Custom Cyclic Redundancy Check dan Manajemen Komunikasi Pengiriman dan Penerimaan Data pada Arsitektur *Delay Tolerant Network*

Griffani Megiyanto Rahmatullah[1], Muhamad Rizki[2]

[1]*Jurusan Teknik Elektro, Politeknik Negeri Bandung 40012*
[1]*Email : griffani.megiyanto@polban.ac.id*
[2]*Email : muhamad.rizki.tkom18@polban.ac.id*

ABSTRAK

Pertukaran informasi melalui jaringan internet saat ini sangat dibutuhkan yang mencakup juga daerah *intermitten* atau daerah putus sambung sinyal. Untuk mendukung teknologi tersebut dibutuhkan jaringan komunukasi yang memadai dari segi konektivitas dan keutuhan informasi. Solusi dari permasalahan tersebut telah dibuat sistem pengumpulan data melalui jaringan *wireless* dengan arsitektur *Delay Tolerant Network* (DTN) yang tetap dapat mengirimkan *file* walaupun terjadi *delay* tinggi. Tujuan dari penilitian yaitu membuat manajemen komunikasi data dengan dilengkapi *custom Cyclic Redudancy Check* (CRC) sebagai algoritma pengecekan *error* pada data. Metode penelitian yang dilakukan yaitu literatur review, perancangan dan pembuatan algoritma, impelementasi sistem dan perbandingan dengan teori dasar. Tahap realisasi menggunakan dua perangkat keras raspberry pi 3 sebagai pengolah data yang berkomunikasi menggunakan jaringan *wireless*. Pada perangkat lunaknya menggunakan IBR-DTN konfigurasi *User Datagram Protocol* sebagai aplikasi layanan DTN,dan bahasa pemrograman python dalam membuat *script* sistem. Pengujian yang dilakukan berupa pengujian fungsionalitas sistem dengan melakukan pengiriman 10 data gambar berbagai ukuran yang dilakukan di dalam ruangan berkisar jarak 1-2 meter antar node.Hasil pengujian menunjukkan 8 dari 10 data gambar berhasil dikirim untuk ukuran <64KB.Sistem handal dalam melakukan pengiriman gambar ukuran <64KB dengan dilengkapi proses pengecekan 8 bit crc dan berhasil melakukan pengiriman ulang apabila terjadi data *error* pada penerima.

Kata Kunci
Delay Tolerant Network, User Datagram Protocol, Cyclic Redundancy Check, Raspberry Pi

1. PENDAHULUAN

Pada saat ini kebutuhan akan pertukaran informasi melalui jaringan internet sangat tinggi. Dibuktikan pada revolusi industri 4.0 khususnya pada sektor pertanian berbasis teknologi menerapkan sistem pertanian cerdas, seperti monitoring kondisi lahan perkebunan untuk mencegah terjadinya kerusakan terhadap lahan dan tanaman yang dapat dipantau dari kejauhan dengan memanfaatkan jaringan internet [1]. Kondisi jaringan pada lahan perkebunan yang merupakan salah satu area terbuka memiliki potensi terjadi putus koneksi, sedangkan untuk menerapkan teknologi tersebut dibutuhkan jaringan internet yang memadai dan tidak mengalami putus koneksi atau *delay* yang lama terutama untuk jaringan *wireless*. Permasalahan tersebut dapat diatasi dengan membangun jaringan yang dapat melakukan komunikasi data pada daerah dengan kondisi jaringan *long delay*, *intermitten*, rentan terjadi putus koneksi yaitu arsitektur jaringan *Delay Tolerant Network* [2].

Dengan menggunakan arsitektur jaringan *Delay Tolerant Network* ini maka pertukaran informasi masih dapat dilakukan pada daerah yang berpotensi terjadi putus koneksi seperti daerah hutan,pegunungan, dan sebagainya[3]. Pada DTN ini dapat digunakan konfigurasi protokol internet yang umum digunakan yaitu TCP/IP dan UDP. Protokol UDP memiliki keunggulan dari parameter kecepatan dibandingkan dengan DTN. Namun dari segi kehandalan pengiriman data DTN lebih baik walaupun saat terjadi putus koneksi [4].

Dalam mendeteksi kesalahan selama transmisi terdapat penelitian mengenai kehandalan dari algoritma *Cyclic Redudancy Check* (CRC) yang mampu mendeteksi kesalahan dalam data[5] [6].

Solusi yang diusulkan penulis adalah realisasi *hardware* dan *software* arsitektur DTN dengan konfigurasi UDP (*User Datagram Protocol)* yang dilengkapi dengan fitur cek kesalahan *Cyclic Redudancy Check* (CRC) dan manajemen komunikasi antara pengirim dan penerima. Penambahan CRC dan manajemen komunikasi pengiriman dan penerimaan data antar perangkat dapat mengatasi karakteristik UDP yang bersifat *connection-loss*, pada penerima dapat dilakukan pengecekan data yang diterima apakah sudah sesuai dengan yang dikirim

dan apabila tidak sesuai maka penerima akan meminta pengiriman ulang data.

Tujuan dari penelitian ini adalah daerah yang terdapat *delay* tinggi masih dapat melakukan pertukaran data, membangun sistem komunikasi pengiriman dan penerimaan data pada protokol *User Datagram Protocol* (UDP) yang bersifat *connection-less* dengan dilengkapi dengan cek kesalahan pengiriman selama transmisi dengan *Cyclic Redudancy Check.*

2. METODE PENELITIAN

Metode penelitian yang dilaksanakan dalam penelitian terdiri dari literatur berkaitan dengan Delay Tolerant Network (DTN), User Datagram Protocol (UDP), dan Cyclic Redudancy Check (CRC) yang menunjang dalam penelitian. Lalu terdapat perancangan dan pembuatan algoritma sistem, implementasi sistem, dan perbandingan hasil algoritma dengan perhitungan berdasarkan teori dasar.

Tahapan pertama yang dilakukan yaitu literatur review yang berkaitan dengan Delay Tolerant Network, User Datagram Protocol dan Cyclic Redudancy Check. Literatur tersebut dibaca dari jurnal dan paper yang telah terbit terkait dengan fungsionalitas masing-masing. Literasi juga dilakukan khususnya perhitungan CRC sebagai pengecekan error berdasarkan CRC 8 bit.

2.1 Persiapan Realisasi

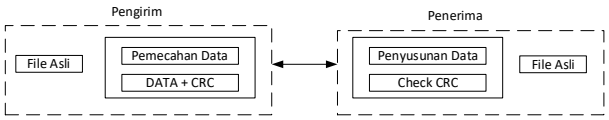
Berdasarkan studi literatur yang telah dilakukan, pada penelitian akan menggunakan perangkat keras raspberry pi, sedangkan perangkat lunak yang akan digunakan adalah IBRDTN dan pemrograman python untuk proses crc.

1. Raspberry Pi, adalah salah satu perangkat keras embedded system berupa komputer seukuran dengan kartu kredit yang dapat menjalankan program, mengolah data, bermain game dan lainnya. Raspberry pi yang digunakan dalam penelitian adalah tipe raspberry pi 3 yang sudah memiliki teknologi pendukung Wireless sehingga menunjang dalam penelitan. Raspberry pi berfungsi sebagai perangkat pengolah data dan tempat diimplementasikan sistem penelitian.
2. IBRDTN, adalah aplikasi perangkat lunak pengembangan dari DTN yang dibuat khusus dalam implementasi DTN untuk embedded system [7]. IBRDTN ini digunakan sebagai aplikasi yang akan menjalankan pengiriman data dengan arsitektur delay tolerant network [8].
3. Cyclic Redudancy Check, merupakan algoritma untuk memastikan integritas suatu data dan mengecek suatu kesalahan pada data yang disimpan atau ditransmisikan[9]. Fungsi CRC dalam penelitian untuk mengecek kesalahan dalam pengiriman data dan memastikan data handal sehingga karakteristik User Datagram Protocol yang tidak handal dapat diatasi.

Fungsi dari sistem yang akan dibuat dapat melakukan pengiriman dengan menggunakan protokol user datagram protocol pada arsitektur delay tolerant network yang dilengkapi dengan bit CRC dan proses pengecekan kesalahan serta pengiriman ulang pada data yang salah.

2.1.1 Diagram Blok

Tahap realisasi dimulai dengan perencanaan blok diagram sistem keseluruhan yang akan dibuat dalam penelitian ini.



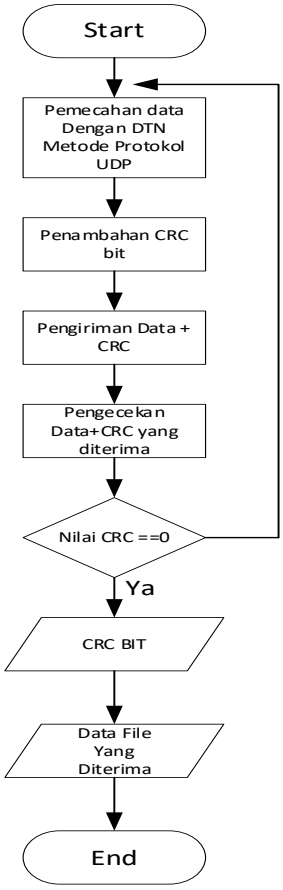
Gambar 1. Diagram Blok Sistem

Bedasarkan gambar 1 sistem terdiri dari bagian pengirim dan penerima. Perangkat yang digunakan dalam komunikasi data ini adalah raspberry pi. Pada kedua raspberry pi sudah terinstall modul IBR-DTN konfigurasi protokol *User Datagram Protocol*. Data file asli akan dikirim berdasarkan arsitektur *Delay Tolerant Network*, pengiriman data disertai CRC bit sebagai deteksi kesalahan, pada bagian penerima data akan di cek CRC nya apakah terdapat data yang hilang atau tidak. Bila terdapat data yang hilang maka penerima akan meminta pengiriman ulang data.

2.2 Flowchart

Alur sistem komunikasi pengiriman dan penerimaan data yang akan dirancang yaitu sebagai berikut :

1. File gambar yang dikirim akan dibaca ke dalam bentuk deretan bit
2. Gambar yang dikirim sudah disertai dengan bit CRC yang ditambahkan pada informasi metadata gambar.
3. Pengiriman data dilakukan dengan aplikasi IBRDTN pada raspberry pi
4. Pada penerima dilakukan pengecekan CRC bit, bila terdapat error pada penerima maka akan dilakukan pengiriman ulang data.



Gambar 2. Flowchart Sistem

Pada Gambar 2 dijelaskan proses pengiriman dan penerimaan data dengan IBRDTN yang disertai dengan CRC bit sebagai pengecekan keutuhan data.

Perancangan perangkat lunak yang akan dibangun untuk merealisasikan sistem komunikasi *delay tolerant network* ini pada kedua raspberry akan diinstal IBR-DTN yang merupakan aplikasi pihak ketiga untuk menjalankan *service delay tolerant network*. Pada sisi pengirim akan ditanamkan program untuk menambahkan CRC bit sebagai bit tambahan untuk pendeteksian kesalahan pengiriman data yang dilakukan. Adapun pada sisi penerima akan ditanamkan program untuk melakukan pengecekan kesalahan atau kegagalan pengiriman file dengan logika XOR. Bila hasil akhir XOR data yang dikirim dengan pembagi CRC bit menunjukkan hasil 0 maka tidak ada *error* yang terjadi, namun bila hasilnya bukan 0 maka penerima akan mendeteksi terjadi *error* dan akan meminta pengiriman ulang pada data yang gagal dikirim tersebut.

Berdasarkan perancangan sistem, Realisasi sistem terbagi menjadi dua bagian yaitu realisasi perangkat keras dan perangkat lunak.

2.3 Realisasi

Realisasi sistem yang dibangun terdiri dari realisasi perangkat keras dan perangkat lunak.

2.3.1 Perangkat Keras

Pada bagian perangkat keras sistem yang akan digunakan adalah raspberry pi 3 model B dan catu daya berupa powerbank 5 v 2,5 A untuk menyalakan raspberry pi. Berikut gambar perangkat keras yang digunakan.

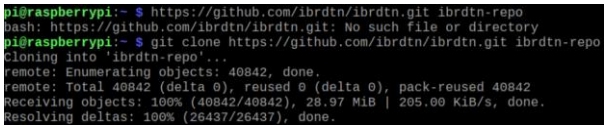
Gambar 3. Raspberry Pi 3

Raspberry pi yang digunakan dalam penelitian ini adalah raspberry pi 3 seperti ditunjukkan pada gambar 3. Pemilihan raspberry pi 3 model B ini dikarenakan pada raspberry jenis ini sudah *support* Wi-Fi sehingga pada realisasi nantinya tidak memerlukan modul Wifi lagi.

2.3.2 Perangkat Lunak

2.3.2.1 Impelementasi IBR-DTN

Pada perealisasian IBR-DTN dilakukan instalasi *software* IBR-DTN pada kedua perangkat raspberry pi yang nantinya digunakan untuk menjalankan arsitektur jaringan *delay tolerant network.* Proses instalasi IBR-DTN pada kedua raspberry pi dilakukan dengan mengambil pada repository DTN di github.



Gambar 4. Proses instalasi IBR-DTN

Tahapan awal dalam penelitian ini adalah menginstall IBR-DTN pada kedua *node* raspberry pi agar dapat melaksanakan komunikasi data dengan arsitektur DTN seperti pada Gambar 4. Kemudian mengubah beberapa konfigurasi dari IBR-DTN agar sesuai dengan penelitian yang akan dilakukan.

```
#
# configuration for a convergence layer named lan0
#
#net_lan0_type = tcp                                # we want to use TCP a$
#net_lan0_interface = wlan0                 # listen on interface eth0
#net_lan0_port = 4556                       # with port 4556 (default)

#
# configuration for a convergence layer named lan1
#
net_lan1_type = udp                                # we want to use UDP as protoc$
net_lan1_interface = wlan0                 # listen on interface eth0
net_lan1_port = 4556                       # with port 4556 (default)
```

Gambar 5. Konfigurasi IBR-DTN

Konfigurasi IBR-DTN yang utama adalah menentukan protokol pengirimannya adalah UDP *convergence layer* seperti yang ditunjukkan pada Gambar 5. Setelah melakukan instalasi IBR-DTN terkonfigurasi UDP pada kedua raspberry pi, perlu dilakukan *test ping* untuk menguji apakah antar node sudah dapat saling berkomunikasi.

```
pi@raspberrypi: ~
File  Edit  Tabs  Help
pi@raspberrypi:~ $ dtnping dtn://node-1/echo
ECHO dtn://node-1/echo 64 bytes of data.
64 bytes from dtn://node-1/echo: seq=1 ttl=30 time=19.79 ms
64 bytes from dtn://node-1/echo: seq=2 ttl=30 time=14.25 ms
64 bytes from dtn://node-1/echo: seq=3 ttl=30 time=14.04 ms
64 bytes from dtn://node-1/echo: seq=4 ttl=30 time=15.29 ms
64 bytes from dtn://node-1/echo: seq=5 ttl=30 time=13.80 ms
64 bytes from dtn://node-1/echo: seq=6 ttl=30 time=14.57 ms
64 bytes from dtn://node-1/echo: seq=7 ttl=30 time=15.10 ms
64 bytes from dtn://node-1/echo: seq=8 ttl=30 time=15.65 ms
64 bytes from dtn://node-1/echo: seq=9 ttl=30 time=14.75 ms
64 bytes from dtn://node-1/echo: seq=10 ttl=30 time=14.07 ms
64 bytes from dtn://node-1/echo: seq=11 ttl=30 time=10.65 ms
64 bytes from dtn://node-1/echo: seq=12 ttl=30 time=15.27 ms
^C
--- dtn://node-1/echo echo statistics ---
12 bundles transmitted, 12 received, 0.00% bundle loss, time 11.80 s
rtt min/avg/max = 10.65/14.77/19.79 ms
pi@raspberrypi:~ $ dtnping dtn://node-1/echo
```

Gambar 6. *Test Ping* Antar *Node DTN*

Pengujian *ping* antar *node* dilakukan untuk mengecek *node* DTN sudah saling terhubung, cara pengujian *ping* dilakukan dengan mengetikkan dtnping dtn//nodeygdituju/echo dengan hasil uji antar *node* DTN sudah saling terhubung seperti ditunjukkan pada Gambar 6, langkah terakhir adalah pengujian pengiriman teks ke *node 2* untuk memastikan bahwa IBR-DTN sudah terinstall dengan baik pada kedua raspberry pi.

```
pi@raspberrypi:~ $ echo Halo Node-2! > CobaKirim
pi@raspberrypi:~ $ dtnsend dtn://node-2/Untukmu CobaKirim
Transfer file "CobaKirim" to dtn://node-2/Untukmu
pi@raspberrypi:~ $
pi@raspberrypi:~ $ dtnrecv --name Untukmu
Halo Node-2!
pi@raspberrypi:~ $
```

Gambar 7. Pengujian pengiriman data berupa text

Untuk menguji IBR-DTN sudah terinstall dengan benar, maka dilakukan pengujian pengiriman data dari *node 1* ke *node 2* dengan pengiriman data yaitu “Halo Node-2 !” dan dihasilkan data diterima oleh *node-2* yang berarti aplikasi IBR-DTN sudah dapat dijalankan seperti pada Gambar 7.

2.3.2.2 Implementasi Program

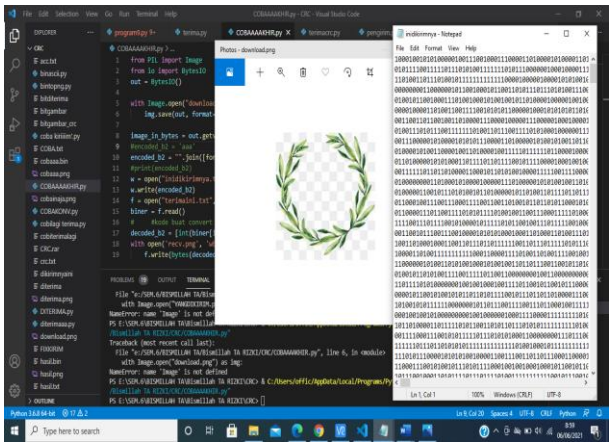
Pada sistem yang dibuat ini terdapat program dengan bahasa pemrograman python. Pada bagian pengirim dibuat program python untuk menambahkan bit data yang dikirim dengan bit CRC. Adapun pada bagian penerima dibuat program python untuk mengecek apakah terjadi *error* saat proses pentransmisian data. seperti pada Gambar .

```
#Program CRC pada Pengirim
def mod2div(hasilbagi,bagi) :
    pick = len(pembagi)
    tmp = hasilbagi[0:pick]
    while pick <len(hasilbagi) :
        if tmp[0]=='1':
            tmp = xor(pembagi,tmp) + hasilbagi[pick]
        else :
            tmp = xor('0'*pick,tmp) + hasilbagi[pick]

        pick +=1
    if tmp[0] == '1':
        tmp = xor(pembagi, tmp)
    else :
        tmp = xor('0'*pick, tmp)

    checkword = tmp
    return checkword

def xor(a,b) :
    #Inisialisasi tempat hasil
    hasil = []
    for i in range(1,len(b)) :
        if a[i] == b[i] :
            hasil.append('0')
        else :
            hasil.append('1')

    return ''.join(hasil)
```

Gambar 8. Program CRC

Program CRC ini berfungsi untuk pengecekan keutuhan data yang dikirimkan selama transmisi data. Dengan adanya CRC ini maka dipastikan data dapat diterima dengan utuh, karena dilakukan proses ada tidaknya *error*, yang mana apabila ada *error* terjadi maka akan dilakukan pengiriman data ulang.

Pada realisasi perangkat lunak dibuat juga program untuk mengkonversi gambar menjadi kumpulan bit-bit yang nantinya akan diproses CRC dan dikonversi kembali menjadi gambar utuh.



Gambar 8. Program Konversi Gambar

Gambar 8 di atas menjelaskan bagaimana program dari pembacaan gambar menjadi deretan bit-bit yang nantinya bit-bit tersebut akan dilakukan proses penambahan CRC bit.

3. HASIL DAN PEMBAHASAN

3.1 Parameter yang Diuji

1. Pengujian Setiap Program
   Pengujian ini diilakukan untuk mengetahui apakah program yang dibuat pada setiap sub program dapat menghasilkan *output* yang diharapkan. Program membaca gambar dalam bentuk deretan biner dapat menghasilkan *output* berupa deretan bener. Adapun program crc pada pengirim dapat menghasilkan bit crc dari proses crc pada deretan biner gambar. Sedangkan pada program pengecekan CRC penerima dapat mendeteksi *error* pada data yang diterima.
2. Pengiriman Data
   Pengujian ini dilakukan untuk memastikan data yang dikirim adalah data gambar yang disertai dengan bit CRC tambahan.
3. Penerimaan Data
   Pengujian ini dilakukan untuk memastikan pada raspberry penerima dapat menerima dan melakukan pengecekan kesalahan pada data gambar yang dikirim dari raspberry pengirim.
4. Fungsionalitas Sistem
   Pada tahap ini dilakukan pengujian terhadap fungsionalitas sistem yang telah dibuat. Pengujian dilakukan dengan melakukan pengiriman 10 gambar dengan berbagai variasi ukuran data gambar.Pengujian memanfaatkan jaringan *wireless* dari *hotspot smartphone* yang dilakukan dalam skala dalam ruangan. Pengujian ini dilakukan untuk mengetahui kehandalan sistem yang telah dibuat, sistem dapat mendeteksi kesalahan pengiriman yang terjadi dan menguji apakah 10 data gambar berhasil dikirim dengan utuh.

3.2 Hasil Pengujian

1. Program Pembacaan Gambar

Gambar 9. Program Pembacaan Gambar

Gambar 10 menunjukkan tiga bagian gambar dengan nomor 1 merupakan gambar yang akan dibaca ke dalam bentuk data biner. Sedangkan nomor 2 adalah hasil *output* program pembacaan gambar nomor 1 ke dalam bentuk deretan bit '1010…'. Adapun nomor 3 merupakan gambar hasil dari konversi deretan bit menjadi gambar utuh kembali. Hasil pengujian program pembacaan gambar menunjukkan program sudah dapat difungsikan.

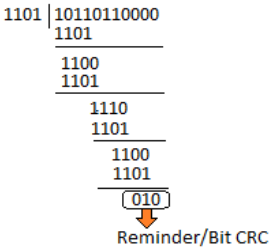
2. Program CRC

1) CRC pada pengirim

```
E:\SEM.6\BISMILLAH TA\Bismillah TA RIZKI\CRC>kirim.py
Data yang akan dikirim = 10110110
10110110
pembagi = 1101
data yang ditransmisikan = 10110110010
```

Gambar 10. Hasil Pengujian Program CRC pada Pengirim

Pada Gambar 11 menunjukkan bagaimana proses pengujian program penambahan bit CRC yang telah dibuat. Pengujian dilakukan dengan data input berupa data yang akan dikirim yaitu 10110110 yang selanjutnya diproses CRC dengan melakukan proses pembandingan dengan bit pembagi 1101 dengan proses pembandingan logika XOR sehingga diperoleh data yang ditransmisikan adalah 10110110010 yang mana merupakan deretan dari data + bit crc. Berikut bagaimana proses perhitungan bit crc secara teori.



Gambar 11. Proses CRC

Berdasarkan hasil perhitungan crc pada Gambar 12 diperoleh nilai bit crc yang akan ditambahkan pada data adalah 010. Hasil tersebut sama dengan hasil proses crc pada Gambar 12 yang mana nilai bit crc nya adalah 010. Hasil pengujian program CRC pada pengirim sesuai dengan perhitungan secara teori menunjukkan program crc pada pengirim dapat difungsikan.

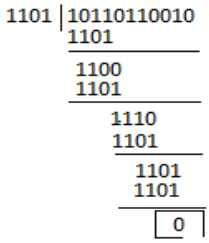
2) CRC pada penerima

Pada penerima dilakukan pengujian pada program dengan mensimulasikan menerima data yang sama seperti pada data yang ditransmisikan pada Gambar 12 yaitu 10110110010. Dengan hasil pengujian sebagai berikut.

```
E:\SEM.6\BISMILLAH TA\Bismillah TA RIZKI\CRC>terima.py
Data yang diterima = 10110110010
10110110010
pembagi = 1101
sisa bagi = 000
Tidak ada error pengiriman

E:\SEM.6\BISMILLAH TA\Bismillah TA RIZKI\CRC>terima.py
Data yang diterima = 10110011001
10110011001
pembagi = 1101
sisa bagi = 110
Terdapat kesalahan pengiriman data
```

Gambar 12. Hasil Pengujian Program CRC Penerima

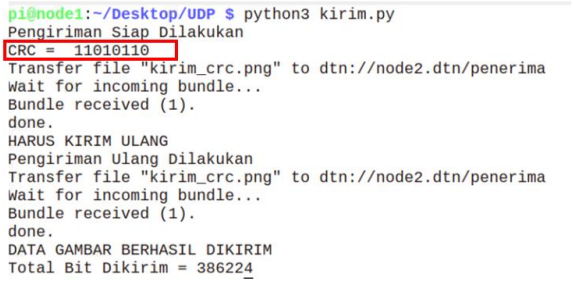
Pada Gambar 13 terdiri dari dua blok gambar dengan warna merah dan kuning. Blok merah adalah pengujian pada program CRC penerima dengan menerima data yang sesuai dengan data yang dikirim yaitu 10110110010, hasil menunjukkan sisa bagi proses CRC adalah 0 yang artinya tidak terdapat *error* pada pengiriman. Adapun pada blok kuning merupakan pengujian dengan data yang diterima tidak sesuai dengan yang dikirim, dengan hasil menunjukkan sisa bagi adalah 110, apabila hasil tidak sama dengan 0 maka terdapat *error* pada keutuhan data yang diterima. Berikut proses pengecekan crc secara teori yang menunjukkan hasilnya sama seperti output program yaitu 0 untuk data yang diterima sesuai.



Gambar 13. Proses pengecekan CRC

3) Fungsionalitas Sistem

Pada pengujian fungsionalitas sistem dilakukan pengujian pada keseluruhan program yang telah dibuat dengan melakukan pengiriman data antar dua perangkat raspberry pi dengan pengujian dilakukan di dalam ruangan memanfaatkan jaringan *wireless* dari *smartphone* dengan berkisar jarak antar raspberry 1-2 meter. Berikut hasil pengujian fungsionalitas sistem.

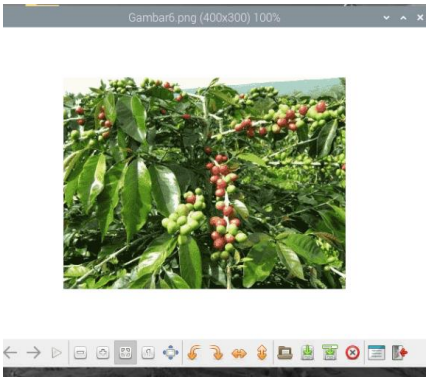


Gambar 14. Proses pengiriman data

Gambar 15 memperlihatkan bagaimana proses pengiriman data yang dilakukan pada raspberry pengirim. Bit crc yang ditambahkan pada data gambar adalah 8 bit CRC yaitu 11010110 (ditunjukkan pada kotak merah) .Data bit crc ini ditambahkan pada metadata gambar ekstensi PNG. Pada proses pengiriman terjadi pengiriman ulang data karena pada raspberry penerima mendeteksi terjadi *error* pada data yang diterima. Dengan adanya manajemen komunikasi dan pengecekan crc maka proses pengiriman tetap berhasil dilakukan dengan total bit yang dikirim adalah 386224. Adapun pada sisi penerima proses yang terjadi adalah sebagai berikut.

Gambar 16. Proses penerimaan data

Gambar 16 menunjukkan bagaimana proses penerimaan data yang terjadi. Pada penerimaan data yang pertama mendeteksi terdapat kesalahan dalam pengiriman data karena sisa bagi dari proses CRC tidak menghasilkan 0, maka penerima meminta proses pengiriman ulang data hingga data yang diperoleh adalah sesuai. Adapun gambar yang diterima sebagai berikut.



Gambar 17. Gambar yang diterima

Berikut hasil pengujian pengiriman 10 gambar yang disajikan dalam tabel.

Tabel 1. Hasil Pengujian Pengiriman Gambar

No	Ukuran Gambar	Status Pengiriman
1	30,9 KB	Berhasil
2	36,4 KB	Berhasil
3	38,3 KB	Berhasil
4	35,7 KB	Berhasil
5	38,4 KB	Berhasil
6	47,1 KB	Berhasil
7	53,1 KB	Berhasil
8	63,7 KB	Berhasil
9	122 KB	Gagal
10	255 KB	Gagal

Tabel 1 merupakan hasil pengujian pengiriman gambar dengan berbeda ukuran. Hasil pengujian menunjukkan bagaimana sistem hanya berhasil melakukan pengiriman gambar sebanyak 8 buah gambar yang mana kedelapan gambar tersebut berukuran <64KB. Untuk dua gambar lainnya dengan ukuran >64 KB gagal terkirim. Hal tersebut terjadi dikarenakan pada pengiriman data menggunakan protokol *user datagram protocol* pada arsitektur *Delay Tolerant Network* memiliki batasan *payload* data sesuai dengan MTU (*Maximum Transmission Unit)* adalah 64KB.

4. KESIMPULAN

Dari hasil pengujian terhadap sistem yang telah direalisasikan pada perangkat raspberry pi dapat disimpulkan bahwa secara fungsionalitas sistem yang telah dibangun telah berhasil difungsikan. Proses pengiriman gambar disertai dengan pengecekan *Cylic Redudancy Check* 8 bit dan terjadi pengiriman ulang data gambar untuk data yang terdeteksi *error* pada penerima.

Sistem handal digunakan untuk melakukan pengiriman gambar dengan ukuran <64KB.Untuk pengembangan selanjutnya dapat menambahkan proses pengiriman data dengan membagi gambar menjadi beberapa potongan gambar agar dapat melakukan transmisi data gambar dengan ukuran >64KB.

UCAPAN TERIMAKASIH

Penulis mengucapkan terima kasih kepada Unit Penelitian dan Pengabdian Masyarakat Politeknik Negeri Bandung sebagai penyandang dana sehingga penelitian ini dapat dilakukan.

DAFTAR PUSTAKA

[1] H. D. Laksana, A. Bhawiyuga, and K. Amron, "Rancang Bangun Perangkat Mobile Berbasis Delay Tolerant Network Sebagai Perantara Pengiriman Data Sensor Dari Lapangan Ke Pusat Data," *J. Pengemb. Teknol. Inf. dan Ilmu Komput. Univ. Brawijaya*, vol. 2, no. 9, pp. 3058–3064, 2018.

[2] K. Fall, "A delay-tolerant network architecture for challenged internets," p. 27, 2003, doi: 10.1145/863955.863960.

[3] P. G.Hutajulu, "Perbandingan Kinerja Routing Multi Copy Dan Routing First Contact Dengan Stationary Relay Node Pada Delay Tolerant Network (DTN)," Universitas Brawijaya, 2017.

[4] G. R Megiyanto and I. Fadillah, "Sistem Komunikasi Jaringan Wireless Menggunakan Raspberry Pi Dengan Arsitektur Delay Tolerant Network," in *Industrial Research Workshop and National Seminar*, 2020, pp. 429–434.

[5] M. Julius and F. Sirait, "Pendeteksian Bit Error Dalam Transmisi Data Dengan Menerapkan Cyclic Redudancy Check," *MEANS (Media Inf. Anal. dan Sist.*, vol. 3, no. 2, pp. 190–193, 2018.

[6] S. Wahyuni, A. Lubis, S. Batubara, and I. K. Siregar, "Implementasi Algoritma CRC 32 Dalam Mengidentifikasi Keaslian File," in *Seminar Nasional Royal (SENAR)*, 2018, pp. 1–6.

[7] Eko Sakti Pramukantoro, *Solusi Kesenjangan Informasi di Daerah Rural : Pendekatan Praktis*, Cetakan Pe. UB Press, 2018.

[8] E. Ismaredah *et al.*, "Perbandingan Kinerja ION-DTN Dan IBR-DTN Menggunakan Raspberry Pi Sebagai Router Delay Tolerant Network," 2019.

[9] A. W. Wardana, E. Firmansyah, and A. Suwastono, "PERANCANGAN DAN IMPLEMENTASI CYCLIC REDUDANCY CHECK-16 SEBAGAI METODE ERROR CHECKING PADA PESAN PROTOKOL MODBUS REMOTE TERMINAL UNIT BERBASIS MICROCONTROLLER UNIT," *J. Nas. Tek. Elektro*, vol. 5, no. 1, pp. 99–109, 2016.