатывает видеопоток, поступающий с камер, находящихся на стенах помещения, распознает положение роботов и направляет им соответствующие команды.

В ходе разработки мы ориентировались на роботов на платформе Raspberry PI. Связь между сервером и роботами осуществляется по протоколу TCP/IP. Робот принимает команду, декодирует её и выполняет. Инновационным подходом в нашей работе является работа робота и сервера под отечественной кибериммунной операционной системой Kaspersky OS.

**Слайд 4. Почему именно KOS?**

Итак, почему же именно Kaspersky OS? В настоящее время остро стоит вопрос импортозамещения, поэтому большим плюсом KasperskyOS является то, что она разработана отечественной компанией «Лаборатория Касперского» и является очень молодой и современной операционной системой.

Она основана на “трех слонах” – архитектуре MILS, архитектуре FLASK и других дополнениях от команды Kaspersky. Благодаря этому она является достаточно защищенной от известных уязвимостей и хорошо подходит для решений в сфере IoT.

**Слайд 5. VFS? EDL?**

**Слайд 6. Alpha Bot (?). Отсылка команд по TCP/IP, потенциально MQTT.**

**Слайд 7. Структурная схема ПО.**

**Слайд 8. Диаграмма классов.**

**Слайд 9. Выводы.**

В результате работы представлен рабочий прототип системы, который демонстрирует возможность разработки защищенных управляющих приложений, использующих FLASK архитектуру для обеспечения контроля выполнения политики безопасности.

Также этот прототип возможно развивать далее, что позволит использовать его в учебных целях, используя вместо настоящей Raspberry PI – её симуляцию под QEMU.

**Заключение**

Спасибо за внимание, звоните пишите, любим 307.