BAB II

LANDASAN TEORI

2. 1 *Generative Adversarial Networks (GAN)*

Generative Adversarial Networks (GAN) adalah sebuah *framework* pembelajaran mesin yang diperkenalkan oleh Ian Goodfellow dan rekan-rekannya pada tahun 2014. *GAN* dirancang untuk menghasilkan data sintetis yang menyerupai data asli dengan cara melibatkan dua jaringan saraf yang saling berkompetisi, yaitu *generator* dan *discriminator* (Goodfellow et al., 2020).

GAN bekerja berdasarkan prinsip *game theory* berupa permainan dua pemain (*minimax game*). *Generator G* bertugas menghasilkan data palsu dari *input noise z*, sedangkan *discriminator D* berfungsi membedakan apakah data yang diterimanya berasal dari data asli atau dari *generator*. Kedua jaringan ini dilatih secara bersamaan dengan tujuan yang berlawanan: *Generator* berusaha memproduksi data yang semakin realistis agar dapat menipu *discriminator*. *Discriminator* berusaha semakin akurat dalam membedakan data asli dan data palsu. Secara matematis, fungsi objektif *GAN* dirumuskan pada persamaan (1).

$$\min_D \max V(D,G) = \mathrm{E}_{x\sim p\mathrm{data}(x)}[\log D(x)] + \mathrm{E}_{z\sim pz(z)}[\log(1-D(G(z)))] \quad (1)$$

Dimana persamaan (1) p data (x) adalah distribusi data asli, $pz(z)$ adalah distribusi *noise* input ke *generator*, $D(x)$ adalah probabilitas bahwa x berasal dari data asli, $G(z)$ adalah data sintetis yang dihasilkan *generator* dari *noise* z (Goodfellow et al., 2020).

Generator berusaha meminimalkan fungsi tersebut dengan menghasilkan data yang dapat menipu *discriminator*, sedangkan *discriminator* berusaha memaksimalkan fungsi tersebut dengan meningkatkan akurasi dalam membedakan data asli dan palsu. Pelatihan *GAN* berlangsung secara iteratif dengan pembaruan parameter kedua model menggunakan algoritma *backpropagation*.

Pada kondisi optimal, discriminator tidak dapat membedakan antara data asli dan data sintetis, sehingga nilai D(x) = 1/2 untuk setiap x*, dan distribusi data yang dihasilkan oleh generator* p_g mendekati distribusi data asli p_data.. Fungsi objektif *GAN* pada kondisi ini terkait dengan minimisasi *Jensen-Shannon divergence* antara kedua distribusi tersebut (Goodfellow dkk., 2020).

2. 2 *Conditional Tabular Generative Adversarial Networks (CT-GAN)*

Conditional Tabular Generative Adversarial Networks (CTGAN) adalah model *deep learning* berbasis *GAN* yang dirancang khusus untuk menghasilkan data sintetis dalam bentuk tabel yang mengandung fitur campuran, yaitu fitur kontinu (numerik) dan kategorikal (diskrit)(Xu dkk., 2019). Model ini dikembangkan oleh MIT Data to AI (DAI) Lab untuk mengatasi tantangan dalam sintesis data tabular yang kompleks, seperti distribusi data yang tidak seimbang dan korelasi antar fitur yang rumit . *CTGAN* memperkenalkan beberapa teknik utamanya atau cara kerjanya, yaitu(Xu et al., 2019):

1. *Conditional GAN*

CTGAN menggunakan *generator* kondisional yang memungkinkan proses generasi data disyaratkan pada nilai tertentu dari fitur kategorikal. Dengan demikian

model dapat secara eksplisit mengontrol distribusi kategori yang dihasilkan, termasuk kategori minoritas, sehingga meningkatkan representasi kelas langka dalam data sintetis.

2. *Mode-Specific Normalization*

Untuk fitur kontinu, *CTGAN* menerapkan normalisasi yang spesifik terhadap mode distribusi lokal, bukan normalisasi global. Hal ini membantu model menangkap distribusi multimodal dari data kontinu yang kompleks.

3. *Training-by-Sampling*

Teknik sampling khusus digunakan selama pelatihan untuk menyeimbangkan frekuensi kategori yang jarang dan umum, sehingga *generator* mendapatkan pelatihan yang seimbang dan tidak bias terhadap kelas mayoritas.

4. *Log-Likelihood Loss* untuk Variabel Kontinu

CTGAN mengoptimalkan *log-likelihood* nilai nyata di bawah distribusi keluaran *generator*, yang membantu menghasilkan nilai kontinu yang lebih realistis dan mengurangi *noise* pada *CTGAN*, *generator* dan *discriminator* dilatih dengan kondisi pada variabel kategorikal c*c*, sehingga fungsi objektifnya yaitu pada persamaan (2).

$$\min_D \max V(D,G)=\mathrm{E}_{x\sim p\mathrm{data}(x)}[\log D(x)]+\mathrm{E}_{z\sim pz(z)}[\log(1-D(G(z|c)|c))] \quad (2)$$

Dimana persamaan (2) artinya data asli *x* dan data sintetis $G(z|c)$ dihasilkan dengan kondisi tertentu *c* (misalnya kategori fitur tertentu). *Discriminator D* juga menerima kondisi *c* sebagai input untuk membedakan data asli dan palsu dalam konteks kondisi tersebut.

CTGAN banyak digunakan untuk augmentasi data dalam berbagai domain, termasuk keamanan siber, kesehatan, dan riset sosial, di mana data asli sulit diakses atau memiliki masalah privasi. Dalam konteks keamanan siber, *CTGAN* dapat digunakan untuk mensintesis *dataset header* paket jaringan yang mengandung fitur tabular kompleks, sehingga mendukung pengembangan model deteksi serangan yang lebih akurat.

2. 3 *Conditional Wasserstein Generative Adversarial Networks (CW-GAN)*

Conditional Wasserstein Generative Adversarial Networks (CWGAN) adalah varian dari GAN yang menggabungkan dua konsep penting, yaitu *Conditional GAN (CGAN)* dan *Wasserstein GAN (WGAN),* untuk menghasilkan data sintetis yang berkualitas tinggi dengan stabilitas pelatihan yang lebih baik serta kemampuan kontrol terhadap kelas data yang dihasilkan (Fabbri, 2019). *CWGAN* banyak digunakan dalam augmentasi data untuk domain dengan data minoritas atau kelas langka, seperti dalam pengenalan citra medis, sintesis data iris, dan keamanan siber untuk menghasilkan *query* serangan *SQL Injection* sintetis yang valid dan bervariasi.

Conditional GAN (CGAN) memperkenalkan informasi kondisi (label atau atribut) ke dalam proses generasi dan diskriminasi data. Informasi kondisi ini, biasanya berupa label kelas y, disisipkan sebagai input tambahan ke *generator* dan *discriminator* sehingga model dapat menghasilkan data yang dikondisikan pada kelas tertentu (misalnya jenis serangan atau normal). Dengan demikian, *GAN* memungkinkan kontrol yang lebih terarah atas keluaran data sintetis(Fabbri, 2019).

Wasserstein GAN (WGAN) menggunakan fungsi *loss* berbasis jarak *Wasserstein (Earth Mover's Distance*) alih-alih *divergence Jensen-Shannon* pada *GAN* konvensional. Pendekatan ini meningkatkan stabilitas pelatihan, mengurangi masalah mode *collapse*, dan memberikan kurva loss yang lebih bermakna untuk *debugging* dan *tuning hyperparameter* (Arjovsky dkk., 2017).

Untuk memastikan pelatihan yang stabil dan memenuhi syarat *Lipschitz continuity* pada *discriminator* (kritikus), *CWGAN* menerapkan *Gradient Penalty (GP)*, yaitu penalti terhadap gradien *discriminator* agar normanya mendekati 1. Teknik ini menghindari pelanggaran *Lipschitz constraint* yang sering terjadi pada *WGAN* standar, sehingga memperbaiki konvergensi dan kualitas data sintetis (LI dkk., 2021). Fungsi objektif *CWGAN* mengoptimalkan nilai ada pada persamaan (3).

$$\min G \max D \; E_{x \sim p_{data}(x|y)}[D(x|y)] - E_{z \sim p_z(z)}[D(G(z|y)|y)] + \lambda \cdot \text{Gradient Penalty} \quad (3)$$

Dimana persamaan (3) x adalah data asli dengan kondisi y (label), $G(z|y)$ adalah data sintetis yang dihasilkan *generator* dari *noise* z dengan kondisi y, $D(\cdot|y)$ adalah *output discriminator* yang juga menerima kondisi y, λ adalah bobot penalti gradien.

2. 4 *SQL Injection*

SQL Injection (SQLi) adalah salah satu jenis serangan siber yang paling umum dan berbahaya terhadap aplikasi web berbasis *database.* Serangan ini terjadi ketika penyerang menyisipkan perintah *SQL* berbahaya ke dalam *query* aplikasi melalui input yang tidak tervalidasi dengan baik, sehingga dapat mengubah perilaku *query* asli yang dimaksudkan oleh aplikasi(Sitharthan S & Sankaran S, 2014). Akibatnya, penyerang

dapat melakukan operasi tidak sah seperti membaca, memodifikasi, atau menghapus data dalam database, bahkan menjalankan perintah administratif pada sistem *database* (OWASP, 2024).

Serangan *SQL injection* memanfaatkan kerentanan di web bidang input aplikasi untuk memanipulasi database aplikasi secara langsung. Dengan menyuntikkan kode *SQL* berbahaya, penyerang dapat memperoleh akses tidak sah ke *database*, mengambil informasi sensitif, mengubah data, atau bahkan menghapus keseluruhannya basis data. Hal ini memungkinkan penyerang untuk mengakses data sensitif, mengubah data, atau bahkan mengambil alih kendali sistem (Bakır, 2025).

Menurut OWASP, serangan SQL Injection dapat menimbulkan berbagai konsekuensi serius pada sistem *database*. Dari sisi kerahasiaan (*confidentiality)*, serangan ini dapat menyebabkan kebocoran data sensitif yang tersimpan dalam database. Pada aspek otentikasi *(authentication)*, penyerang berpotensi masuk sebagai pengguna lain tanpa perlu mengetahui kata sandi yang sah. Dari segi otorisasi *(authorization)*, SQL Injection dapat dimanfaatkan untuk mengubah hak akses maupun data otorisasi yang tersimpan dalam sistem. Sementara itu, pada aspek integritas *(integrity)*, penyerang dapat melakukan perubahan atau bahkan penghapusan data secara tidak sah, sehingga mengganggu keandalan dan validitas informasi dalam database OWASP (2024).

Jenis-jenis *SQL Injection* dapat diklasifikasikan berdasarkan mekanisme injeksi dan tujuan serangan, seperti injeksi melalui input pengguna, variabel *server*, atau *stored procedures*. Salah satu jenis yang sering terjadi adalah *Second-Order SQL*

Injection, yaitu serangan yang melibatkan penyimpanan input berbahaya di *database* yang kemudian dieksekusi pada waktu lain(Salih Abdullah & Mohsin Abdulazeez, 2024).

Serangan ini sangat umum terjadi karena banyak aplikasi web yang mengandalkan *database* dan seringkali tidak menerapkan validasi input yang ketat. Studi menunjukkan bahwa rata-rata sebuah aplikasi web dapat mengalami puluhan hingga ribuan percobaan serangan *SQL Injection* dalam waktu singkat(Sitharthan S & Sankaran S, 2014).

2. 5 *IBM X-Force Threat Intelligence Indeks*

IBM X-Force Threat Intelligence Index adalah laporan tahunan yang dirilis oleh *IBM Security* untuk memberikan wawasan mendalam mengenai tren, pola, dan taktik serangan siber global berdasarkan pemantauan lebih dari 150 miliar peristiwa keamanan setiap hari di lebih dari 130 negara. Laporan ini menjadi salah satu acuan utama bagi para profesional keamanan siber dalam memahami lanskap ancaman yang terus berkembang dan membantu organisasi memperkuat pertahanan mereka terhadap serangan siber.

Laporan *IBM X-Force Threat Intelligence Index* 2025 menyoroti bahwa serangan terhadap aplikasi publik-aplikasi yang dapat diakses secara langsung dari *internet* seperti portal web, layanan *cloud*, dan sistem login menjadi salah satu vektor serangan utama yang dimanfaatkan pelaku kejahatan siber sepanjang tahun 2024 hingga awal

2025. Berikut adalah temuan-temuan utama terkait serangan publik berdasarkan laporan tersebut(IBM, 2025):

1. Eksploitasi Kerentanan pada Aplikasi Publik

30% dari insiden yang ditangani *X-Force* pada 2024 melibatkan eksploitasi aplikasi publik (*public-facing applications*). Banyak organisasi menghadapi tantangan dalam manajemen *patch*, sehingga aplikasi yang terhubung ke internet menjadi target empuk bagi penyerang yang memanfaatkan kerentanan yang belum ditambal. Setelah berhasil mendapatkan akses awal, penyerang sering melakukan *active scanning* menggunakan alat otomatis untuk menemukan kerentanan tambahan dan memperluas akses di lingkungan korban.

2. Lonjakan dan Penyebaran Kerentanan

Sejak 1993, *IBM X-Force* telah mencatat lebih dari 300.000 kerentanan unik, dengan hampir 65.000 di antaranya memiliki *exploit* publik yang siap digunakan penyerang. Artinya, hampir seperempat dari seluruh kerentanan yang tercatat sudah memiliki *exploit* yang dapat langsung dipakai. Jumlah kerentanan yang ditemukan meningkat tiga kali lipat dalam delapan tahun terakhir, didorong oleh adopsi *cloud* dan layanan infrastruktur bersama, sehingga aplikasi publik semakin rentan dieksploitasi pada skala besar.

3. Pencurian Kredensial dan Penyalahgunaan Identitas

Pencurian kredensial menjadi vektor serangan publik paling dominan. Hampir satu dari tiga insiden pada 2024 berujung pada pencurian kredensial, didorong oleh lonjakan 84% *email phishing* yang mengirimkan *infostealer* di tahun tersebut.

Penyerang menggunakan *infostealer* untuk mencuri identitas pengguna yang kemudian diperjual belikan di *dark* web.

4. Serangan Berbasis Otomasi dan Bot

Penyerang memanfaatkan bot dan alat otomatis yang tersedia di *dark* web untuk menargetkan aplikasi dan layanan publik organisasi. Dengan alat ini, mereka dapat melakukan serangan secara masif dan sistematis, termasuk *brute force login*, *scanning* kerentanan, dan eksploitasi otomatis.

2. 6 *Exploit Public-Facing Applications*

Menurut *MITRE ATT&CK* Eksploitasi aplikasi publik (*Exploit Public-Facing Application*) adalah teknik yang digunakan oleh pelaku ancaman untuk mendapatkan akses awal ke jaringan dengan memanfaatkan kelemahan pada sistem atau aplikasi yang dapat diakses secara langsung melalui internet (*public-facing*). Teknik ini dikategorikan dalam taktik *Initial Access* pada *MITRE ATT&CK Framework* dengan kode teknik T1190**.**

Eksploitasi aplikasi publik menjadi vektor serangan utama yang memungkinkan pelaku kejahatan siber masuk ke dalam sistem tanpa harus melewati lapisan keamanan yang lebih dalam, terutama di lingkungan *cloud* dan infrastruktur bersama yang semakin kompleks. Selain itu, serangan ini sering dikombinasikan dengan pencurian kredensial yang meningkat drastis, yang juga menjadi metode utama untuk mendapatkan akses awal.

Temuan *IBM X-Force Threat Intelligence Index* 2025 menguatkan pentingnya teknik ini sebagai vektor serangan utama, dengan sekitar 30% insiden yang ditangani melibatkan eksploitasi aplikasi publik. Eksploitasi ini menjadi pintu masuk bagi banyak serangan lanjutan, terutama pada organisasi dengan aplikasi *internet-facing* yang rentan(IBM, 2025).

2. 7 *SQL Injection Attack Netflow Datasets*

Dataset ini berisi data *NetFlow* yang merekam lalu lintas jaringan yang mengandung serangan *SQL Injection (SQLI)* sebagai data berbahaya (*malicious traffic*). Serangan yang dilakukan meliputi dua jenis utama, yaitu *Union Query SQL Injection* dan *Blind SQL Injection*. Untuk melancarkan serangan ini, digunakan alat otomatis bernama *SQLMAP*, yang merupakan *tool* populer untuk mendeteksi dan mengeksploitasi kerentanan *SQL Injection*.

NetFlow adalah protokol jaringan yang dikembangkan oleh *Cisco* untuk mengumpulkan dan memonitor data aliran lalu lintas jaringan (*network traffic flow*). Dalam konteks *dataset* ini, *NetFlow* versi 5 digunakan, yang mendefinisikan sebuah *flow* sebagai rangkaian paket satu arah yang melewati perangkat jaringan dengan properti umum.

Data *NetFlow* dikumpulkan menggunakan *DOROTHEA* (*Docker-based framework for gathering netflow traffic*), sebuah *framework* berbasis *Docker* yang memungkinkan pembuatan jaringan virtual terhubung untuk menghasilkan dan mengumpulkan data *flow* menggunakan protokol *NetFlow*. *DOROTHEA*

menggunakan modul kernel *Linux* dengan *iptables* untuk memproses paket dan mengubahnya menjadi *flow NetFlow*.

Dataset terdiri dari dua bagian utama (Ignacio & Campazas, 2022):

1. D1 (*Training Dataset*): Berisi 400.003 sampel dengan rasio trafik benign dan malicious seimbang 50:50. *Dataset* ini digunakan untuk melatih model deteksi.

2. D2 (*Testing Dataset*): Berisi 57.239 sampel dengan rasio yang sama, digunakan untuk menguji dan menggeneralisasi model. *Dataset* ini berisi jenis serangan yang berbeda dari D1.

Serangan *SQL Injection* dilakukan pada 16 *node* penyerang yang meluncurkan *SQLMAP* ke 200 node korban. *Node* korban menjalankan layanan web dengan *form* yang rentan terhadap serangan *Union-type SQL Injection*, terhubung ke *database MySQL* dan *SQL Server* (masing-masing 50%). Pada D2, serangan *Blind SQL Injection* dilakukan terhadap *form* web yang terhubung ke *database PostgreSQL*, dengan alamat IP yang berbeda dari D1. *Server database* yang digunakan adalah *MariaDB 10.4.12* untuk *MySQL*, *Microsoft SQL Server 2017 Express*, dan *PostgreSQL 13*. Layanan web berjalan pada *port* standar 80 dan 443.

Trafik *benign* mensimulasikan aktivitas pengguna nyata seperti *browsing* web, mengirim *email*, dan koneksi *SSH*, yang dikontrol melalui skrip *Python*. Sedangkan trafik *malicious* dihasilkan dari serangan *SQL Injection* yang otomatis dijalankan oleh *SQLMAP* dengan parameter konfigurasi seperti enumerasi pengguna, *database*, tabel, dan lain-lain.

2. 8 *SQL Injection Query Dataset*

SQL Injection Dataset yang dipublikasikan oleh Abu Syeed Sajid Ahmed di *platform Kaggle* dengan *username* sajid576 ini berisi kumpulan *query SQL* yang terdiri dari dua kelas utama, yaitu *query benign* (normal) dan *query* yang merupakan serangan *SQL Injection*. Setiap *query* diberi label untuk membedakan antara permintaan yang valid dan permintaan yang mengandung pola serangan. *Query* berbahaya dalam *dataset* ini mengandung pola umum serangan *SQL Injection* seperti ' OR 1=1 --, UNION SELECT, DROP TABLE, serta teknik manipulasi lainnya yang sering digunakan oleh pelaku serangan siber (Kaggle, 2021).

Dataset ini digunakan oleh Abu Syeed Sajid Ahmed dan rekan-rekannya dalam penelitiannya yang berjudul "*A Hybrid Approach to Detect Injection Attacks on Server-Side Applications Using Data Mining Techniques*". Dalam penelitian tersebut, *dataset SQL Injection* digunakan sebagai sumber data utama untuk melatih dan menguji model deteksi serangan injeksi pada aplikasi sisi *server*. *Dataset* ini berisi ribuan *query SQL* yang telah diberi label sebagai *query* normal (*benign*) dan *query* berbahaya (*malicious*) yang mengandung pola serangan *SQL Injection*.

Penelitian ini mengembangkan pendekatan *hybrid* yang menggabungkan teknik validasi input tradisional dengan metode *machine learning* berbasis data *mining* untuk meningkatkan akurasi deteksi serangan. *Dataset* tersebut diproses dengan langkah-langkah pra-pemrosesan seperti verifikasi label, pembersihan data duplikat, dan pemisahan data menjadi set pelatihan dan pengujian. Dengan menggunakan *dataset* ini,

model yang dikembangkan mampu mengenali pola serangan *SQL Injection* secara efektif, meningkatkan ketahanan aplikasi *server-side* terhadap serangan injeksi.

2. 9 Paket Jaringan

Paket jaringan adalah unit data yang dikirim melalui jaringan komputer dan terdiri dari beberapa bagian utama, yaitu *header* dan *payload.* Dalam konteks serangan *SQL Injection*, pemahaman tentang struktur paket ini penting karena *payload* berisi *query SQL* yang bisa dimanipulasi oleh penyerang, sementara *header* berisi informasi kontrol yang mengatur pengiriman paket tersebut.

1. *Header*

Header paket berisi informasi kontrol penting yang digunakan untuk mengarahkan dan mengelola pengiriman data. Informasi ini meliputi alamat sumber dan tujuan (*IP address* dan *port*), protokol yang digunakan, serta nomor urut paket. *Header* memastikan paket dapat mencapai tujuan dengan benar dan data dapat dirakit ulang secara tepat setelah diterima (ITBOX, 2024).Pada lapisan jaringan dan *transport*, *header* berfungsi untuk menentukan jalur pengiriman dan memastikan keutuhan data, seperti yang dijelaskan pada lapisan *Network* dan *Transport* dalam model OSI (Lie, 2018).

2. *Payload*

Payload adalah bagian dari paket yang membawa data aktual, dalam hal ini *query SQL* yang dikirimkan dari *client* ke *server*. Pada serangan *SQL Injection*, *payload* ini yang disisipkan kode berbahaya untuk mengeksploitasi celah keamanan pada aplikasi

database (Bangun, 2012). *Payload* dapat berupa teks *query SQL* yang dimanipulasi agar *server* menjalankan perintah yang tidak diinginkan.

Dalam proses komunikasi *HTTP* atau protokol lain yang digunakan aplikasi web, *query SQL* yang dikirim sebagai bagian dari *request* akan berada di *payload* paket jaringan. *Intrusion Detection System (IDS)* seperti *Snort* dapat menganalisis paket ini dengan memeriksa baik *header* maupun *payload* untuk mendeteksi pola serangan, termasuk *SQL Injection*. *Snort* menggunakan aturan yang memeriksa konten *payload* dan anomali pada *header* paket untuk mengidentifikasi serangan (Fathoni, 2015).

Sebagai contoh, *payload* yang berisi *query SQL* dengan pola tertentu (misalnya mengandung kata kunci seperti "*union select*" atau tanda kutip yang tidak biasa) dapat menandakan adanya serangan *SQL Injection*. Sementara *header* paket membantu menentukan asal dan tujuan paket tersebut serta memastikan paket tersebut valid dan dalam konteks komunikasi yang benar.

Analisis paket data dari serangan *SQL Injection* menunjukkan informasi detail pada *header* seperti alamat *MAC*, alamat *IP* sumber dan tujuan, *port* sumber dan tujuan, serta protokol yang digunakan (biasanya *TCP*). *Payload* berisi *query SQL* yang dimodifikasi oleh penyerang. Contoh data *header* dan *payload* dapat dilihat dalam penelitian yang menganalisis paket serangan *SQL Injection* pada komunikasi jaringan (Unikom, 2021).

2. 10 *Kolmogorov-Smirnov Test*

Kolmogorov-Smirnov Test adalah metode statistik nonparametrik yang digunakan untuk menguji kesamaan distribusi antara dua sampel data atau antara

sampel data dengan distribusi teoritis tertentu (misalnya distribusi normal) (Leni dkk., 2023). Dalam konteks evaluasi data sintetis, *Kolmogorov-Smirnov Test* membandingkan fungsi distribusi kumulatif (CDF) dari data asli dan data sintetis untuk setiap variabel atau fitur.

Menurut penelitian sebelumnya, *Kolmogorov-Smirnov test* merupakan metode yang digunakan untuk menilai kesamaan distribusi variabel antara data sintetis dan data asli (Leni et al., 2023). Uji ini menghasilkan dua ukuran utama yang saling melengkapi (Nielsen, 2019). Pertama adalah *Kolmogorov-Smirnov Statistic*, yaitu nilai yang mengukur jarak maksimum antara dua fungsi distribusi kumulatif (CDF). Nilai ini berada pada rentang 0 hingga 1, di mana nilai mendekati 0 menunjukkan bahwa kedua distribusi hampir identik, sementara nilai yang lebih tinggi mengindikasikan adanya perbedaan distribusi yang lebih besar. Selanjutnya adalah *p-value*, yang menunjukkan tingkat signifikansi statistik dari perbedaan distribusi tersebut. Apabila nilai p lebih besar dari 0,05, maka tidak terdapat bukti yang cukup untuk menyatakan bahwa distribusi data sintetis berbeda dengan distribusi data asli, sehingga dapat disimpulkan bahwa data sintetis mampu merepresentasikan distribusi data asli dengan baik.

2. 11 *Jensen-Shannon Divergence*

Jensen-Shannon Divergence adalah metrik yang digunakan untuk mengukur kesamaan antara dua distribusi probabilitas. *Jensen-Shannon Divergence* merupakan versi simetris dari *Kullback-Leibler Divergence (KL Divergence)*, sehingga memiliki

beberapa keunggulan penting seperti selalu bernilai terbatas dan simetris, berbeda dengan *KL Divergence* yang bisa tak hingga dan tidak simetris (Nielsen, 2019).

Cara kerja dan definisi *Jensen-Shannon Divergence* yaitu ada pada persamaan (4) dan (5)

1. *Jensen-Shannon Divergence* mengukur jarak antara dua ditribusi probabilitas *P* dan *Q* dengan membandingkan keduanya terhadap distribusi rata-rata

$$M = \frac{1}{2}(P + Q). \quad (4)$$

2. Rumus dasar *Jensen-Shannon Divergence*

$$JSD\ (P||Q) = \frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M) \quad (5)$$

Di mana D_{KL} adalah *Kullback-Leibler Divergence.*

3. Nilai *Jensen-Shannon Divergence* selalu berada di antara 0 dan log 2 (atau 1 jika menggunakan basis logaritma 2), dengan nilai 0 menunjukkan distribusi identik dan nilai lebih tinggi menunjukkan perbedaan yang lebih besar (Nielsen, 2019).

2.12 Penelitian Terdaulu

Tabel 2.1 Penelitian Terdahulu

Penelitian 1	
Judul	*Enhancing SQL Injection Detection and Prevention Using Generative Models*
Peneliti dan Tahun	(Dasari dkk., 2025)
Link	doi.org/10.48550/arXiv.2502.04786
Metode	Model generatif (VAE, CWGAN-GP, U-Net) untuk augmentasi data
Hasil	Meningkatkan akurasi deteksi *SQL Injection* dengan data sintetis, mengurangi *false positive/negative*, dan adaptif terhadap serangan baru
Penelitian 2	
Judul	*FlowGAN - Synthetic Network Flow Generation using Generative Adversarial Networks*
Peneliti dan Tahun	(Manocchio dkk., 2021)
Link	10.1109/CSE53436.2021.00033
Metode	*FlowGAN* dengan *Manifold Guided GAN* (*MGGAN*) untuk mengatasi modal collapse pada data flow jaringan
Hasil	*FlowGAN* mampu menghasilkan data *flow* jaringan sintetis yang lebih realistis dan beragam dibandingkan metode *GAN* sebelumnya, diukur dengan *Wasserstein distance*
Penelitian 3	
Judul	*IDSGAN: Generative Adversarial Networks for Attack Generation against Intrusion Detection*
Peneliti dan Tahun	(Lin dkk., 2022)
Link	https://doi.org/10.48550/arXiv.1809.02077

Metode	*Wasserstein GAN* yang dimodifikasi
Hasil	Penelitian ini mengembangkan *IDSGAN, framework GAN* yang menghasilkan *adversarial malicious traffic* untuk mengecoh sistem deteksi intrusi (IDS). Dengan memanfaatkan *Wasserstein GAN* dan yang di modifikasi terbatas, *IDSGAN* mampu menghasilkan traffic berbahaya yang mempertahankan fungsionalitas serangan asli sekaligus menghindari deteksi.
Penelitian 4	
Judul	*Generative Adversarial Networks for Network Traffic Feature Generation*
Peneliti dan Tahun	(Anande dkk., 2023)
Link	https://doi.org/10.1080/1206212X.2023.2191072
Metode	*GAN* dengan transformasi data untuk menangani atribut kategorikal pada data trafik jaringan.
Hasil	Berhasil menghasilkan fitur trafik jaringan sintetis yang realistis dan beragam, meningkatkan kualitas data untuk pelatihan sistem keamanan jaringan.
Penelitian 5	
Judul	*Enhancing UAV Network Security: A Human-in-the-Loop and GAN-Based Approach to Intrusion Detection*
Peneliti dan Tahun	(Zeng & Farid Nait-Abdesselam, 2025)
Link	10.1109/JIOT.2025.3545389
Metode	*CTGAN* untuk augmentasi data tabular IDS
Hasil	*CTGAN* berhasil menghasilkan data sintetis berkualitas tinggi yang memperbaiki keseimbangan kelas dan meningkatkan performa IDS pada jaringan UAV
Penelitian 6	

Judul	*Synthetic Data Generation in Cybersecurity: A Comparative Analysis*
Peneliti dan Tahun	(Ammara dkk., 2024)
Link	arXiv:2410.16326v1
Metode	*CTGAN* dan *GAN* untuk data tabular
Hasil	*CTGAN* unggul dalam menghasilkan data sintetis dengan fidelitas dan utilitas tinggi dibanding metode lain, khususnya untuk data trafik keamanan siber

Berdasarkan tabel 2.12 menunjukkan efektivitas model generatif dalam bidang keamanan jaringan, meskipun fokus dan cakupannya masih terbatas. Penelitian Dasari berhasil meningkatkan deteksi SQL Injection dengan berbagai model generatif, namun lebih menekankan pada sistem deteksi dan pencegahan, bukan pengembangan dataset lengkap (Dasari et al., 2025).

FlowGAN yang dikembangkan (Manocchio dkk., 2021) mampu menghasilkan aliran jaringan sintetis yang realistis, tetapi tidak diarahkan pada serangan spesifik. Penelitian (Lin dkk., 2022) dengan IDSGAN menekankan pada pembuatan *adversarial traffic* untuk mengecoh IDS, sementara penelitian (Anande dkk., 2023) berfokus pada pembuatan fitur trafik jaringan yang realistis tanpa membangun dataset serangan tertentu.

Disisi lain, (Zeng & Farid Nait-Abdesselam, 2025) serta (Ammara dkk., 2024) menunjukkan keunggulan CTGAN dalam menghasilkan data tabular sintetis dengan kualitas tinggi, namun masih terbatas pada data IDS atau data tabular umum.

Penelitian ini mengisi celah (gap) yang ada pada riset-riset sebelumnya. Penelitian oleh Dasari telah memanfaatkan model generatif untuk deteksi, namun lebih menekankan pada sistem pencegahan daripada pengembangan dataset paket jaringan yang lengkap. Riset lainnya oleh Ammara menggunakan *CTGAN* untuk menghasilkan data trafik yang berkualitas tinggi, tetapi masih terbatas pada data tabular umum dan tidak diarahkan pada jenis serangan spesifik seperti *SQL Injection*. Berbeda dengan penelitian tersebut, riset ini menggabungkan *CTGAN* untuk header tabular dan *CWGAN-GP* untuk payload sekuensial guna menghasilkan dataset serangan *SQL Injection* yang menyeluruh, bervariasi, dan dapat dignakan dalam format PCAP (Dasari et al., 2025) (Ammara et al., 2024).