Smart Camera

Naveen Ravi, Pavankumar BL, Pranay Ponnappa, Anthony Vazhappilly
Computer Science
Umass Lowell
Lowell, United States of America

***Abstract*—We have designed a smart camera system for home security. It detects an intruder with the help of motion sensor. On sensing the motion, it triggers the camera to take a picture of the intruder and send a warning to the client along with the image captured.**

I. Introduction

Home security systems are generally connected via the cloud to a mobile device or the web for remote monitoring, and come with a variety of features such as motion detectors, door and window sensors and video cameras with recording capabilities. Although "the intent of these systems is to provide security and remote monitoring to a home" given the vulnerabilities, the owner of the home security system may not be the only one monitoring the home. This is the motivation and the driving force behind our project, to assure people that their house is secured while they are away. Our aim is to build a device that can provide a layer of security against intruders as well as be easy to manage by the client.

II. Smart Camera Design

A. Requirements

Hardware Requirements:

- Raspberry Pi Model B+
- PIR motion sensor
- Camera

Software Requirements:

- Raspbian Jessie Pixel
- Mosquitto MQTT Client/server /broker
- Paho Mqtt

B. Smart Camera:

The PIR motion sensor is connected to the Raspberry Pi GPIO ports via the breadboard . When the PIR sensor detects a motion/intruder, it will trigger the Camera to capture an image of the intruder and save it to a dedicated folder in the local machine (Raspberry Pi).

Once the image is captured, it is published to the mqtt broker which publishes it to all the devices subscribed to that topic. The Publisher will have access to the files stored in this folder and it will create a Topic, publish it over the secured tunnel created by the broker. If a subscriber needs access to the topic, then it would send a request to it via the broker. Once an intruder is detected it also send an alert via email to the user.

The image file that was published by the publisher was in the form of a long sequence of random code, we didn't know how to convert that back to image file nor how to save it to the subscriber. We overcame the problem by saving the payload to a file and named it using time stamp (to give unique file name to each file saved to the subscriber). Every image file transmitted over tunnel is secure.

C. Security Analysis:

IOT inherently comes with certain security flaws in regard with the communication, fortunately MQTT offers certain protocol such as ssl communication over port 8883 which we utilized in our project to obtain network encryption.

Security in MQTT is divided in multiple layers, each layer prevents different kind of attacks. The Ultimate aim of it is to provide a lightweight and easy to use communication protocol for the internet of things.

Network Level: Physically secured network or VPN as the foundation for any communication between clients and broker is one way to provide a secure and trustworthy connection. This would be suitable for gateway applications, where the gateway is connected to devices on the one hand with the broker over VPN on the other side.

Transport Level: TLS/SSL is being used for transport encryption. It provides a secure and proven way to make sure nobody can read along and even authenticate both sides, when using client certification authentication.

Application Level: On the transport level it can be ensured that the communication is encrypted and the identity is authenticated. The MQTT protocol provides a client identifier and username/password credentials, which can also be used to authenticate devices on the application level. The broker controls the authorization and what each device is allowed to do. It also provides payload encryption on the application level.

D. Security Evaluation

We have effectively secured our communications using the ssl protocol in mqtt. Each device that wants to access the broker has to provide a username and password to authenticate themselves. We ensure that the username and password is sent over an encrypted network to protect them against man in the middle attacks. The network is encrypted using client side and server side certifications. We have updated the mqtt config file to request any client for certificates and verify them before allow them to any topics on the broker.

E. Conclusions

We Implemented the Smart Camera system using Raspberry Pi, PIR sensor and a camera. This very step we have taken is just a beginning, the demand for smart cameras will steadily increase. Research interest, economic and social factors will drive continuous technological and product development. In future a more secured and reliable system can be developed with these logics as the base.

F. References:

[1] J. Dunmire, SSL/TLS Client Certs to Secure MQTT, 2016

[2] HuyITF, Configure SSL/TLS for MQTT broker mosquitto, Jun 2, 2016

[3] Primal Cortex, MQTT Mosquitto broker with SSL/TLS transport security, March 31, 2016

[4] http://lukse.lt/uzrasai/2015-02-internet-of-things-messaging-mqtt-1-installing-mosquitto-server/