IJAI (Indonesian Journal of Applied Informatics)
Vol. 7 No. 1 Tahun 2022 pISSN: 2548-3846, eISSN: 2598-5981

Analisis Performa Transmisi Data *Log* Berbasis IoT *Cloud* Pada Kunci Pintu Pintar Menggunakan Rekognisi Wajah

Dania Putri Nur'Aini*, Sri Lestari
Prodi Diploma IV Teknologi Rekayasa Internet Universitas Gadjah Mada
*Email: daniaputri99@mail.ugm.ac.id

Info Artikel

Kata Kunci :
Amazon Web Service (AWS);
Cloud;
ESP32-CAM;
Internet of Things (IoT);
Smart Door Lock

Keywords :
Amazon Web Service (AWS);
Cloud;
ESP32-CAM;
Internet of Things (IoT);
Smart Door Lock

Tanggal Artikel
Dikirim : 22 Juli 2022
Direvisi : 24 September 2022
Diterima : 30 November 2022

Abstrak

Data *logging* yang diterapkan pada suatu alat bertujuan untuk mencatat setiap kondisi yang terjadi. Oleh karena itu dibutuhkan sistem *log* yang bersifat *real time* dan fleksibel agar pengguna dapat melakukan monitoring perangkat dengan mudah. *Internet of Things* (IoT) yang merupakan suatu konsep dimana suatu sistem dapat terhubung dengan perangkat lain melalui jaringan internet yang terintegrasi dapat dikembangkan untuk memaksimalkan fungsionalitasnya dengan menggabungkan antara IoT dan *cloud*. Penelitian ini bertujuan melakukan penerapan IoT dengan membuat sistem keamanan kontrol kunci pintu rumah dengan sistem *log* berbasis *cloud*. Sistem keamanan kontrol kunci pintu rumah menggunakan keakuratan pengenalan biometrik berupa wajah sebagai verifikasi penghuni rumah. Ketika verifikasi wajah dilakukan maka akan dikirimkan *log* informasi berupa hasil dari verifikasi tersebut ke *database* dan Aplikasi Telegram menggunakan *service cloud*. Hasil Penelitian ini telah berhasil dalam merancang dan membangun sistem *log* berbasis *cloud* menggunakan *service* dari *Amazon Web Service* (AWS) berupa AWS IoT *Core* yang digunakan sebagai broker untuk meng-*handle* data *log* dari ESP32-*cam* ke DynamoDB dan Amazon SNS. Dalam melakukan penyimpanan data log didapatkan nilai *successful rate* sebesar 100% dan notifikasi berhasil dikirim ke Aplikasi Telegram dengan *delay* 2 hingga 3 detik. Pengujian performa pengiriman data didapatkan rata-rata *delay* sebesar 245,9374 mili detik dengan kategori Bagus, rata-rata nilai *throughput* 3.686 bps dengan kategori Sangat Bagus dan *packet loss* sebesar 0% dengan kategori Sangat Bagus berdasarkan standar TIPHON.

Abstract

Data logging applied to a device aims to record every condition that occurs. Therefore, a log system that is real time and flexible is needed so that users can monitor the device easily. IoT (Internet of things) which is a concept where a system can connect with other devices through an integrated internet network can be developed to maximize its functionality by combining IoT and Cloud. This research aims to implement IoT by creating a home door lock control security system with a cloud-based log system. To be able to send logs from IoT devices to the cloud requires good data transmission performance so that users can monitor home conditions. The home door lock control security system uses the accuracy of biometric recognition in the form of faces as verification of home occupants. When face verification is done, information logs will be sent in the form of the results of the verification to the database and Telegram application using cloud services. The results of this study have succeeded in designing and building a cloud-based log system using services from AWS (Amazon Web Service) in the form of AWS IoT Core which is used as a broker to handle log data from ESP32-cam to DynamoDB and Amazon SNS. In storing log data, the successful rate value is

100% and the notification is successfully sent to the Telegram application with a delay of 2 to 3 seconds. Testing the performance of data transmission obtained an average delay of 245.9374 milliseconds in the Good category, the average throughput value of 3,686 bps in the Very Good category and packet loss of 0% in the Very Good category based on the TIPHON standard.

1. PENDAHULUAN

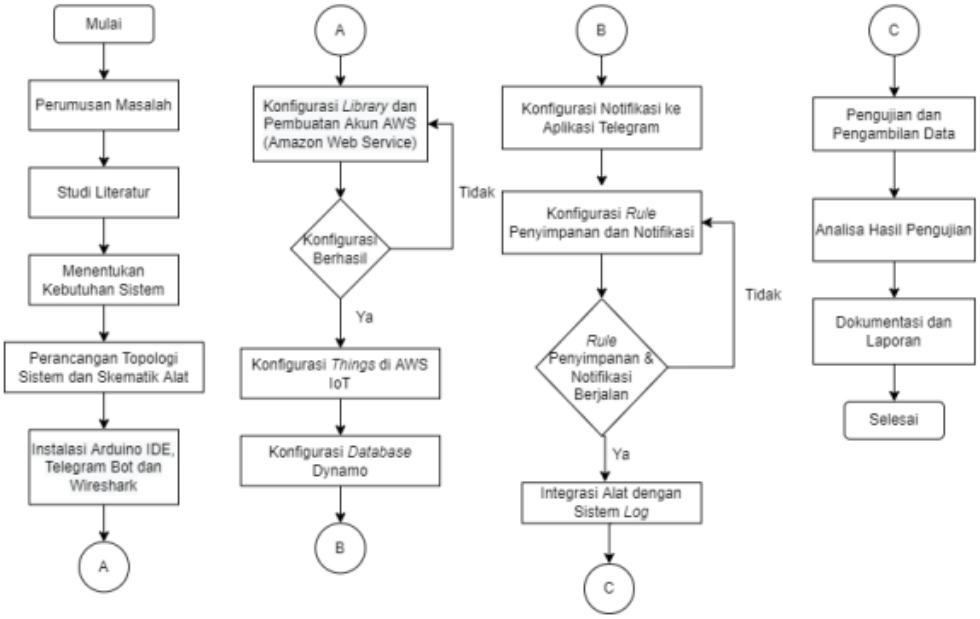
Berdasarkan data dari laman Dataku Bappeda Jogja kasus pencurian dari tahun 2020 hingga tahun 2021 di Daerah Istimewa Yogyakarta mengalami kenaikan sebanyak 1.131 menjadi 1.219 kasus [1]. Kasus pencurian dapat terjadi di luar maupun di dalam ruangan tertutup seperti di dalam rumah. Tindak kejahatan tersebut dapat terjadi kapan saja ketika anggota keluarga berada di dalam rumah maupun ketika rumah dalam keadaan kosong. Salah satu faktor yang dapat diantisipasi adalah dengan melakukan kontrol akses pintu keluar masuk secara *real time* dengan membuat data *logger*. Data *logger* yang diterapkan bertujuan untuk mencatat setiap kondisi yang terjadi. Oleh karena itu dibutuhkan sistem *log* yang bersifat *real time* dan fleksibel agar pengguna dapat melakukan *monitoring* pada kondisi riil dan dapat diakses dengan mudah. Sehingga anggota keluarga dapat mengetahui siapa saja orang yang masuk maupun yang mencoba berusaha masuk ke dalam rumah. Untuk mendapatkan kunci pintu rumah dengan sistem keamanan tersebut, saat ini sangat memungkinkan dengan berkembangnya teknologi *Internet of Things* (IoT).

Cloud computing merupakan model komputasi dimana sumber dayanya dijalankan sebagai layanan. Mulai dari *storage, network dan software* dapat diakses dimana saja selama terkoneksi dengan internet. IoT akan terintegrasi dengan *cloud* untuk menyimpan data *log* di *database* dan mengolahnya agar dapat tersampaikan ke pengguna lewat notifikasi ke Aplikasi Telegram. Penelitian ini bertujuan melakukan penerapan IoT dengan membuat sistem keamanan kontrol kunci pintu rumah dengan sistem *log* berbasis *cloud*. Untuk dapat mengirimkan *log* dari perangkat IoT menuju *cloud* diperlukan performa pengiriman data yang baik agar *user* dapat memantau kondisi rumah. Pada penelitian ini dibangun sistem *log* yang terintegrasi dengan *cloud* pada kunci pintu pintar menggunakan keakuratan pengenalan biometrik berupa wajah sebagai verifikasi penghuni rumah. Ketika verifikasi wajah dilakukan maka akan dikirimkan log informasi berupa hasil dari verifikasi tersebut ke *database* dan Aplikasi Telegram menggunakan *service cloud*. Pengujian kinerja pengiriman data *log* dilakukan dengan melakukan pengukuran nilai *delay, throughput,* dan *packet loss* ketika mengirimkan data *log* yang dilakukan dari perangkat ESP32-*cam* menuju AWS sebagai penyedia layanan *cloud*. Sehingga akan diketahui kualitas dari pengiriman data log ke AWS yang diaplikasikan dalam sistem *door lock* yang telah dibuat. Sistem ini diharapkan dapat meminimalkan pencurian dan pelanggaran dalam hal pembatasan akses dan mengurangi peran manusia dengan menggunakan sistem perpaduan antara IoT dan *cloud computing*.

2. METODE PENELITIAN

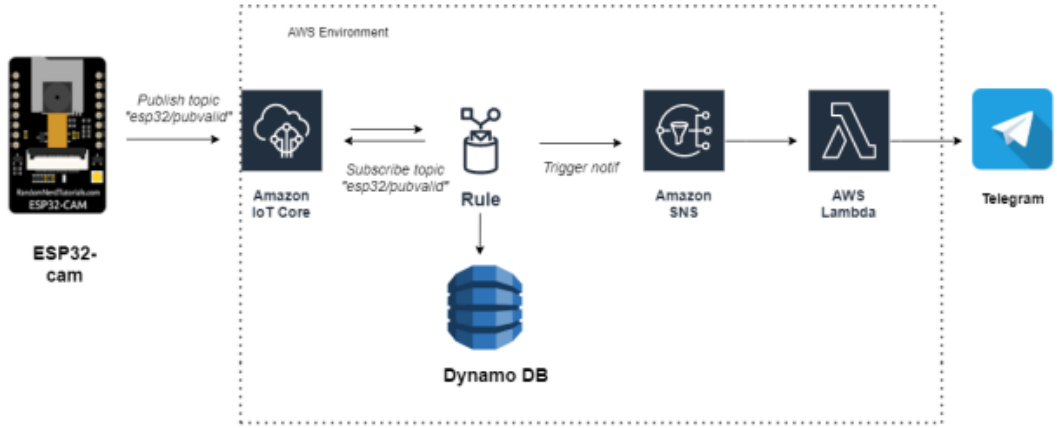
Tahapan dalam penelitian proyek akhir ini dilakukan dengan melakukan perumusan masalah pada sistem kunci pintu pintar, kemudian dari rumusan masalah tersebut dilakukan studi literatur berkaitan dengan kebutuhan sistem berupa teknologi dan skema yang akan diimplementasikan. Berdasarkan studi literatur maka dilakukan perancangan topologi sistem dan arsitektur perangkat yang digunakan. Tahapan selanjutnya melakukan instalasi perangkat lunak yang akan menunjang terlaksananya penelitian ini. Setelah semua terinstal dilakukan konfigurasi pada Arduino IDE dengan menambahkan *library* yang digunakan dan membuat Bot Telegram dan pembuatan akun AWS. Pada tahapan ini dilakukan percobaan untuk melakukan pengecekan apakah konfigurasi sudah dilakukan dengan benar sehingga dapat melanjutkan ke tahapan berikutnya. Selanjutnya dilakukan pembuatan things pada console AWS via *Graphic User Interface* (GUI) dan konfigurasi *database* Dynamo dengan membuat tabel yang akan digunakan untuk menyimpan *log*. Kemudian membuat fungsi *trigger* untuk mengirim log notifikasi ke Aplikasi Telegram pada *Simple Notification Service* (SNS). Masuk ke tahap berikutnya adalah membuat *rule* yang berfungsi merutekan data *log* agar tersimpan ke *database* dan terkirim ke Aplikasi Telegram. Kemudian akan dilakukan percobaan *rule* yang telah dibuat. Jika pesan dapat tersimpan ke *database* dan terkirim ke Telegram maka dilanjutkan ke tahapan berikutnya yaitu integrasi perangkat dengan sistem yang telah dibuat. Masuk ke tahapan selanjutnya yaitu pengujian dan pengambilan data dengan memanfaatkan metode analisis QoS untuk mengetahui performa pengiriman data *log* dari ESP32-cam

ke AWS IoT *Core*. Dokumentasi dan pembuatan laporan merupakan langkah terakhir pada penelitian proyek akhir ini. Peneliti menunjukkan diagram alir metode penelitian pada Gambar 1.



Gambar 1. Diagram Alir Penelitian

Pada Gambar 2 terdapat gambar ilustrasi sistem menggunakan protokol MQTT. ESP32-*cam* akan melakukan rekognisi pada wajah orang yang berusaha untuk membuka pintu, ketika wajah terdeteksi sebagai pemilik rumah ataupun tidak maka ESP32-*cam* mengirimkan *log* informasi ke Amazon IoT *Core* dengan metode *publish* dan *subscribe message*. Paket data akan diteruskan menuju AWS IoT Core yang bertindak sebagai broker. Melalui broker paket data akan diberikan identitas topik yang kemudian akan dilanjutkan ke *subscriber*. Pada penelitian ini *database* Dynamo dan aplikasi Telegram bertindak sebagai *subscriber*. Data *log* berupa notifikasi akan disimpan di dalam *database* yang dapat diakses pada AWS Console. Kemudian Telegram akan mendapatkan pesan dari AWS melalui *Simple Notification Service* pada AWS dengan fungsi yang dijalankan oleh AWS Lambda.



Gambar 2. Ilustrasi Sistem

Berikut merupakan landasan teori dari penelitian ini :

2.1 *Internet of Things*

Menurut [2] *Internet of Things* merupakan sebuah konsep yang bertujuan untuk memperluas manfaat dari konektivitas internet yang tersambung secara terus menerus. Hal ini memungkinkan adanya komunikasi antara komputer dengan peralatan elektronik untuk bertukar informasi di antara mereka sehingga mengurangi interaksi manusia [3]. IoT dapat menghubungkan perangkat sensor, aktuator, server secara jarak jauh secara *real-time*. Suatu

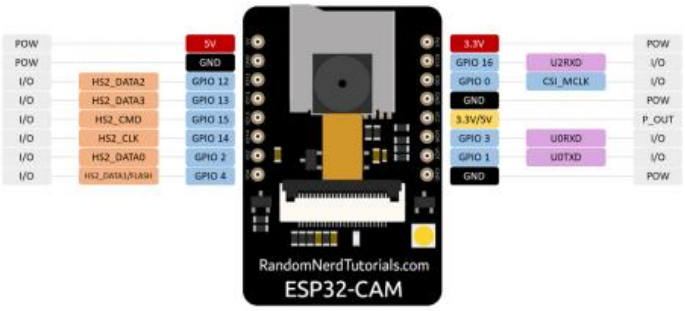
sistem keamanan yang dapat dipantau dan dikendalikan dari jarak jauh dapat dibuat dengan adanya teknologi IoT. Penggunaan IoT dapat diaplikasikan dalam *smart home, smart city, smart garden* dan sebagainya sesuai dengan kebutuhan. Prinsip kerja perangkat IoT adalah dengan memberikan pengenal berupa identitas unik pada perangkat sehingga dapat dikenali oleh sistem yang kemudian dapat direpresentasikan dalam bentuk data. Menurut [4] cara kerja dari IoT mengacu pada 3 elemen utama yaitu barang fisik yang dilengkapi dengan modul IoT, perangkat koneksi ke internet dan data center untuk menyimpan aplikasi beserta *database*.

2.2. *Face Recognition*

Terdapat beberapa jenis pemrosesan wajah (*face processing*) yaitu pengenalan wajah (*face recognition*), autentikasi wajah (*face authentication*), lokalisasi wajah (*face localization*), penjejakan wajah (*face tracking*) dan pengenalan ekspresi wajah (*facial expression recognition*). Menurut [5] *face recognition* merupakan proses membandingkan citra wajah masukan dengan suatu *database* wajah dan menemukan wajah yang paling cocok dengan citra masukan tersebut. Sebelum memasuki tahap *face recognition* wajah terlebih dahulu harus terdeteksi sebagai objek wajah dengan *face detection*. Untuk melakukan *face recognition* pada ESP32-*cam* dilakukan proses pencocokkan wajah. Sample citra wajah yang ada pada *database* akan diekstraksi lalu dilakukan pencocokan. Pada proses ini dilakukan perhitungan wajah pada citra masukan seperti mata, hidung, mulut, lebar dan tinggi jarak antar komposisi wajah kemudian dicocokkan dengan jarak komposisi sampel wajah yang disimpan pada *database* [6].

2.3 ESP32-CAM

Modul ESP32-*cam* merupakan mikrokontroler yang memiliki fitur berupa *bluetooth*, Wi-Fi, kamera dan *slot micro*-SD. Pada umumnya modul ini digunakan untuk projek IoT karena memiliki fitur yang lengkap dengan harga yang murah. ESP32-*cam* dapat digunakan secara luas di berbagai aplikasi IoT. Seperti pada perangkat *smart home*, kontrol nirkabel industri, pemantauan nirkabel, QR identifikasi, dan aplikasi IoT lainnya. Mikrokontroler ini mudah digunakan dan diprogram dimana pemrogramannya menggunakan Arduino IDE [7]. Tampilan dari ESP-32*cam* dapat dilihat pada Gambar 3.

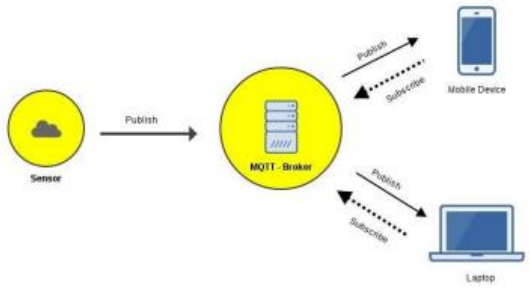


Gambar 3. ESP32-CAM

Modul ESP32-*cam* tidak memiliki *port* USB sehingga untuk mengirim program dari *port* USB komputer ke board membutuhkan USB TTL atau dapat menambahkan modul tambahan berupa *downloader* khusus untuk ESP32-*cam*. Kamera yang terpasang pada ESP32-cam dapat dilepas ataupun dipasang sesuai kebutuhan.

2.4 MQTT (*Message Queueing Telemetry Transport*)

MQTT merupakan protokol pesan ringan yang berjalan di atas protokol TCP/IP. Protokol MQTT digunakan untuk keperluan IoT karena dapat berjalan dengan daya minimum, delay rendah dan akurasi pengiriman mencapai 100% [8]. Sistem kerja MQTT menggunakan model *publish-subscribe*, dimana yang bertindak sebagai pengirim pesan adalah *publisher* dan *subscriber* sebagai penerima. Dalam proses pengiriman pesan terdapat broker yang bertindak sebagai penyampai pesan dari pengirim ke penerima berdasarkan topik yang telah diikuti oleh *subscriber*. *Publisher* dan *subscriber* adalah klien MQTT yang hanya berkomunikasi dengan broker MQTT. Klien MQTT dapat berupa perangkat seperti mikrokontroler atau aplikasi yang memiliki koneksi ke broker MQTT melalui sebuah jaringan. Berikut ilustrasi dari konsep *publish-subscribe* dalam MQTT seperti pada Gambar 4.



Gambar 4. Ilustrasi *publish-subscribe* MQTT

Dalam menyampaikan pesan MQTT memiliki level kualitas keamanan. Level-level tersebut memberikan garansi dari konsistensi (*reliability*) dari pengiriman pesan. Semakin tinggi level QoS maka data akan semakin reliable. Berikut merupakan 3 level QoS MQTT :

1. QoS level 0 (*at most once delivery*) : Broker/Klien akan mengirim pesan satu kali, tanpa menerima respon konfirmasi.
2. QoS level 1 (*at least once delivery*) : Broker/Klien akan mengirim pesan paling sedikit satu kali, jika subscriber tidak menerima pesan maka broker akan mengirimkan respon terhadap *publisher* bahwa pesan gagal dikirim. Sehingga dapat terjadi duplikasi pesan karena pengiriman pesan yang berulang kali.
3. QoS level 2 (*exactly once delivery*) : Broker/Klien akan mengirim pesan satu kali dengan menggunakan 4 langkah *handshake*.

2.5 *Cloud Computing*

Cloud computing merupakan model komputasi yang sumber dayanya dapat dikonfigurasi sesuai kebutuhan pengguna secara mudah melalui jaringan internet. Mulai dari sumber daya *server, storage, network, software, analytic* dan sebagainya yang dapat diakses dimana saja selama terkoneksi dengan internet [9]. Layanan yang diberikan oleh *cloud computing* dapat dibedakan menjadi 3 model bagian menurut [10] yaitu :

1. *Infrastructure as a service* (IaaS)

   IaaS merupakan layanan yang menyewakan sumberdaya teknologi seperti sistem operasi, *server*, media penyimpanan, *memory*, kapasitas jaringan dan berbagai layanan lain yang dapat digunakan penyewa untuk menjalankan aplikasinya. IaaS memberikan kemudahan pada pengguna untuk menggunakan perangkat fisik secara *logic* tanpa perlu membeli, perangkat fisik, melakukan pemeliharaan rutin dan konfigurasi perangkat. Contoh dari layanan IaaS adalah Amazon Web Service, Digital Ocean, Azure dan lain sebagainya.

2. *Platform as a Service* (PaaS) PaaS merupakan layanan penyedia *platform* siap pakai yang digunakan untuk mengembangkan sebuah aplikasi yang hanya bisa berjalan pada platform tersebut. PaaS meliputi semua yang ada di IaaS seperti *server, storage, networking resources* dan sebagainya yang ada di IaaS dimana telah dikelola oleh perusahaan atau penyedia pihak ketiga. Contoh dari layanan PaaS adalah Red Hat, Heroku, Microsoft Azure dan sebagainya.
3. *Software as a service* (SaaS) merupakan layanan yang difokuskan pada *user* individual, yaitu dengan memanfaatkan *web-based interface* yang diakses melalui web browser. User tidak perlu memikirkan tentang kode bahkan proses instalasi untuk menggunakan aplikasi. Sehingga user hanya perlu tau bagaimana cara menggunakan aplikasi dengan benar. Contoh dari layanan SaaS adalah Google Docs, Gmail, Google Slides dan sebagainya.

2.6 *Amazon Web Service* (AWS)

AWS merupakan penyedia layanan *service cloud* yang memberikan lebih dari 200 layanan yang dapat digunakan. AWS memiliki banyak layanan mulai dari teknologi infrastruktur seperti penghitungan, penyimpanan, basis data hingga teknologi yang berkembang, seperti *machine learning* dan *Internet of Things*. Berikut merupakan beberapa service yang disediakan AWS :

1. AWS IoT *Core* adalah layanan *cloud* yang dapat menghubungkan antara perangkat fisik dengan aplikasi *cloud* atau aplikasi lain dengan interaksi yang aman dan mudah. AWS IoT *Core* dapat terhubung dengan banyak perangkat dan merutekan pesan yang didapat ke perangkat lain atau endpoint AWS [11].
2. DynamoDB Amazon merupakan basis data NoSQL yang didesain untuk menjalankan aplikasi dengan performa tinggi. DynamoDB menawarkan keamanan, *backup* data berkelanjutan, replikasi *server* otomatis, *export* dan analisa data. DynamoDB dapat menangani lebih dari 10 triliun permintaan per hari dengan 20 juta permintaan per detik [11].
3. *Simple Notification Service* (SNS) merupakan layanan pesan dari AWS. SNS menyediakan metode *publish* dan *subscribe* untuk menghubungkan aplikasi yang berbeda. Amazon SNS dapat memberikan notifikasi ketika terdapat data baru yang masuk sebagai *trigger*. SNS menyediakan pilihan protokol seperti *http, https, email, sms, application protocol, lambda protocol, dan firehose protocol* [12].
4. AWS Lambda merupakan layanan komputasi yang memungkinkan pengguna untuk menjalankan suatu kode tanpa harus menyediakan dan mengelola *server*. Pengguna hanya cukup membayar berdasarkan waktu komputasi yang berjalan. Kemudian AWS akan menjalankan kode tersebut dengan *availability* yang baik [11].

2.7 *Quality of Service* (QoS)

Quality of service (QoS) adalah teknik yang digunakan untuk mengukur seberapa baik jaringan dan mendefinisikan karakteristik dari suatu *service* [13]. QoS dirancang untuk meningkatkan produktivitas dengan memastikan bahwa sistem mendapatkan kinerja yang andal dari aplikasi berbasis jaringan. Parameter yang digunakan pada penelitian ini menggunakan standar TIPHON (*Telecommunications and Internet Protocol Harmonization Over Networks*) [14] yang diterbitkan oleh ETSI (*European Telecommunications Standards Institude*). Berikut merupakan parameter QoS berdasarkan standar TIPHON yang digunakan pada penelitian ini :

1. *Throughput* merupakan kecepatan rata-rata transfer data dalam satu waktu transmisi. Cara menghitung nilai dari *throughput* adalah dengan cara melakukan pembagian jumlah data yang dikirim dengan waktu transmisi data. Kategori dalam *throughput* tertera pada Tabel 1.

Tabel 1. Kategori *Throughput*

Kategori Throughput	***Throughput***	***Indeks***
Sangat Bagus	100	4
Bagus	75	3
Sedang	50	2
Jelek	<25	1

2. *Delay* didefinisikan sebagai waktu yang dibutuhkan data untuk menempuh jarak dari asal ke tujuan. Delay dipengaruhi jarak, media fisik, dan waktu proses yang lama. *Delay* didapatkan dari selisih waktu kirim antara satu paket TCP dengan paket lainnya yang direpresentasikan dalam satuan detik. Kategori dalam *delay* tertera pada Tabel 2.

Tabel 2. Kategori *Delay*

Kategori Latensi	**Besar *Delay***	**Indeks**
Sangat Bagus	<150 ms	4
Bagus	150 s/d 300 ms	3
Sedang	300 s/d 450 ms	2
Jelek	>450 ms	1

3. *Packet Loss Ratio* (PLR) didefinisikan sebagai kegagalan transmisi paket mencapai tujuannya yang disebabkan oleh beberapa kemungkinan. PLR merupakan indikator kesuksesan pengiriman data dan menjadi parameter dalam kehandalan layanan pengiriman data. Kemungkinan penyebab terjadinya *packet loss* yaitu *overload* trafik dan perlambatan trafik (*congestion*) dalam jaringan, *error* yang terjadi pada media fisik, dan kegagalan yang terjadi pada sisi penerima. Kategori dalam packet loss tertera pada Tabel 3.

Tabel 3. Kategori *Packet Loss*

Kategori Degradasi	***Packet Loss***	**Indeks**
Sangat Bagus	0%	4

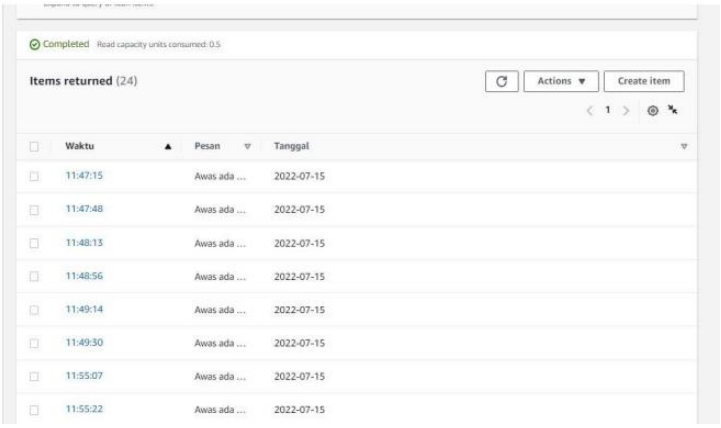
Bagus	3%	3
Sedang	15%	2
Jelek	25%	1

Indeks parameter QoS akan dijumlah dan dirata-rata hingga diperoleh nilai indeks performa seperti pada Tabel 3.

3. HASIL DAN PEMBAHASAN

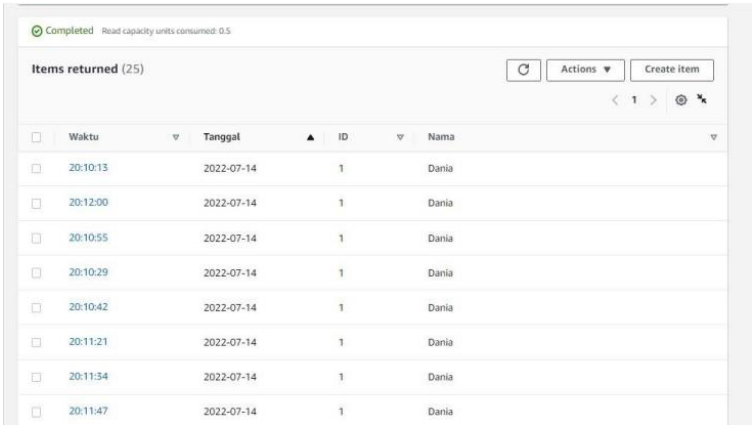
3.1. Pengujian *Database*

Pada *database* telah dibuat dua tabel yang masing-masing akan menyimpan data *log* berdasarkan hasil rekognisi wajah. Ketika wajah berhasil terverifikasi sebagai penghuni rumah maka data *log* akan tersimpan pada tabel *valid_access* seperti pada Gambar 5 dengan kolom Waktu, Tanggal dan Pesan.



Gambar 5. *Database Valid Access*

Kemudian jika wajah terverifikasi sebagai orang asing, maka data log akan tersimpan pada tabel invalid_access seperti pada Gambar 6 dengan kolom Waktu, Tanggal, ID dan Nama.



Gambar 6. *Database Invalid Access*

Hasil pengujian *database* Dynamo dapat dilihat pada Tabel 4 pengujian dilakukan dengan melakukan rekognisi wajah sebanyak 10 kali. Metode pengujian dilakukan 5 kali menggunakan wajah yang sudah terdaftar pada sistem sehingga data akan disimpan pada tabel *valid_access* dan 5 kali menggunakan wajah yang belum terdaftar pada sistem sehingga data akan disimpan pada tabel *invalid_access*.

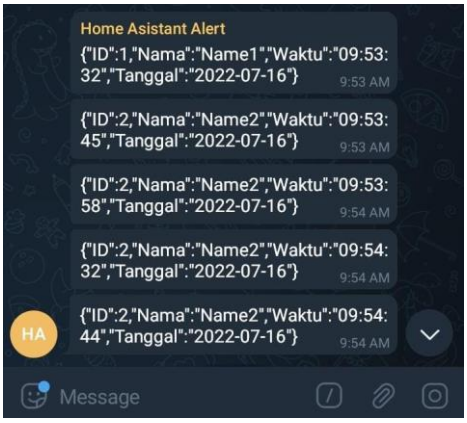
Tabel 4. Hasil Pengujian *Database*

Pengujian	Tabel *Database* (*valid_access / invalid_access*)	Status	*Successful Rate* (%)
1	valid_access	Tersimpan	100%
2	valid_access	Tersimpan	100%
3	valid_access	Tersimpan	100%
4	valid_access	Tersimpan	100%
5	valid_access	Tersimpan	100%
6	invalid_access	Tersimpan	100%
7	invalid_access	Tersimpan	100%
8	invalid_access	Tersimpan	100%
9	invalid_access	Tersimpan	100%
10	invalid_access	Tersimpan	100%

Dari 10 kali pengujian yang dilakukan, 5 data log berhasil tersimpan pada tabel *valid_access* dan 5 data *log* berhasil tersimpan pada tabel *invalid_access*. Kemudian dilakukan perhitungan successful rate pada setiap data yang berhasil tersimpan dengan didapatkan nilai rata-rata sebesar 100%. Sehingga dapat dikatakan bahwa sistem penyimpanan *database* Dynamo berfungsi dengan baik.

3.2. Pengujian Notifikasi Aplikasi Telegram

Pada Gambar 7 dapat dilihat notifikasi yang masuk ke Aplikasi Telegram berupa dua jenis data *log* yang berbeda. *Log* yang dikirim yaitu ketika wajah berhasil teridentifikasi sebagai penghuni rumah dan ketika wajah tidak teridentifikasi sebagai penghuni rumah.



Gambar 7. Tampilan Notifikasi Telegram

Waktu respon dihitung menggunakan *stopwatch* sejak wajah berhasil teridentifikasi hingga pesan muncul di Aplikasi Telegram. Hasil pengujian dapat diamati pada Tabel 5. Dari Tabel 5 dapat diketahui bahwa 5 pesan log yang dikirim dapat tersampaikan sebagai notifikasi di Telegram. Dari 5 kali pengujian yang dilakukan didapatkan waktu respon paling lama adalah 3 detik dan paling cepat adalah 2 detik.

Tabel 5. Hasil Pengujian Notifikasi

Pengujian	Data Terkirim / Tidak Terkirim	Waktu Respon (detik)
1	Terkirim	3
2	Terkirim	3
3	Terkirim	3
4	Terkirim	3
5	Terkirim	2

Menurut [15] respon Bot Telegram terhitung *relative* sangat cepat ketika kurang dari 1 menit. Pada pengujian yang telah dilakukan penulis dapat disimpulkan bahwa waktu respon Bot Telegram untuk mengirimkan *log* berupa notifikasi termasuk kategori tersebut, yaitu *relative* sangat cepat. Kemudian pada tabel terdapat perbedaan waktu respon, hal ini dapat terjadi karena beberapa faktor yang mempengaruhi. Beberapa faktor tersebut yaitu kecepatan data, kemampuan dari perangkat, banyaknya data yang dikirim dan performa *server*.

3.3. Analisis *Quality of Service* (QoS)

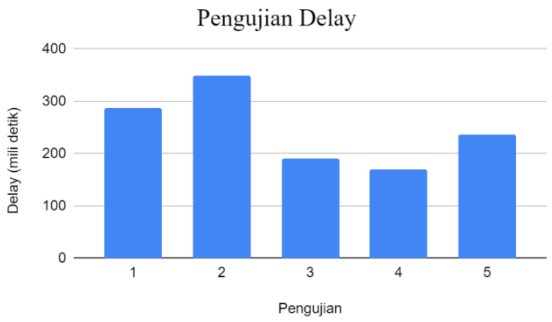
3.3.1 Hasil Pengujian *Delay*

Berdasarkan tabel 6 diperoleh delay dari setiap pengujian yang dilakukan dengan melakukan penambahan data sebanyak 10 data *log* secara berurutan. Pengujian dilakukan 5 kali dimana setiap percobaan berkisar 3 menit. Data log dikirim dari ESP32-*cam* ke AWS IoT Core dengan melakukan rekognisi wajah yang telah terdaftar pada sistem.

Tabel 6. Hasil Pengujian *Delay*

Pengujian	Banyak Pengiriman Data	*Delay* (ms)	Waktu Pengujian
1	10	285,9186	3 Menit
2	10	348,3796	
3	10	190,8763	
4	10	168,7381	
5	10	235,7745	

Pada Tabel 6 dapat dilihat bahwa *delay* terbesar ada pada pengujian 2 sebesar 348,3796 mili detik dan *delay* terkecil ada pada pengujian 4 sebesar 168,7381. Berdasarkan data tersebut, nilai rata-rata *delay* pengiriman dari ESP32-*cam* ke AWS IoT pada 5 kali pengujian adalah 245,9374 mili detik. Menurut kategori *delay* pada Tabel 2 nilai tersebut termasuk dalam kategori Bagus. Sehingga didapatkan nilai indeks sebesar 3 poin karena nilai *delay* masih pada rentang 150 mili detik hingga 300 mili detik. Adapun grafik hasil dari perhitungan nilai *delay* dapat dilihat pada Gambar 8.



Gambar 8. Grafik Hasil Pengujian *Delay*

Menurut [13] semakin kecil nilai *delay* pada saat transmisi data maka proses pengiriman data akan cepat. Dapat dilihat pada Gambar 8 bahwa nilai *delay* paling besar ada pada pengujian 2 sehingga proses transmisi data pada pengujian 2 lebih lama dan proses transmisi data pada pengujian 4 paling cepat karena memiliki nilai *delay* paling kecil. Selain itu terlihat grafik mengalami kenaikan pada pengujian 2 dan mengalami penurunan yang signifikan pada pengujian 3. Variasi nilai *delay* pada pengujian dapat terjadi karena beberapa faktor yaitu, jarak antara klien dengan *server*, antrian paket yang panjang, media fisik yang digunakan dan pengambilan rute dalam menghindari kemacetan *routing*.

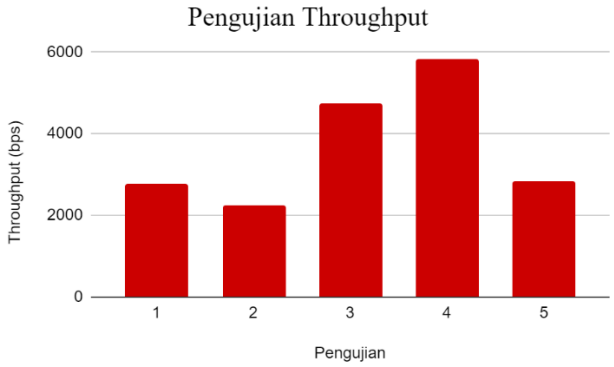
3.3.2 Hasil Pengujian *Throughput*

Berdasarkan tabel 7 diperoleh throughput dari setiap pengujian yang dilakukan dengan melakukan penambahan data sebanyak 10 data *log* secara berurutan. Pengujian dilakukan 5 kali dimana setiap percobaan berkisar 3 menit. Data *log* dikirim dari ESP32-*cam* ke AWS IoT Core dengan melakukan rekognisi wajah yang telah terdaftar pada sistem.

Tabel 7. Hasil Pengujian *Packet Loss*

Pengujian	*Throughput* (bps)	Waktu Pengujian
1	2.770	3 Menit
2	2.257	
3	4.732	
4	5.834	
5	2.839	

Berdasarkan data nilai *throughput* yang ditampilkan pada Tabel 7 nilai *throughput* mengalami perubahan pada 5 kali pengujian. Nilai *throughput* paling tinggi berada pada pengujian 4 sebesar 5.834 bps dan paling rendah berada pada pengujian 2 sebesar 2.257 bps. Berdasarkan data tersebut, nilai rata-rata *throughput* pengiriman dari ESP32-*cam* ke AWS IoT pada 5 kali pengujian adalah 3.686 bps dengan mendapatkan nilai indeks sebesar 4 dengan kategori Sangat Bagus berdasarkan standar TIPHON. Adapun grafik hasil dari perhitungan nilai throughput dapat dilihat pada Gambar 9.



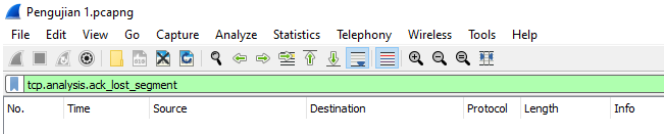
Gambar 9. Grafik Hasil Pengujian *Throughput*

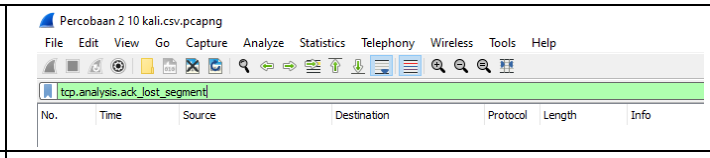
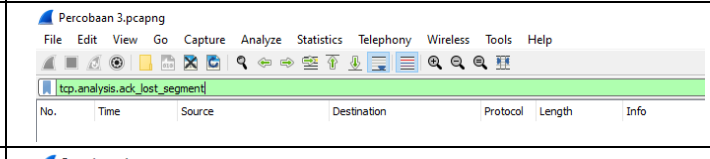
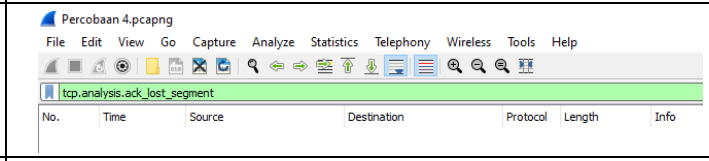
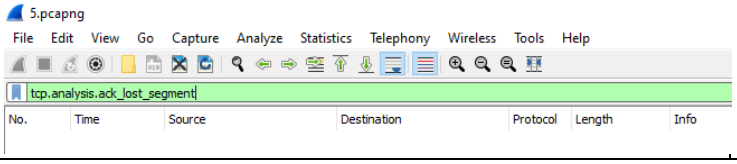
Dapat dilihat pada Gambar 9 bahwa nilai *throughput* paling tinggi ada pada pengujian 4 dan terendah ada pada pengujian 2. Jika dibandingkan dengan nilai *delay* pada Gambar 9 maka nilai *throughput* merupakan kebalikan dari nilai *delay*. Sehingga dapat disimpulkan bahwa semakin besar nilai *delay* maka akan semakin kecil nilai *throughput* yang dihasilkan.

3.3.2 Hasil Pengujian *Throughput*

Berdasarkan tabel 8 diperoleh nilai *packet loss* dari setiap pengujian yang dilakukan dengan melakukan penambahan data sebanyak 10 data *log* secara berurutan. Pengujian dilakukan 5 kali dimana setiap percobaan berkisar 3 menit. Data *log* dikirim dari ESP32-*cam* ke AWS IoT Core dengan melakukan rekognisi wajah yang telah terdaftar pada sistem.

Tabel 8. Hasil Pengujian *Packet Loss*

Pengujian	Filter	Packet Loss (%)
1		0%

2		0%
3		0%
4		0%
5		0%

Berdasarkan Tabel 8 menunjukkan bahwa hasil dari pengujian *packet loss* bernilai 0% karena dari pengujian pengiriman data pada Aplikasi Wireshark tidak ditemukan adanya *packet loss*. Persentase pada tabel menunjukkan paket yang hilang selama transmisi data. Hal ini menunjukkan bahwa pada pengujian 1 hingga pengujian 5 tidak ada paket data yang hilang karena tidak sampai pada tujuan. Kemudian dari nilai 0% yang didapat, disimpulkan bahwa *packet loss* untuk pengiriman data dari ESP32-*cam* ke AWS IoT Core dikatakan sangat baik menurut pada Tabel 3 dengan mendapatkan nilai indeks 4.

4. KESIMPULAN

Berdasarkan penelitian analisis performa transmisi data *log* berbasis IoT *cloud* pada kunci pintu pintar menggunakan rekognisi wajah, maka dapat diambil beberapa kesimpulan sebagai berikut :

1. AWS IoT *Core* dapat digunakan sebagai *broker* untuk meng-*handle* data *log* dari ESP32-*cam* ke DynamoDB dan Amazon SNS.
2. Sistem penyimpanan menggunakan *service cloud* dari AWS berupa DynamoDB memiliki nilai *successful rate* 100% untuk menyimpan data sehingga *database* dapat digunakan untuk menyimpan *log*.
3. *Delay* pada pengujian *respon time* notifikasi telegram bernilai kisaran 2 – 3 detik, yang mana masih termasuk dalam kategori *relative* sangat cepat dan *real time*.
4. Berdasarkan pengujian yang telah dilakukan didapat rata-rata nilai delay sebesar 245,9374 mili detik yang termasuk dalam kategori Bagus. Untuk parameter throughput dikategorikan Sangat Bagus dengan nilai rata-rata 3.686 bps. Kemudian untuk nilai *packet loss* menunjukkan bahwa tidak ada paket data yang hilang selama proses transmisi sehingga mendapat persentase sejumlah 0% dengan kategori Sangat Bagus. Semua pengkategorian dianalisis berdasarkan standar TIPHON.

REFERENSI

[1] "List Data Dasar | Aplikasi Dataku." http://bappeda.jogjaprov.go.id/dataku/data_dasar/index/447-jumlah-kasus-pencurian?id_skpd=39 (accessed Mar. 15, 2022).

[2] M. Mehta, "(PDF) ESP 8266: A BREAKTHROUGH IN WIRELESS SENSOR NETWORKS AND INTERNET OF THINGS | IAEME Publication - Academia.edu," *IJECET*, 2015. https://www.academia.edu/27613529/ESP_8266_A_BREAKTHROUGH_IN_WIRELESS_SENSOR_NETWORKS_AND_INTERNET_OF_THINGS (accessed Apr. 13, 2022).

[3] F. Adani and S. Salsabil, "INTERNET OF THINGS: SEJARAH TEKNOLOGI DAN PENERAPANNYA," *ISU TEKNOLOGI STT MANDALA* , vol. 14, pp. 92–99, 2019, Accessed: Mar. 13, 2022. [Online]. Available: https://www.ejournal.sttmandalabdg.ac.id/index.php/JIT/article/view/162/141

[4] Y. Efendi, "INTERNET OF THINGS (IOT) SISTEM PENGENDALIAN LAMPU MENGGUNAKAN RASPBERRY PI BERBASIS MOBILE," *Jurnal Ilmiah Ilmu Komputer*, vol. 4, no. 1, 2018, Accessed: Apr. 13, 2022. [Online]. Available: http://ejournal.fikom-unasman.ac.id

[5] F. N. Afandi, R. P. Sinaga, Y. Aprilinda, and F. Ariani, "IMPLEMENTASI FACE DETECTION PADA SMART CONFERENCE MENGGUNAKAN VIOLA JONES," *Explore: Jurnal Sistem Informasi dan Telematika (Telekomunikasi, Multimedia dan Informatika)*, vol. 10, no. 2, Oct. 2019, doi: 10.36448/JSIT.V10I2.1320.

[6] A. Putra, M. Susilo, D. Darlis, and D. A. Nurmantris, "PENGENALAN WAJAH BERBASIS ESP32-CAM UNTUK SISTEM KUNCI SEPEDA MOTOR ESP32-CAM-BASED FACE RECOGNITION FOR MOTORCYCLE LOCK SYSTEM," vol. 8, no. 2, pp. 1091–1103, 2021, doi: 10.25124/jett.v8i2.4199.

[7] M. F. Wicaksono and M. D. Rahmatya, "Implementasi Arduino dan ESP32 CAM untuk Smart Home," *Jurnal Teknologi dan Informasi*, vol. 10, no. 1, pp. 40–51, Feb. 2020, doi: 10.34010/JATI.V10I1.2836.

[8] G. Yudha Saputra, A. Denhas Afrizal, F. Khusnu Reza Mahfud, F. Angga Pribadi, and F. Jati Pamungkas, "Penerapan Protokol MQTT Pada Teknologi Wan (Studi Kasus Sistem Parkir Univeristas Brawijaya)," *Informatika Mulawarman : Jurnal Ilmiah Ilmu Komputer*, vol. 12, no. 2, pp. 69–75, Aug. 2017, doi: 10.30872/JIM.V12I2.653.

[9] C. F. Permatasari and H. Dhika, "JISA (Jurnal Informatika dan Sains) Optimasi Jalur Transfer Data dari HTTP menjadi MQTT pada IoT menggunakan Cloud Services," vol. 01, no. 02, 2018.

[10] I. N. 'Abidah, M. A. Hamdani, and Y. Amrozi, "Implementasi Sistem Basis Data Cloud Computing pada Sektor Pendidikan," *KELUWIH: Jurnal Sains dan Teknologi*, vol. 1, no. 2, pp. 77–84, Aug. 2020, doi: 10.24123/saintek.v1i2.2868.

[11] M. M. Usman and X. B. N. Najoan, "Rancang Bangun Aplikasi Monitoring," *Jurnal Teknik Informatika*, vol. 9, pp. 73–80, 2020.

[12] D. C. Marinescu, "Cloud Application Development," *Cloud Computing*, pp. 317–359, Jan. 2013, doi: 10.1016/B978-0-12-404627-6.00011-7.

[13] H. Fahmi, "ANALISIS QOS (QUALITY OF SERVICE) PENGUKURAN DELAY, JITTER, PACKET LOST DAN THROUGHPUT UNTUK MENDAPATKAN KUALITAS KERJA RADIO STREAMING YANG BAIK ANALYSIS QOS (QUALITY OF SERVICE) MEASUREMENT OF DELAY , JITTER, PACKET LOST AND THROUGHPUT TO GET GOOD QUALITY OF RADIO STREAMING WORK," 2018.

[14] "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON); General aspects of Quality of Service (QoS)," 1999, Accessed: Jul. 16, 2022. [Online]. Available: http://www.etsi.org

[15] H. Mulyo and G. Mohammad, "INTEGRASI SISTEM INFORMASI AKADEMIK MAHASISWA DENGAN BOT TELEGRAM SEBAGAI MESIN PENJAWAB OTOMATIS INTEGRATION OF STUDENT ACADEMIC INFORMATION SYSTEMS WITH TELEGRAM BOT AS AN AUTOMATIC ANSWERING MACHINE," vol. 13, no. 1, pp. 2548–4168, 2022, doi: 10.34001/jdpt.v12i2.