Муниципальное бюджетное общеобразовательное учреждение

Лицей №153

Кировского района городского округа город Уфа

Республики Башкортостан

## 

Секция: «Радиотехники и компьютерных технологий»

## Безопасный умный дом

Авторы: Опякин Роман Сергеевич,

ученик 11 «Г» класса,

Сатаев Эмиль Робертович,

ученик 11 «Г» класса.

Руководитель:

Коряковцев Алишер Муродович, студент МГТУ им. Баумана

2017 г.

Содержание

**Введение3**

**Цель проекта3**

**Задачи проекта3**

**Глава 1. Обзор литературы4**

1.1 Основные понятия4

1.1.1 Умный дом4

1.1.2 Интернет вещей4

1.1.3 Raspberry Pi4

1.1.4 HTTPS4

1.2 Обзор современных Умных Домов5

1.2.1 Apple HomeKit5

1.2.2 Samsung Smart Home5

**Глава 2. Опыт создания прототипа безопасного Умного Дома6**

2.1 Средства разработки6

2.1.1 Сервер6

2.1.2 Периферийные устройства6

2.2 Реализация7

2.2.1 Периферийные устройства7

2.2.2 Сервер8

2.2.3 Веб-интерфейс9

2.2.4 Мобильное приложение10

**Выводы11**

**Список литературы11**

Введение

В Европе и мире, где уже развит рынок умных домов, остро стоит проблема взломов и атак на них. В связи с популяризацией умных домов в России, ожидается, что возникнет похожая проблема. Для решения этой проблемы за рубежом разрабатываются стандарты безопасного взаимодействия между устройствами интернета вещей. Аналогичные действия необходимы и в рамках российского рынка.

Цель проекта

Разработка и создание умного дома с повышенной информационной и физической безопасностью.

Задачи проекта

1. Создание защищенного сервера умного дома
2. Создание функциональных датчиков и устройств, взаимодействующих с сервером
3. Разработка веб-интерфейса и мобильного приложения для управления умным домом
4. Обеспечение информационной безопасности умного дома

**Глава 1. ОБЗОР ЛИТЕРАТУРЫ**

**1.1 Основные понятия**

1.1.1 Умный дом

Умный дом - система [высокотехнологичных](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%81%D0%BE%D0%BA%D0%B8%D0%B5_%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D0%B8) устройств в жилом [доме](https://ru.wikipedia.org/wiki/%D0%96%D0%B8%D0%BB%D0%B8%D1%89%D0%B5) современного типа, организованная для наиболее комфортного проживания или работы людей.

Под «умным» домом следует понимать систему, которая обеспечивает [безопасность](https://ru.wikipedia.org/wiki/%D0%91%D0%B5%D0%B7%D0%BE%D0%BF%D0%B0%D1%81%D0%BD%D0%BE%D1%81%D1%82%D1%8C) и [ресурсосбережение](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D1%81%D1%83%D1%80%D1%81%D0%BE%D1%81%D0%B1%D0%B5%D1%80%D0%B5%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5) (в том числе комфорт) для всех пользователей. В простейшем случае она должна уметь распознавать конкретные ситуации, происходящие в здании, и соответствующим образом на них реагировать: например, одна из систем может управлять поведением других по заранее выработанным [алгоритмам](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC).

1.1.2 Интернет вещей

Интернет вещей - концепция [вычислительной сети](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C) физических предметов («вещей»), оснащённых встроенными технологиями для взаимодействия друг с другом или с внешней средой, рассматривающая организацию таких сетей как явление, способное перестроить экономические и общественные процессы, исключающее из части действий и операций необходимость участия человека.

1.1.3 Raspberry Pi

Raspberry Pi - [одноплатный компьютер](https://ru.wikipedia.org/wiki/%D0%9E%D0%B4%D0%BD%D0%BE%D0%BF%D0%BB%D0%B0%D1%82%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80) размером с банковскую карту. Изначально разработанный как бюджетная система для обучения информатике, впоследствии получил намного более широкое применение и популярность, чем ожидали его авторы.

В нашем проекте мы используем Raspberry Pi 3 в качестве центрального сервера.

1.1.4 HTTPS

HTTPS (HyperText Transfer Protocol Secure) - расширение [протокола](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) [HTTP](https://ru.wikipedia.org/wiki/HTTP), для поддержки [шифрования](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) в целях повышения безопасности. Данные в протоколе HTTPS передаются поверх криптографических протоколов [SSL](https://ru.wikipedia.org/wiki/SSL) или [TLS](https://ru.wikipedia.org/wiki/TLS). В отличие от HTTP с TCP-портом 80, для HTTPS по умолчанию используется [TCP](https://ru.wikipedia.org/wiki/TCP)-[порт](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%80%D1%82_%28TCP/IP%29) 443.

[<https://ru.wikipedia.org/>]

**1.2 Обзор современных Умных домов**

1.2.1 Apple HomeKit

Apple HomeKit – платформа для объединения умных устройств.

Впервые Apple заговорила о HomeKit на конференции WWDC 2014, собственно тогда она и представила концепцию своей платформы. Все устройства с поддержкой HomeKit – безопасны, просты и работают с несколькими гаджетами Apple (в том числе iPhone и Apple TV).

HomeKit работает с любыми iPhone, iPad и iPod touch под управлением iOS 8.1 или выше. Так же можно управлять умными устройствами с помощью Apple Watch под управлением WatchOS2 и Apple TV (3-го поколения и выше) – для управления устройствами с помощью команд Siri, когда никого нет дома.

Все производители, которые планируют выпускать устройства с поддержкой HomeKit, должны принимать участие в программе сертификации Apple Made for iPhone (MFI).

Apple проверяет их API, написанные для устройств, для того чтобы удостовериться в их соответствии с требованиями App Store, и после этой проверки либо разрешает, либо запрещает их производство. Устройства, которые прошли эту сертификацию, имеют специальный MFI логотип на упаковке, только устройства с этим логотипом, могут считаться проверенными и безопасными.

Однако при достаточном количестве достоинств, система имеет довольно непривлекательную для большинства населения нашей страны цену.

[<https://geektimes.ru/post/280584/>]

1.2.2 Samsung Smart Home

Samsung Smart Home — это новый сервис, который позволяет объединять в домашнюю сеть smart-телевизоры, смартфоны, бытовую и цифровую технику и управлять ими с помощью единой интегрированной платформы.

С помощью сервиса Smart Home пользователь сможет управлять всей техникой в доме, используя лишь одно приложение, интегрированную платформу и сервер, которые объединяют в домашнюю сеть все устройства: от холодильников и стиральных машин до телевизоров, камер и смартфонов.

Три основные функции Samsung Smart Home: Управление устройствами (Device Control), Наблюдение за домом (Home View) и Пользовательский smart-сервис (Smart Customer Service) дадут возможность подключаться к устройствам в любое время и в любом месте.

Из минусов стоит отметить не очень стабильное мобильное приложение и отсутствие в открытом доступе результатов аудита на безопасность.

[<https://habrahabr.ru/company/samsung/blog/209786/>]

**Глава 2. Опыт создания прототипа безопасного Умного Дома**

**2.1 Средства разработки**

2.1.1 Сервер

***Django***

В качестве центрального веб-сервера, управляющего всем комплексом умных устройств мы использовали Django. Он прост в настройке, имеет качественную и подробную документацию, а также постоянно развивается.

***Nginx + gunicorn***

Для обеспечения безопасности на уровне сетевого взаимодействия нам необходимо было использовать защищенный протокол HTTPS. Но Django не умеет работать с этим протоколом. Поэтому мы используем Nginx в качестве прокси-сервера и gunicorn как WSGI HTTP сервер.

***SQLite3 + Django ORM***

Для хранения данных мы используем базу данных SQLite 3. Также возникла необходимость использовать ORM для работы с БД.

***Bootstrap + JQuery***

Для адаптации нашего веб-интерфейса под различные экраны мы использовали CSS-фреймворк Bootstrap. Для динамической связи с сервером мы используем технологию JQuery.

2.1.2 Периферийные устройства

***NodeMCU***

В качестве основного контроллера для датчиков выбран NodeMCU Devkit V1.0 запрограммированный с помощью Arduino IDE. Плата NodeMCU основана на WiFi-модуле ESP8266, что позволяет производить обмен данных по WiFi, а производительности достаточно для произведения необходимых математических операций. Также плата имеет один аналоговый вход, что позволяет считывать данные с термистора для получения температуры в помещении.

**2.2 Реализация**

2.2.1 Периферийные устройства



При подаче питания на NodeMCU плата подключается к WiFi сети сервера. Затем поднимается собственный сервер на ESP8266, который принимает запросы, зашифрованные алгоритмом AES256, от главного сервера (см. исходный код ниже).

server.on ("/set", request);

**void** request(**char** req)

{

aes256\_dec\_single(key, req);

**switch** (req)

{

**case** "lighton":

// включаем свет

**break**;

...

}

}

В зависимости от запроса контроллер выполняет определенные команды (например включает свет). Также NodeMCU может отправлять данные на сервер по протоколу HTTPS. Например устройство отправляет данные о температуре в помещении (см. исходный код ниже).

**void** temperature()

{

**double** temp;

temp = **log**(((10240000/analogRead(A0)) - 10000));

temp = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 \* temp \* temp ))\* temp );

temp = temp - 273.15;

HTTPClient http;

http.begin(main\_ip);

http.POST(**String**(temp));

http.writeToStream(&Serial);

http.end();

}

**2.2 Сервер**

Мы изучили возможности распространенных веб-серверов. Решили использовать Django. Этот фреймворк позволяет поднимать сервера одной командой из консоли, и при этом он очень гибок и удобен. Но он не умеет работать с защищенными протоколами. Нам необходим был прокси-сервер, обеспечивающий работу с протоколом HTTPS. Мы остановились на Nginx. Это веб-сервер, позволяющий проксировать пришедшие HTTPS пакеты на Django, в виде обычных HTTP-запросов. Для удобства связи этих двух компонент мы используем WSGI-сервер gunicorn.

Для аутентификации мы используем встроенный механизм авторизации пользователей Django, который использует механизм сессий. Это позволяет нам не беспокоиться о безопасности аутентификации.

Например, вот так выглядит проверка авторизованности:

**if** request.user.is\_authenticated():

# если авторизован

**else**:

# если не авторизован

Полный исходный код сервера выложен на Github:

<https://github.com/Sinketsu/smarthome-site>

Также на нашем сервере мы используем Django ORM для упрощения работы с базой данных. Этот механизм позволяет транслировать записи из БД в объекты языка Python.

Например сейчас, модель умной вещи выглядит так:

**class** Thing(models.Model):

**type** = models.CharField(max\_length = 50)

crypto\_key = models.CharField(max\_length = 40)

TYPE\_CONNECTION = (

('bl', 'BLUETOOTH'),

('wf', 'WIFI'),

('rd', 'RADIO\_433')

)

conn\_type = models.CharField(max\_length=2,

choices=TYPE\_CONNECTION,

default="wf")

ip\_addr = models.GenericIPAddressField()

port = models.IntegerField()

bluetooth\_name = models.CharField(max\_length = 40)

channel = models.IntegerField()

На сервере мы реализовали API для прозрачной работы с умными устройствами. Например, GET-запрос по адресу:

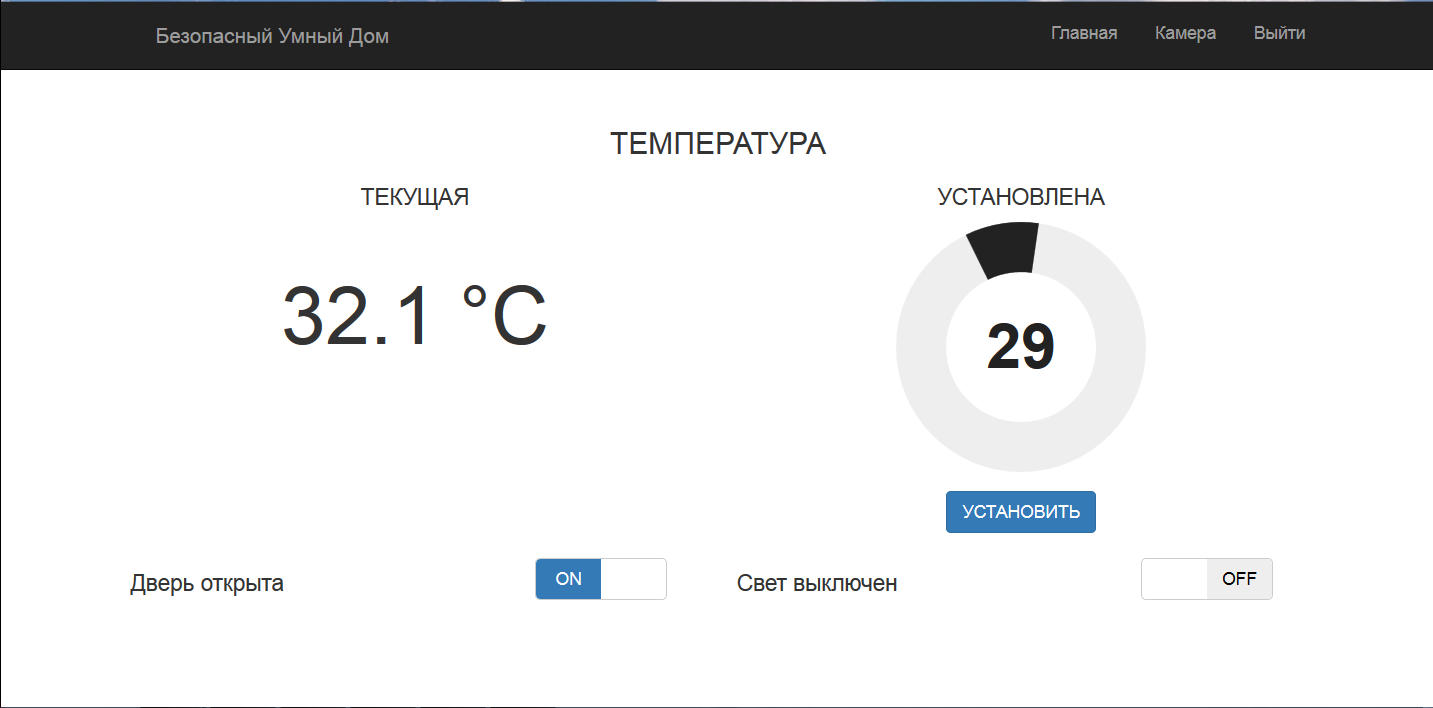
<https://ваш_ip/api/get/temperature> - вернет текущую температуру дома.

Также, мы используем open-source разработку mjpg-streamer для реализации видеонаблюдения в реальном времени.

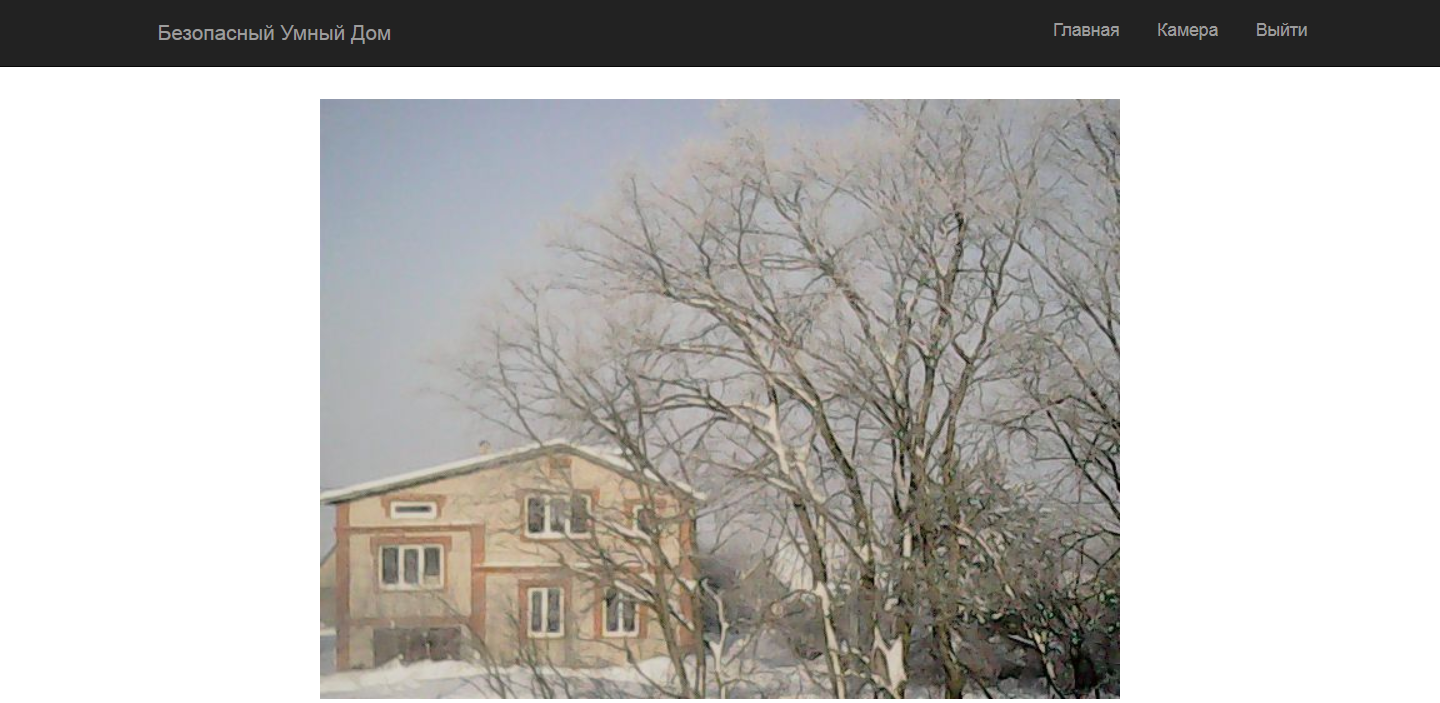
**2.3 Веб-интерфейс**

Для разработки веб-интерфейса нашего дома мы решили использовать CSS-фреймворк Bootstrap для адаптивной верстки под разные платформы. Для синхронизации с сервером у нас используются технологии JQuery и Django templates.

Сейчас наш сайт выглядит так:



Интерфейс видео-наблюдения:



**2.4 Мобильное приложение**

Для удобства использования умного дома, а также с целью возможности использования умного дома людьми с ограниченными возможностями было создано мобильное приложение с голосовым интерфейсом.

Приложение написано на Java, для распознавания речи был использован сервис Google Speech API.

Исходный код доступен на Github:

<https://github.com/Deltax2016/Qwert>

Внешний вид мобильного приложения:



**Выводы**

1. Разработан защищенный сервер умного дома.
2. Собраны и запрограммированы датчики и устройства
3. Разработаны удобный веб-интерфейс и мобильное приложение
4. Реализована информационная защита умного дома

**Список литературы**

* Брюс Шнайер «Прикладная криптография»
* Эндрю Таненбаум «Компьютерные сети»
* Vivek Ramachandran «[BackTrack 5 Wireless Penetration Testing Beginner's Guide](http://www.amazon.com/BackTrack-Wireless-Penetration-Testing-Beginners/dp/1849515581/ref=sr_1_1?s=books&ie=UTF8&qid=1461675276&sr=1-1&keywords=BackTrack+5+-+Wireless+Penetration+Testing)»
* Dafydd Stuttard and Marcus Pinto «[The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws](http://www.amazon.com/Web-Application-Hackers-Handbook-Exploiting/dp/1118026470)»

# [Collin Mulliner](http://www.allitebooks.com/author/collin-mulliner/), [Georg Wicherski](http://www.allitebooks.com/author/georg-wicherski/), [Joshua J. Drake](http://www.allitebooks.com/author/joshua-j-drake/), [Pau Oliva Fora](http://www.allitebooks.com/author/pau-oliva-fora/), [Stephen A. Ridley](http://www.allitebooks.com/author/stephen-a-ridley/), [Zach Lanier](http://www.allitebooks.com/author/zach-lanier/) «Android Hacker’s Handbook»

* Кевин Митник «Искусство обмана»

Также следующие интернет-ресурсы:

* <https://habrahabr.ru/>
* <https://geektimes.ru/>
* <https://tproger.ru/>

И доклады со следующих конференций:

* ZeroNights 2015
* ZeroNights 2014
* PHDays VI
* PHDays V