DETEKSI SERANGAN PADA PROTOKOL MQTT IOT MENGGUNAKAN *RANDOM FOREST*

SKRIPSI

Oleh :
GALUH MUHAMMAD IMAN AKBAR
NIM. 17650016

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2023

DETEKSI SERANGAN PADA PROTOKOL MQTT IOT MENGGUNAKAN *RANDOM FOREST*

SKRIPSI

Oleh :
GALUH MUHAMMAD IMAN AKBAR
NIM. 17650016

Diajukan Kepada:
Fakultas Sains dan Teknologi
Universitas Islam Negeri (UIN) Malang
untuk Memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2023

HALAMAN PERSETUJUAN

DETEKSI SERANGAN PADA PROTOKOL MQTT IOT MENGGUNAKAN *RANDOM FOREST*

SKRIPSI

Oleh:

GALUH MUHAMMAD IMAN AKBAR
NIM. 17650016

Telah Diperiksa dan Disetujui untuk Diuji
Tanggal: 15 Juni 2023

Pembimbing I	Pembimbing II
Dr. M. Amin Hariyadi, M.T	Ajib Hanini, M.T
NIP. 19670018 200501 1 001	NIDT. 19840731 20160801 1 076

Mengetahui,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Fachrul Kurniawan, M.MT., IPM
NIP. 19771020 200912 1 001

HALAMAN PENGESAHAN

DETEKSI SERANGAN PADA PROTOKOL MQTT IOT MENGGUNAKAN *RANDOM FOREST*

SKRIPSI

Oleh:
GALUH MUHAMMAD IMAN AKBAR
NIM. 17650016

Telah Dipertahankan di Depan Dewan Penguji
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Tanggal: 15 Juni 2023

Susunan Dewan Penguji

Ketua Penguji	:	Johan Ericka Wahyu Prakasa, M.Kom NIP. 19831213 201903 1 004	( )
Anggota Penguji I	:	Dr. Cahyo Crysdian NIP. 19740424 200901 1 008	( )
Anggota Penguji II	:	Dr. M. Amin Hariyadi, M.T NIP. 19670018 200501 1 001	( )
Anggota Penguji III	:	Ajib Hanani, M.T NIDT. 19840731 20160801 1 076	( )

Mengetahui,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Fachrul Kurniawan, M.MT., IPM
NIP. 19771020 200912 1 001

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan dibawah ini:

Nama	: Galuh Muhammad Iman Akbar
NIM	: 17650016
Fakultas	: Sains dan Teknologi
Program Studi	: Teknik Informatika
Judul Skripsi	: Deteksi Serangan Pada Protokol MQTT IoT Menggunakan *Random Forest*

Menyatakan dengan sebenarnya bahwa Skripsi yang saya tulis ini benar-benar merupakan hasil karya sendiri, bukan merupakan pengambilalihan data, tulisan atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan Skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 15 Juni 2023
Yang membuat pernyataan,

Galuh Muhammad Iman Akbar
NIM. 17650016

HALAMAN MOTTO

“FOKUS APA YANG MENJADI KELEBIHANMU KEKURANGAN CUKUP DISYUKURI SAJA”

HALAMAN PERSEMBAHAN

Penulis mempersembahkan hasil skripsi yang telah dibuat kepada keluarga penulis sendiri yaitu ayah dan mama yang selalu memberikan motivasi, berdoa siang dan malam, dan memberikan contoh teladan terbaik dalam membentuk karakterku menjadi anak yang saleh dan berbakti, meskipun semua itu harus dilakukan dengan penuh perjuangan dan air mata yang semata-mata ingin menjadikan aku anak yang berakhlak dan mempunyai ilmu pengetahuan yang tinggi, tanpa ridho dan restu dari kedua orang tuaku penulis bukanlah siapa-siapa. Ucapan terima kasih juga kepada diri sendiri karena sudah berjuang sampai detik ini.

KATA PENGANTAR

Assalamu’alaikum Warahmatullahi Wabarakatuh

Puji syukur penulis ucapkan kepada Allah Subhanahu Wa Ta'ala, yang telah memberikan karunianya dan juga kesehatannya kepada penulis, sehingga penulis dapat menyelesaikan pendidikan di Universitas Islam Negeri Maulana Malik Ibrahim Malang, program studi Teknik Informatika dengan judul skripsi “**Deteksi Serangan Pada Protokol MQTT IoT Menggunakan Metode *Random Forest***” sebagai syarat kelulusan jenjang pendidikan S1, sholawat serta salam penulis haturkan kepada Baginda Nabi Muhammad SAW yang telah membawa dari agama islam dari zaman jahiliyah menuju zaman terang-menderang yakni (Ad-dinul islam).

Keberhasilan penulisan skripsi ini tidak lepas dari bantuan dan juga bimbingan dari segala pihak. Maka dari itu pada kesempatan ini penulis ingin mengucapkan terima kasih kepada:

1. Prof. Dr. H. M. Zainuddin, MA, selaku rektor UIN Maulana Malik Ibrahim Malang.
2. Dr. Sri Harini, M.Si, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Dr. Fachrul Kurniawan, M.MT, selaku Ketua Program Studi Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
4. Dr. Ir. Mokhamad Amin Hariyadi selaku dosen pembimbing 1 skripsi, yang selalu memberikan arahan, pengalaman dan juga bimbingan dengan ilmu yang beliau miliki.
5. Ajib Hanani, M.T selaku dosen pembimbing 2 skripsi, yang selalu meluangkan waktunya dan memberikan ilmu beserta masukan-masukan terhadap skripsi penulis.
6. Johan Ericka Wahyu Prakasa, M.Kom selaku ketua penguji, penulis ucapkan terima kasih sebanyak-banyaknya karena selalu memberikan

masukan dan juga ilmu yang beliau miliki kepada penulis, dan juga beliau sering memberikan ilmu dalam contoh nyata dibidang *cyber security.*

7. Dr. Cahyo Crysdian selaku penguji 1, penulis ucapkan terima kasih, karena selalu memberikan saran dan juga kritik positif terhadap skripsi yang telah dibuat oleh penulis.
8. Pemilik NIM 16650043, terima kasih telah membantu dan memberi dukungan kepada penulis.
9. Segenap sivitas akademika Program Studi Teknik Informatika, terutama seluruh dosen, terima kasih atas segenap ilmu dan bimbingannya.
10. Ayahanda dan Ibunda tercinta yang senantiasa memberikan doa dan restunya kepada penulis dalam menuntut ilmu.
11. Anggota keluarga dan kerabat yang selalu memberikan do'a dan semangat kepada penulis.
12. Semua pihak yang ikut membantu dalam menyelesaikan skripsi ini baik berupa materiil maupun moril yang tidak bisa penulis sebutkan satu persatu tanpa mengurasi rasa hormat dan terimakasih.

Penulis menyadari bahwa tulisan skripsi ini masih banyak kekurangan dan penelitian yang telah dilakukan masih jauh dari kata sempurna, maka dari itu penulis memohon maaf jika ada kekurangan selama proses pembuatan skripsi ini.

Wassalamu'alaikum Warahmatullahi Wabarakatuh

Malang, 15 Juni 2023

Penulis

DAFTAR ISI

HALAMAN JUDUL ..... **ii**
HALAMAN PERSETUJUAN ..... Error! Bookmark not defined.
HALAMAN PENGESAHAN ..... Error! Bookmark not defined.
PERNYATAAN KEASLIAN TULISAN ..... Error! Bookmark not defined.
HALAMAN MOTTO ..... **v**
HALAMAN PERSEMBAHAN ..... **vii**
KATA PENGANTAR ..... **viii**
DAFTAR ISI ..... **x**
DAFTAR GAMBAR ..... **xii**
DAFTAR TABEL ..... **xiii**
ABSTRAK ..... **xiv**
ABSTRACT ..... **xv**
الملخص ..... **xvi**
BAB I PENDAHULUAN **1**
1.1 Latar Belakang 1
1.2 Pernyataan Masalah 3
1.3 Tujuan Penelitian 3
1.4 Batasan Masalah 4
1.5 Manfaat Penelitian 4
BAB II STUDI PUSTAKA **5**
2.1 Internet of Things (IoT) 5
2.2 Protokol MQTT 6
2.3 *Random Forest* 10
BAB III METODE PENELITIAN **13**
3.1 Desain Penelitian 13
3.2 Desain Penelitian 13
3.2.1 Konfigurasi *Lab Hacking Environment* 14
3.2.2 Simulasi Peretasan pada *service* MQTT 14
3.2.3 Perekaman Lalu Lintas Jaringan 15
3.2.4 Ekstraksi Trafik Jaringan 15
3.2.5 Pelabelan 16
3.2 Perancangan Sistem dan *Lab Hacking Environment* 17
3.3 Praproses Data 19

3.4 Random Forest Pembuatan Model 20
3.5 *Random Forest* Deteksi Serangan MQTT 27
3.6 *Lab Hacking Environment* 30
BAB IV UJI COBA DAN PEMBAHASAN 32
4.1 Simulasi Uji Coba 32
4.1.1 Akuisisi Data 32
4.1.1.1 Konfigurasi *Lab Hacking Environment* 32
4.1.1.2 Skenario Peretasan 41
4.1.1.3 Perekaman Data Lalu Lintas Jaringan IoT 43
4.1.1.4 Ekstraksi Lalu Lintas Jaringan 45
4.1.1.5 Pelabelan 47
4.1.2 Praproses Data 48
4.1.3 *Random Forest* 48
4.1.4 Evaluasi Model 50
4.2 Hasil Uji Coba 51
4.3 Pembahasan 53
BAB V KESIMPULAN DAN SARAN 58
5.1 Kesimpulan 58
5.2 Saran 59
DAFTAR PUSTAKA

DAFTAR GAMBAR

Gambar 2. 1 Alur Random Forest 11
Gambar 3. 1 Perancangan Penelitian 13
Gambar 3. 2 Blok Diagram Perancangan Sistem 18
Gambar 3. 3 Diagram Alur Model *Random Forest* Data Training 21
Gambar 3. 4 *Script* untuk pembuatan model training 27
Gambar 3. 5 Diagram Alur Model *Random Forest* Data Testing 28
Gambar 3. 6 *Script* Proses Deteksi Serangan pada Data Testing 29
Gambar 3. 7 Desain Lab *Hacking Environment* 30
Gambar 4. 1 Dashboard VPS 33
Gambar 4. 2 Login menggunakan SSH *Service* 34
Gambar 4. 3 Instalasi Mosquitto 35
Gambar 4. 4 *Service* mosquito *Running* 35
Gambar 4. 5 *Setting Password* 36
Gambar 4. 6 Konfigurasi file mosquito 36
Gambar 4. 7 Modul ESP 8266 dan DHT11 38
Gambar 4. 8 Konfigurasi Koneksi di MQTT IoT Panel 38
Gambar 4. 9 Konfigurasi MQTT Suhu dan Kelembapan 39
Gambar 4. 10 Hasil dari pembacaan sensor DHT11 40
Gambar 4. 11 *Tools* Mqttsa untuk melakukan serangan *bruteforce* 41
Gambar 4. 12 Hasil Scanning *Tools Nmap* 42
Gambar 4. 13 Hasil *Bruteforce* 42
Gambar 4. 14 Data primer normal 43
Gambar 4. 15 Data primer *bruteforce* 44
Gambar 4. 16 Tampilan GUI CICflowMeter 45
Gambar 4. 17 Tampilan Menu *Offline* 46
Gambar 4. 18 Rescalling Baris Angka 48
Gambar 4. 19 Sampel Bootsrapping pada Data Primer 49
Gambar 4. 20 Model *Random Forest* 49

DAFTAR TABEL

Tabel 3. 1 Sampel Dataset MQTT-IOT-IDS2020 17
Tabel 3. 2 Nilai Baris Data *TotLen Bwd Pkts* 23
Tabel 3. 3 Daftar Nilai Berdekatan 24
Tabel 3. 4 Data *Partition* 24
Tabel 3. 5 Hasil Perhitungan 26
Tabel 3. 6 *Lab Hacking Environment* 31
Tabel 4. 1 *Script* Modul untuk ESP8266 Dalam Bentuk JSON 37
Tabel 4. 2 Ruang lingkup Perangkat dan Jaringan 40
Tabel 4. 3 Data Primer 45
Tabel 4. 4 Hasil Ekstraksi Data Primer 46
Tabel 4. 5 Parameter Protokol MQTT 47
Tabel 4. 6 Informasi Data Lengkap Penelitian 47
Tabel 4. 7 Matriks Konfusi *Actual* dan *Prediction* 51
Tabel 4. 8 Hasil Uji Coba Random Forest *Primary Data* 52
Tabel 4. 9 Hasil *Output* Random Forest *Primary Data* 52
Tabel 4. 10 Hasil Uji Coba Random Forest *Secondary Data* 53
Tabel 4. 11 Hasil *Output* Random Forest *Secondary* Data 53
Tabel 4. 12 Hasil Uji Coba *Primary Data* dan *Secondary Data* 54
Tabel 4. 13 *Measurement Results Primary Data* dan *Secondary Data* 54

ABSTRAK

Akbar, Galuh Muhammad Iman. 2023. **Deteksi Serangan Pada Protokol MQTT IOT Menggunakan *Random Forest***. Skripsi. Program Studi Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang. Pembimbing: (1): Dr. M. Amin Hariadi M. T. (II): Ajib Hanani, M.T.

Kata Kunci: *Bruteforce*, Deteksi Serangan IoT, *Random Forest*.

Bruteforce merupakan salah satu metode dari teknik *hacking* yang melancarkan serangan dengan cara *guessing* username dan password pada sistem yang menjadi target penyerangan. Serangan *bruteforce* pada protokol MQTT merupakan serangan yang sering terjadi pada IoT, sehingga perlu adanya deteksi serangan pada protokol MQTT untuk mengetahui *traffic* normal dan *traffic bruteforce*. *Random Forest* dipilih karena metode tersebut dapat melakukan klasifikasi data secara banyak dalam waktu yang relatif singkat dan hasil dari *Random Forest* dapat meningkatkan akurasi dan dapat mencegah *overfitting* pada proses klasifikasi data. Penelitian ini menggunakan dua jenis data yaitu: data primer yang diambil dari lab *hacking environment* dan data sekunder yang diambil dari *dataset* IEEE Data Port MQTT-IOT-IDS2020, proses uji coba menggunakan 7 parameter penelitian yaitu *totlen bwd pkts, bwd pkt len max, bwd pkt len mean, bwd seg size avg, subflow bwd byts, init bwd win byts, label.* Pada data primer didapatkan nilai dari hasil pengukuran yaitu akurasi 99,55%, presisi 100%, recall 99,54% dan f-measure 99,77% durasi yang dibutuhkan untuk mendapatkan hasil tersebut dengan baris data sebanyak 1796 baris yaitu selama 0 detik. Sedangkan untuk data sekunder peneliti mendapatkan akurasi sebesar 99.77%, presisi sebesar 100%, recall sebesar 99.43%, f-measure sebesar 98.71% durasi yang dibutuhkan untuk mendapatkan hasil tersebut dengan baris data sebanyak 85002 baris yaitu selama 62 detik.

ABSTRACT

Akbar, Galuh Muhammad Iman. 2023. **Attack Detection MQTT IOT Protocol Using *Random Forest***. Thesis. Informatics Engineering Study Program, Faculty of Science and Technology, State Islamic University (UIN) Maulana Malik Ibrahim Malang. Supervisors: (1): Dr. M. Amin Hariadi M. T. (II): Ajib Hanani, M.T.

Bruteforce is a hacking technique in which the target system's username and password are guessed before the attack is launched. Because a *bruteforce* attack on the MQTT protocol is a common IoT attack, it is necessary to identify both normal and *bruteforce* traffic by detecting attacks on the MQTT protocol. The results of *Random Forest* can improve accuracy and prevent overfitting in the data classification process, so it was chosen because it can classify a lot of data in a short amount of time. There are two types of data used in this study: The trial process makes use of 7 research parameters, which are *total bwd pkts, bwd pkt len max, bwd pkt len mean, bwd seg size avg, subflow bwd byts, init bwd win byts, label*. Secondary data from the IEEE Data Port MQTT-IOT-IDS2020 dataset. The values derived from the measurement results for the primary data are 99,55% accuracy, 100% precision, 99,54% recall, and 99,77% f-measure. Concerning the secondary data, the researcher was able to obtain an f-measure of 98,71%, an accuracy of 99,77%, a precision of 100%, a recall of 99,43%, and the duration necessary to obtain these results with 85002 data lines, which was 62 seconds.

Keywords: *Bruteforce*, IoT *Attack Detection, Random Forest*.

الملخص

أكبر، غالوه محمد إيمان. 2023. كشف الهجوم على بروتوكول MQTT IOT باستخدام الغابة العشوائية (*Random Forest*). البحث الجامعي. قسم الهندسة المعلوماتية كلية العلوم والتكنولوجيا جامعة مولانا مالك إبراهيم الإسلامية الحكومية مالانج. المشرف: (١) الدكتور محمد أمين هاريادي الماجستير. (٢) عجيب حناني الماجستير. **الكلمات المفتاحية:** هجوم القوة الغاشمة (*Bruteforce*)، الغابة العشوائية(*Random Forest*)، كشف الهجوم على IoT

هي طريقة لتقنية القرصنة التي تشن هجومًا عن طريق تخمين اسم المستخدم وكلمة المرور على النظام المستهدف للهجوم. هجوم Bruteforce هو هجوم يحدث غالبًا على إنترنت الأشياء ، لذلك من الضروري اكتشاف الهجمات على بروتوكول MQTT على بروتوكول Bruteforce لأن هذه الطريقة يمكن أن تصنف الكثير Random Forest لاكتشاف حركة المرور العادية وحركة المرور القاسية. تم اختيار MQTT الدقة ويمكن أن تمنع التلاعب الزائد في عملية تصنيف Random Forest من البيانات في وقت قصير نسبيًا ويمكن أن تحسن النتائج من البيانات. تستخدم هذه الدراسة نوعين من البيانات ، وهما: البيانات الأولية المأخوذة من معمل بيئة القرصنة والبيانات الثانوية المأخوذة من مجموعة بيانات IEEE Data Port MQTT-IOT-IDS2020 ، وتستخدم العملية التجريبية 7 معلمات بحثية وهي totlen bwd pkts و bwd pkt len max، bwd pkt len mean، bwd seg size avg، subflow bwd byts، init bwd win byts، label. في البيانات الأولية ، القيم التي تم الحصول عليها من نتائج القياس هي دقة 99.55٪ ، دقة 100٪ ، استرجاع 99.54٪ و f أما بالنسبة للبيانات الثانوية ، فقد حصل الباحث على دقة 99.77٪ ، دقة 100٪ ، استدعاء 99.43٪ ، مقياس .f قياس ٪99.77 المدة المطلوبة للحصول على هذه النتائج بـ 85002 خط بيانات ، أي 62. ثواني ، ٪98.71.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Teknologi merupakan sebuah sarana yang digunakan oleh manusia untuk membantu dalam kelangsungan dan kenyamanan hidup. Perkembangan teknologi yang ada saat ini akan mengalami perkembangan yang sangat pesat hingga masa yang akan datang. Saat ini perkembangan teknologi mengarah ke basis internet yang bertujuan untuk meningkatkan manfaat dari koneksi yang berguna untuk *sharing file*, melakukan *remote control*, melakukan analisis. Teknologi yang mengarah ke basis internet yaitu IoT (*Internet of Things*). IoT merupakan sebuah konsep di mana suatu objek diberikan akses ke internet yang berguna untuk transfer data, melakukan kontrol pada sistem, dan lain-lain yang berguna untuk meringankan pekerjaan. *Internet of Things* (IoT) adalah sebuah konsep yang menggunakan sebuah jaringan internet sebagai jaringan infrastruktur utama yang mengkoneksikan objek – objek tertentu (Widagdo & Rofik, 2019).

Perkembangan teknologi selalu tidak selalu berbanding lurus dengan keamanan informasi, ketika teknologi semakin berkembang maka dibutuhkan teknologi yang menjamin data pengguna dapat diakses tanpa adanya gangguan. Setiap tahun dampak dari serangan siber banyak memberikan kerugian pada sektor ekonomi berdasarkan data dari penyedia layanan antivirus Norton Symantec dari tahun 2015 sampai dengan Februari 2016, tercatat kejahatan online terjadi di Indonesia menyebabkan total kerugian Rp 194.6 Miliar (Symantec,2016) dan kejahatan yang sering terjadi yaitu seperti kejahatan *carding* (credit card fraud),

ATM/EDC *skimming* pada awal tahun 2010, *cracking*, *hacking*, *malware* (*virus/worm/trojan/bots*), *cyber-squatting*, pornografi, perjudian online, *phising* (internet banking fraud), transnasional *crime* (mafia, perdagangan narkoba, terorisme, *human trafficking*, *money laundering*, *underground economy*) (IDSIRTII/CC, 2017).

Pemanfaatan salah satu bidang teknologi IoT dan juga penerapan *cyber security* diharapkan dapat membantu para IoT dan juga praktisi *cyber security* dalam mendeteksi serangan *cyber* yang sering terjadi khususnya pada bidang IoT. Salah satu teknik untuk mendeteksi serangan menggunakan metode *Random Forest.*

Random Forest (RF) merupakan sebuah metode yang dipakai untuk melakukan klasifikasi dengan membuat beberapa pohon klasifikasi untuk menyelesaikan masalah. RF dapat meningkatkan tingkat akurasi dikarenakan pada proses tersebut terdapat pemilihan data secara acak dalam membangkitkan simpul *node child* untuk setiap *node* diatasnya dan dilakukan akumulasi hasil klasifikasi dari setiap pohon, kemudian hasil dari klasifikasi yang paling sering muncul yang akan dipilih.

Metode ini umumnya digunakan untuk mengejar opsi yang memiliki ukuran atau standar yang berbeda. Untuk tujuan pelatihan, Random Forest memanfaatkan objek berupa lalu lintas data pada jaringan sebelumnya. Ini memastikan bahwa semakin banyak data ditambahkan, akurasi yang dicapai akan meningkat secara signifikan.

Dalam Al-Qur’an telah ditunjukan tentang balasan kebaikan dan keburukan, seperti firman Allah SWT dalam surah Al-Qashash [28] ayat ke 84:

مَنْ جَآءَ بِالْحَسَنَةِ فَلَهٗ خَيْرٌ مِّنْهَاۚ وَمَنْ جَآءَ بِالسَّيِّئَةِ فَلَا يُجْزَى الَّذِيْنَ عَمِلُوا السَّيِّاٰتِ اِلَّا مَا كَانُوْا يَعْمَلُوْنَ

“*Barangsiapa datang dengan (membawa) kebaikan, maka dia akan mendapat (pahala) yang lebih baik daripada kebaikannya itu; dan barang siapa datang dengan (membawa) kejahatan, maka orang-orang yang telah mengerjakan kejahatan itu hanya diberi balasan (seimbang) dengan apa yang dahulu mereka kerjakan.*”*(Q.S. Al-Qashash:84).*

Pada ayat diatas dijelaskan bahwa setiap orang yang melakukan kebaikan akan selalu mendapatkan ganjaran berupa pahala, dan setiap kejahatan maka akan diberikan balasan sesuai dengan apa yang diperbuat. Maka dari itu kejahatan dalam bentuk apapun harus dicegah dan juga meminimalisir dampaknya merupakan bentuk dari tolong menolong.

1.2 Pernyataan Masalah

Berikut adalah pernyataan masalah peneliti yang didasarkan pada uraian masalah sebelumnya:

Seberapa tinggi akurasi, presisi, recall, f-measure dan berapa lama rentang perhitungan durasi dengan menggunakan algoritma *Random Forest* dalam membedakan serangan terhadap protokol MQTT IoT?

1.3 Tujuan Penelitian

Seberapa tinggi algoritma *Random Forest* dalam mengukur durasi, akurasi, presisi, recall, dan f-measure pada serangan protokol MQTT IoT.

1.4 Batasan Masalah

1. Penelitian menggunakan dua jenis data yaitu data primer dan data sekunder. Data primer akan dibagi menjadi dua bagian yaitu data untuk pelatihan dan data untuk pengujian. Data sekunder kemudian akan dibagi menjadi dua bagian yaitu data untuk pelatihan dan data untuk pengujian.
2. Dataset menggunakan dari IEEE Data Port (MQTT-IOT-IDS2020: MQTT Internet of Things Intrusion Detection Dataset).
3. Simulasi Peretasan dan deteksi serangan pada protokol MQTT IoT.

1.5 Manfaat Penelitian

Perhitungan *Random Forest* menghasilkan nilai akurasi yang tinggi, Sehingga penelitian ini diharapkan bermanfaat bagi beberapa pihak sebagai berikut:

1. Manfaat bagi developer IoT, dengan adanya sistem deteksi serangan pada perangkat IoT diharapkan developer dapat melakukan evaluasi pada sistem dengan cara melakukan *hardening* pada perangkat IoT sehingga ancaman pada serangan IoT dapat diminimalisir.
2. Manfaat bagi developer *software enginnering*, menjadi masukan bagi developer untuk mengembangkan sistem yang bertujuan melakukan deteksi serangan pada sensor IoT yang dapat digunakan secara meluas.

BAB II

STUDI PUSTAKA

2.1 Internet of Things (IoT)

Internet of Things atau IoT adalah sebuah konsep di mana sebuah objek dapat memiliki akses ke internet yang berguna untuk transfer data, melakukan kontrol pada sistem yang memungkinkan untuk menghubungkan sebuah peralatan atau mesin dengan jaringan internet sehingga ada proses timbal balik antara *user* dan mesin yang memungkinkan mesin untuk bertindak berdasarkan informasi yang diperoleh secara independen (Arafat et al., 2016)

Internet dapat digambarkan sebagai jaringan komunikasi yang menghubungkan individu ke informasi, sedangkan *Internet of Things* (IoT) adalah sistem yang saling berhubungan yang secara khusus menangani item fisik yang mampu dengan berbagai tingkat kemampuan pemrosesan, penginderaan, dan aktuasi yang berbagi kemampuan untuk beroperasi dan berkomunikasi melalui Internet sebagai platform bersama (Hussein, 2019)

Tujuan utama dari *Internet of Things* adalah untuk memungkinkan objek terhubung dengan objek lain, individu, kapan saja atau dimana saja menggunakan jaringan, jalur, atau layanan apa pun. *Internet of Things* (IoT) secara bertahap dianggap sebagai fase berikutnya dalam evolusi Internet. IoT akan memungkinkan perangkat biasa dipautkan ke internet untuk mencapai tujuan berbeda yang tak terhitung jumlahnya. Saat ini, diperkirakan baru 0,6% perangkat yang bisa menjadi bagian dari IoT yang telah terkoneksi sejauh ini (Ryan & Watson, 2017).

2.2 Protokol MQTT

Penelitian ini menjelaskan bahwa solusi untuk melindungi saluran komunikasi dan mengusulkan Value-to-Keyed-Hash Message Authentication Code (Value-to-HMAC) sebagai metode alternatif untuk mencapai kerahasiaan informasi. Dengan cara ini, kerahasiaan informasi, integritas, peningkatan kinerja, dan kecepatan data dicapai, sambil memastikan bahwa node target dapat membaca pesan tersebut, dengan menggunakan "paho mqtt" dan Onion Omega2 sebagai client di MQTT untuk mengakses mosquito, dan menggunakan python untuk mengimplementasikan pengujian, dan penyebaran metode pemetaan Value-to-HMAC untuk verifikasi. Metode ini menggunakan data sumber dan kunci untuk membuat tanda tangan melalui fungsi HMAC dan mengirim hash ke tujuan. Pengirim menggunakan kunci dan data untuk menghasilkan kode verifikasi, dan penerima menggunakan tabel yang dihasilkan untuk memetakan nilai ke tanda tangan untuk memulihkan data. Tabel berisi nilai, data, dan ringkasan nilai HMAC. Penerima mengambil nilai data dengan mencari di tabel untuk menemukan kecocokan. Bahkan jika klien berlangganan ke topik yang sama, kunci yang berbeda perlu didistribusikan. Untuk penyerang, ketika kunci tidak diketahui, maka perlu untuk menggabungkan semua kemungkinan nilai transmisi kunci dalam sistem untuk menghasilkan hash untuk mendapatkan pemetaan tabel, yang hampir tidak mungkin dicapai. Metode pemetaan HMAC cocok untuk lingkungan pesan yang dapat diprediksi. Ini diperoleh dengan membandingkan waktu nilai data yang berbeda. Sekalipun jumlah datanya sangat besar, waktu eksekusi fungsi pengambilan tidak akan memengaruhi kinerja (Dinculeană & Cheng, 2019).

Penelitian ini menjelaskan pembuatan klasifikasi model dan pengembangan model pembelajaran mesin IDS yang diterapkan ke IoT. Metode deteksi utama adalah Untuk deteksi berbasis tanda tangan dan deteksi berbasis anomali. Pembelajaran mesin menggunakan tiga metode yang berbeda untuk mengklasifikasikan kumpulan data di *Internet of Things* yang secara bersamaan mengalami tiga serangan berbeda dan data normal. Untuk memenuhi klasifikasi anomali di lingkungan IoT, peneliti menggunakan MQTT untuk membangun kumpulan data dan melakukan serangan DOS, serangan man-in-the-middle, dan membuat tiga file dengan format CSV untuk menandai apakah data tersebut diserang atau tidak. Metode klasifikasi menggunakan XGBoost, LSTM, GRU, dan menggunakan SGD untuk mengoptimalkan metode integratif berfungsi untuk menyelesaikan masalah perpindahan variable internal. Skor F-Beta dan akurasi klasifikasi digunakan sebagai indikator evaluasi. Akhirnya, melalui eksperimen dapat disimpulkan bahwa model ini dapat digunakan pada IDS untuk IoT (Alaiz-Moreton et al., 2019).

Penelitian ini menjelaskan tentang analisis data anomali. *Anomali Detection Engine (ADE)* merupakan perangkat kerja yang diuraikan sebagai berikut: 1. Data mentah 2. Representasi deret waktu 3. Transformasi rangkaian waktu 4. Membuat model deret waktu 5. Evaluasi 6. Pelabelan 6. Peringatan. Data mentah biasanya dalam bentuk format CSV, TTL, JSON kemudian menangkap representasi deret waktu diikuti oleh algoritma khusus yang mengarah pada evaluasi, deteksi anomali, dan pelabelan. Dalam beberapa situasi yaitu pada aplikasi dalam wujud nyata deteksi diikuti oleh peringatan. ADE yang akurat dan andal bergantung pada tiga

faktor pemilihan utama yaitu kualitas titik data, transformasi deret waktu, dan tempat analitik dijalankan sifat lingkungan jaringan yang heterogen, konvergensi komunikasi dan protocol data di IoT memerlukan perhatian khusus dalam hal pengembangan perangkat lunak deteksi anomali. Properti ini tidak mendukung deret waktu nyata karena lebih banyak komputasi pada tingkat ADE akan mempengaruhi keakuratan ADE. Dari perspektif pengembangan perangkat lunak, triknya mirip alat *mining* data yang terletak di *server* database. Setelah dataset di kompilasi, pengguna dapat memilih alat statistik yang paling sesuai. Namun, penyedia solusi ERP mungkin akan mengusulkan ADE sebagai modul yang dapat disesuaikan yang paling sesuai dengan kebutuhan pelanggan. Pekerjaan masa depan akan menyelidiki tantangan dari eksperimen empiris dan bagaimana deteksi anomali dapat diterjemahkan sebagai layanan dalam komputasi awan (Mohamudally & Peermamode-Mohaboob, 2018).

Penelitian ini mengusulkan pendekatan baru untuk deteksi anomali di IoT berdasarkan kombinasi dua algoritma pembelajaran mesin, *Inverse Weight Clustering* (IWC) dan algoritma keputusan C4.5. IWC adalah versi algoritma k-means yang disempurnakan yang dapat digunakan untuk mengelompokkan data secara efektif ke dalam grup berdasarkan kesamaan antara data ini. C4.5 merupakan algoritma pohon keputusan yang dapat digunakan untuk membangun pohon keputusan untuk mengklasifikasikan data. Untuk membangun model deteksi anomali yang diusulkan dan memenuhi tujuan kedua dalam penelitian ini, dataset untuk IoT yang tidak berlabel diperoleh dari Intel Lab. Data telah diproses sebelumnya sehingga hanya berisi dua atribut; suhu dan kelembapan. Jumlah record

dalam dataset juga dikurangi menjadi 7526 record. Dataset dikelompokkan menggunakan IWC menjadi dua kelompok; setiap kelompok diidentifikasi dengan ID dengan nilai "1" atau "2". Dataset tersebut kemudian dipecah menjadi dua set data; dataset training yang berisi record berlabel dan mewakili 66% dataset, dan data testing yang berisi record *unlabeled* dan mewakili 34% dari kumpulan data. Dataset pelatihan digunakan oleh C4.5 untuk membangun pohon keputusan, yang kemudian diterapkan pada data pengujian. Hasil penelitian menunjukkan bahwa akurasi keseluruhan sistem yang diusulkan dalam memprediksi anomali dan normalitas pada IoT adalah 97%, dan tingkat kesalahan klasifikasi sebesar 2,1%. Sensitivitas sistem dalam mendeteksi anomali melebihi 98% dan rasio *false positif* 2,1% (Alghuried, 2017).

Penelitian ini memperkenalkan metode untuk mendeteksi komunikasi anomali dalam jaringan IoT menggunakan autoencoders sparse. Autoencoder sparse yang berbeda dilatih untuk mempelajari profil komunikasi yang sah dari setiap jenis perangkat IoT yang ada di jaringan. Fitur yang digunakan adalah statistik ukuran paket N pertama yang dikirim dan diterima, bersama dengan statistik waktu kedatangan antar paket. Jaringan rumah pintar eksperimental menyiapkan model nilai kerja yang bergantung pada nilai N, model ini mencapai tingkat deteksi serangan mulai dari 86,9% hingga 91,2% dan tingkat *false positif* mulai dari 0,1% hingga 0,5%. Setelah komunikasi terdeteksi sebagai anomali, komunikasi dapat terputus tanpa harus benar-benar mengganggu layanan yang disediakan oleh perangkat (Gkoulalas-Divanis et al., n.d.).

2.3 *Random Forest*

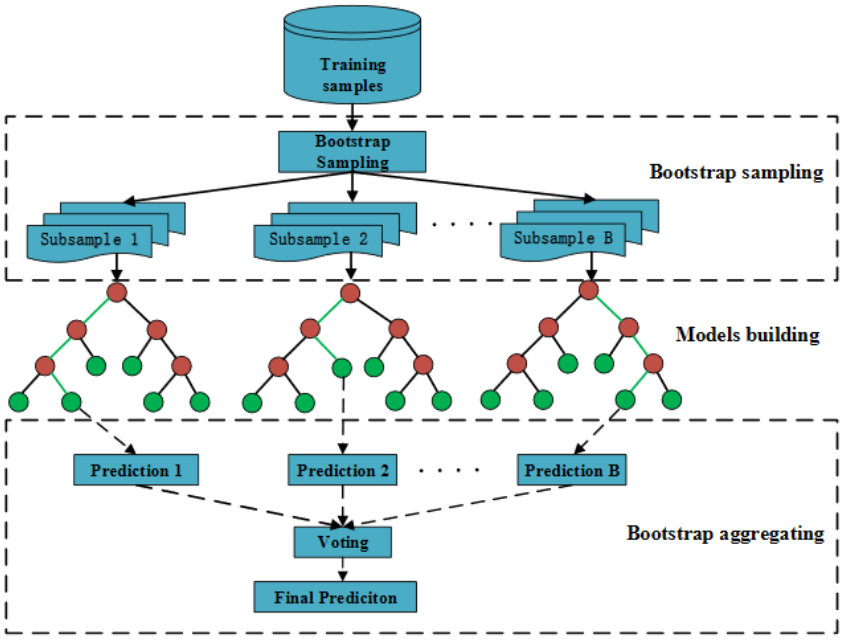
Random Forest adalah algoritma pembelajaran mesin dengan metode ansambel yang dapat digunakan untuk klasifikasi dan regresi. Hutan Acak terdiri dari kumpulan pohon keputusan yang terkait dengan sampel bootstrap yang dikumpulkan dari dataset (Amalia et al., 2022).

Metode ini digunakan untuk membuat sebuah pohon keputusan yang terdiri dari *internal, node root node, dan leaf node* dengan mengambil beberapa atribut dan data secara acak sesuai ketentuan yang diberlakukan. *Internal node* adalah simpul percabangan, dimana *node* ini mempunyai output minimal dua dan hanya ada satu input. *Root node* merupakan simpul yang terletak paling atas, atau biasa disebut sebagai akar dari pohon keputusan. Sedangkan *leaf node* atau *terminal node* ialah simpul terakhir yang hanya mempunyai satu input tanpa memiliki output. Nilai *entropy* dihitung pada saat pohon keputusan dibuat dan digunakan sebagai penentu tingkat ketidakmurnian atribut dan nilai *information gain.* (Sulistyo Nugroho & Nova Emiliyawati, n.d.).

Ada tiga aspek penting dalam sebuah metode *random forest* (Primajaya & Sari, 2018), yaitu:

1. Melakukan bootstrap sampling untuk membangun pohon prediksi.
2. Masing-masing pohon keputusan memprediksi dengan prediktor acak.
3. *Random Forest* melakukan prediksi dengan mengombinasikan hasil dari setiap pohon keputusan dengan cara majority vote untuk klasifikasi atau rata-rata untuk regresi.

Gambaran alur *random forest* dapat dilihat *pada* Gambar 2.1 berikut.



Gambar 2. 1 Alur Random Forest

Berdasarkan studi lengkap ditemukan bahwa seseorang harus menggunakan teknik *random forest* pada jenis dataset ini untuk menyelesaikan serangan cyber pada jaringan IoT karena *random forest* memprediksi serangan *Data Type Probing (DTP), Malicious Control (MC), Malicious Operation (MO), Scan (SC), Spying (SP), Wrong Setup (WS)* secara akurat dibandingkan dengan pendekatan yang lain. Pada kasus DoS dan Normal, dapat juga memprediksi sampel lebih akurat daripada teknik lain. Oleh karena itu, berdasarkan estimasi ini, dapat disimpulkan bahwa *random forest* adalah teknik terbaik untuk penelitian ini. Tidak ada algoritma baru yang dibuat pada kumpulan data pada penelitian ini. Oleh karena itu, studi lebih lanjut sangat diperlukan untuk mengembangkan algoritma pendeteksian yang kuat. Lebih banyak analisis harus diberikan pada keseluruhan desain kerangka kerja. Selain itu, pekerjaan ini didasarkan pada data lingkungan virtual. Dalam kasus data

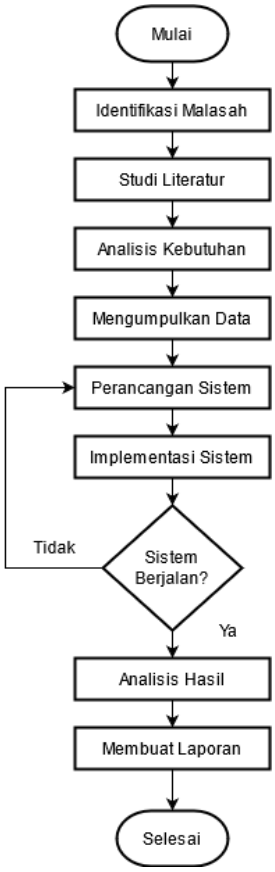
real-time, mungkin ada masalah yang berbeda. Diperlukan studi yang lebih empiris pada masalah ini dengan fokus pada data real-time. Pada jaringan loT, layanan mikro berperilaku berbeda pada waktu yang berbeda yang menyebabkan penyimpangan perilaku normal pada layanan IoT sehingga menimbulkan anomali. Studi lebih lanjut diperlukan untuk menafsirkan masalah ini secara lebih mendalam. Dalam studi ini, *random forest* bekerja secara komparatif lebih baik dengan akurasi 99,4%. Namun, tidak menjamin bahwa dalam kasus data besar dan masalah lain yang tidak diketahui, *random forest* akan bekerja dengan cara ini. Oleh karena itu, dibutuhkan lebih banyak studi (Hasan et al., 2019).

BAB III

METODE PENELITIAN

3.1 Desain Penelitian

Tujuan dari perancangan penelitian ini adalah untuk memudahkan peneliti dalam mencatat langkah-langkah yang akan dilakukan.



Gambar 3. 1 Perancangan Penelitian

3.2 Desain Penelitian

Tahap ini yaitu melakukan akuisisi data peneliti menggunakan dua data yaitu data primer yang peneliti ambil datanya dari lab yang telah dibuat, sedangkan untuk data sekunder peneliti ambil dari penelitian sebelumnya. Data primer akan dibagi menjadi 2 bagian yaitu data *training* dan data *testing*, begitu juga dengan data

sekunder akan dibagi menjadi 2 bagian yaitu data *training* dan data *testing*. Data *training* akan digunakan untuk membuat model yang dibangun oleh peneliti sedangkan data *testing* akan digunakan untuk menguji model yang telah dibuat.

Berikut merupakan konfigurasi *Lab Hacking Environment* untuk melakukan akuisisi data:

3.2.1 Konfigurasi *Lab Hacking Environment*

Bangun sebuah broker *service* MQTT pada Ubuntu *server* dan melakukan konfigurasi pada Ubuntu *server*. *Service* MQTT ini akan menghubungkan antara broker dengan client. Dalam praktik peretasan peneliti akan menggunakan sistem operasi Kali Linux berbasisis Debian sebagai peretas yang akan melakukan penyerangan ke *service* MQTT.

3.2.2 Simulasi Peretasan pada *service* MQTT

Setelah selesai dikonfigurasi, Langkah berikutnya yaitu melakukan simulasi peretasan ke *service* MQTT dengan cara melakukan serangan *Bruteforce*. Ubuntu *server* digunakan dalam menjalankan *service* MQTT mosquito dan sistem operasi peretas menggunakan Kali Linux. Ketika *service* broker sudah dijalankan peneliti akan melihat *service* pada mosquito. Pada sistem operasi Kali Linux yang dijalankan oleh peneliti untuk menjalankan *tool*s *mqttsa* yang berfungsi untuk menjalankan *Bruteforce Credential* dengan cara menembak *credential* pada *service* MQTT dengan menggunakan *directory attack.*

3.2.3 Perekaman Lalu Lintas Jaringan

Proses *capture traffic* menggunakan *Tools* Wireshark. Ketika proses peretasan *service* MQTT berlangsung, peretas menggunakan *tools mqttsa*. Selama proses peretasan berjalan aplikasi Wireshark akan merekam trafik jaringan yang berjalan pada *service* MQTT tersebut. Lakukan konfigurasi untuk mengatur jalannya *capturing* lalu lintas pada perangkat yang terhubung ke WiFi, hal ini karena *service* MQTT akan terhubung ke WiFi.

Wireshark akan melakuka proses perekaman lalu lintas jaringan antara *service* MQTT sebagai broker dengan perangkat mobile apps IoT sebagai penerima, dalam hal ini yang bertindak sebagai client adalah *tools mqttsa,* dengan begitu tools ini menajalankan peran yang ganda yaitu sebagai peretas.

Perekaman pada trafik jaringan menggunakan *tools* wireshark bertujuan mengambil data pengujian, karena dibutuhkannya data dari peretas, response dari *service* MQTT dan data normal. Data normal diperoleh dari *subscribe* normal client ke *service* MQTT dan response dari *service* MQTT tanpa menggunakan *tools mqttsa* trafik tersebut direkam oleh wireshark yang kemudian data tersebut diperlukan sebagai trafik normal pada jaringan. Semua data yang berhasil di *capture* akan disimpan dengan format extension PCAP.

3.2.4 Ekstraksi Trafik Jaringan

Setelah dilakukan proses capturing data pada lab yang telah dibuat oleh peneliti, data tersebut disimpan dalam bentuk ektensi dengan format .PCAP file yang sudah tersimpan tersebut selanjutnya akan digunakan proses extraksi dengan menggunakan tools CICFlowmeter tools tersebut berguna untuk melakukan

ekstrasi data yang bersifat anomali, hasil dari ekstraksi tersebut akan memberikan output berupa file dengan format CSV pada file tersebut terdapat banyak parameter peneliti akan menggunakan hanya 7 parameter penelitian saja.

3.2.5 Pelabelan

Data *record* lalu lintas yang diperolah dan yang telah diubah menjadi berformat CSV, akan diubah kembali menjadi XLSX. Data yang didapatkan dari hasil perekaman lalu lintas pada jaringan masih belum memliki label untuk membedakan *trafik* data normal dan juga *trafik* data peretasan. Setelah dilakukannya tahap peretasan peneliti masuk ke tahap pelabelan secara manual. Peneliti mengambil lalu lintas jaringan secara independen pada lalu lintas normal dan lalu lintas peretasan, sehingga peneliti akan mendapatkan lalu lintas yang sesuai dengan aktifitas *real* serangan, penamaan adalah langkah terakhir menuju akuisisi jaringan.

Data yang dikumpulkan dari penelitian sebelumnya disebut data sekunder. Peneliti memutuskan untuk menggunakan *Dataset* dari IEEE Data Port (MQTT-IOT-IDS2020: MQTT *Internet of Things* Intrusion Detection Dataset). *Dataset* tersebut berisi lalu lintas jaringan yang normal dan juga data serangan pada protokol MQTT, peneliti sebelumnya membuat lab untuk pengambilan data MQTT IDS data tersebut mirip seperti data serangan pada dunia nyata. Peneliti akan memakai data dalam format .PCAPNG yang kemudian dikonversi menjadi format .PCAP data tersebut akan dilakukan ekstraksi dengan *tools* CICFlowmeter sehingga memiliki *output* format dengan ektensi CSV kemudian akan di konfersi menjadi data XLSX untuk melakukan training data. Berikut merupakan sampel dataset yang diambil

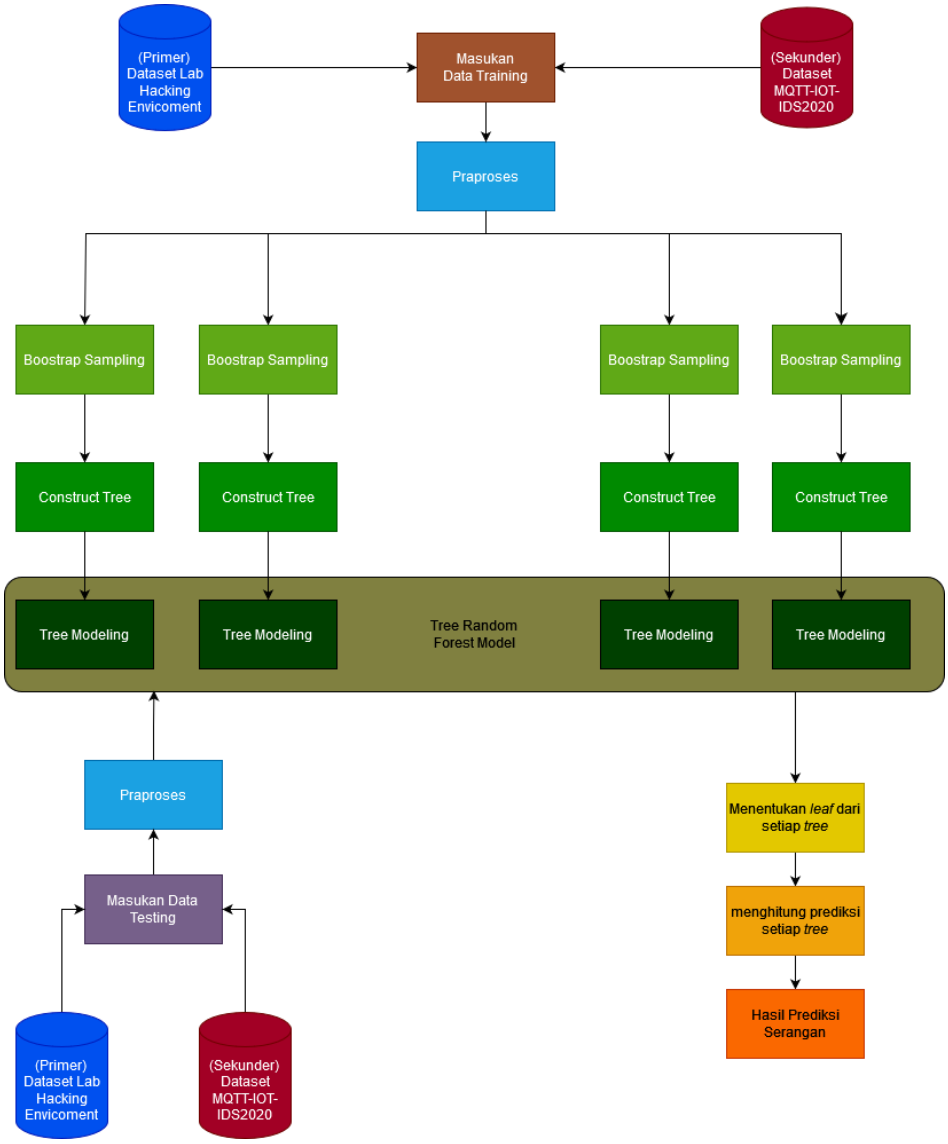
dari *dataset* MQTT-IOT-IDS2020 pada Tabel 3.1

Tabel 3. 1 Sampel Dataset MQTT-IOT-IDS2020

TotLen Bwd Pkts	Bwd Pkt Len Max	Bwd Pkt Len Mean	Bwd Seg Size Avg	Subflow Bwd Byts	Init Bwd Win Byts	Label
54.0	50.0	27.0	27.0	54	1810	Normal
54.0	50.0	27.0	27.0	54	1810	Normal
54.0	50.0	27.0	27.0	54	1810	Normal
53.0	49.0	26.5	26.5	53	1810	Normal
66.0	30.0	6.6	6.6	66	1810	Bruteforce
45.0	29.0	9.0	9.0	45	1810	Bruteforce
66.0	30.0	6.6	6.6	66	1810	Bruteforce
48.0	32.0	9.6	9.6	48	1810	Bruteforce
63.0	31.0	7.0	7.0	63	1810	Bruteforce
41.0	29.0	10.25	10.25	41	1810	Bruteforce
43.0	31.0	10.75	10.75	43	1810	Bruteforce
68.0	32.0	6.8	6.8	68	1810	Bruteforce

3.2 Perancangan Sistem dan *Lab Hacking Environment*

Pada sub-bab ini akan membahas perancangan sistem penelitian yang dibuat oleh peneliti. Dapat dilihat pada Gambar 3.2 berikut ini:



Gambar 3. 2 Blok Diagram Perancangan Sistem

Pada diagram blok perancangan sistem yang telah disajikan pada Gambar 3.2, peneliti memisahkan antara data primer yang dibuat oleh peneliti dengan data sekunder yang diambil dari penelitian sebelumnya, jika data yang diolah sebagai *training* diambil dari data primer maka data *testing* akan diambil dari primer begitu juga sebaliknya jika data *training* diambil dari data sekunder maka data *testing* diambil dari data sekunder juga, data primer akan dibagi menjadi dua yaitu data *training* dan data *testing*, begitu juga dengan data sekunder. Langkah pertama

peneliti melakukan mengumpulkan dataset sekunder digunakan untuk melakukan training, kemudian pada tahap melakukan praproses terdiri dari beberapa tahap ini peneliti perlu melakukan skema normalisasi data yaitu melakukan pengumpulan data dalam rentang yang sama, kemuian skema *boostrap* mengambil beberapa sampel data untuk melakukan skema boostrap dengan tujuan untuk meletakan data asli pada sampel baru yang sedang dibangun, Dari sampel terbaru maka akan terbuat pohon keputusan sebagai model *training*.

Data pengujian dimasukan dari pohon keputusan yang telah selesai terbuat. Pada tahap pengujian dilakukan hal yang sama yaitu dengan normalisasi sama seperti pada proses data *training*. Data pengujian yang telah dilakukan normalisasi dimasukan ke dalam model pelatihan. Hasil yang akan diperoleh dari proses pengujian tersebut yaitu berupa nilai *leaf*, dari nilai yang telah terbangun maka akan dikumpulkan nilai yang paling banyak muncul untuk dijadikan sebagai nilai acuan pada hasil prediksi sistem.

3.3 Praproses Data

Pada sub-bab ini penelitian tidak dapat langsung diolah dengan metode random forest perlu adanya pelatihan model terlebih dahulu. Maka sebelum melakukan pemodelan, peneliti menggunakan data primer yang sudah peneliti dapatkan dari hasil pengambilan data pada lalu lintas jaringan, untuk data normal sebanyak 46 baris dan data *bruteforce* sebanyak 1750 baris dalam hal ini, proporsi yang digunakan untuk data pelatihan yaitu 75% sedangkan untuk data pengujian sebanyak 25%. Sedangkan untuk data sekunder yang peneliti dapatkan dari dataset MQTT-IOT-IDS2020 yang akan digunakan yaitu data normal 52082 baris dan

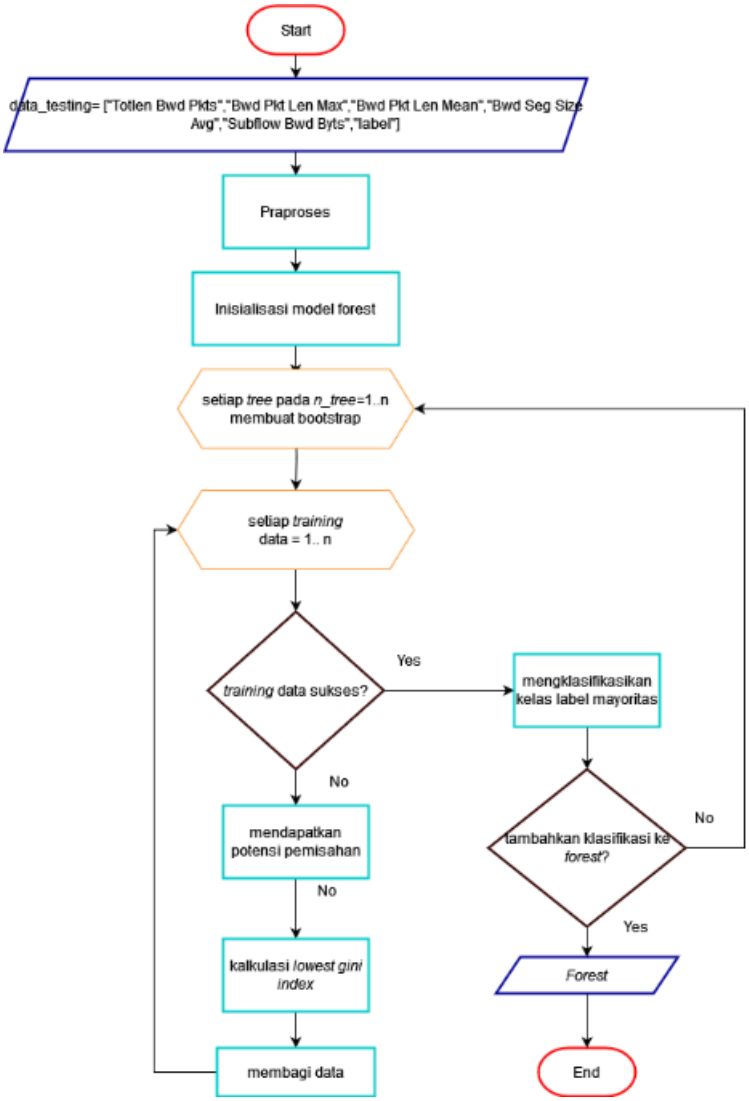
untuk data *bruteforce* sebanyak 32920 baris, proporsi yang digunakan untuk data pelatihan yaitu 75% sedangkan untuk data pengujian sebanyak 25%. Dari kedua data set yang telah dibuat oleh peneliti maka akan dilakukan permeriksaan terlebih dahulu sehingga semua data dapat berjalan sesuai dengan prosedur yang dibuat oleh peneliti. Pada praproses data dilakukan tahapan *normalisasi* data numerik. Normalisasi yaitu mengubah data numerik menjadi skala yang sama, berikut merupakan perhitungan data numerik.

$$normalisasi(\mathrm{x}) = \frac{x_i - x_i min}{x_i max - x_i min} \quad (3.1)$$

Keterangan:
x_i = merupakan data ke *i*
x_i *max* dan x_i *min* = data maximum dan minimum pada data ke *i*

3.4 Random Forest Pembuatan Model

Pembuatan sistem model *training* yang akan dibuat oleh peneliti dengan menggunakan algoritma *Random Forest* untuk membangun sistem tersebut. Langkah tersebut dibuat dalam sebuah diagram alur Model *Random Forest* Data Training seperti Gambar 3.3:



Gambar 3. 3 Diagram Alur Model *Random Forest* Data Training

Pada Gambar 3.3 untuk membuat *data training* pada sistem dijelaskan langkah-langkah yang dilakukan oleh peneliti, tahap awal yaitu melakukan input data training dan kemudian pengambilan *sample* secara acak yang terdapat pada dataset. Dataset perlu disiapkan ketika melakukan *bootstrap* yang kemudian dilakukan pengambilan sample secara acak. Pada kolom *dataset* akan dijaga sesuai dengan aslinya, baris dalam kumpulan data dapat dipilih secara acak dan baris yang sama dapat dimasukkan dalam sampel baru.

Pada tahap pembuatan *Bootstrap* dilakukan dengan cara membuat sampel ulang pada setiap n_tree pada kasus ini saya membuat n_tree sebanyak 200, setelah di dapatkan pembentukan dalam bentuk *bootstrap* atau *sample data* kemudian dilakukan pembentukan *tree* pada setiap *bootstrap. Decision tree*, seperti pohon keputusan, terdiri dari tiga bagian: simpul akar, simpul internal, dan simpul daun. Metode Indeks Gini digunakan oleh peneliti dalam penelitian ini untuk membangun pohon.

Dalam membuat pohon keputusan dibutuhkan adanya metode untuk membangun pohon tersebut. Peneliti menggunakan metode *Gini Index.*

Berikut merupakan rumus untuk mendapatkan nilai *impurity*, sebagai berikut:

$$Impurity = 1 - \sum P_i^2 \tag{3.2}$$

Keterangan:
P_i = nilai yang sering muncul pada kolom
P = *node* dipilih dan dibagi total keseluruhan *node*.

Sementara itu, persamaan berikut dapat digunakan untuk menentukan nilai Gini Index:

$$Gini(x) = 1 - \sum \frac{ni}{n} * Impurity \tag{3.3}$$

Keterangan:
n_i = banyak data (jumlah *tupel* milik kelas *i*)
n = banyak seluruh data himpunan (jumlah node pada *child*)

Pada Tabel 3.2. Merupakan data perhitungan manual Gini Index dengan kasus nilai baris data pada kolom *TotLen Bwd Pkts.*

Tabel 3. 2 Nilai Baris Data *TotLen Bwd Pkts*

No	***TotLen Bwd Pkts***	**Label**
1	54	Normal
2	53	Normal
3	54	Normal
4	54	Normal
5	40	Bruteforce
6	69	Bruteforce
7	66	Bruteforce
8	45	Bruteforce
9	58	Bruteforce
10	74	Bruteforce

1. Langkah pertama yaitu dengan cara mengelompokan data yang nilainya berdekatan. Seperti pada Tabel 3.3 Daftar Nilai Berdekatan berikut:

   Perhitungan rerata dilakukan dengan cara sebagai berikut:

$$\text{Rerata} = \frac{Data\ sebelum + Data\ sesudah}{2} \quad (3.4)$$

Tabel 3. 3 Daftar Nilai Berdekatan

No	**Data**	***TotLen Bwd Pkts***
1	54	
2	53	53,5
3	54	53,5
4	54	54
5	40	47
6	69	54,5
7	66	67,5
8	45	55,5
9	58	51,5
10	74	66

2. Nilai normal dan *bruteforce* digunakan untuk menghitung nilai kolom dari rata-rata sekelompok baris yang terkait erat. sesuai dengan Tabel 3.4 berikut ini.

Tabel 3. 4 Data *Partition*

No	Data *Partition*	Normal	*Bruteforce*	Nilai n_i	Nilai *n*
1	<=53,5	53	53	106	
2	>53,5	54	54	108	214
3	<=53,5	52	53	105	
4	>53,5	54	54	108	105
5	<=54	53	54	107	
6	>54	54	55	109	216
7	<=47	0	47	47	
8	>47	54	48	102	149
9	<=54,5	54	54	108	
10	>54,5	0	55	55	163
11	<=67,5	54	67	121	
12	>67,5	0	68	68	189
13	<=55,5	54	55	109	
14	>55,5	0	60	60	169
15	<=51,5	0	51	51	

16	>51,5	52	52	104	155
17	<=66	54	66	120	
18	>66	0	67	67	187

Kolom *Bruteforce* di dapatkan dengan cara menghitung jumlah data yang bisa masuk pada kolom partisi data yang diambil dari Tabel 3.1

3. Langkah selanjutnya melakukan kalkulasi dengan *gini impurity* dan juga *gini index* pada Tabel 3.3 tentang data partisi. Data akan dihitung mulai dari yang terkecil.

$$\textit{Gini Impurity (<=1,5)} = 1 - \left(\left(\frac{53}{106}\right)^2\right) + \left(\left(\frac{53}{106}\right)^2\right)$$
$$= 1 - (0{,}25 + 0{,}25)$$
$$= 0{,}5$$

$$\textit{Gini Impurity (>1,5)} = 1 - \left(\left(\frac{54}{108}\right)^2\right) + \left(\left(\frac{54}{108}\right)^2\right)$$
$$= 1 - (0{,}25 + 0{,}25)$$
$$= 1 - 0.5$$
$$= 0.5$$

$$\textit{Gini Index(1,5)} = 1 - \left(\left(\frac{106}{214} * 0{,}5\right) + \left(\frac{108}{214} * 0{,}5\right)\right)$$
$$= 1 - (0{,}24 + 0{,}25)$$
$$= 0{,}51$$

4. Ulangi setiap perhitungan pada semua baris menggunakan metode *gini impurity* dan *gini index*.
5. Berikut merupakan hasil perhitungan pada Tabel 3.5 berikut:

Tabel 3. 5 Hasil Perhitungan

No	Data *Partision*	Normal	*Bruteforce*	Nilai n_i	Nilai *n*	Gini Impurity	Gini Index
1	<=53,5	53	53	106		0,5	
2	>53,5	54	54	108	214	0,5	0,5
3	<=53,5	52	53	105		0,499954649	
4	>53,5	54	54	108	105	0,5	1,01095622
5	<=54	53	54	107		0,499956328	
6	>54	54	55	109	216	0,499957916	0,49995713
7	<=47	0	47	47		0	
8	>47	54	48	102	149	0,498269896	0,34109751
9	<=54,5	54	54	108		0,5	
10	>54,5	0	55	55	163	0	0,33128834
11	<=67,5	54	67	121		0,494228536	
12	>67,5	0	68	68	189	0	0,31641086
13	<=55,5	54	55	109		0,499957916	
14	>55,5	0	60	60	169	0	0,32245806
15	<=51,5	0	51	51		0	
16	>51,5	52	52	104	155	0,5	0,33548387
17	<=66	54	66	120		0,495	
18	>66	0	67	67	187	0	0,31764706

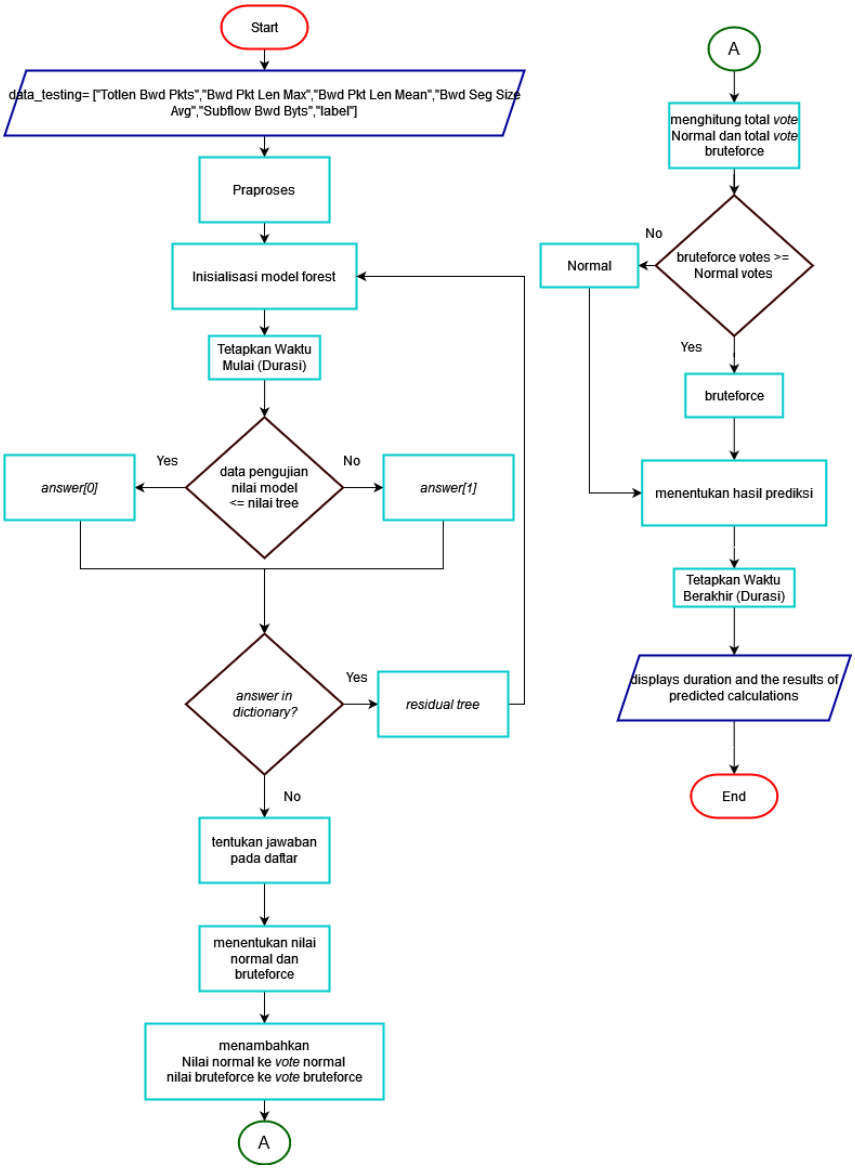
Nilai *gini index* berguna untuk melakukan pemisahan pada setiap pohon keputusan sehingga dapat dengan mudah mendapatkan nilai dari *leaf*, nilai dari *leaf* dikumpulkan menjadi nilai prediksi yang sering muncul pada akhir. Berikut merupakan *Script* untuk pembuatan model training pada Gambar 3.4 berikut.

```
def decision_tree_algorithm(df, counter=0, min_samples=0, max_depth=5, random_subspace=None):
    if counter == 0:
        global COL_HEADERS, FEATURE_TYPES
        COL_HEADERS = df.columns
        FEATURE_TYPES = determine_type_of_feature(df)
        data = df.values
    else:
        data = df
    if (check_purity(data)) or (len(data) < min_samples) or (counter == max_depth):
        classification = clasify_data(data)
        return classification
    else:
        counter += 1
        potential_splits = get_potential_splits(data, random_subspace)
        split_col, split_value, _ = determine_best_split(data, potential_splits)
        data_below, data_above = split_data(data, split_col, split_value)

        if len(data_below) == 0 or len(data_above) == 0:
            classification = clasify_data(data)
            return classification
        feature_name = COL_HEADERS[split_col]
        type_of_feature = FEATURE_TYPES[split_col]
        if type_of_feature == "continuous":
            question = "{} <= {}".format(feature_name, split_value)
        else:
            question = "{} = {}".format(feature_name, split_value)
        sub_tree = {question: []}
        yes_answer = decision_tree_algorithm(data_below, counter, min_samples, max_depth, random_subspace)
        no_answer = decision_tree_algorithm(data_above, counter, min_samples, max_depth, random_subspace)

        if yes_answer == no_answer:
            sub_tree = yes_answer
        else:
            sub_tree[question].append(yes_answer)
            sub_tree[question].append(no_answer)

    return sub_tree


def bootstrapping(data, n_bootstrap):

    bootstrap_indices = np.random.randint(low=0, high=len(data), size=n_bootstrap)
    df_bootstrapped = data.iloc[bootstrap_indices]

    return df_bootstrapped

def random_forest_algorithm(data, n_trees, n_bootstrap, n_features, dt_max_depth):
    forest = []
    for i in range(n_trees):
        df_bootstrapped = bootstrapping(data, n_bootstrap)
        tree = decision_tree_algorithm(df_bootstrapped, max_depth=dt_max_depth, random_subspace=n_features)
        forest.append(tree)

    return forest
```

Gambar 3. 4 *Script* untuk pembuatan model training

3.5 *Random Forest* Deteksi Serangan MQTT

Setelah dibuatkannya model data *training*, Langkah berikutnya yaitu melakukan proses deteksi serangan MQTT pada sistem tersebut. Dibawah ini merupakan gambar diagram alur deteksi serangan MQTT seperti pada Gambar 3.5.



Gambar 3. 5 Diagram Alur Model *Random Forest* Data Testing

Proses memasukkan data *testing* merupakan langkah awal dari pohon model yang telah dibuat sebelumnya pada gambar diagram alur sistem deteksi serangan MQTT. Tahap normalisasi dari proses praproses harus diselesaikan pada data pengujian yang dimasukkan. data yang dimasukkan ke dalam model deteksi akan menghasilkan kumpulan *list* dari *leaf* pohon keputusan. Setelah itu ketika proses mau dimulai, akan ada waktu yang di tetapkan. Kemudian dilakukannya pemisahan

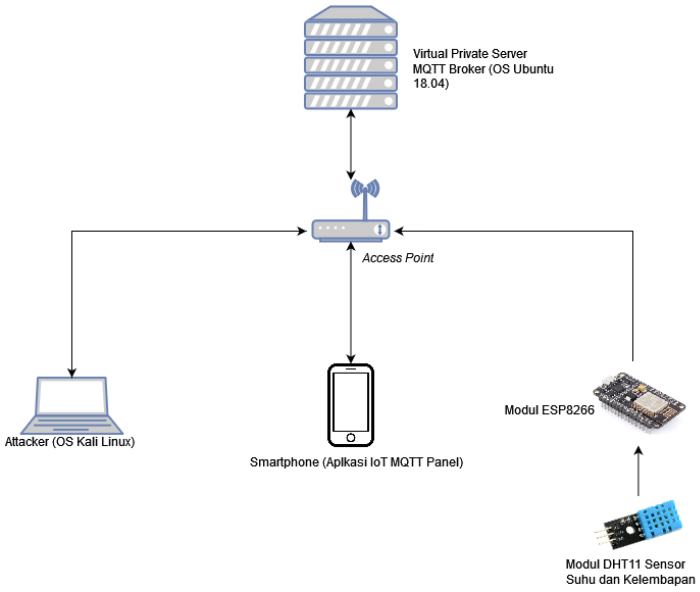
antara data Normal dengan data *bruteforce* MQTT pada *leaf* tersebut. Setelah itu akan terdapat proses hasil perhitungan waktu yang dihasilkan selama proses *testing* Kemudian hasil akhir ditentukan oleh *leaf* yang sering muncul dari deteksi lalu *network traffic* dengan algortima *Random Forest* dalam mendeteksi *bruteforce* pada protokol MQTT dan durasi waktu pengujian yang diperlukan. *Script* program Random Forest pada Gambar 3.6 di bawah ini digunakan untuk melakukan proses pendeteksian serangan terhadap data testing.

```
def predict_example(example, tree):

  question = list(tree.keys())[0]
  feature_name, comparison_operator, value = question.split(" ")


  if comparison_operator == "<=":
      if example[feature_name] <= float(value):
        answer = tree[question][0]
      else:
        answer = tree[question][1]


  else:
      if str(example[feature_name]) == value:
          answer = tree[question][0]
      else:
          answer = tree[question][1]


  if not isinstance(answer, dict):
    return answer


  else:
    residual_tree = answer
    return predict_example(example, residual_tree)


def decision_tree_predictions(df, tree):
    predictions = df.apply(predict_example, args=(tree,), axis=1)
    return predictions

def random_forest_predictions(data, forest):
    df_predictions = {}
    for i in range(len(forest)):
        col_name = "tree_{}".format(i)
        predictions = decision_tree_predictions(data, tree=forest[i])
        df_predictions[col_name] = predictions

    df_predictions = pd.DataFrame(df_predictions)
    random_forest_predictions = df_predictions.mode(axis=1)[0]
```

Gambar 3. 6 *Script* Proses Deteksi Serangan pada Data Testing

3.6 *Lab Hacking Environment*

Pada Gambar 3.7 berikut merupakan Desain *Lab Hacking Environment* yang dibuat oleh peneliti:



Gambar 3. 7 Desain Lab *Hacking Environment*

Pada Desain *Lab Hacking Environment* pada *server* akan dijalankan broker MQTT menggunakan *mosquitto* dan juga melakukan *capture traffic* data pada *server*. pada *access point* digunakan untuk menghubungkan device pada satu jaringan. *Attacker* akan melakukan serangan ke *service* pada protokol MQTT dengan menggunakan tools yaitu *tools mqttsa* digunakan untuk melakukan *bruteforce* pada broker MQTT. Pada device *smartphone* akan mengirimkan *subscribe* ke *service* MQTT dan juga membaca hasil sensor suhu dan kelembapan. Pada modul ESP8266 akan melakukan pemrograman untuk membaca sensor suhu dan kelembapan dan melakukan *publish* ke *service* MQTT Broker melewati *access point*. Kebutuhan bahan dalam membuat *Lab Hacking Environment* disajikan pada Tabel 3.6 sebagai berikut:

Tabel 3. 6 *Lab Hacking Environment*

No	Peran	Spesifikasi
1	VPS MQTT (Broker *Mosquitto*)	- Ubuntu 18.04 - 1 GB RAM - Location: Indonesia
2	IoT Device	- Modul ESP8266 - Modul DHT11 (Sensor suhu dan Kelembapan)
3	Smartphone	- Aplikasi IoT MQTT Panel
4	Penyerang	- *Nmap (Network Mapper)* - *Mqttsa* digunakan untuk melakukan *bruteforce* pada broker MQTT.

BAB IV

UJI COBA DAN PEMBAHASAN

4.1 Simulasi Uji Coba

Pada simulasi yang akan peneliti lakukan yaitu melakukan penyerangan sesuai dengan skema topologi yang telah dibuat pada *environment* lab *hacking*, peneliti akan mengambil *network traffic* normal dan *bruteforce.* Pada skenario ini peneliti menggunakan beberapa tools untuk melakukan melakukan *enumerasi*, penyerangan dan melakukan extraksi data.

4.1.1 Akuisisi Data

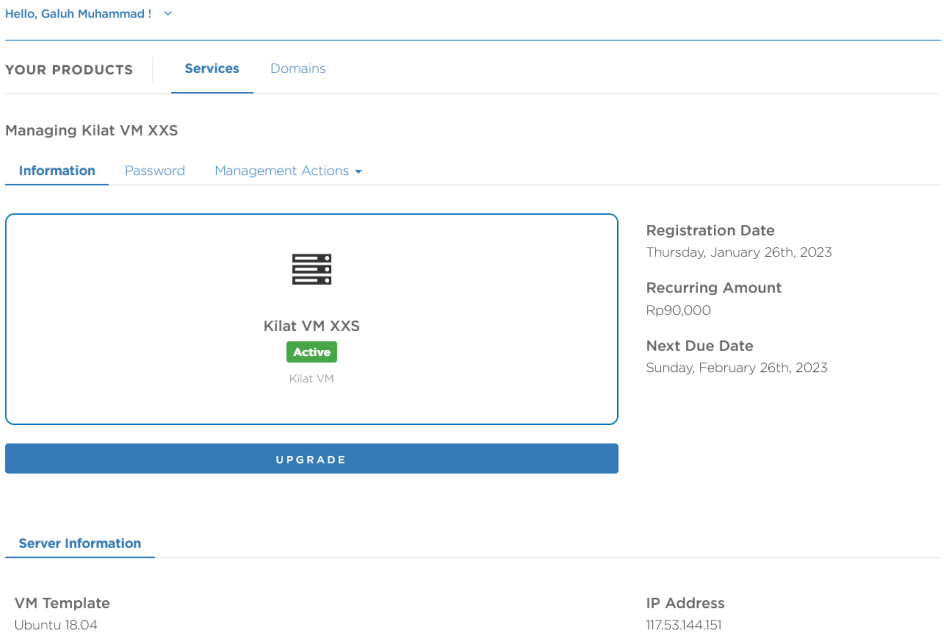
Peneliti akan melakukan skenario hacking pada broker *service* MQTT, untuk skenario yang penguji lakukan berada pada ruang lingkup lab yang telah dibuat. Berikut merupakan Langkah-langkah penguji mengambil data primer yang akan digunakan pada penelitian:

4.1.1.1 Konfigurasi *Lab Hacking Environment*

Membuat lingkungan pengujian yang mirip dengan skenario aslinya merupakan langkah pertama yang dilakukan peneliti. yaitu membuat lingkungan peretasan. Peneliti membutukan sebuah *virtual machine* yang berguna untuk melakukan konfigurasi pada pada sistem operasi untuk menjalankan MQTT *service* yang *hackable* untuk melakukan simulasi peretasan. Peneliti membutukan sebuah software untuk menjalankan *virtual machine* yaitu *Virtual Private Server* (VPS) milik peneliti. VPS ini akan menjalankan sistem operasi yang berperan sebagai Broker MQTT *service hackable* yang aman untuk diretas.

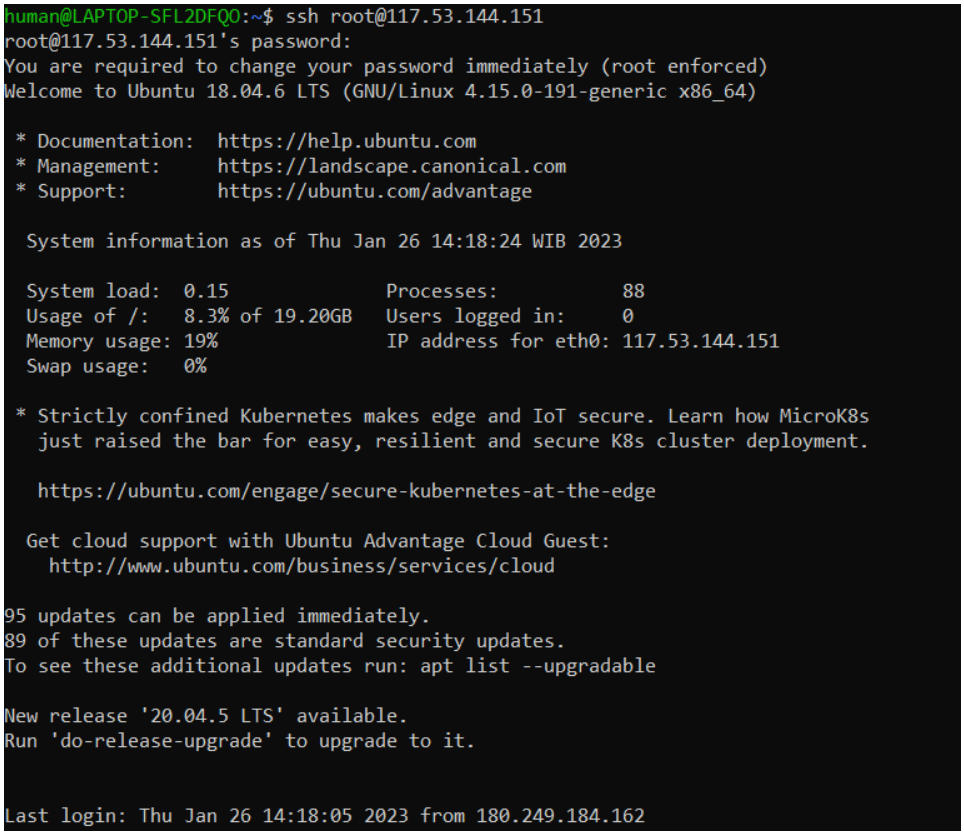
Pada *virtual machine* akan dijalankan sistem operasi Ubuntu untuk menjalankan MQTT *service* sebagai broker yang kemudian akan dilakukan konfigurasi untuk koneksi ke IoT device (ESP 8266).

Berikut merupakan Gambar 4.1 Menampilkan *interface* VPS yang telah terpasang sistem operasi Ubuntu 18.04. Berikut merupakan Gambar 4.1 Dashboard VPS



Gambar 4. 1 Dashboard VPS

Peneliti Melakukan login ke VPS dengan menggunakan *service* SSH untuk melakukan konfigurasi pada MQTT broker. Berikut merupakan Gambar 4. 2 Login menggunakan SSH *Service*



Gambar 4. 2 Login menggunakan SSH *Service*

Kemudian peneliti melakukan instalasi broker mosquitto *service* dengan cara seperti berikut ini "*apt-get install mosquitto*". Berikut merupakan Gambar 4. 3 Instalasi Mosquitto

```
root@galuh:~# apt-get install mosquitto
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libev4 libwebsockets8
The following NEW packages will be installed:
  libev4 libwebsockets8 mosquitto
0 upgraded, 3 newly installed, 0 to remove and 92 not upgraded.
Need to get 214 kB of archives.
After this operation, 584 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://archive.ubuntu.com/ubuntu bionic/universe amd64 libev4 amd64 1:4.22-1 [26.3 kB]
Get:2 http://archive.ubuntu.com/ubuntu bionic/universe amd64 libwebsockets8 amd64 2.0.3-3build1 [71.8 kB]
Get:3 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 mosquitto amd64 1.4.15-2ubuntu0.18.04.3 [116 kB]
Fetched 214 kB in 2s (126 kB/s)
Selecting previously unselected package libev4.
(Reading database ... 90819 files and directories currently installed.)
Preparing to unpack .../libev4_1%3a4.22-1_amd64.deb ...
Unpacking libev4 (1:4.22-1) ...
Selecting previously unselected package libwebsockets8:amd64.
Preparing to unpack .../libwebsockets8_2.0.3-3build1_amd64.deb ...
Unpacking libwebsockets8:amd64 (2.0.3-3build1) ...
Selecting previously unselected package mosquitto.
Preparing to unpack .../mosquitto_1.4.15-2ubuntu0.18.04.3_amd64.deb ...
Unpacking mosquitto (1.4.15-2ubuntu0.18.04.3) ...
Setting up libev4 (1:4.22-1) ...
Setting up libwebsockets8:amd64 (2.0.3-3build1) ...
Setting up mosquitto (1.4.15-2ubuntu0.18.04.3) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Processing triggers for ureadahead (0.100.0-21) ...
Processing triggers for libc-bin (2.27-3ubuntu1.6) ...
Processing triggers for systemd (237-3ubuntu10.53) ...
```

Gambar 4. 3 Instalasi Mosquitto

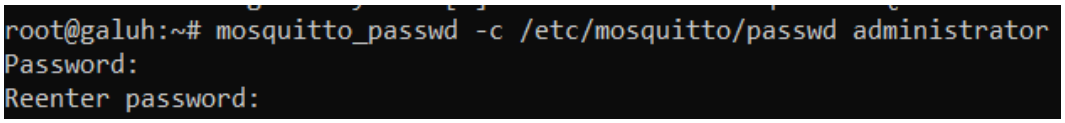
Kemudian jalankan *service* mosquito dengan commandline sebagai berikut: "*Systemctl start mosquitto*" dan kemudian lakukan pengecekan status terhadap *service* tersebut dengan menggunakan perintah "*systemctl status mosquitto*". Berikut merupakan Gambar 4.4 *Service* mosquito *Running*

```
root@galuh:~# systemctl status mosquitto
● mosquitto.service - LSB: mosquitto MQTT v3.1 message broker
   Loaded: loaded (/etc/init.d/mosquitto; generated)
   Active: active (running) since Thu 2023-01-26 14:23:02 WIB; 1min 22s ago
     Docs: man:systemd-sysv-generator(8)
    Tasks: 1 (limit: 1143)
   CGroup: /system.slice/mosquitto.service
           └─1990 /usr/sbin/mosquitto -c /etc/mosquitto/mosquitto.conf

Jan 26 14:23:02 galuh systemd[1]: Starting LSB: mosquitto MQTT v3.1 message broker...
Jan 26 14:23:02 galuh mosquitto[1974]:  * Starting network daemon: mosquitto
Jan 26 14:23:02 galuh mosquitto[1974]:    ...done.
Jan 26 14:23:02 galuh systemd[1]: Started LSB: mosquitto MQTT v3.1 message broker.
```

Gambar 4. 4 *Service* mosquito *Running*

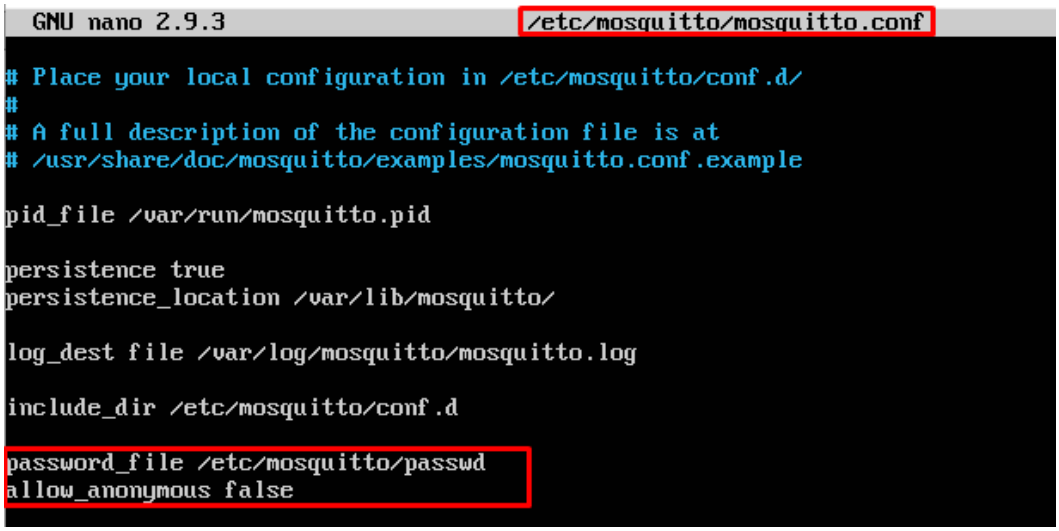
Setelah itu lakukuan konfigurasi kredensial berupa otorisasi username dan password yang valid untuk mengakses *service* broker MQTT, berikut merupakan konfigurasinya, peneliti membuat username menggunakan **administrator** dan menggunakan password **qwerty123#**.

Berikut merupakan Gambar 4.4 *Setting Password*



Gambar 4. 5 *Setting Password*

Lakukan konfigurasi pada file mosquito.conf dengan menambahkan baris password_file dan allow_anonymous, password_file merujuk pada lokasi path password yang telah peneliti buat sedangkan allow_anonymous *false* merupakan konfigurasi supaya semua user harus terautentikasi terlebih dahulu untuk mengakses data yang ada pada broker MQTT sebagai berikut. Berikut merupakan Gambar 4.6 Konfigurasi file mosquitto



Gambar 4. 6 Konfigurasi file mosquito

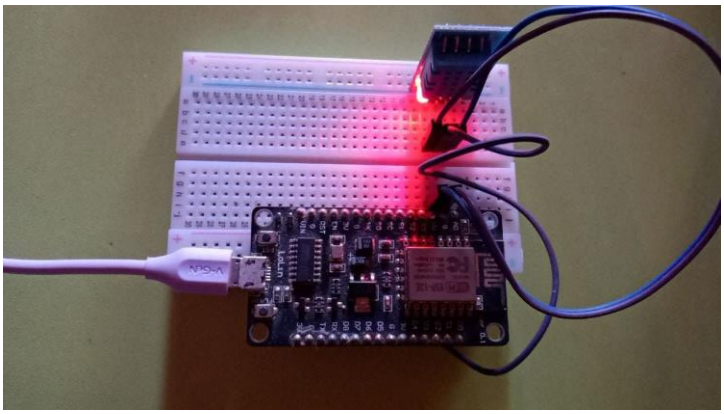
Kemudian lakukan restart pada broker *service* MQTT dengan begitu *service* sudah menggunakan kredensial otorisasi username dan password.

Kemudian buat *script* untuk modul ESP8266 untuk mengirimkan data hasil sensor DHT11 ke broker pada *service*, berikut merupakan *script* yang peneliti buat, untuk

mengirimkan data yang dibaca oleh sensor ke *service* MQTT pengiriman data ke *service* menggunakan tipe data JSON dengan index 0 sebagai nilai dari suhu, dan index ke-1 sebagai nilai dari kelembapan. Berikut merupakan Tabel 4. 1 *Script* Modul untuk ESP8266 Dalam Bentuk JSON

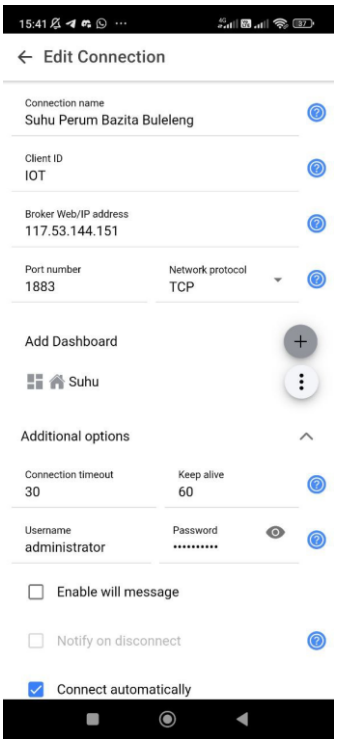
Tabel 4. 1 *Script* Modul untuk ESP8266 Dalam Bentuk JSON

```
  float h = dht.readHumidity(true);
  float t = ((dht.readTemperature(true) -32) * 5/9);
  float f = dht.readTemperature();
  Serial.print("Temperature : ");
  Serial.print(t);
  Serial.print(" °C ");
  Serial.print(" | ");
  Serial.print("Humidity : ");
  Serial.print(h);
  Serial.println(" RH ");

  char result[100];
  char* arrayBuka = "{\"cikarang\":{\"suhu_kelembapan\":[";
  char* tagSatu = "{\"tag\":\"suhu\",\"value\":";
  sprintf(msg, "%d.%02d", (int)t, (int)(t*100)%100);
  char* tutupJson = "}";
  char* tagDua = ",{\"tag\":\"kelembapan\",\"value\":";
  sprintf(msg2, "%d.%02d", (int)h, (int)(h*100)%100);
  char* arrayTutup = "]}}";
  strcpy(result, arrayBuka);
  strcat(result, tagSatu);
  strcat(result, msg);
  strcat(result, tutupJson);
  strcat(result, tagDua);
  strcat(result, msg2);
  strcat(result, tutupJson);
  strcat(result, arrayTutup);
  Serial.println(result);

  mqtt.publish("v2/IOT2021/HP/json", result);
  delay(1000);
 }
```



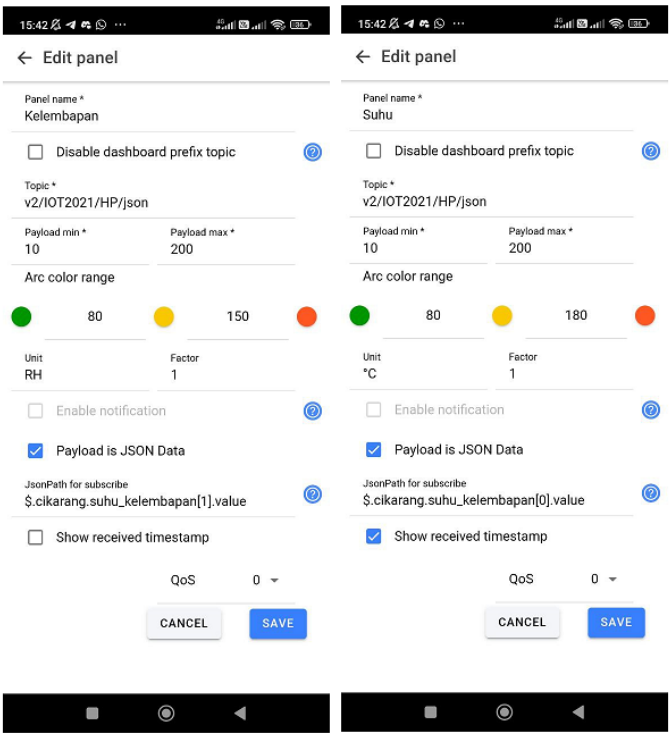
Gambar 4. 7 Modul ESP 8266 dan DHT11

Setelah itu lakukan setting pada aplikasi IOT MQTT Panel, sesuaikan dengan IP *service*, port, username dan password yang digunakan pada *service* setelah itu klik *save*. Berikut merupakan Gambar 4.8 konfigurasi Koneksi pada MQTT IoT Panel



Gambar 4. 8 Konfigurasi Koneksi di MQTT IoT Panel

Kemudian buat juga sub-menu untuk suhu dan kelembapan pada aplikasi tersebut, dan sesuaikan dengan data yang JSON yang dikirimkan oleh *service* ke aplikasi. Berikut meurpakan Gambar 4.9 Konfigurasi MQTT Suhu dan Kelembapan



Gambar 4. 9 Konfigurasi MQTT Suhu dan Kelembapan

Setelah melakukan konfigurasi pada IoT MQTT Panel, maka data yang ada pada service broker MQTT akan terlihat pada apps IoT MQTT Panel pada mobile. Seperti pada Gambar 4. 10 Hasil dari pembacaan sensor DHT11

Gambar 4. 10 Hasil dari pembacaan sensor DHT11

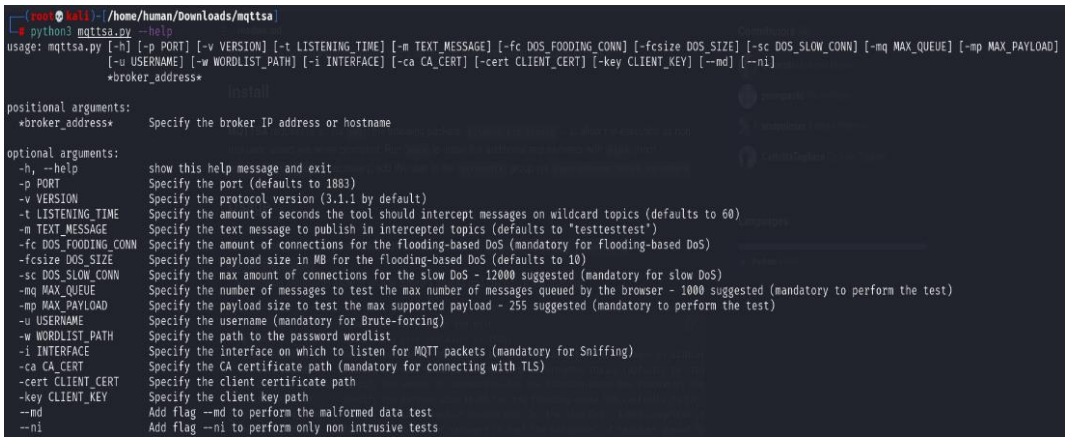
Berikut merupakan alamat IP untuk klien, modul ESP8266, dan mesin virtual pada Tabel 4.2 Ruang lingkup Perangkat dan Jaringan.

Tabel 4. 2 Ruang lingkup Perangkat dan Jaringan

No	Nama perangkat	Perangkat	Peran	*IP address*
1	VPS Ubuntu 18.04	*Virtual machine*	MQTT (Broker Mosquitto)	117.53.144.151
2	OS Kali Linux	*Virtual machine*	Penyerang	192.168.43.160
3	ESP8266 dan DHT11	Modul	Pengirim data sensor (suhu dan kelembapan) ke *service* MQTT	192.168.43.44
4	Mobile (IoT MQTT Panel)	Smartphone	Penerima data dari *service* MQTT	192.168.43.2

Proses peretasan membutuhkan tools yang akan digunakan seperti *mqttsa* yang digunakan untuk melakukan beberapa macam teknik seperti *bruteforce*

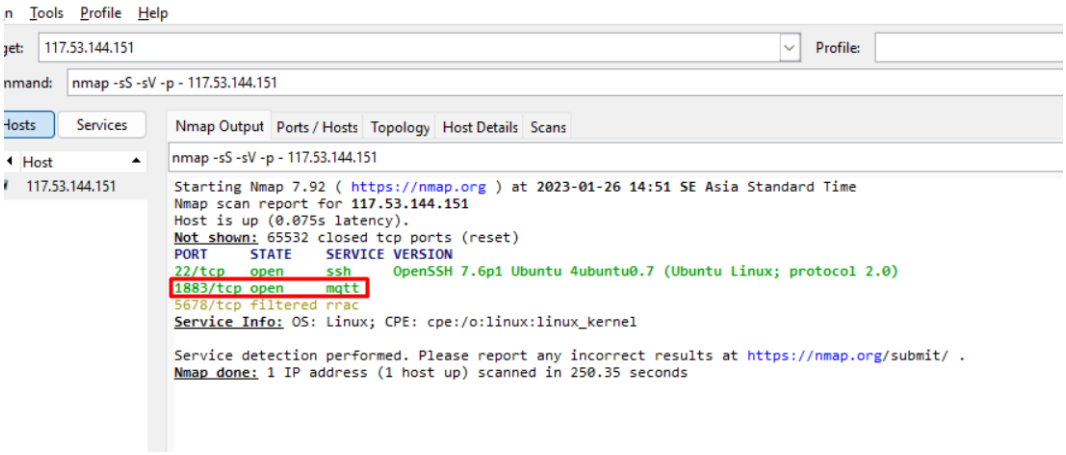
kredensial yang ada pada *service* MQTT. Berikut merupakan Gambar 4.11 *Tools* Mqttsa untuk melakukan serangan bruteforce



Gambar 4. 11 *Tools* Mqttsa untuk melakukan serangan *bruteforce*

4.1.1.2 Skenario Peretasan

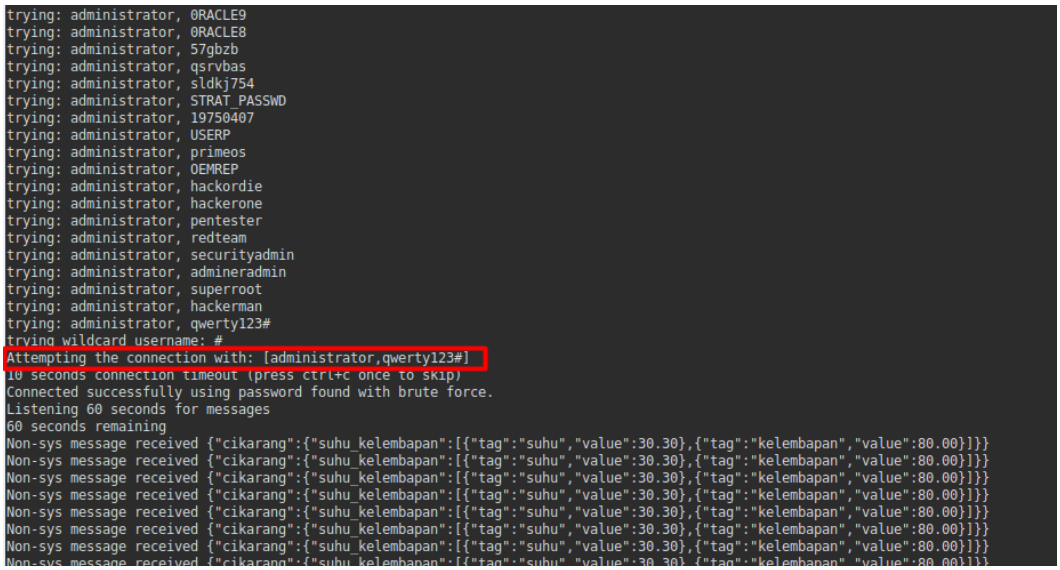
Skenario peretasan dilakukan dengan cara menjalankan *script mqttsa* sebagai eksekutor yang akan melakukan serangan dengan cara menebak username dan password yang ada pada *service* mosquitto. Langkah pertama lakukan *scanning* dengan bantuan tools nmap untuk mengentahui *service* yang sedang berjalan pada *server*, dengan perintah "*nmap -p 1883 <ip-server>*" seperti pada tabel diatas ip *server* mempunyai nilai 117.53.144.151. berikut merupakan hasil *enumarasi* nya. Berikut merupakan Gambar 4.12 Hasil Scanning *tools Nmap*



```
Starting Nmap 7.92 ( https://nmap.org ) at 2023-01-26 14:51 SE Asia Standard Time
Nmap scan report for 117.53.144.151
Host is up (0.075s latency).
Not shown: 65532 closed tcp ports (reset)
PORT      STATE    SERVICE VERSION
22/tcp    open     ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.7 (Ubuntu Linux; protocol 2.0)
1883/tcp open     mqtt
5678/tcp filtered rrac
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 250.35 seconds
```

Gambar 4. 12 Hasil Scanning *Tools Nmap*

Berdasarkan hasil *scanning* diketahui bahwa *server* tersebut memiliki *service* MQTT yang berjalan, tetapi *service* tersebut mempunyai kredensial yang harus diketahui oleh penyerang tersebut. Jadi dilakukan teknik *bruteforce*, peneliti membuat *wordlist* sebanyak 500 kata kemudian menjalankan *script* mqttsa dengan perintah "*python3 mqttsa.py 117.53.144.151 -p 1883 -u administrator -w wordlist.txt*". berikut merupakan hasilnya. Berikut merupakan proses pengerangan bruteforce pada broker MQTT, seperti pada Gambar 4.13 Hasil *Bruteforce*



```
trying: administrator, 0RACLE9
trying: administrator, 0RACLE8
trying: administrator, 57gbzb
trying: administrator, qsrvbas
trying: administrator, sldkj754
trying: administrator, STRAT_PASSWD
trying: administrator, 19750407
trying: administrator, USERP
trying: administrator, primeos
trying: administrator, OEMREP
trying: administrator, hackordie
trying: administrator, hackerone
trying: administrator, pentester
trying: administrator, redteam
trying: administrator, securityadmin
trying: administrator, admineradmin
trying: administrator, superroot
trying: administrator, hackerman
trying: administrator, qwerty123#
trying wildcard username: #
Attempting the connection with: [administrator,qwerty123#]
10 seconds connection timeout (press ctrl+c once to skip)
Connected successfully using password found with brute force.
Listening 60 seconds for messages
60 seconds remaining
Non-sys message received {"cikarang":{"suhu_kelembapan":[{"tag":"suhu","value":30.30},{"tag":"kelembapan","value":80.00}]}}
Non-sys message received {"cikarang":{"suhu_kelembapan":[{"tag":"suhu","value":30.30},{"tag":"kelembapan","value":80.00}]}}
Non-sys message received {"cikarang":{"suhu_kelembapan":[{"tag":"suhu","value":30.30},{"tag":"kelembapan","value":80.00}]}}
Non-sys message received {"cikarang":{"suhu_kelembapan":[{"tag":"suhu","value":30.30},{"tag":"kelembapan","value":80.00}]}}
Non-sys message received {"cikarang":{"suhu_kelembapan":[{"tag":"suhu","value":30.30},{"tag":"kelembapan","value":80.00}]}}
Non-sys message received {"cikarang":{"suhu_kelembapan":[{"tag":"suhu","value":30.30},{"tag":"kelembapan","value":80.00}]}}
Non-sys message received {"cikarang":{"suhu_kelembapan":[{"tag":"suhu","value":30.30},{"tag":"kelembapan","value":80.00}]}}
Non-sys message received {"cikarang":{"suhu_kelembapan":[{"tag":"suhu","value":30.30},{"tag":"kelembapan","value":80.00}]}}
```

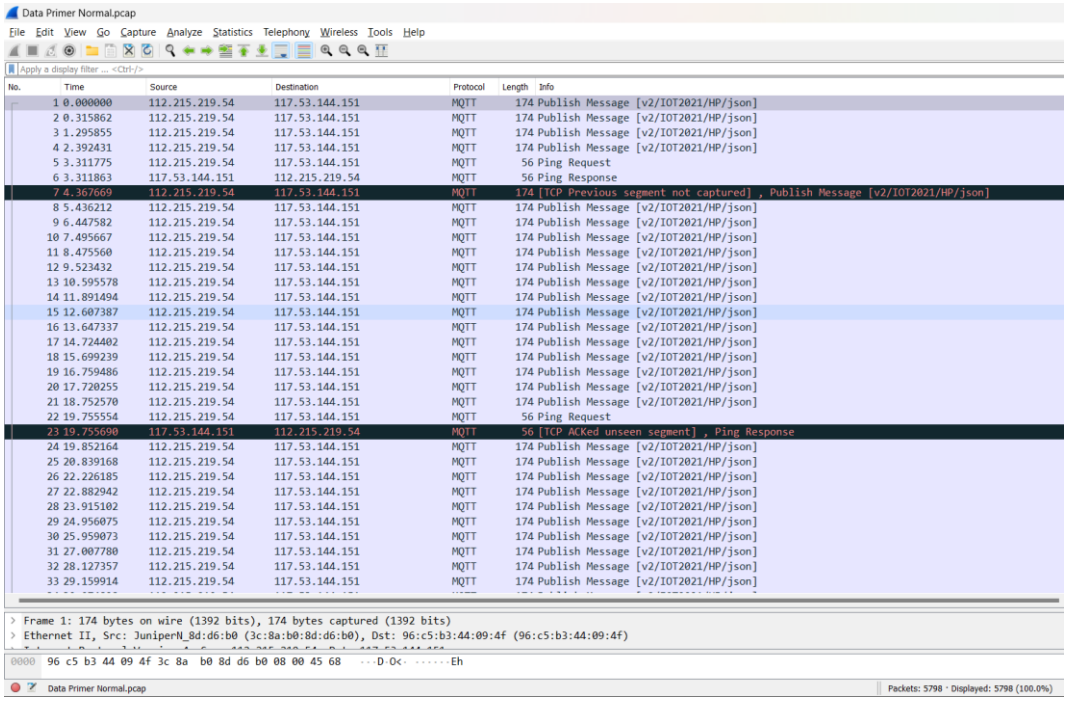
Gambar 4. 13 Hasil *Bruteforce*

4.1.1.3 Perekaman Data Lalu Lintas Jaringan IoT

Alat yang membantu merekam lalu lintas jaringan yang sedang berlangsung sangat penting untuk melakukan proses perekaman data lalu lintas jaringan IoT, *tools* yang digunakan yaitu *WireShark*. Karena *WireShark* adalah *tools* yang digunakan untuk merekam dan juga melihat lalu lintas jaringan. Untuk mendapatkan data tersebut *tools WireShark* harus dijalankan terlebih dahulu sabelum melakukan serangan ke jaringan, dengan begitu lalu lintas yang dikirimkan dari penyerang ke *server* akan terlihat pada lalu lintas jaringan tersebut.

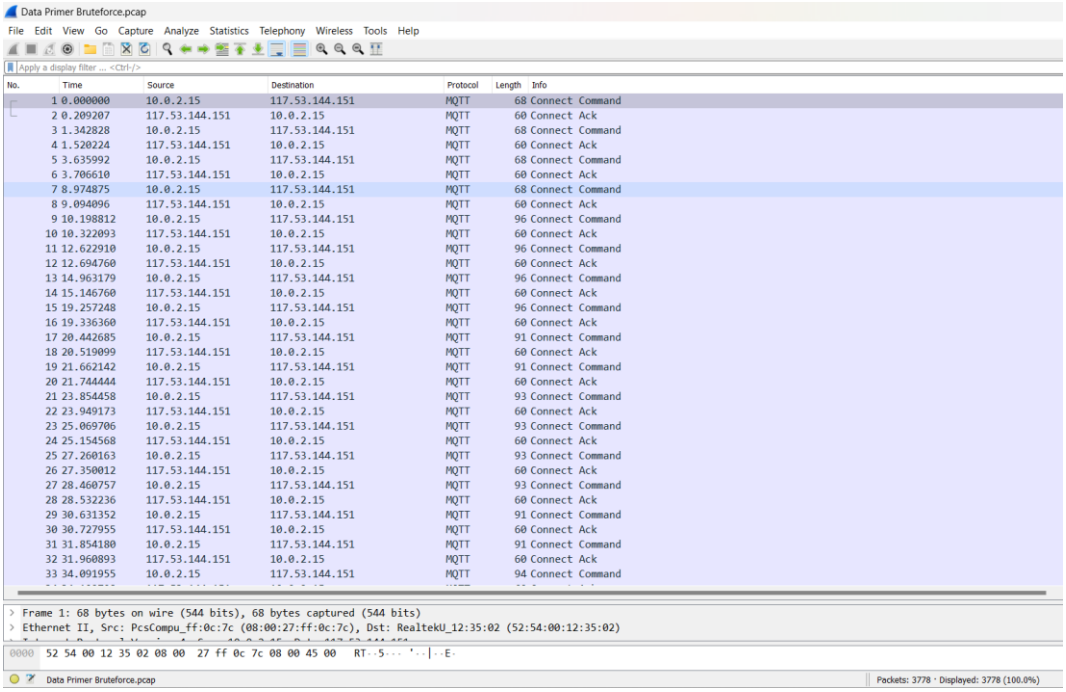
4.1.1.3.1 Pengambilan Data Primer

Peneliti melakukan pengambilan data lalu lintas jaringan IoT dari mulai *traffic* yang normal. Sesuai pada gambar ditampilkan proses rekaman serangan belum diluncurkan. Pada Gambar 4.14 Data primer normal



Gambar 4. 14 Data primer normal

Ketika WireShark sudah berjalan, kemudian penyerang mencoba untuk melakukan serangan ke sisi broker MQTT dengan melakukan serangan *bruteforce attack* dan berikut merupakan gambar yang dilakukan serangan *bruteforce attack, traffic* pada tools wireshark diketahui ada kolom info dengan baris "*Connect Command*" dimana penyerang mencoba melakukan koneksi ke sisi broker MQTT dan "*Connect Ack*" merupakan response dari broker MQTT. Pada Gambar 4.15 Data primer *bruteforce*



Gambar 4. 15 Data primer *bruteforce*

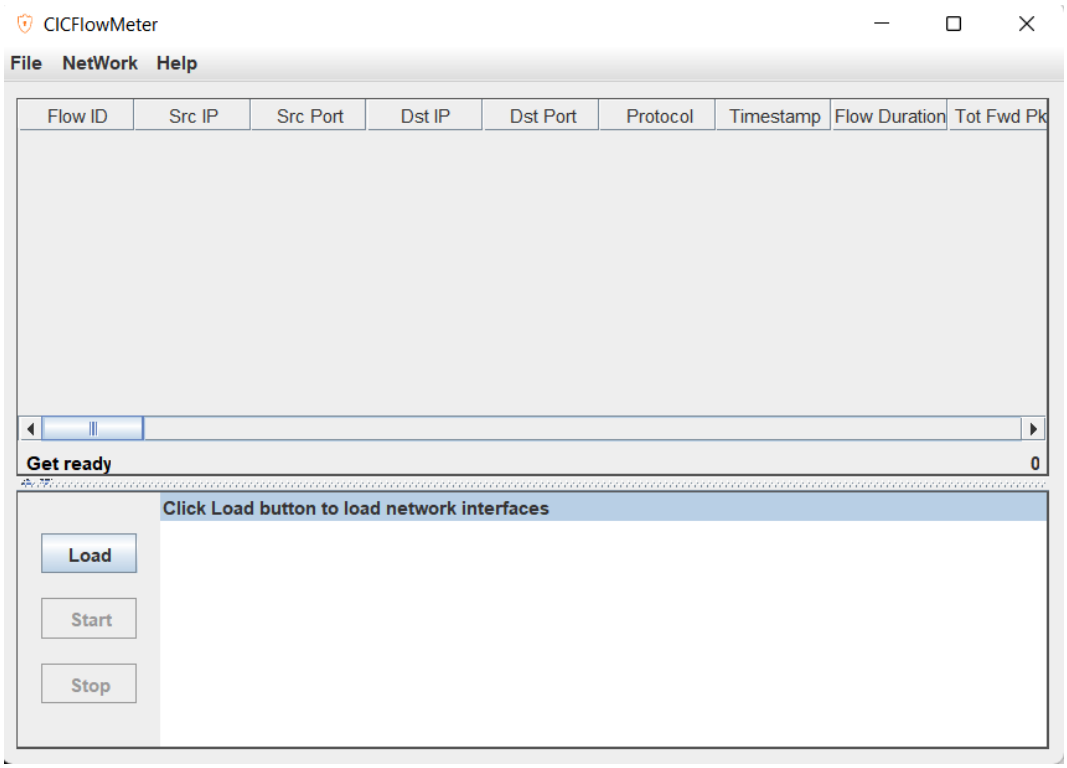
Peneliti tidak menggabungkan data normal dan data serangan untuk mendapatkan data yang akurat dan murni, untuk data normal dilakukan pengambilan data terlebih dahulu hingga selesai kemudian di simpan dengan nama "*Data Primer Normal*" dan juga pengambilan data pada serangan dipisah antara serangan "*Data Primer Bruteforce"*. Tabel 4.3 di bawah menyajikan jumlah baris yang tercatat secara efektif selama proses pengambilan data.

Tabel 4. 3 Data Primer

Primary data		
No	*Network Traffic*	*Data*
1	Normal	5798 baris
2	*Bruteforce*	3778 baris

4.1.1.4 Ekstraksi Lalu Lintas Jaringan

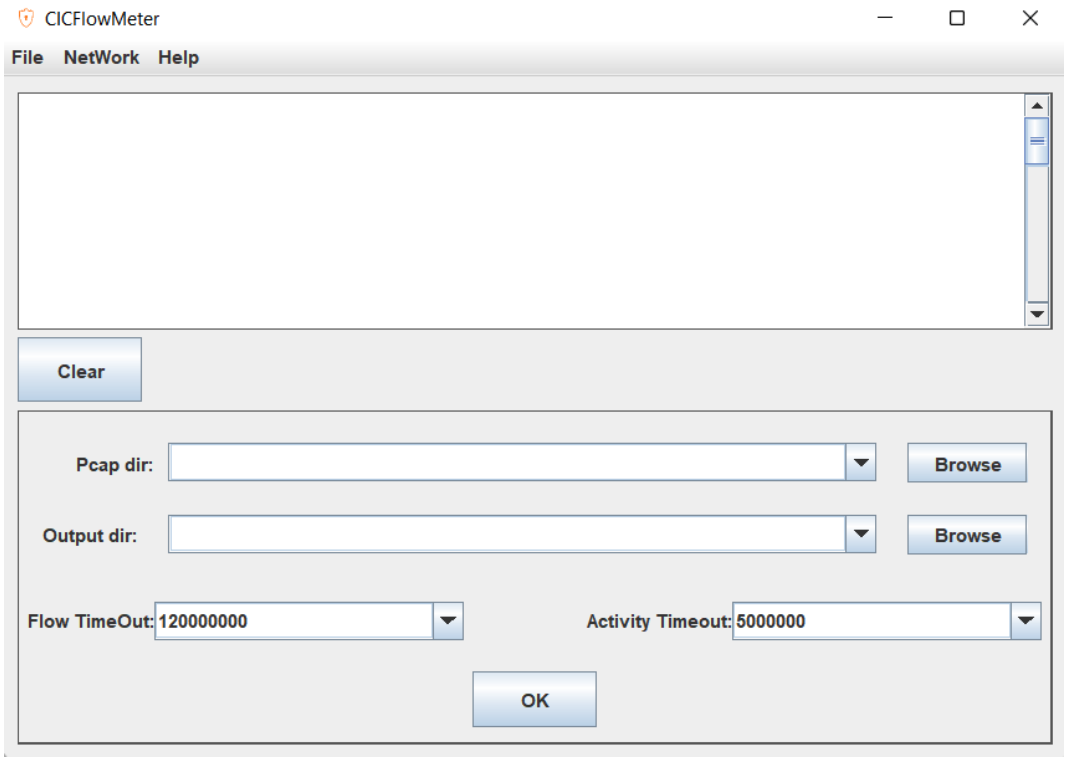
Setelah melakukan pengambilan data pada proses perekaman lalu lintas jaringan, maka tahap selanjutnya yaitu melakukan ekstraksi data pada lalu lintas jaringan yang sudah diambil oleh peneliti. Peneliti menggunakan tools yaitu CICFlowMeter GUI. Seperti Gambar 4.16 Tampilan GUI CICflowMeter berikut:



Gambar 4. 16 Tampilan GUI CICflowMeter

Untuk melakukan ekstraksi dibutuhkan data wireshark dalam bentuk ekstension *pcap* yang data tersebut akan diekstrak dan disimpan menjadi ekstension .csv. untuk melakukan ekstraksi data yang sudah peneliti miliki, pada menu utama

terdapat *NetWork>Offline* kemudian silahkan pilih data yang ingin di ekstrak. Berikut merupakan Gambar 4.17 Tampilan Menu *Offline*



Gambar 4. 17 Tampilan Menu *Offline*

Berikut merupakan tabel hasil ekstraksi lalu lintas jaringan dengan menggunakan tools CICFlowMeter. Disajikan Tabel 4.4 hasil ekstraksi dari data primer.

Tabel 4. 4 Hasil Ekstraksi Data Primer

No	*Network Traffic*	*Extraction Data*
1	Normal	46 baris
2	*Bruteforce*	1750 baris

4.1.1.5 Pelabelan

Hal berikutnya yang dilakukan peneliti adalah memisahkan lalu lintas serangan dari lalu lintas di jaringan. Peneliti menggunakan pemisahan data untuk menjaga keaslian data lalu lintas dan tidak tercampur adukan dengan data yang lain. Lalu lintas *traffic* normal berisi 46 baris, sedangkan lalu lintas *traffic* serangan berisi 1750 baris. Setelah itu, kedua lalu lintas jaringan digabungkan menjadi satu file XLSX sebagai data utama untuk algoritma *Random Forest*.

Dataset MQTT-IOT-IDS2020 dimanfaatkan oleh peneliti sebagai data penelitian sekunder. Beberapa parameter dari dataset MQTT-IOT-IDS2020 digunakan oleh peneliti. Seperti Tabel 4.5 Parameter Protokol MQTT.

Tabel 4. 5 Parameter Protokol MQTT

No	*Datasets feature*	*Research Parameters*
1	TotLen Bwd Pkts	TotLen Bwd Pkts
2	Bwd Pkt Len Max	Bwd Pkt Len Max
3	Bwd Pkt Len Mean	Bwd Pkt Len Mean
4	Bwd Seg Size Avg	Bwd Seg Size Avg
5	Subflow Bwd Byts	Subflow Bwd Byts
6	Init Bwd Win Byts	Init Bwd Win Byts
7	label	label

Rincian baris yang digunakan dalam penelitian ditunjukkan pada Tabel 4.6 Informasi Data Lengkap Penelitian berikut:

Tabel 4. 6 Informasi Data Lengkap Penelitian

Data Type	*Network Traffic Type*	*Number of Data Rows*	*Total (Row x Column)*
Primary data	Normal	46	1796 x 7
	Bruteforce	1750	
Secondary data	Normal	52082	85002 x 7
	Bruteforce	32920	

4.1.2 Praproses Data

Peneliti melakukan praproses data sebelum melanjutkan ke tahap data pembelajaran *Random Forest*. Peneliti kemudian menggunakan metode min-max untuk melakukan rescaling data, dengan tujuan untuk memberikan nilai minimum dan maksimum pada setiap variabel yang akan diubah menjadi *range* 0 sampai 1. Hasil rescaling data numerik ditunjukkan Gambar 4.18. Rescalling baris angka

```
[[6.13585854e-03 9.00439239e-02 1.20000000e-15 7.72909091e-02
  6.13585854e-03 1.00000000e+00]
 [6.52236931e-03 9.00439239e-02 4.80000000e-16 7.67546218e-01
  6.52236931e-03 0.00000000e+00]
 [6.42574162e-03 9.00439239e-02 1.13196000e-12 7.73538462e-01
  6.42574162e-03 0.00000000e+00]
 ...
 [1.20784617e-03 1.83016105e-02 7.50000000e-16 7.50000000e-16
  1.20784617e-03 1.00000000e+00]
 [1.15953232e-03 1.75695461e-02 7.20000000e-16 7.20000000e-16
  1.15953232e-03 1.00000000e+00]
 [1.15953232e-03 1.75695461e-02 7.20000000e-16 7.20000000e-16
  1.15953232e-03 1.00000000e+00]]
```

Gambar 4. 18 Rescalling Baris Angka

4.1.3 *Random Forest*

Pada tahap ini akan dibangun model pelatihan dan data yang telah peneliti kumpulkan akan diuji pada sistem yang telah dibuat. Langkah pertama yang dilakukan oleh peneliti yaitu membuat *resampling* mengacak data-data *training* dengan cara, tahap ini disebut sebagai *bootsraping. Boostraping* dibuat secara berulang sebanyak pohon jumlah yang ditentukan. Untuk tahap ini peneliti menggunakan jumlah pohon yang sudah ditentukan. Gambar 4.19 berikut ini merupakan contoh hasil dari *bootstraping.*

	TotLen_Bwd_Pkts	Bwd_Pkt_Len_Max	Bwd_Pkt_Len_Mean	Bwd_Seg_Size_Avg	Subflow_Bwd_Byts	Init_Bwd_Win_Byts	Label
185	0.001727	0.024691	1.134328e-15	4.596774e-16	0.001727	1.0	Bruteforce
865	0.001604	0.023320	1.074627e-15	4.354839e-16	0.001604	1.0	Bruteforce
114	0.001665	0.024005	1.104478e-15	4.475806e-16	0.001665	1.0	Bruteforce
808	0.001665	0.024005	1.104478e-15	4.475806e-16	0.001665	1.0	Bruteforce
1298	0.001727	0.024691	1.134328e-15	4.596774e-16	0.001727	1.0	Bruteforce
...	...	...	...	...	...	...	...
725	0.001542	0.022634	1.044776e-15	4.233871e-16	0.001542	1.0	Bruteforce
382	0.001727	0.024691	1.134328e-15	4.596774e-16	0.001727	1.0	Bruteforce
770	0.001665	0.024005	1.104478e-15	4.475806e-16	0.001665	1.0	Bruteforce
1723	0.001604	0.023320	1.074627e-15	4.354839e-16	0.001604	1.0	Bruteforce
112	0.001665	0.024005	1.104478e-15	4.475806e-16	0.001665	1.0	Bruteforce

Gambar 4. 19 Sampel Bootsrapping pada Data Primer

Kumpulan *Random Forest* akan dibangun menggunakan data bootstrap yang telah diperoleh. Model yang dibangun akan berfungsi sebagai model utama untuk baris data pelatihan. Sesuai dengan Gambar 4.20, berikut ini adalah ilustrasi Model Random Forest.

```
{'TotLen_Bwd_Pkts <= 0.0017392984829452123': ['Bruteforce',
                                              {'Subflow_Bwd_Byts <= 0.006522369311044545': ['Normal',
                                                                                            'Bruteforce']}]},
{'Bwd_Seg_Size_Avg <= 9.710619600000045e-10': ['Bruteforce',
                                               {'Bwd_Pkt_Len_Max <= 0.016105417276720352': ['Bruteforce',
                                                                                            'Normal']}]},
{'Subflow_Bwd_Byts <= 0.0017392984829452123': ['Bruteforce',
                                               {'Bwd_Pkt_Len_Mean <= 1.1319600000000052e-12': ['Normal',
                                                                                              'Bruteforce']}]},
{'TotLen_Bwd_Pkts <= 0.0017392984829452123': ['Bruteforce', 'Normal']},
{'Bwd_Pkt_Len_Max <= 0.02635431918008785': ['Bruteforce',
                                            {'Bwd_Pkt_Len_Max <= 0.09004392386530015': ['Normal',
                                                                                        'Bruteforce']}]},
{'Subflow_Bwd_Byts <= 0.0017392984829452123': ['Bruteforce',
                                               {'TotLen_Bwd_Pkts <= 0.006522369311044545': ['Normal',
                                                                                            'Bruteforce']}]},
{'Bwd_Seg_Size_Avg <= 9.710619600000045e-10': ['Bruteforce',
                                               {'Bwd_Pkt_Len_Max <= 0.016105417276720352': ['Bruteforce',
                                                                                            'Normal']}]},
{'Subflow_Bwd_Byts <= 0.0017392984829452123': ['Bruteforce', 'Normal']},
```

Gambar 4. 20 Model *Random Forest*

Tahap ini juga disebut sebagai tahap prediksi karena model pelatihan dibangun sebagai *tree forest*. atau tahap pendeteksian *traffic* jaringan. Data pengujian pada data sekunder dideteksi menggunakan model *Random Forest* yang dibuat dari data pelatihan sekunder. Selain itu, Model *Random Forest* dibangun dari data *training* dan akan diuji pada data *testing*. Kemudian disaat proses deteksi sedang berjalan penguji juga memberikan waktu yang digunakan sebagai acuan

durasi proses dari deteksi tersebut berlangsung.

4.1.4 Evaluasi Model

Peneliti akan memberikan penjelasan mengenai prosedur untuk mendapatkan output dari algoritma *Random Forest* pada sub-bab ini. pada hasil prediksi serta akurasi waktu yang diperoleh dari data pengujian dan pelatihan. Nilai akurasi adalah rasio prediksi akurat model terhadap jumlah total prediksi. Peneliti menentukan nilai akurasi sesuai dengan rumus 4.1 (Elnakib et al., 2023).

$$Accuracy = \frac{\mathrm{TP+TN}}{\mathrm{TP+TN+FP+FN}} x100\% \quad (4.1)$$

Nilai Presisi menghitung rasio antara sampel positif yang diprediksi dengan benar dengan jumlah total prediksi positif. Perhitungan *Precision* sesuai dengan rumus 4.2 (Elnakib et al., 2023).

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP+FP}} x100\% \quad (4.2)$$

Nilai *Recall* menghitung rasio antara sampel positif yang diprediksi benar dengan jumlah total sampel positif. Perhitungan *recall* sesuai dengan rumus 4.3 (Elnakib et al., 2023).

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP+FN}} x100\% \quad (4.3)$$

Nilai *f-measure* khususnya melihat nilai presisi dan *recall* dalam menentukan keefektifan nilai. Rumus tersebut digunakan untuk menentukan nilai f-measure. 4.4 (Elnakib et al., 2023).

$$f\text{-}measure = 2.\frac{precision.recall}{precision+recall} x100\% \quad (4.4)$$

Berikut ini adalah penjelasan singkat tentang istilah dan artinya:

1. *True Negative* (TN) apabila baris data normal diperlakukan sebagai baris data normal.
2. *False Negative* (FN) apabila baris data serangan diperlakukan sebagai baris data normal.
3. *True Positif* (TP) apabila baris data serangan diperlakukan sebagai baris data serangan.
4. *False Positif* (FP) apabila baris data normal diperlakukan sebagai baris data serangan.

Data dari Matriks Konfusi dapat digunakan sebagai panduan untuk lebih memahami perbandingan data aktual dan prediksi, yang ditampilkan dalam bentuk tabel Matriks Konfusi *Actual* dan *Prediction* pada Tabel 4.9.

Tabel 4. 7 Matriks Konfusi *Actual* dan *Prediction*

Actual / *Prediction*	*Bruteforce*	Normal
Bruteforce	TP	FP
Normal	FN	TN

4.2 Hasil Uji Coba

Sesuai dengan prosedur dan tahapan yang telah dijelaskan penguji pada sub bab sebelumnya, peneliti menyampaikan hasil uji coba sesuai dengan tahapan yang telah dibuat. Pada data primer dan sekunder dilakukan pengujian sistem. Data

pengujian primer yang digunakan untuk menguji metode Random Forest ditunjukkan pada Tabel 4.8.

Tabel 4. 8 Hasil Uji Coba Random Forest *Primary Data*

Nomor Baris Data	Data *Prediction*	Data *Acctual*
758	Bruteforce	Bruteforce
345	Bruteforce	Bruteforce
1645	Bruteforce	Bruteforce
1566	Bruteforce	Bruteforce
1551	Bruteforce	Bruteforce
…….	……	……
809	Bruteforce	Bruteforce
909	Bruteforce	Bruteforce
1782	Bruteforce	Bruteforce
1488	Bruteforce	Bruteforce
649	Bruteforce	Bruteforce
Duration: 0 detik ***Akurasi***: 99.55456570155901 ***Presisi***: 100.0 ***Recall***: 99.54648526077098 ***F-measure***: 99.7727272727273		

Menggunakan data pengujian dari data primer dengan metode Random Forest, dari hasil pengujian peneliti mengambil kesimpulan dari hasil uji coba tersebut pada Tabel 4.9 disajikan Hasil Output Random Forest *Primary Data.*

Tabel 4. 9 Hasil *Output* Random Forest *Primary Data*

Duration	0 detik
Akurasi	99,55%
Presisi	100%
Recall	99.54%
f-Measure	99,77%

Selanjutnya uji coba menggunakan data uji menggunakan data sekunder menggunakan metode *Random Forest*. Berikut dalam Tabel 4.10 disajikan hasil uji coba random forest data sekunder.

Tabel 4. 10 Hasil Uji Coba Random Forest *Secondary Data*

Nomor Baris Data	Data *Prediction*	Data *Acctual*
72457	Normal	Normal
36287	Normal	Normal
65354	Normal	Normal
12682	Bruteforce	Bruteforce
40409	Normal	Normal
…….	……	……
23579	Bruteforce	Bruteforce
4802	Bruteforce	Bruteforce
16435	Bruteforce	Bruteforce
84572	Normal	Normal
58549	Normal	Normal
Duration: 62 detik ***Akurasi***: 99.77883393722648 ***Presisi***: 100.0 ***Recall***: 99.43407585791691 ***F-measure***: 99.71623498158544		

Dari hasil tabel tersebut peneliti mengambil kesimpulan dari uji coba yang telah dilakukan yang disajikan pada Tabel 4.11 Hasil Output Random Forest *Secondary* Data.

Tabel 4. 11 Hasil *Output* Random Forest *Secondary* Data

Duration	62 detik
Akurasi	99.77%
Presisi	100%
Recall	99.43%
f-Measure	98.71%

4.3 Pembahasan

Dalam tahap pembahasan peneliti pertama-tama menampilkan hasil ini dalam tabel matriks konfusi dari setiap hasil percobaan. Hasil pengujian yang dilakukan terhadap data primer dan sekunder berdasarkan hasil pengujian tidak ditemukan adanya *False Positives* (FP) karena pada dasarnya metode Random Forest memiliki kemampuan untuk mengatasi overfitting pada data pengujian. Berikut adalah hasil

dari masing-masing kumpulan data yang tertera pada Tabel 4.12 yaitu hasil uji coba data primer dan sekunder.

Tabel 4. 12 Hasil Uji Coba *Primary Data* dan *Secondary Data*

***Confusion Matrix Primary Data* Random Forest**		
Actual / **Prediction**	***Bruteforce***	**Normal**
Bruteforce	439	0
Normal	2	8
***Confusion Matrix Secondary Data* Random Forest**		
Aktual / **Prediction**	***Bruteforce***	**Normal**
Bruteforce	8258	0
Normal	47	12946

Berikut adalah perbandingan hasil pengujian kedua dataset data primer dan sekunder dan juga durasi yang dihasilkan menggunakan metode *Random Forest* dengan menggunakan data pengujian dari primer dan sekunder yang menjelaskan bahwa algoritma *Random Forest* dapat dipercaya untuk mendeteksi serangan pada jaringan. Berikut ditampilkan *Measurement Results Primary Data* dan *Secondary Data* pada Tabel 4.13 berikut:

Tabel 4. 13 *Measurement Results Primary Data* dan *Secondary Data*

	Primary	***Secondary***
Duration	0 detik	62 detik
Akurasi	99,55%	99.77%
Presisi	100%	100%
Recall	99.54%	99.43%
F-measure	99,77%	98.71%

Munculnya error pada pengukuran hasil akurasi pada *Random Forest* dikarenakan kurangnya representasi kelas yang tidak seimbang (*imbalanced class*) dikarenakan model *Random Forest* cenderung memiliki kesulitan dalam memprediksi kelas minoritas jika data target memiliki kelas yang tidak seimbang.

Hal ini dapat menyebabkan kesalahan dalam memprediksi kelas minoritas. Seperti data yang penguji sajikan diatas diketahui bahwa pada data primer terdapat jumlah data normal yaitu 46 baris dan jumlah data *bruteforce* yaitu 1750 baris. Sedangkan untuk data sekunder jumlah data normal sebanyak 52082 baris sedangkan data *bruteforce* 32920 baris. Sehingga data primer dan data sekunder tidak memiliki kelas yang seimbang antara data normal dengan data *bruteforce*.

Melakukan *hacking* untuk kepuasan pribadi (*blackhat*) maupun untuk keuntungan pribadi merupakan jalan yang batil, Maka dari itu perlu adanya solusi-solusi pencegahan maupun deteksi peretasan untuk mengurangi kejahatan yang ada di dunia maya. Sejalan dengan bahasan penelitian yang diambil yaitu mengenai deteksi serangan pada protokol MQTT IoT, yang mana IoT merupakan sebuah perangkat keras yang sering mengalami serangan dan sering terjadi setiap hari, maka perlu dibuat dan dirancang solusi untuk dapat mencegah maupun mendeteksi serangan tersebut.

Melakukan deteksi serangan pada protokol IoT merupakan sebuah *ikhtiar* yang dilakukan oleh peneliti, peneliti melakukan pengelompokan dan juga pemisahan antara data serangan maupun data normal. Tindakan *hacking* merupakan aktifitas *illegal* yang mana tujuan dari *hacking* biasanya untuk melakukan pencurian data, manipulasi data, dan merusak reputasi dari pihak yang ditargetkan. Tindakan *hacking* secara diam-diam tanpa persetujuan dari pihak terkait dapat menyebabkan seseorang terkena Pasal 30 ayat 1, ayat 2, dan atau ayat 3 UU No 11/2008 tentang Informasi dan Transaksi Elektronik (ITE), berbunyi (1) Setiap orang dengan sengaja dan tanpa hak atau melawan hukum mengakses Komputer

dan/atau Sistem Elektronik milik orang lain dengan cara apa pun. Ini sejalan dengan ajaran Islam yang menjelaskan tentang perbuatan yang menyimpang dari seseorang. Hal ini disampaikan pada Al-Qur'an telah ditunjukan tentang balasan kebaikan dan keburukan, seperti firman Allah SWT dalam surah An-Nur [24] ayat ke 27:

يَا أَيُّهَا الَّذِينَ آمَنُوا لَا تَدْخُلُوا بُيُوتًا غَيْرَ بُيُوتِكُمْ حَتَّىٰ تَسْتَأْنِسُوا وَتُسَلِّمُوا عَلَىٰ أَهْلِهَا ۚ ذَٰلِكُمْ خَيْرٌ لَكُمْ لَعَلَّكُمْ تَذَكَّرُونَ

"*Hai orang-orang yang beriman, janganlah kamu memasuki rumah yang bukan rumahmu sebelum meminta izin dan memberi salam kepada penghuninya. Yang demikian itu lebih baik bagimu, agar kamu (selalu) ingat.*" *(Q.S. An-Nur:24).*

Dijelaskan dalam Tafsir Jalalain (Jalaluddin al-Mahalli & Jalaluddin as-Suyuthi):

1. Pada lafal (**يَا أَيُّهَا الَّذِينَ آمَنُوا لَا تَدْخُلُوا بُيُوتًا غَيْرَ بُيُوتِكُمْ حَتَّى تَسْتَأْنِسُوا**) Hai orang-orang yang beriman! Janganlah kalian memasuki rumah yang bukan rumah kalian sebelum meminta izin) maksudnya sebelum kalian meminta izin kepada empunya **وَتُسَلِّمُوا عَلَى أَهْلِهَا** (dan memberi salam kepada penghuninya). Seseorang jika mau memasuki rumah orang lain hendaknya ia mengucapkan, "Assalaamu Alaikum, bolehkah aku masuk?" demikianlah menurut tuntunan hadist (Al-Mahalli & Al-Imam Jalaluddin Muhammad).
2. Pada lafal **ذَلِكُمْ خَيْرٌ لَّكُمْ** (Yang demikian itu lebih baik bagi kalian) daripada masuk tanpa izin **لَعَلَّكُمْ تَذَكَّرُونَ** (agar kalian selalu ingat) lafal Tadzakkaruuna dengan mengidgamkan huruf Ta kedua kepada huruf Dzal; maksudnya supaya kalian mengerti akan kebaikan meminta izin itu, kemudian kalian

mengerjakannya (Al-Mahalli & Al-Imam Jalaluddin Muhammad).

Pada penejelasan tafsir Jalalain peneliti mengambil kesimpulan bahwa umat Islam memerintahkan untuk tidak memasuki rumah orang lain tanpa izin atau tanpa memberi salam terlebih dahulu. Tindakan seperti itu dianggap sebagai tindakan yang tidak sopan dan tidak pantas dilakukan. Dalam ayat ini juga terkandung pesan bahwa tindakan sopan santun dan menghargai hak privasi orang lain sangatlah penting dalam Islam.

Alhasil, tujuan dari penelitian ini adalah untuk menemukan serangan pada protokol MQTT IoT. Diharapkan para peneliti dapat membedakan dan merespon upaya peretasan jaringan IoT di perusahaan dengan lebih tepat dan cepat jika menggunakan sistem yang dikembangkan.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Peneliti telah melakukan penelitian mengenai deteksi serangan pada protokol MQTT IoT menggunakan metode *Random Forest* data yang digunakan untuk menjadi bahan penelitian menggunakan data lalu lintas jaringan normal dan juga data serangan *bruteforce*. Dalam penelitian ini menggunakan dua data yaitu data primer yang diambil dari *Lab Hacking Environment* yang peneliti buat dan untuk data sekunder peneliti mengguanakan dataset dari IEEE MQTT-IOT-IDS2020 data tersebut dikumpulkan dari *lab hacking* para ahli untuk tujuan penelitian pada bidang keamanan IoT. Dua data tersebut akan dibagi menjadi *data training* dan *data testing* pada setiap dataset. Data primer akan dilakukan splitting menjadi dua bagian yaitu *data training* dengan proporsi 75% dan *data testing* dengan proporsi 25% sedangkan untuk data sekunder akan dilakukan splitting menjadi dua bagian yaitu yaitu *data training* dengan proporsi 75% dan *data testing* dengan proporsi 25%. Parameter yang diguanakan dalam penelitian ini ada 7 parameter yaitu *TotLen Bwd Pkts, Bwd Pkt Len Max, Bwd Pkt Len Mean, Bwd Seg Size Avg, Subflow Bwd Byts, Init Bwd Win Byts dan label*. Pada Model pelatihan yang dibagun oleh peneliti setiap data primer dan sekunder pada data training akan digunakan sebagai model untuk data testing.

Peneliti melakukan uji coba pada data primer dan hasil yang didapatkan yaitu akurasi 99,55%, presisi 100%, recall 99,54% dan f-measure 99,77% durasi yang dibutuhkan untuk mendapatkan hasil tersebut dengan baris data sebanyak 1796

baris yaitu selama 0 detik. Sedangkan untuk data sekunder peneliti mendapatkan akurasi 99.77%, presisi 100%, recall 99.43%, f-measure 98.71% durasi yang dibutuhkan untuk mendapatkan hasil tersebut dengan baris data sebanyak 85002 baris yaitu selama 62 detik.

5.2 Saran

Dari hasil penelitian yang telah dilakukan dan program yang telah dibuat masih banyak yang harus diperbaiki dan diterapkan pada penelitian selanjutnya.

1. Pengambilan data lab yang telah dibuat oleh peneliti masih sedikit terlebih untuk data primer pada jaringan normal, dan juga perlu untuk menerapkan beberapa prosedur dan juga kompleksitas untuk membuat *lab hacking environment* yang sesuai dan tepat.
2. Program yang digunakan dalam penelitian ini sifatnya masih belum bisa melakukan deteksi secara dini (*realtime detection*) peneliti berharap penelitian selanjutnya dapat melakukan deteksi secara realtime serangan siber pada jaringan IoT.

DAFTAR PUSTAKA

Alaiz-Moreton, H., Aveleira-Mata, J., Ondicol-Garcia, J., Muñoz-Castañeda, A. L., García, I., & Benavides, C. (2019). Multiclass Classification Procedure for Detecting Attacks on MQTT-IoT Protocol. *Complexity*, *2019*. https://doi.org/10.1155/2019/6516253

Alghuried, A. (2017). *A Model for Anomalies Detection in Internet of Things (IoT) Using Inverse Weight Clustering and Decision Tree*. https://doi.org/10.21427/D7WK7S

Amalia, S., Deborah, I., & Yulita, I. N. (2022). Comparative analysis of classification algorithm: Random Forest, SPAARC, and MLP for airlines customer satisfaction. *SINERGI*, *26*(2), 213. https://doi.org/10.22441/sinergi.2022.2.010

Arafat, S., Kom, M., & Kom. (2016). SISTEM PENGAMANAN PINTU RUMAH BERBASIS Internet Of Things (IoT) Dengan ESP8266. *Technologia : Jurnal Ilmiah*, *7*(4). https://doi.org/10.31602/TJI.V7I4.661

Dinculeană, D., & Cheng, X. (2019). Vulnerabilities and Limitations of MQTT Protocol Used between IoT Devices. *Applied Sciences 2019, Vol. 9, Page 848*, *9*(5), 848. https://doi.org/10.3390/APP9050848

Elnakib, O., Shaaban, E., Mahmoud, M., & Emara, K. (2023). EIDM: deep learning model for IoT intrusion detection systems. *Journal of Supercomputing*, 1–21. https://doi.org/10.1007/S11227-023-05197-0/TABLES/4

Gkoulalas-Divanis, A., Marchetti, M., Avresky, D. R. (Dimitri R., IEEE Computer Society, IEEE Computer Society. TC on Distributed Processing, & Institute of Electrical and Electronics Engineers. (n.d.). *2019 IEEE 18th International Symposium on Network Computing and Applications (NCA) : September 26-28, 2019*.

Hasan, M., Milon Islam, M., Ishrak Islam Zarif, M., & Hashem, M. (2019). *Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches*. https://doi.org/10.1016/j.iot.2019.10

Hussein, A. R. H. (2019). Internet of Things (IOT): Research Challenges and Future Applications. *International Journal of Advanced Computer Science and Applications*, *10*(6), 77–82. https://doi.org/10.14569/IJACSA.2019.0100611

Mohamudally, N., & Peermamode-Mohaboob, M. (2018). Building An Anomaly Detection Engine (ADE) For IoT Smart Applications. *Procedia Computer Science*, *134*, 10–17. https://doi.org/10.1016/J.PROCS.2018.07.138

Primajaya, A., & Sari, B. N. (2018). Random Forest Algorithm for Prediction of Precipitation. *Indonesian Journal of Artificial Intelligence and Data Mining (IJAIDM)*, *1*(1), 27–31.

Ryan, P. J., & Watson, R. B. (2017). Research Challenges for the Internet of Things: What Role Can OR Play? *Systems 2017, Vol. 5, Page 24*, *5*(1), 24. https://doi.org/10.3390/SYSTEMS5010024

Sulistyo Nugroho, Y., & Nova Emiliyawati, dan. (n.d.). *Sistem Klasifikasi Variabel Tingkat Penerimaan Konsumen Terhadap Mobil Menggunakan Metode Random Forest*. http://archive.ics.uci.edu/ml/

Widagdo, B., & Rofik, M. (2019). Internet of Things as Engine of Economic Growth in Indonesia. In *INDONESIAN JOURNAL OF BUSINESS AND ECONOMICS* (Vol. 2, Issue 1). https://journal.uniku.ac.id/index.php/ijbe

Muhammad, Damar Adyun (2020). *Detect distributed denial of service using Random Forest*. Undergraduate thesis, State Islamic University of Maulana Malik Ibrahim.

Al-Mahalli, A. J. M., & Al-Imam Jalaluddin Abdurrahman bin Abu Bakar As-Sayuthi, N. J. Tafsir jalalain / Al-Imam Jalaluddin Muhammad Al-Mahalli, Al-Imam Jalaluddin Abdurrahman bin Abu Bakar As-Suyuthi ; penerjemah, Najib Junaidi, Lc (Edisi Buku 131.). Surabaya : Pustaka eLBA, 2015.

Kementrian Komunikasi dan Informatika. 2008. Undang-Undang Republik Indonesia Nomor 11 tahun 2008 Tentang Informasi dan Transaksi Elektronik. Kementrian Komunikasi dan Informatika. Jakarta: Direktorat Jenderal Informasi dan Komunikasi Publik

Symantec, N. (2016). Norton Cyber Security Insights Report Global Corporation. Symantec Corporation.

http://www.idsirtii.or.id/halaman/tentang/sejarah-id-sirtii-cc.html (diakses pada tanggal 6 Januari 2021)