Analisis Resiliensi Protokol Kriptografi dan SSL/TLS pada *Tor Browser* Terhadap Serangan *Website Fingerprinting* Berbasis *Machine Learning*

Nakeisha Valya Shakila - 18223133
Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: nakeishavalya@gmail.com , 18223133@std.stei.itb.ac.id

***Abstrak*—Tor Browser menerapkan mekanisme *Onion Routing* dan HTTPS berbasis TLS untuk menjaga anonimitas dan kerahasiaan komunikasi pengguna. Namun, metadata lalu lintas jaringan masih berpotensi dimanfaatkan dalam serangan *Website Fingerprinting*. Penelitian ini menganalisis tingkat keteridentifikasian metadata lalu lintas sebagai indikator resiliensi mekanisme kriptografi dan protokol SSL/TLS pada Tor Browser terhadap serangan tersebut menggunakan algoritma Decision Tree dan Random Forest. Data lalu lintas dari enam situs web direkam menggunakan Wireshark dan diekstraksi menjadi lima fitur statistik. Hasil eksperimen menunjukkan bahwa Random Forest memperoleh akurasi sebesar 91,67 persen dan *weighted F1-score* sebesar 0,9111, lebih tinggi dibandingkan Decision Tree. Temuan ini menunjukkan bahwa meskipun isi komunikasi telah dienkripsi, metadata lalu lintas masih mengandung pola statistik yang dapat dimanfaatkan dalam serangan *Website Fingerprinting*.**

Kata Kunci*—*Tor Browser, Website Fingerprinting, Machine Learning, Random Forest, Analisis Trafik, SSL/TLS

I. PENDAHULUAN

Dalam era digital saat ini, privasi pengguna internet menghadapi ancaman yang semakin besar akibat pengawasan dalam skala besar dan analisis lalu lintas jaringan (*network traffic analysis*). Aktivitas ini memungkinkan pihak ketiga memantau pola komunikasi serta melacak identitas individu. Untuk mengurangi risiko tersebut, Tor (*The Onion Router*) banyak digunakan guna menjaga anonimitas melalui komunikasi terenkripsi berlapis pada jaringan terdistribusi [1].

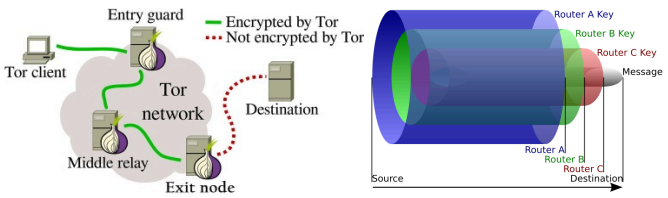
Tor Browser mengamankan komunikasi pengguna melalui mekanisme *Onion Routing*, yaitu dengan meneruskan lalu lintas melalui sirkuit yang terdiri atas *Entry Relay*, *Middle Relay*, dan *Exit Relay* sehingga tidak ada satu *node* pun yang mengetahui identitas pengguna sekaligus tujuan akhirnya [1]. Untuk melindungi isi komunikasi secara *end-to-end*, Tor Browser tetap mengandalkan HTTPS yang diimplementasikan melalui protokol TLS guna menjamin kerahasiaan, integritas, dan autentikasi data [2]. Namun, sebagai jaringan anonimitas berlatensi rendah (*low-latency anonymity networ*k), Tor memiliki keterbatasan dalam menyamarkan karakteristik lalu lintas jaringan secara menyeluruh. Akibatnya, metadata seperti ukuran paket, arah aliran trafik, volume data, dan pola waktu transmisi masih dapat diamati serta dimanfaatkan dalam serangan *Website Fingerprinting* (WF) untuk mengidentifikasi situs web yang diakses pengguna tanpa perlu mendekripsi isi komunikasi [3].

Ancaman terhadap privasi pengguna Tor semakin meningkat seiring berkembangnya teknik WF berbasis *machine learning*. Metode WF tradisional umumnya memanfaatkan fitur statistik lalu lintas yang diklasifikasikan menggunakan berbagai algoritma *machine learning*, sedangkan pendekatan modern telah berkembang menuju model *deep learning* yang mampu mengekstraksi karakteristik lalu lintas secara otomatis dari data mentah [4]. Meskipun demikian, algoritma berbasis pohon keputusan seperti Decision Tree dan Random Forest masih banyak digunakan karena memiliki efisiensi komputasi yang tinggi, waktu pelatihan yang singkat, serta tingkat interpretabilitas yang lebih baik dibandingkan model *deep learning* yang cenderung bersifat *black-box* [5]. Karakteristik tersebut menjadikan kedua algoritma relevan untuk menganalisis pola lalu lintas yang masih dapat diamati pada jaringan Tor.

Oleh karena itu, penelitian ini bertujuan untuk menganalisis tingkat keteridentifikasian metadata lalu lintas sebagai indikator resiliensi mekanisme kriptografi dan TLS pada Tor Browser terhadap serangan *Website Fingerprinting* berbasis *machine learning*, serta mengevaluasi efektivitas algoritma Decision Tree dan Random Forest dalam mengidentifikasi pola lalu lintas yang berpotensi mengungkap sidik jari situs web.

II. DASAR TEORI

A. *Tor Browser*



Gambar. 1. Arsitektur Jaringan Tor dan Mekanisme Onion Routing

Jaringan Tor (*The Onion Router*) menggunakan arsitektur terdesentralisasi berbasis ribuan *relay* yang dioperasikan oleh berbagai pihak di seluruh dunia. Saat pengguna mengakses internet melalui Tor Browser, data pengguna tidak langsung dikirim ke server tujuan, melainkan dilewatkan melalui sirkuit acak yang terdiri dari *Entry/Guard Relay*, *Middle Relay*, dan *Exit Relay*. Karena setiap *relay* hanya mengetahui sebagian informasi jalur komunikasi, identitas pengirim dan tujuan akhir data tetap terjaga kerahasiaannya [6].

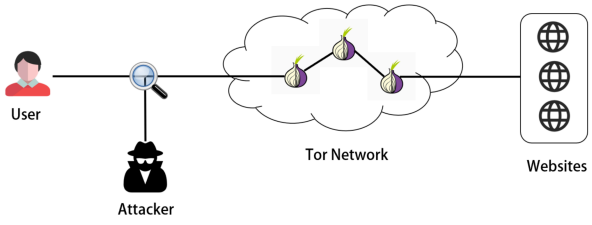
Mekanisme anonimitas pada jalur tersebut digerakkan oleh teknik *Onion Routing*, di mana data pengguna dibungkus oleh beberapa lapisan enkripsi sekaligus sebelum dikirimkan. Berdasarkan pada analogi pengupasan lapisan bawang secara bertahap, setiap relay di sepanjang sirkuit hanya memiliki kunci kriptografi untuk membuka satu lapisan terluar yang spesifik [7]. Proses dekripsi di setiap node hanya mengungkap instruksi rute untuk lompatan (*hop*) berikutnya, sehingga data asli di lapisan terdalam tetap tersembunyi dan aman sepanjang perjalanan.

B. SSL/TLS dan Kriptografi Tor

Dalam operasionalnya, Tor mengombinasikan kriptografi kunci asimetris (*public key*) dan kunci simetris. Pada tahap pembentukan sirkuit, Tor Client melakukan *key exchange* dengan setiap relay menggunakan protokol berbasis Diffie-Hellman untuk menghasilkan kunci sesi simetris yang digunakan untuk mengenkripsi dan mendekripsi lalu lintas data selama komunikasi berlangsung [8]. Melalui mekanisme *Onion Routing*, data pengguna dilindungi oleh beberapa lapisan enkripsi saat melewati rangkaian relay Tor. Namun, ketika data keluar melalui *Exit Node*, lapisan enkripsi Tor berakhir sehingga data dapat diamati oleh operator *Exit Node* maupun pihak lain apabila layanan tujuan tidak menggunakan enkripsi tambahan seperti HTTPS [6].

Oleh karena itu, SSL/TLS berperan melengkapi mekanisme keamanan Tor. Istilah SSL/TLS digunakan untuk merujuk pada protokol komunikasi aman yang saat ini diimplementasikan melalui Transport Layer Security (TLS) sebagai penerus Secure Sockets Layer (SSL). Tor menyembunyikan identitas dan rute komunikasi pengguna melalui *Onion Routing*, sedangkan TLS menjamin kerahasiaan, integritas, dan autentikasi data secara *end-to-end*. Pada akses HTTPS, data terlebih dahulu diamankan menggunakan TLS sebelum dibungkus kembali oleh lapisan enkripsi Tor selama transmisi [8]. Dengan demikian, meskipun enkripsi Tor berakhir di *Exit Node*, isi komunikasi tetap terlindungi oleh TLS hingga mencapai server tujuan. Untuk konsistensi penulisan, istilah tersebut selanjutnya ditulis sebagai TLS.

C. Website Fingerprinting



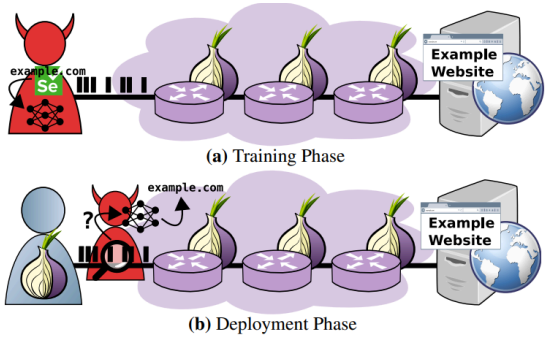
Gambar. 2. Skenario Serangan Website Fingerprinting pada Jaringan Tor

Serangan *Website Fingerprinting* (WF) dikonseptualisasikan sebagai model ancaman pasif lokal di mana *attacker* bertindak sebagai pengamat yang mengawasi lalu lintas data antara pengguna dan jaringan Tor, seperti ISP atau operator Wi-Fi lokal. Dalam model ini, penyerang tidak perlu mendekripsi isi komunikasi maupun meretas struktur internal sirkuit Tor. Sebagai gantinya, teknik analisis trafik ini memanfaatkan karakteristik metadata paket yang masih dapat diamati, seperti ukuran paket, arah aliran data, volume transmisi, dan pola waktu kedatangan paket (*inter-packet timing*) [9]. Karakteristik tersebut kemudian diekstraksi dan digunakan untuk membentuk sebuah *fingerprint* yang merepresentasikan pola lalu lintas unik dari suatu situs web.

Ketika diterapkan pada jaringan Tor, serangan WF bertujuan untuk memperkirakan situs web yang sedang diakses pengguna berdasarkan pola lalu lintas yang diamati. Meskipun Tor dirancang untuk menyembunyikan identitas pengguna dan tujuan komunikasi melalui mekanisme *Onion Routing*, metadata trafik tetap dapat diamati oleh penyerang pasif. Akibatnya, situs web yang dikunjungi pengguna berpotensi diidentifikasi dengan tingkat akurasi yang tinggi tanpa perlu memecahkan mekanisme enkripsi yang digunakan oleh Tor maupun TLS [9]. Untuk mengurangi efektivitas serangan tersebut, berbagai mekanisme pertahanan telah dikembangkan, salah satunya adalah *traffic padding*, yaitu teknik penyisipan paket tambahan untuk mengaburkan pola asli lalu lintas sehingga lebih sulit dikenali oleh penyerang.

D. Machine Learning

Machine Learning merupakan pendekatan komputasional yang memungkinkan sistem mempelajari pola dari data pelatihan untuk membangun model yang mampu mengenali dan mengklasifikasikan karakteristik trafik jaringan secara otomatis [9]. Dalam konteks *Website Fingerprinting*, *machine learning* digunakan untuk mempelajari hubungan antara pola lalu lintas jaringan dan situs web yang diakses pengguna melalui pendekatan *supervised learning*, yaitu metode pembelajaran yang memanfaatkan data pelatihan yang telah diketahui label kelasnya untuk membangun model klasifikasi. Setelah model dilatih, data baru dapat dianalisis dan diklasifikasikan ke dalam kelas yang paling sesuai berdasarkan pola yang telah dipelajari sebelumnya [9].



Gambar. 3. Tahapan pada Website Fingerprinting Berbasis Machine Learning

Dalam *Website Fingerprinting* pada Tor Browser, klasifikasi trafik dipandang sebagai permasalahan *supervised multi-class classification*. Prosesnya diawali dengan tahap ekstraksi fitur yang mengubah data mentah berupa urutan

paket menjadi vektor fitur numerik yang merepresentasikan karakteristik suatu situs web. Penerapan *machine learning* umumnya terdiri atas dua tahap utama, yaitu *training phase* dan *deployment phase*. Pada *training phase*, sampel trafik dari sejumlah situs web target digunakan untuk melatih model agar dapat mempelajari perbedaan pola lalu lintas antar situs web. Setelah model terlatih, *deployment phase* dilakukan dengan menganalisis lalu lintas yang diamati dan membandingkannya dengan pola yang telah dipelajari sebelumnya sehingga model dapat memperkirakan situs web yang sedang diakses pengguna berdasarkan tingkat kemiripan pola trafik yang terdeteksi [10].

E. Decision Tree dan Random Forest

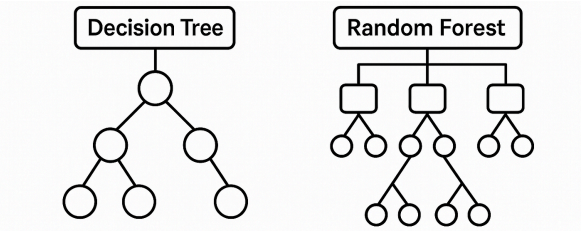
Decision Tree (Pohon Keputusan) adalah algoritma *machine learning* terawasi (*supervised learning*) yang digunakan untuk mengklasifikasikan data berdasarkan serangkaian aturan keputusan. Algoritma ini bekerja dengan membagi data secara bertahap berdasarkan fitur tertentu hingga diperoleh kelompok data yang semakin homogen. Pemilihan fitur terbaik pada setiap percabangan dilakukan dengan mengukur tingkat ketidakpastian (*impurity*) data menggunakan metrik *Entropy* atau *Gini Impurity* [11]. *Entropy* digunakan untuk mengukur tingkat ketidakpastian atau ketidakteraturan distribusi kelas dalam suatu himpunan data [11].

$$Entropy(S) = -\sum_{i=1}^{c} p_i \log_2(p_i)$$

Di mana S melambangkan himpunan sampel data, c merupakan total jumlah kelas target, dan P_i menyatakan probabilitas sampel yang termasuk dalam kelas ke-i. Nilai *Entropy* digunakan untuk menghitung *Information Gain*, yaitu ukuran yang menunjukkan seberapa baik suatu atribut dapat mengurangi ketidakpastian data setelah proses pembagian (*splitting*) dilakukan [11]. Selain *Entropy*, *Gini Impurity* juga banyak digunakan karena lebih sederhana dan efisien secara komputasi. *Gini Impurity* mengukur kemungkinan terjadinya kesalahan klasifikasi apabila suatu sampel diambil dan diberi label secara acak, yang dirumuskan secara matematis sebagai berikut:

$$Gini(S) = 1 - \sum_{i=1}^{c} (p_i)^2$$

Meskipun transparan dan mudah diinterpretasikan sebagai *white-box model*, Decision Tree memiliki kelemahan berupa sensitivitas yang tinggi terhadap perubahan kecil pada data pelatihan [12]. Kondisi ini dapat menyebabkan overfitting, yaitu ketika model terlalu menyesuaikan diri dengan data pelatihan sehingga kemampuan generalisasinya terhadap data baru menurun.



Gambar. 4. Ilustrasi Struktur Algoritme Decision Tree dan Random Forest

Untuk mengatasi kelemahan tersebut, dikembangkan Random Forest, yaitu metode *ensemble learning* yang membangun banyak pohon keputusan dan menentukan prediksi akhir melalui *majority voting*. Random Forest menerapkan teknik *Bagging* (*Bootstrap Aggregating*) dan pemilihan subset fitur secara acak (*Feature Randomness*), sehingga meningkatkan keragaman model, kemampuan generalisasi, dan mengurangi risiko *overfitting* [13]. Sebagai *rule of thumb* pada permasalahan klasifikasi, jumlah fitur acak (m) yang digunakan pada setiap percabangan node ditentukan berdasarkan total jumlah fitur tersedia (M) melalui persamaan

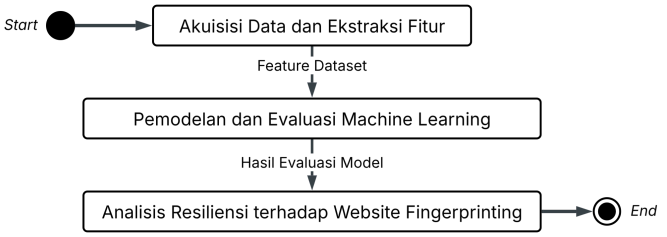
$$m = \sqrt{M}$$

Ketika seluruh pohon digabungkan melalui mekanisme *majority voting*, bias model secara umum tetap relatif konstan, sedangkan variansnya menurun seiring bertambahnya jumlah pohon. Akibatnya, Random Forest memiliki kemampuan generalisasi yang lebih baik serta lebih tangguh terhadap noise dibandingkan Decision Tree tunggal [13].

III. Metodologi Penelitian dan Eksperimen

A. Skema Penelitian dan Eksperimen

Penelitian ini menggunakan metode eksperimen untuk menganalisis resiliensi mekanisme kriptografi dan protokol TLS pada Tor Browser terhadap serangan *Website Fingerprinting* berbasis *machine learning*. Eksperimen dilakukan dengan memanfaatkan metadata lalu lintas jaringan yang diperoleh selama proses akses situs web melalui jaringan Tor. Metadata tersebut kemudian diproses melalui serangkaian tahapan eksperimen yang saling terintegrasi untuk mengevaluasi tingkat resiliensi Tor Browser terhadap serangan *Website Fingerprinting*.



Gambar. 5. Tahapan Penelitian dan Eksperimen

Penelitian terdiri atas tiga tahapan utama, yaitu akuisisi data dan ekstraksi fitur, pemodelan dan evaluasi *machine learning*, serta analisis resiliensi terhadap serangan *Website Fingerprinting*. Ketiga tahapan tersebut membentuk alur eksperimen yang digunakan untuk mengevaluasi tingkat keteridentifikasian lalu lintas jaringan Tor terhadap serangan *Website Fingerprinting*.

TABEL I.

Komponen	Konfigurasi
Browser	Tor Browser
Network Capture	Wireshark
Jumlah Website	6
Jumlah Sampel	60

Komponen	Konfigurasi
Jumlah Fitur	5
Train-test Split	80:20
Algoritme	Decision Tree dan Random Forest
Optimasi	Hyperparameter Tuning dengan GridSearchCV
Validasi	5-Fold Stratified Cross-Validation
Metrik Evaluasi	Accuracy, Precision, Recall, F1-Score, dan Confusion Matrix

Konfigurasi eksperimen pada Tabel I dirancang untuk menghasilkan representasi lalu lintas yang seimbang serta menyediakan mekanisme evaluasi yang objektif terhadap kemampuan model dalam mengidentifikasi pola trafik pada jaringan Tor. Penjelasan lebih rinci mengenai proses pembentukan *feature dataset*, pemodelan *machine learning*, dan evaluasi hasil klasifikasi pada setiap tahapan eksperimen dibahas pada subbab berikutnya.

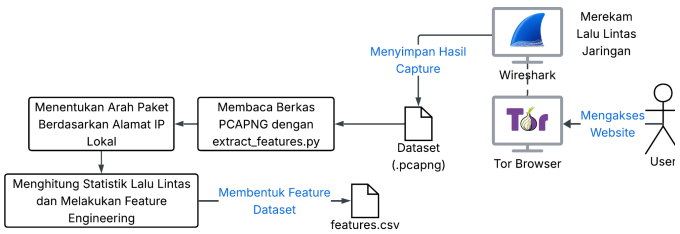
B. Akuisisi Data dan Ekstraksi Fitur

Tahap akuisisi data dan ekstraksi fitur bertujuan mengubah lalu lintas jaringan saat mengakses situs web melalui Tor Browser menjadi dataset numerik untuk pemodelan *machine learning*. Data diperoleh dengan mengakses enam situs web target menggunakan Tor Browser dan merekam lalu lintas paket menggunakan Wireshark. Setiap situs direkam sebanyak sepuluh kali sehingga diperoleh enam puluh berkas *Packet Capture Next Generation* (PCAPNG) dengan distribusi sampel yang seimbang. Selanjutnya, berkas PCAPNG diproses melalui ekstraksi dan rekayasa fitur (*feature engineering*) untuk menghasilkan dataset terstruktur sebagai masukan model *machine learning*.

TABEL II. DISTRIBUSI DATASET HASIL AKUISISI DATA

Website	Jumlah Sampel	Website	Jumlah Sampel
Amazon	10	ITB	10
CNN	10	Wikipedia	10
Github	10	Youtube	10
Total		**60**	

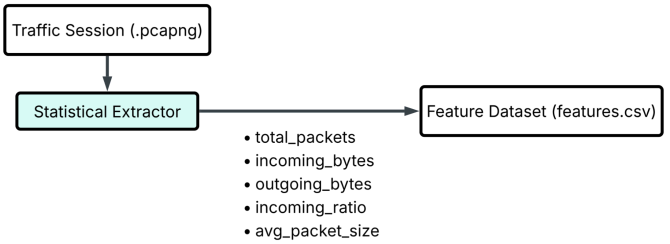
Seluruh hasil perekaman disimpan dalam format PCAPNG dan dikelompokkan ke dalam direktori pcapng_data/.



Gambar. 6. Tahap Akuisisi Data dan Ekstraksi Fitur

Proses diawali dengan perekaman lalu lintas jaringan menggunakan Tor Browser dan Wireshark untuk menghasilkan dataset dalam format PCAPNG. Selanjutnya, setiap berkas PCAPNG dibaca menggunakan skrip extract_features.py, kemudian dilakukan identifikasi arah paket berdasarkan alamat IP lokal untuk menentukan paket masuk dan paket keluar secara akurat. Informasi tersebut selanjutnya digunakan untuk menghitung karakteristik statistik lalu lintas dan membentuk dataset fitur yang akan digunakan pada tahap pemodelan *machine learning*.

Oleh karena itu, dilakukan proses ekstraksi fitur untuk mengubah data mentah menjadi representasi numerik yang menggambarkan karakteristik statistik suatu sesi komunikasi.



Gambar. 7. Representasi Proses Ekstraksi Fitur

Setiap sesi lalu lintas jaringan diproses menggunakan *Statistical Extractor* untuk menghasilkan lima fitur, yaitu total_packets, total_incoming_bytes, total_outgoing_bytes, incoming_ratio, dan avg_packet_size. Dua fitur terakhir merupakan *derived features* hasil *feature engineering* yang dirancang untuk merepresentasikan proporsi arah aliran data dan ukuran rata-rata paket. Pemilihan fitur tersebut didasarkan pada asumsi bahwa karakteristik lalu lintas masih dapat diamati meskipun isi komunikasi (*payload*) telah dilindungi oleh mekanisme kriptografi dan TLS pada Tor Browser.

Perhitungan *derived features* dilakukan dengan terlebih dahulu menentukan total volume lalu lintas (total_bytes) sebagai penjumlahan data masuk dan data keluar. Selanjutnya, nilai incoming_ratio dihitung dari perbandingan antara data masuk dan total lalu lintas, sedangkan avg_packet_size diperoleh dari rata-rata ukuran data pada setiap paket. Implementasi perhitungan *derived features* pada penelitian ini ditunjukkan pada Gambar 8.

```
total_bytes = incoming + outgoing
incoming_ratio = incoming / total_bytes if total_bytes > 0 else 0.0
avg_packet_size = total_bytes / count if count > 0 else 0.0
```

Gambar. 8. Perhitungan derived features

Kondisi if ... else 0.0 diterapkan untuk menghindari kesalahan pembagian (*division by zero*) ketika tidak terdapat paket atau lalu lintas yang tercatat selama proses pengamatan. Selain itu, atribut total_bytes tidak digunakan sebagai fitur masukan model karena memiliki hubungan linear langsung dengan total_incoming_bytes dan total_outgoing_bytes, sehingga berpotensi menimbulkan redundansi informasi pada proses pembelajaran model.

C. Pemodelan dan Evaluasi Machine Learning

Tahap pemodelan dan evaluasi *machine learning* bertujuan untuk membangun model klasifikasi yang mampu mengenali pola lalu lintas jaringan berdasarkan *feature dataset* yang telah dibentuk pada tahap sebelumnya. Dataset features.csv yang

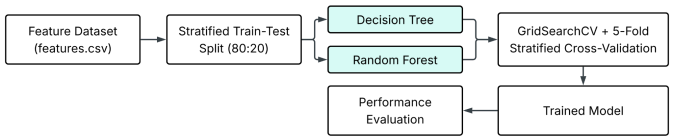
terdiri atas enam puluh sampel dan lima atribut numerik digunakan sebagai masukan untuk melatih dua algoritma klasifikasi, yaitu Decision Tree dan Random Forest. Karena setiap sampel telah memiliki label situs web yang diketahui, proses pembelajaran dilakukan menggunakan pendekatan *supervised learning*.

Sebelum proses pelatihan dilakukan, dataset terlebih dahulu dipisahkan menjadi data pelatihan (*training set*) dan data pengujian (*testing set*) menggunakan metode *Stratified Train-Test Split* dengan rasio 80:20. Pendekatan ini dipilih untuk mempertahankan proporsi jumlah sampel pada setiap kelas sehingga distribusi data pada data pelatihan dan data pengujian tetap seimbang.

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
```

Gambar. 9. Pembagian Dataset Menggunakan Stratified Train-Test Split

Sebanyak 80% data digunakan sebagai data pelatihan dan 20% sisanya digunakan sebagai data pengujian. Parameter stratify=y diterapkan untuk memastikan proporsi jumlah sampel pada setiap kelas tetap terjaga selama proses pembagian dataset.



Gambar. 10. Tahap Pemodelan dan Evaluasi Machine Learning

Setelah proses pembagian dataset selesai dilakukan, data pelatihan digunakan untuk membangun model klasifikasi menggunakan algoritma Decision Tree dan Random Forest. Implementasi kedua algoritma dilakukan menggunakan pustaka Scikit-learn.

```
grid_search_dt = GridSearchCV(
    DecisionTreeClassifier(random_state=42),
    param_grid_dt,
    cv=StratifiedKFold(n_splits=5, shuffle=True, random_state=42),
    scoring='f1_weighted',
    n_jobs=-1,
    verbose=1
)
```

Gambar. 11. Implementasi GridSearchCV pada Decision Tree

Untuk memperoleh model Decision Tree yang optimal, penelitian ini menerapkan hyperparameter tuning menggunakan GridSearchCV dan 5-Fold Stratified Cross-Validation untuk mengevaluasi berbagai kombinasi parameter yang didefinisikan pada param_grid_dt. Validasi dilakukan menggunakan *5-Fold Stratified Cross-Validation*, di mana data pelatihan dibagi menjadi lima lipatan dengan proporsi kelas yang tetap seimbang. Parameter scoring='f1_weighted' digunakan sebagai fungsi evaluasi karena mempertimbangkan keseimbangan antara nilai *precision* dan *recall* pada seluruh kelas, sedangkan n_jobs=-1 memungkinkan proses komputasi dijalankan secara paralel pada seluruh inti prosesor yang tersedia.

```
grid_search_rf = GridSearchCV(
    RandomForestClassifier(random_state=42, n_jobs=-1),
    param_grid_rf,
    cv=StratifiedKFold(n_splits=5, shuffle=True, random_state=42),
    scoring='f1_weighted',
    n_jobs=-1,
    verbose=1
)
```

Gambar. 12. Implementasi GridSearchCV pada Random Forest

Seperti pada Decision Tree, algoritma Random Forest juga dioptimasi menggunakan GridSearchCV dan *5-Fold Stratified Cross-Validation*. Random Forest dibangun sebagai metode *ensemble learning* yang terdiri atas banyak pohon keputusan independen, sedangkan penggunaan parameter n_jobs=-1 memungkinkan proses pelatihan dan pencarian parameter dilakukan secara pararel sehingga waktu komputasi menjadi lebih efisien. Setelah proses pencarian parameter selesai dilakukan, metode fit() digunakan untuk melatih model menggunakan data pelatihan dan menghasilkan model klasifikasi terbaik berdasarkan nilai *F1-Score*.

Tahap selanjutnya adalah evaluasi model menggunakan data pengujian yang sebelumnya tidak digunakan selama proses pelatihan. Evaluasi dilakukan menggunakan metrik *Accuracy*, *Precision*, *Recall*, dan *F1-Score*, serta *Confusion Matrix* untuk menganalisis distribusi prediksi yang benar dan salah pada setiap kelas situs web. Hasil evaluasi tersebut kemudian digunakan sebagai dasar untuk menganalisis tingkat resiliensi mekanisme kriptografi dan TLS pada Tor Browser terhadap serangan *Website Fingerprinting*.

IV. Hasil dan Analisis

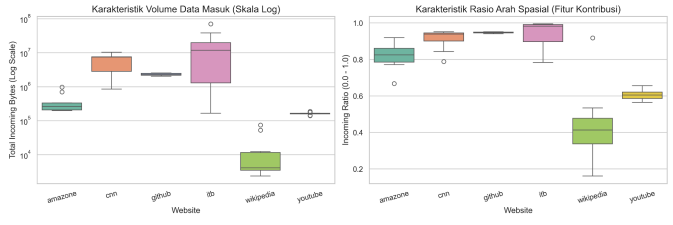
A. *Hasil Ekstraksi dan Karakteristik Dataset*

Tahap ekstraksi fitur menghasilkan sebuah *feature dataset* dalam berkas features.csv yang digunakan sebagai masukan pada proses pemodelan *machine learning*. Dataset tersebut memiliki dimensi **60 × 7** yang terdiri atas lima fitur numerik, yaitu total_packets, total_incoming_bytes, total_outgoing_bytes, incoming_ratio, dan avg_packet_size, serta label kelas (*class_label*) berdasarkan *website*. Selanjutnya dilakukan analisis deskriptif untuk mengidentifikasi karakteristik statistik lalu lintas yang dihasilkan oleh masing-masing situs web.

TABEL III. Profil Statistik Dataset Berdasarkan Label Kelas

Website	Total Incoming Bytes		Total Outgoing Bytes		Mean Incoming Ratio	Mean Avg Packet Size
	Mean	Max	Mean	Max		
Amazone	369.931,7	981.017	69.359,5	166.685	0,821494	891,993
CNN	6.091.393,5	10.414.864	396.905,7	538.819	0,912824	1.199,319
GitHub	2.322.546,7	2.541.210	129.646,9	153.438	0,947128	1.070,932
ITB	17.518.582,8	70.345.519	162.523,7	383.687	0,936223	1.033,965
Wikipedia	16.967,4	74.545	33.454,7	272.560	0,440338	341,346
YouTube	161.724,4	188.616	105.324,1	127.593	0,605777	776,190

Setiap situs web menghasilkan karakteristik lalu lintas yang berbeda secara signifikan. ITB memiliki rata-rata total_incoming_bytes terbesar, yaitu sekitar 17,52 MB, diikuti CNN sebesar 6,09 MB dan GitHub sebesar 2,32 MB. Sebaliknya, Wikipedia menghasilkan rata-rata total_incoming_bytes terkecil, yaitu sekitar 16,97 KB. Selain itu, variasi pada nilai incoming_ratio dan avg_packet_size menunjukkan bahwa setiap situs memiliki karakteristik lalu lintas yang berbeda selama proses pemuatan halaman



Gambar. 13. Analisis Metadata Trafik Antar Kelas Website

Berdasarkan Gambar 13, ITB dan CNN menghasilkan volume data masuk yang lebih tinggi dan bervariasi karena memuat lebih banyak sumber daya, seperti dokumen HTML, gambar, *stylesheet*, dan berkas JavaScript. Sebaliknya, Wikipedia menghasilkan volume data yang relatif kecil dan lebih konsisten karena struktur halamannya cenderung lebih sederhana.

Perbedaan karakteristik juga terlihat pada fitur incoming_ratio. GitHub dan ITB memiliki rasio data masuk di atas 0,93, yang menunjukkan dominasi transfer data dari server ke klien selama proses pemuatan halaman. Sebaliknya, Wikipedia memiliki rasio sekitar 0,44, sehingga proporsi lalu lintas masuk dan keluarnya relatif lebih seimbang. Selain itu, CNN, GitHub, dan ITB memiliki rata-rata ukuran paket di atas 1 KB per paket, sedangkan Wikipedia hanya sekitar 341 byte per paket. Kondisi ini menunjukkan bahwa ukuran *payload* dan pola lalu lintas setiap situs menghasilkan karakteristik statistik yang berbeda.

Secara keseluruhan, setiap situs menghasilkan pola metadata yang cukup khas untuk dibedakan. Meskipun isi komunikasi telah dilindungi oleh *Onion Routing* dan TLS, metadata seperti volume data, ukuran paket, dan arah aliran trafik tetap dapat diamati karena tidak termasuk bagian *payload* yang dienkripsi. Kondisi ini menunjukkan bahwa metadata lalu lintas masih berpotensi dimanfaatkan dalam serangan *Website Fingerprinting* dan digunakan sebagai masukan model *machine learning*.

B. Hasil Pemodelan dan Evaluasi Machine Learning

Tahap pemodelan dilakukan menggunakan algoritma Decision Tree dan Random Forest untuk mengklasifikasikan metadata lalu lintas jaringan hasil ekstraksi fitur serta mengevaluasi tingkat keteridentifikasian situs web yang diakses melalui Tor Browser. Dataset hasil ekstraksi kemudian digunakan pada proses pelatihan dan pengujian model guna mempelajari pola statistik yang membedakan karakteristik lalu lintas setiap situs web. Untuk memperoleh konfigurasi yang optimal, dilakukan *hyperparameter tuning* menggunakan *GridSearchCV* dan *5-Fold Stratified Cross-Validation* guna meningkatkan kemampuan generalisasi dan stabilitas model.

TABEL IV. PARAMETER OPTIMAL HASIL GRIDSEARCHCV

Algoritma	Parameter Terbaik
Decision Tree	criterion = gini, max_depth = 5, min_samples_leaf = 1, min_samples_split = 2
Random Forest	max_depth = 5, min_samples_leaf = 1, min_samples_split = 2, n_estimators = 30

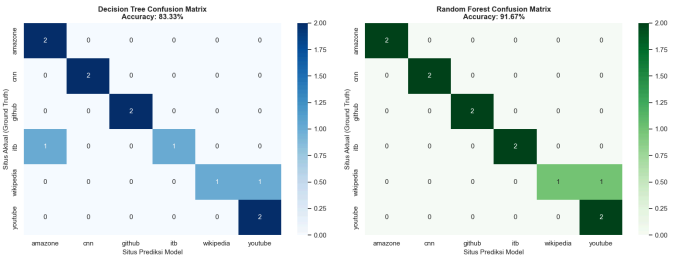
Algoritma Decision Tree memperoleh konfigurasi terbaik menggunakan kriteria pemisahan (*criterion*) berupa *Gini Impurity* dengan kedalaman maksimum pohon sebesar lima tingkat. Sementara itu, Random Forest menghasilkan konfigurasi optimal dengan membangun 30 pohon keputusan (*n_estimators* = 30) dan kedalaman maksimum yang sama. Konfigurasi tersebut menunjukkan bahwa kompleksitas model yang relatif moderat sudah mampu menangkap karakteristik metadata lalu lintas pada jaringan Tor tanpa memerlukan struktur pohon yang terlalu kompleks.

Kualitas model selanjutnya dievaluasi menggunakan *weighted F1-score* selama proses *cross-validation*. Decision Tree memperoleh nilai *weighted* F1-score sebesar 0,7985, sedangkan Random Forest menghasilkan nilai yang lebih tinggi, yaitu 0,8870. Hasil ini menunjukkan bahwa pendekatan *ensemble learning* pada Random Forest menghasilkan model yang lebih stabil dan memiliki kemampuan generalisasi yang lebih baik dibandingkan penggunaan satu Decision Tree.

TABEL V. PERBANDINGAN KINERJA MODEL

Model	Accuracy	Weighted F1-Score
Decision Tree	83,33%	0,8222
Random Forest	91,67%	0,9111

Random Forest menghasilkan akurasi sebesar 91,67% dengan *weighted F1-score* sebesar 0,9111, lebih tinggi dibandingkan Decision Tree yang memperoleh akurasi 83,33% dan *weighted F1-score* sebesar 0,8222. Hasil tersebut menunjukkan bahwa penggabungan beberapa pohon keputusan pada Random Forest menghasilkan model yang lebih stabil dan memiliki kemampuan generalisasi yang lebih baik dalam mengenali pola metadata lalu lintas jaringan Tor.



Gambar. 14. Perbandingan Confusion Matrix Decision Tree dan Random Forest

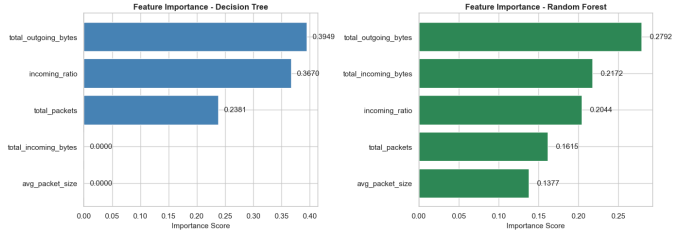
Berdasarkan Gambar 14, Decision Tree menghasilkan dua kesalahan klasifikasi, yaitu sampel ITB yang diprediksi sebagai Amazon dan sampel Wikipedia yang diprediksi sebagai YouTube. Sebaliknya, Random Forest hanya menghasilkan satu kesalahan klasifikasi, yaitu sampel Wikipedia yang diprediksi sebagai YouTube. Berkurangnya jumlah misklasifikasi menunjukkan bahwa pendekatan *ensemble learning* pada Random Forest lebih efektif dalam membedakan pola metadata antar situs web.

Hasil *classification report* menunjukkan bahwa Random Forest memperoleh nilai *precision*, *recall*, dan *F1-score* sempurna pada kelas Amazon, CNN, GitHub, dan ITB. Penurunan kinerja hanya terjadi pada kelas Wikipedia dan YouTube, yang mengindikasikan bahwa kedua situs memiliki karakteristik metadata yang relatif mirip sehingga lebih sulit dibedakan oleh model.

Secara keseluruhan, hasil evaluasi menunjukkan bahwa metadata lalu lintas jaringan Tor masih mengandung pola statistik yang konsisten untuk dipelajari oleh algoritma *machine learning*. Akurasi Random Forest yang mencapai lebih dari 90% menunjukkan bahwa karakteristik seperti volume data, jumlah paket, dan arah aliran trafik masih dapat dimanfaatkan untuk mengidentifikasi situs web yang diakses pengguna. Temuan ini menunjukkan bahwa metadata lalu lintas Tor masih berpotensi dimanfaatkan dalam serangan *Website Fingerprinting* meskipun isi komunikasi telah dilindungi oleh mekanisme kriptografi dan TLS.

C. *Analisis Resiliensi Tor Browser terhadap Serangan Website Fingerprinting*

Dalam penelitian ini, resiliensi Tor Browser diinterpretasikan berdasarkan tingkat keberhasilan *Website Fingerprinting* dalam mengidentifikasi situs web melalui metadata lalu lintas yang masih dapat diamati. Random Forest memperoleh akurasi 91,67% dan weighted F1-score 0,9111, lebih tinggi dibandingkan Decision Tree yang mencapai akurasi 83,33% dan weighted F1-score 0,8222. Hasil tersebut menunjukkan bahwa metadata lalu lintas Tor masih mengandung pola statistik yang dapat dipelajari oleh algoritma *machine learning*. Dengan demikian, meskipun Tor Browser menerapkan Onion Routing dan TLS, karakteristik eksternal lalu lintas jaringan masih dapat dimanfaatkan untuk mengidentifikasi situs web yang diakses pengguna.



Gambar. 15. Tingkat Kepentingan Fitur pada Decision Tree dan Random Forest

Fitur total_outgoing_bytes merupakan fitur paling dominan pada kedua model. Pada Decision Tree, nilai *importance* fitur tersebut sebesar 0,3949, sedangkan pada Random Forest sebesar 0,2792. Selain itu, fitur incoming_ratio, total_incoming_bytes, total_packets, dan avg_packet_size juga berkontribusi dalam proses klasifikasi. Temuan ini menunjukkan bahwa volume data, jumlah paket, dan arah aliran trafik masih membentuk pola yang berbeda pada setiap situs meskipun isi komunikasinya telah dienkripsi.

Dominasi fitur tersebut mengindikasikan bahwa serangan *Website Fingerprinting* tidak bergantung pada dekripsi *payload*, melainkan memanfaatkan metadata yang masih dapat diamati, seperti volume transfer data, jumlah paket, dan arah aliran trafik. Setiap situs menghasilkan pola pemuatan sumber daya yang berbeda, seperti dokumen HTML, gambar, berkas JavaScript, dan objek multimedia, sehingga membentuk sidik jari (*fingerprint*) yang khas pada metadata lalu lintas jaringan.

Hasil analisis juga menunjukkan bahwa mekanisme kriptografi dan TLS pada Tor Browser tetap menjalankan fungsi utamanya dalam menjaga kerahasiaan, integritas, dan autentikasi isi komunikasi sehingga *payload* tidak dapat dibaca oleh pihak ketiga. Namun, mekanisme tersebut tidak menyembunyikan seluruh karakteristik statistik lalu lintas jaringan. Akibatnya, informasi mengenai volume data, arah aliran paket, dan jumlah paket masih dapat diamati tanpa perlu mendekripsi komunikasi.

Temuan ini menunjukkan bahwa resiliensi Tor Browser terhadap serangan *Website Fingerprinting* masih terbatas. Akurasi klasifikasi yang melebihi 90% mengindikasikan bahwa metadata lalu lintas masih cukup informatif untuk membedakan pola akses antar situs web. Dengan demikian, meskipun Onion Routing dan TLS melindungi identitas pengguna dan isi komunikasi, anonimitas pengguna belum sepenuhnya terlindungi dari analisis trafik berbasis *machine learning*. Hasil ini juga perlu diinterpretasikan dengan mempertimbangkan keterbatasan jumlah sampel dan variasi website.

Repository Kode Sumber

https://github.com/NakeishaValya/tor-wf-analyzer

Tautan Video Demonstrasi

https://youtu.be/dvFEQTDnSiY

Ucapan Terima Kasih (Acknowledgment)

Penulis mengucapkan terima kasih kepada Bapak Rinaldi Munir selaku dosen pengampu mata kuliah Kriptografi atas materi, arahan, dan wawasan yang diberikan selama proses pembelajaran dan pengerjaan penelitian ini, serta kepada Kak Diero Arga Purnama dan Kak Ahmad Naufal Ramadan atas arahan dan pendampingan dalam pelaksanaan penugasan. Penulis juga menyampaikan apresiasi kepada pengembang *Tor Project*, *Wireshark*, dan komunitas *open-source* yang telah menyediakan perangkat lunak, dokumentasi, serta berbagai sumber referensi yang mendukung proses implementasi dan penyelesaian penelitian mengenai resiliensi Tor Browser terhadap serangan *Website Fingerprinting* berbasis *machine learning* ini.

Referensi

[1] Jansen, R., Wails, R., & Johnson, A. (2024). *A Measurement of Genuine Tor Traces for Realistic Website Fingerprinting*. arXiv preprint arXiv:2404.07892. https://arxiv.org/pdf/2404.07892

[2] Rescorla, E. (2018). *The Transport Layer Security (TLS) Protocol Version 1.3*. RFC 8446. Tautan: https://www.rfc-editor.org/info/rfc8446/

[3] Basyoni, L., Fetais, N., Erbad, A., Mohamed, A., & Guizani, M. (2020). *Traffic Analysis Attacks on Tor: A Survey*. In *Proceedings of the 2020 IEEE International Conference on Informatics, IoT, and Enablers (ICIIET)* (pp. 183-188). IEEE. https://css.csail.mit.edu/6.858/2023/readings/tor-traffic-analysis.pdf

[4] Rimmer, V., Preuveneers, D., Juarez, M., Van Goethem, T., & Joosen, W. (2018). *Automated Website Fingerprinting through Deep Learning*. In *Proceedings of the Network and Distributed System Security Symposium (NDSS) 2018*. https://www.ndss-symposium.org/wp-content/uploads/2018/02/ndss2018_03A-1_Rimmer_paper.pdf

[5] Ramadhan, R. F., & Ashari, W. M. (2024). *Performance Comparison of Random Forest and Decision Tree Algorithms for Anomaly Detection in Networks*. Journal of Applied Informatics and Computing (JAIC), 8(2), 367-375. https://garuda.kemdiktisaintek.go.id/documents/detail/4806979

[6] The Tor Project. *Tor Relay Guide: Join the Tor Network and Help Protect Privacy Online*. https://community.torproject.org/relay/

[7] Reed, M. G., Syverson, P. F., & Goldschlag, D. M. (1998). *Anonymous Connections and Onion Routing*. IEEE Journal on Selected Areas in Communications, 16(4), 482-494. https://www.onion-router.net/Publications/JSAC-1998.pdf

[8] The Tor Project. *Tor Protocol Specification (tor-spec)*. https://spec.torproject.org/tor-spec/

[9] Wang, T. (2015). *Website Fingerprinting: Attacks and Defenses* (Doctoral dissertation, University of Waterloo). https://dspacemainprd01.lib.uwaterloo.ca/server/api/core/bitstreams/e003fce9-c868-465d-8876-c31531614249/content

[10] Cherubin, G., Jansen, R., & Troncoso, C. (2022). *Online Website Fingerprinting: Evaluating Website Fingerprinting Attacks on Tor in the Real World*. 31st USENIX Security Symposium. https://www.usenix.org/system/files/sec22-cherubin.pdf

[11] Salzberg, S. L. (1994). *Book Review: C4.5: Programs for Machine Learning by J. Ross Quinlan*. Machine Learning, 16(3), 235-240. https://scispace.com/pdf/book-review-c4-5-programs-for-machine-learning-by-j-ross-20r79kvcpo.pdf

[12] Tan, P. N., Steinbach, M., Karpatne, A., & Kumar, V. (2018). *Introduction to Data Mining* (2nd Global Edition). Pearson. https://api.pageplace.de/preview/DT0400.9780273775324_A37747616/preview-9780273775324_A37747616.pdf

[13] Breiman, L. (2001). *Random Forests*. Machine Learning, 45(1), 5-32. Kluwer Academic Publishers. https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 19 Juni 2026

Nakeisha Valya Shakila 18223133