Предварительная версия текста

**Вступление**

Добрый день, мы студенты Московского Авиационного Института, кафедры 307. (Представиться?). Сегодня мы расскажем вам о нашем проекте – разработке прототипа системы управления автономным роботом на базе Kaspersky OS.

**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 уже реализованных решений 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 сервера, камер видеонаблюдения, подключенных через Ethernet 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. Почему же мы выбрали именно её?

**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.**

На данном слайде представлено программное устройство робота. Каждый робот содержит в себе Command Receiver и некоторое количество моторов, которые приводят его в движение. Для более удобной отладки мы создали по шаблону проектирования Strategy 2 типа двигателей. Console Engine подразумевает то, что работа мотора будет отображена в виде вывода в консоль. GPIO Engine подразумевает отправку команды через GPIO на реальный мотор робота.

Аналогично для удобства отладки были реализованы классы CommandReceiver и мы можем переключаться между ручным вводом команд и принятие команд по TCP/IP.

**Слайд 9. Выводы.**

Подводя итоги, 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 и далее. Например это позволит использовать его в учебных целях, используя вместо настоящей Raspberry PI – её симуляцию под QEMU.

**Заключение**

На этом все. Спасибо за внимание. Мы готовы ответить на ваши вопросы.