Bit (Fakultas Teknologi Informasi Universitas Budi Luhur) p-ISSN: 1693-9166
Vol. 21, No. 2, September 2024, hlm. 145-152 e-ISSN: 2685-127X

DETEKSI *WEBSITE PHISHING* DARI ANALISIS URL MENGGUNAKAN ALGORITMA *RANDOM FOREST*

Damar Bambang Suwarno[1*], Mardi Hardjianto [2]

[1*,2] Teknik Informatika, Fakultas Teknologi Informasi, Universitas Budi Luhur, Kota Jakarta Selatan, Indonesia
Email: [11*]sdamar321@gmail.com, [2]mardi.hardjianto@budiluhur.ac.id

(Naskah masuk: 7 Agustus 2024, diterima untuk diterbitkan: 7 September 2024)

Abstrak

Serangan digital, atau yang lebih dikenal sebagai *cybercrime*, semakin sering terjadi di era modern ini, seiring dengan pesatnya perkembangan teknologi. Metode serangan yang paling sering dipakai adalah *phishing*, yang pertama kali muncul pada tahun 1996. *Phishing* merupakan salah satu bentuk kejahatan siber di mana penyerang berusaha mengelabui pengguna agar secara tidak sadar memberikan informasi sensitif seperti *username, password*, atau data keuangan. Berbagai penelitian telah dilakukan untuk mencari solusi dalam menangani serangan *phishing*, mulai dari penggunaan *tools* keamanan seperti menggunakan OS *Kali Linux* hingga mengedukasi para pegawai agar lebih waspada dan menggunakan alat bantu keamanan seperti antivirus. Salah satu solusi yang semakin relevan adalah penggunaan kecerdasan buatan (*Artificial Intelligence*) untuk secara otomatis mengidentifikasi apakah sebuah URL yang diakses aman atau berpotensi berbahaya. Penelitian ini bertujuan untuk mengembangkan aplikasi berbasis web yang mampu mendeteksi serangan *phishing* dengan menggunakan teknik *machine learning*, serta memanfaatkan *dataset* yang cukup besar dan representatif untuk melatih model deteksi URL *phishing*. Dalam penelitian ini kontribusi yang dilakukan ialah menggunakan algoritma *random forest* dalam pendeteksian *website phishing* dan penambahan fitur deteksi yang di integrasikan pada *website* yang membahas *phishing*, algoritma klasifikasi *Random Forest* digunakan karena kemampuannya yang tinggi dalam memproses sejumlah besar fitur deteksi. Dengan menggunakan 30 fitur deteksi, hasil pengujian menunjukkan bahwa sistem yang dibangun mampu mencapai kinerja yang optimal, dengan tingkat prediksi sebesar 96%, *Recall* 92%, Akurasi 94%, dan F-*Score* 93%. Hasil ini menunjukkan bahwa metode yang diusulkan efektif dalam mendeteksi serangan *phishing* dengan tingkat akurasi yang tinggi, menjadikannya alat yang sangat berguna dalam mencegah pengguna dari ancaman siber dan dinilai dapat menyelesaikan permasalahan yang ada karena dapat bekerja dengan optimal.

Kata kunci: *cybercrime, phishing, random forest, machine learning*

PHISHING WEBSITE DETECTION FROM URL ANALYSIS USING RANDOM FOREST ALGORITHM

Abstract

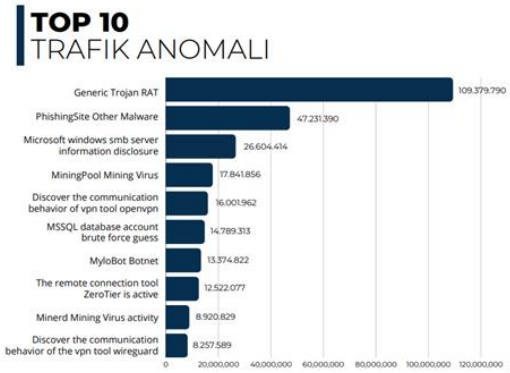
Digital attacks, or what is better known as cybercrime, are increasingly occurring in this modern era, along with the rapid development of technology. The most frequently used attack method is phishing, which first appeared in 1996. Phishing is a form of cybercrime in which attackers attempt to trick users into unknowingly providing sensitive information such as usernames, passwords, or financial data. Various studies have been carried out to find solutions for dealing with phishing attacks, starting from using security tools such as using the Kali Linux OS to educating employees to be more alert and use security tools such as anti-virus. One solution that is increasingly relevant is the use of artificial intelligence to automatically identify whether a URL to be accessed is safe or potentially dangerous. This research aims to develop a web-based application that is capable of detecting phishing attacks using machine learning techniques, as well as utilizing a fairly large and representative dataset to train a phishing URL detection model. In this research, the Random Forest classification algorithm is used because of its high ability to process a large number of detection features. By using 30 detection features, the test results show that the system built is able to achieve optimal performance, with a prediction level of 96%, Recall 92%, Accuracy 94%, and F-Score 93%. These results show that the proposed method is effective in detecting phishing attacks with a high degree of accuracy, making it a very useful tool in protecting users from cyber threats.

Keywords*: cybercrime, phishing, random forest, machine learning*

1. PENDAHULUAN

Internet adalah kepanjangan dari istilah "*Interconnected Network*", yang berarti hubungan komputer dengan berbagai jenis jaringan. Internet pada awalnya adalah proyek yang dirancang untuk kepentingan militer dari suatu negara [1]. Seiring cepatnya perkembangan internet terdapat pula kejahatan dunia maya yang terjadi atau biasa disebut *cybercrime*. *Cybercrime* adalah aktivitas kriminal yang melibatkan penggunaan komputer, perangkat digital, atau jaringan untuk melakukan tindakan ilegal [2]. Terdapat beberapa jenis serangan *cybercrime* yang umum meliputi. *Malware*, *ransomware*, *denial of service* (*DoS*) *attack*, *phishing*, dan lainnya.

Phishing adalah aktivitas kriminal yang menggunakan teknik rekayasa sosial [3]. *Phishing* pertama kali muncul pada tahun 1996. Jumlah anomali *phishing* yang tercatat sebanyak 47.231.390 pada tahun 2023 dari Badan Siber dan Sandi Negara Republik Indonesia Gambar 1.



Gambar 1. Top 10 trafik anomali [5]

Phishing telah menjadi masalah keamanan siber yang semakin mengkhawatirkan. Berbagai pihak telah melakukan upaya untuk mengatasi ancaman ini. Dijelaskan bahwa peneliti telah mengembangkan berbagai teknik deteksi *phishing*. Di antaranya URL di cek menggunakan API *integration* ke *website phistank* dan menggunakan algoritma *XGboost* [4] pada penelitian ini terdapat beberapa kekurangan *dataset* yang digunakan dan juga data yang tidak *real time* pada fitur deteksi dikarenakan tidak terintegrasi oleh *website* terpercaya yang membahas *phishing* juga, lalu lembaga keuangan dan penyedia layanan *online* juga terus meningkatkan keamanan sistem mereka dan mengedukasi pengguna tentang cara mengenali dan menghindari serangan *phishing.*

Kesimpulan yang didapat bahwa *phishing* adalah salah satu kasus *cyber crime* yang paling umum dan sering terjadi akhir-akhir ini. Serangan *phishing* menduduki peringkat dua berdasarkan data Badan Siber dan Sandi Negara Republik Indonesia pada tahun 2023 [5]. Peneliti mengusulkan bahwa detektor *phishing* dianggap dapat menyelesaikan masalah ini. Peneliti menggunakan data set API *public* yaitu *website similar web* dan *open page rank*, penggunaan API digunakan agar data set selalu ter *update* dan juga penambahan *dataset* menjadi 11.055 data *phishing* dan *non-phishing*. Algoritma *random forest* juga dipakai di dalam penelitian kali ini bertujuan agar hasil lebih akurat dari sebelumnya.

2. METODE PENELITIAN

Pada metode penelitian ini peneliti menggunakan beberapa tahapan dalam penelitian di antaranya :

2.1 *Understanding the problem*

Pada tahap ini peneliti mencari berbagai jurnal terkait *phishing*, berbagai macam algoritma yang dipakai di antaranya algoritma KNN, *naive bayes*, *XGBoost* [6], dan lainya. Dalam hal ini peneliti memilih algoritma *random forest* karena dinilai lebih cocok untuk menjadi algoritma dalam penelitian ini. Tahap ini juga tahap yang berfungsi untuk memahami masalah yang ada dan untuk menetapkan tujuan dari penelitian ini dengan masalah yang ada [7]. Keberhasilan dari penelitian ini diambil dari akurasi sistem mendeteksi suatu URL *phishing* dengan di ukur menggunakan *confusion matrix*, semakin tinggi nilai akurasi semakin efektif pula model yang dibuat untuk mendeteksi dan mencegah URL *phishing*.

2.2 *Understanding the* data

Pada tahap ini pencarian data set dilakukan pada *website* phistank.com, openphish.com, similarweb.com, archive.ics.uci.edu, dan lainya yang berhubungan dengan *phishing* [8]. Data dari archive.ics.uci.edu menjadi acuan program peneliti karena data set yang di hasilkan menjadi bahan belajar *machine learning* karena sudah di *extract* menjadi bilangan 1,0,-1 sesuai dengan *feature* deteksi yang digunakan mengeluarkan hasil yang serupa yaitu 1,0,-1, dan melihat isi data.

2.3 Pemodelan Algoritma *Random Forest*

Tahap ini merupakan tahap mengimplementasikan algoritma *random forest* ke dalam penelitian yang dibuat setelah memahami masalah yang ada dan mengumpulkan *dataset* yang ada selanjutnya proses algoritma *random forest* dilakukan yaitu membaca *dataset*, pembagian data, menggunakan klasifikasi *random forest*, dan penggunaan 30 fitur [9] pengetesan URL merupakan salah satu cara untuk mencari ciri dari suatu URL *phishing*, dan juga sebagai penentuan hasil klasifikasi. Pada Tabel 1 menampilkan 30 fitur deteksi dengan *value* dari fitur yang ada dan penjelasannya.

2.3.1 *Having IP Address*

Jika sebuah DNS diganti menjadi sebuah IP *address* ketika di akses seperti HTTP://123.45.6.789/fake.html maka itu bisa dibilang *phishing* dan dapat mencuri informasi pribadi *user* yang mengakses URL tersebut. Selain itu, IP *address* terkadang diubah menjadi

hexadecimal, seperti HTTP://www.0x57.0xAA.0xyT.0x83.ox7t/3/paypal.ca/index.html jika URL berubah alamat IP ketika diakses, nilai -1. Sebaliknya, diberi nilai satu.

Tabel 1. Fitur Deteksi

Parameter	Fitur	Nilai *Feature*
	Have IP *address*	(-1,1)
	Panjang URL	(1,0,-1)
	URL pendek	(1,-1)
	Mempunyai simbol @	(-1,1)
	Menggunakan //	(-1,1)
	Menggunakan simbol -	(-1,1)
	Mempunyai *subdomain*	(-1,0,1)
	Mempunyai sertifikat SSL	(1,0,-1,-1)
	Panjang *Domain*	(-1,1,-1,-1-1)
	Favicon	(1,-1,-1)
	Port	(1,-1,1,1)
	Mempunyai HTTPS	(1,-1)
	Request URL	(1, -1)
	Mempunyai *Anchor*	(1,0,-1,-1)
1 = Valid	*Link* pada *tags*	(-1,1)
0 = *Suspicious*	*SFH*	(1,-1)
-1 = *Phishing*	*Submit email*	(-1,1,-1)
	Link URL mencurigakan	(-1, 1)
	Redirect	(1,-1)
	On mouse over	(1,-1)
	Klik kanan	(1,-1)
	Pop up window	(1,-1)
	I frame	(1,-1)
	Umur *domain*	(1,-1,-1,-1)
	DNS record	(1,-1,-1,-1)
	Web traffic	(1,0,-1)
	Page rank	(1,-1,-1)
	Google indeks	(1,-1,-1)
	Link to page	(1,0,-1,-1)
	Statistic report	(-1,1,-1,-1)

2.3.2 URL *Length*

Hacker memakai URL yang panjang untuk menipu contohnya HTTP://www.paypal.ca/3f/aze/ab3ed5f46s64gasd65a8sd7/?cmd=homemp;dispatch=110278asd124379asd27134foe4ij123749asoieh1236419adfjsabeo19237@Phishing.html. Jika URL memiliki panjang kurang dari 54 maka diberi nilai 1, Jika URL memiliki panjang antara 54 dan 75 diberi nilai nol, dan selebihnya diberi nilai -1.

2.3.3 *Shortening Service*

URL dibuat menjadi pendek dan mengarahkan ke web yang dimaksud. Hal ini terjadi karena adanya "HTTP *Redirect*" pada domain URL dan mengarahkannya ke web yang sama yang memiliki URL panjang. Contoh, HTTP://www.sample.com dapat di *Shortening* menjadi HTTP://bit.ly/8as46e. Jika URL menggunakan *Shortening service* maka nilai -1. Jika tidak, nilai 1.

2.3.4 *Having At Symbol*

Mengecek URL apakah memiliki simbol '@', Contoh, HTTP://con@toh.com, maka mengabaikan HTTP://con dan mengarahkannya ke "toh.com." Jika URL mempunyai simbol '@' diberi nilai -1. Kalau tidak diberi nilai satu.

2.3.5 *Double Slash Redirecting*

Jika ada '//' pada URL setelah 6 karakter di dalam URL maka diarahkan ke *website* lain. Contohnya HTTP://contoh.com//HTTP://www.phishing.com maka diarahkan ke *phishing*.com, bukan "contoh.com." Pada URL dengan HTTP, simbol '//' ditemukan pada posisi keenam, sedangkan pada HTTPS, ditemukan pada posisi ketujuh. Jika posisi simbol '//' pada URL berada pada posisi lebih dari tujuh maka nilainya -1 kalau tidak, maka diberi nilai satu.

2.3.6 *Prefix Suffix*

Penggunaan simbol (-) jarang dipakai pada URL sah *Hacker* menambahkan imbuhan atau akhiran dan memisahkannya dengan (-) untuk menipu, sehingga menganggap mereka mengakses URL yang asli. Contoh, HTTP://www.confirmed-paypal.com. Jika simbol '-' ada pada URL maka nilai -1. Jika tidak, maka nilai satu.

2.3.7 *Having Sub Domain*

Contoh, HTTP://www.*school*.ac.uk/students URL tersebut mungkin punya *country-code*-top-level-domain (*ccTLD*), yaitu "UK". Bagian "AC" merupakan kependekan dari "academic", apabila digabungkan "ac.uk" membentuk *second*-level domain (SLD) dan "*school*" merupakan nama domain tersebut. Agar tahu, maka bagian "WWW" harus diabaikan, kemudian menghilangkan *ccTLD* dan menghitung jumlah "." tersisa. Jika jumlahnya > 1, maka URL dapat dikelompokkan "*Suspicious*" karena hanya mempunyai 1 *subdomain*. Jika jumlahnya > dua, maka dikelompokkan "*Phishing*" karena seharusnya memiliki *multiple* sub domain. Jika URL tidak punya sub domain, maka dapat dikelompokkan "*legitimate*.

2.3.8 *SSL Final_State*

Adanya HTTPS pada URL memberikan kesan kepada *user*s bahwa *website* yang diakses *legitimate*, tapi itu belum cukup. Masih disarankan untuk mengecek *certificate* pada HTTPS, seperti otoritas pemberi *certificate* dan umur *certificate*. Pemberi *certificate* yang terkenal biasanya *GeoTrust*, *GoDaddy*, *Doster* dan *VeriSign*. Selain itu, umur *certificate* biasanya dua tahun. Jika URL memiliki HTTPS, *certificate* SSL terpercaya dan umur *certificate* memiliki umur satu tahun maka diberi nilai 1. Jika URL memiliki HTTPS tetapi *certificate* SSL tidak terpercaya, maka diberi nilai 0. Selebihnya, diberi nilai -1.

2.3.9 *Domain Registration Length*

Website phishing biasanya memiliki jangka umur yang pendek, sedangkan *website* yang asli biasanya dibayar secara *regular* untuk beberapa tahun ke depan. Umur *website phishing* terpanjang yang ditemukan adalah hanya satu tahun. Jika umur

website kurang dari satu tahun maka diberi nilai -1. Sebaliknya, maka diberi nilai 1, selebihnya -1.

2.3.10 *Favicon*

Favicon adalah sebuah icon yang terhubung dengan sebuah *webpage* yang spesifik. Kebanyakan *graphical browsers* dan *newsreaders* menampilkan *favicon* sebagai pengingat visual dari identitas *website* pada *address* bar. Jika *favicon* yang ditampilkan berbeda dengan yang muncul di *address* bar, maka kemungkinan *website* tersebut merupakan *phishing*. Jika *website* menggunakan *favicon external* diberi nilai -1 selebihnya 1.

2.3.11 *Port*

Fitur ini untuk validasi *service* tertentu seperti HTTP digunakan pada server. Disarankan hanya memakai *port* yang dibutuhkan dengan tujuan mengurangi adanya gangguan pada server. Maksudnya adalah, jika semua *port* dibuka, maka *Hacker* bisa menjalankan berbagai *service* yang diinginkan. Jika sebuah *website* hanya membuka *port* tertentu saja seperti *port* 21,22,23,80,443 maka diberi nilai satu. Sebaliknya, jika *port* tidak sesuai dengan *port* yang telah ditentukan maka diberi nilai -1.

2.3.12 *HTTPS token*

Hacker bisa saja menambahkan HTTPS di depan URL *phishing* dengan tujuan mengelabui pengguna. Contoh HTTP://HTTPS-www-paypalwebapps.software.com. Jika URL mempunyai HTTPS di depan DNS maka diberi nilai satu. Sebaliknya, maka diberi nilai -1.

2.3.13 *Request* URL

Fitur ini mengecek apakah ada objek eksternal yang dimasukkan ke dalam *website*, seperti images, *videos* dan *sounds* berasal dari domain lain. Dalam *website* yang *legitimate*. Jika sebuah *website* melakukan *Request* objek eksternal kurang dari 22% maka diberi nilai satu. Jika *website* melakukan *Request* objek eksternal >= 22% dan < 61% diberi nilai nol. Selebihnya, maka diberi nilai -1.

2.3.14 URL *of Anchor*

Anchor adalah simbol <a> yang ada dalam *website*. Fungsi fitur ini yaitu *Request* URL. Tetapi fitur ini sebenarnya bertujuan untuk memeriksa: A. Jika *tags* <a> dan *website* memiliki domain yang berbeda. B. Jika *Anchor* tidak terhubung dengan *webpage* apa pun. Contohnya: - <*ahref*="#"> - <*ahref*="#*content*"> - <a *href*="#skip"> - <a *href*="*JavaScript* ::*void*(0)"> Jika *tags* <a> memiliki domain yang berbeda kurang dari 31% maka diberi nilai satu. Jika *tags* <a> memiliki domain yang berbeda >= 31% dan <= 67% maka diberi nilai nol. Selebihnya, maka diberi nilai -1.

2.3.15 URL in *tags*

Umumnya *website* sah mempunyai beberapa seperti *tag* <meta> yang menyediakan meta data dokumen HTML, *tag* <*script*> untuk membuat *script* dari sisi klien, dan *tag* <URL> untuk mengambil *resource* dari web lain. Ketiga, *tag* tersebut biasanya terhubung dari domain yang sama. Jika *tag*<meta>, *tag*<*script*>, dan *tag*<URL> terhubung dari domain yang berbeda kurang dari 17% maka diberi nilai nol. Jika *tag*<meta>, *tag*<*script*>, dan *tag*<URL> terhubung dari domain yang berbeda lebih dari sama dengan 17% dan kurang dari 81% maka diberi nilai nol. Selebihnya maka diberi nilai -1.

2.3.16 SFH (*Server from Handler*)

SFH mempunyai "*about*:*blank*" bisa dicurigai karena harus ada *action* yang dilakukan terhadap informasi yang di *submit*. Jika domain *name* dalam SFH berbeda dengan domain *name* dari *webpage*, patut dicurigai karena biasanya informasi yang di *submit* jarang dikendalikan *external domains*. Jika SFH pada URL berisi "*about*:*blank*" maka diberi nilai -1. Jika SFH pada URL berasal dari domain yang berbeda maka diberi nilai nol. Sebaliknya, maka diberi nilai satu.

2.3.17 *Submitting to email*

Web *form* biasanya diperbolehkan *submit* informasi pribadinya yang kemudian dikirim ke server. Tapi *Hacker* kemungkinan mengirim informasi pribadi *user* ke email pribadinya. Server - *side script language* seperti "*mail*()" *function* di PHP dan "mailto:". Jika *website* mengandung *script* "*mail*()" atau "*mailto*" maka nilai -1. Selebihnya nilai satu.

2.3.18 Abnormal URL

Fitur ini mengambil *database* WHOIS. *Website* yang *legitimate* biasanya mempunyai identitas di bagian URL. Jika *hostname* tidak terdapat pada URL maka diberi nilai -1. Sebaliknya, maka diberi nilai satu.

2.3.19 *Redirect*

Perbedaan dari *website phishing* dan *website* asli adalah seberapa banyak jumlah *website* di *Redirect*. Dalam *datasets* yang digunakan, *website legitimate* biasanya di *Redirect* maksimal satu kali, sedangkan *website phishing* di *Redirect* paling sedikit empat kali. Jika, *website* hanya melakukan *Redirect* setidaknya sebanyak satu kali maka diberi nilai 1. Jika *website* melakukan *Redirect* sebanyak <=2 dan >4 diberi nilai nol. Selebihnya, maka diberi nilai -1.

2.3.20 On Mouseover

Peretas biasanya menggunakan *JavaScript* untuk menunjukkan URL palsu di status bar *user*s. Untuk memakai fitur ini, maka harus mencari dari *source code* terutama di *event* "*onMouseOver*" untuk mengecek apakah terjadi perubahan di status bar. Jika

URL pada status bar berubah atau berbeda diberi nilai -1. Jika tidak, diberi nilai satu.

2.3.21 *Right Click*

Peretas biasanya memakai *JavaScript* untuk menghilangkan fungsi *right click* sehingga *user*s tidak dapat melihat dan menyimpan *source code* dari *webpage* palsu tersebut. Fitur ini dapat digunakan dengan "*Using onMouseOver to hide the* URL". Jika *website* melakukan *disable right click* diberi nilai -1. Selebihnya nilai satu.

2.3.22 *PopUp Window*

Jarang sekali ditemukan *website legitimate* meminta *user* memasukkan informasi pribadinya melalui sebuah *pop-up window*. Hanya saja, fitur ini telah digunakan oleh beberapa *website legitimate*. Jika pada sebuah *website* terdapat pop *up*, diberi nilai -1. Selebihnya nilai satu.

2.3.23 *Iframe*

Iframe ialah *tag* HTML dipakai agar bisa menampilkan halaman web tambahan jadi satu dengan web yang sedang dibuka. Peretas menggunakan *tag* "*iframe*" dan biasanya tidak terlihat atau tanpa menggunakan border. Dalam hal ini, pelaku *phishing* menggunakan atribut "*frame* Border" yang membuat browser menampilkan penggambaran visual. Jika *website* menggunakan *iframe* diberi nilai -1. Selebihnya nilai satu.

2.3.24 *Age_Of_Domain*

Fitur ini dapat diekstrak dari *database* WHOIS. Kebanyakan situs web *phishing* biasanya memiliki umur yang lebih singkat, di mana *website legitimate* biasanya berumur enam bulan atau lebih. Jika umur domain setidaknya berumur enam bulan diberi nilai 1. Selebihnya nilai -1

2.3.25 *DNSRecord*

Identitas *website* tidak ditemukan pada *database* WHOIS dan *hostname* juga tidak ditemukan maka bisa dikategorikan sebagai *phishing*. Jika domain tidak mempunyai DNS *Record*, maka diberi nilai -1. Sebaliknya, maka diberi nilai satu.

2.3.26 *Web Traffic*

Fitur ini mengukur popularitas *website* berdasarkan jumlah pengunjung dan jumlah halaman yang dikunjungi. Dari *datasets* yang digunakan, seburuk-buruknya *rank website legitimate*, masih masuk dalam Top 100.000. Jika sebuah domain tidak dikenali oleh *similar* web, maka domain tersebut patut dicurigai. Jika peringkat *website* kurang dari 100.000 maka diberi nilai satu. Jika peringkat *website* lebih dari sama dengan 100.000 maka diberi nilai nol. Sedangkan, jika *website* tidak terdaftar atau dikenali oleh *similar* web maka diberi nilai -1.

2.3.27 *Page Rank*

PageRank nilai yang berkisar dari "nol" hingga "satu". *PageRank* berguna mengukur seberapa penting halaman web di Internet. Semakin besar nilainya, semakin penting web tersebut. Peneliti menemukan bahwa sekitar 95% halaman web *phishing* tidak memiliki *PageRank*. Selain itu, peneliti menemukan bahwa sisa 5% dari laman web *phishing* dapat mencapai nilai *PageRank* hingga "2". Jika *website* memiliki *PageRank* kurang dari 2, maka diberi nilai -1. Sebaliknya, maka diberi nilai satu.

2.3.28 *Google Index*

Fitur ini memeriksa apakah ada pada indeks Google atau tidak. Jika sebuah situs web terdapat pada indeks Google maka ditampilkan hasilnya. Biasanya, halaman web *phishing* hanya dapat diakses dalam waktu yang singkat dan akibatnya, banyak halaman web *phishing* mungkin tidak ditemukan di dalam indeks Google. Jika *website* terdapat di indeks Google diberi nilai 1. Selebihnya nilai -1

2.3.29 URL *Pointing to Page*

Jumlah URL yang mengarah ke web menampilkan tingkat sahnya meskipun beberapa tautan berada di domain yang sama. Dalam kumpulan *datasets* yang digunakan, karena umur pakainya yang pendek, ditemukan bahwa 98% *website phishing* tidak memiliki tautan URL yang mengarah ke sana. Di sisi lain, *legitimate website* memiliki setidaknya dua tautan URL eksternal yang mengarah ke sana. Jika *website* tidak memiliki tautan URL eksternal maka diberi nilai -1. Jika *website* memiliki tautan URL kurang dari tiga maka diberi nilai nol. Selebihnya, diberi nilai satu.

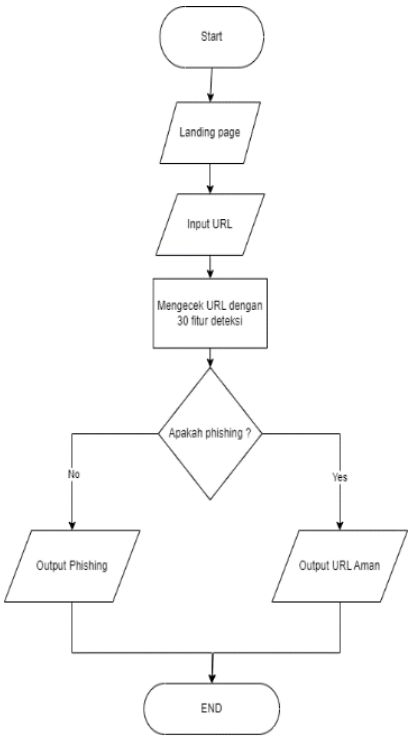
2.3.30 *Statistical Report*

Beberapa situs seperti *Phish* Tank dan *StopBadware* sering kali merilis berbagai laporan statistik mengenai *website phishing* dalam jangka waktu tertentu. Jika *website* termasuk dalam Top *phishing* IP atau Top *phishing* Domain maka diberi nilai -1. Selebihnya nilai 1

3. HASIL DAN PEMBAHASAN

3.1 *Flowchart*

Flowchart di tunjukan pada Gambar 2 berisi langkah dari proses mendeteksi URL *phishing* menggunakan algoritma *random forest*. Untuk mendeteksi harus memasukkan URL saja setelah itu sistem mengecek ke 30 fitur deteksi dan menentukan hasil menggunakan algoritma *random forest.*



Gambar 2. Flowchart

3.2 Implementasi Metode

Pada tahap ini algoritma *random forest* digunakan untuk membuat pemodelan di antaranya yaitu :

a. Membaca *dataset*. Data dibaca menggunakan *library pandas* dengan ketentuan. X membaca semua kolom kecuali kolom terakhir, dan Y label membaca semua kolom kecuali kolom terakhir seperti Gambar 3.

```
data = pd.read_csv('Dataset/dataset.csv')
X = data.iloc[:, :-1].values
y = data.iloc[:, -1].values
```

Gambar 2. Membaca data menggunakan library pandas

b. Memisahkan data menggunakan *library sklearn* dengan kode *python from sklearn.model_selection import train_test_split* Gambar 4 menjelaskan *code*

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

Gambar 4. Memisahkan data uji dan latih

Pada Gambar 4 dijelaskan bahwa data dibagi menjadi 4 bagian yaitu *X_train, X_test, Y_train, Y_test*. Lalu data di *split* menjadi 20% data uji dan 80% data latih. Dengan fungsi *test_size*=0.2. Pembagian data model algoritma yang digunakan di sini umum karena memberikan cukup data untuk pelatihan model sambil menyisakan cukup data untuk evaluasi yang akurat [10].

c. Penggunaan *random forest classifier* data yang sudah dibagi menjadi dua bagian x dan y disimpan pada folder *rf_final.pkl* untuk pembelajaran sistem berikut *code python* pada Gambar 5

```
from sklearn.ensemble import RandomForestClassifier
```

Gambar 3. Memanggil fungsi random forest classifier

Lalu setelah dipanggil *random forest classifier* maka ini dipanggil kembali di dalam *code* berikut pada Gambar 6.

```
classifier = RandomForestClassifier()
classifier.fit(X_train, y_train)
joblib.dump(classifier, 'Last_pkl/rf_final.pkl')
```

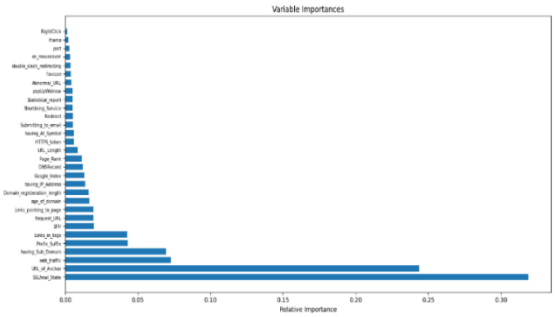
Gambar 6. Fungsi random forest

d. Pengetesan *dataset* URL dengan fitur deteksi pada percobaan ini adalah salah satu contoh dari hasil deteksi pada satu URL HTTPS://google.com pada *link* ini *output* seperti Tabel 2

Tabel 1. Nilai pengecekan https://google.com

No.	Fitur deteksi	Tipe Data	URL aman	*Phishing*
1	*Have* IP *address*	*Interger*	1	
2	Panjang URL	*Interger*	1	
3	URL pendek	*Interger*	1	
4	*Having Symbol*	*Interger*	1	
5	Menggunakan //	*Interger*	1	
6	*Prefix Suffix*	*Interger*	1	
7	*Having subdomain*	*Interger*		-1
8	*SSL final state*	*Interger*	1	
9	*Domain Length*	*Interger*	1	
10	*Favicon*	*Interger*	1	
11	*Port*	*Interger*	1	
12	Mempunyai *HTTPS*	*Interger*	1	
13	*Merequest URL*	*Interger*		-1
14	Mempunyai *Anchor*	*Interger*	1	
15	*Link in tags*	*Interger*	1	
16	*SFH*	*Interger*		-1
17	*Submit email*	*Interger*	1	
18	*Abnormal URL*	*Interger*	1	
19	*Redirect*	*Interger*		-1
20	*On mouse over*	*Interger*		-1
21	Klik kanan	*Interger*		-1
22	*Pop up window*	*Interger*		-1
23	*I frame*	*Interger*		-1
24	Umur domain	*Interger*	1	
25	*DNS* catatan	*Interger*	1	
26	*Web traffic*	*Interger*	1	
27	*Page rank*	*Interger*	1	
28	*Google indeks*	*Interger*	1	
29	*Link to page*	*Interger*	1	
30	*Statistic report*	*Interger*		-1
	Hasil Deteksi			URL aman (1)

e.
```
            data_set.append ( 1) #phishing
            logging.debug("%% : -1")
        except Exception as e:
            data_set.append (-1)
            logging.debug("%% : 1")
```
Penentuan hasil klasifikasi tahap ini setelah mengecek URL dan mengetahui hasil deteksi yang di cek pada 30 fitur deteksi yang ada maka sistem menentukan hasil prediksi, dengan cara menghitung jumlah dari banyaknya fitur deteksi jika hasil yang di tunjukan oleh sistem dominan -1 maka sistem mengindikasikan URL yang di cek adalah *phishing*, begitu pun sebaliknya. Tetapi ada beberapa faktor juga yang menentukan hasil prediksi dari sistem yaitu nilai dari 30 fitur deteksi yang ada untuk mengetahui nilai dari 30 fitur peneliti membuat grafik nilai fitur *dataset* pada Gambar 7, dan cara membacanya.



Gambar 7. Nilai Fitur Deteksi

Grafik nilai pada Gambar 7 menunjukkan bahwa fitur *SSL_final_state* yang memiliki nilai deteksi paling besar di karena kan fitur ini sangat harus dipunyai oleh kebanyakan *website* yaitu fitur sertifikat *security* dari suatu *website*, karena *website* yang sah biasanya diawali dengan HTTPS dan situs *phishing* hanya diawali dengan HTTP, Jika terdapat situs yang mempunyai HTTPS tetapi tidak memiliki sertifikat SSL maka dilabeli *phishing*. Pada posisi kedua di duduki oleh fitur *URL_Anchor* umumnya situs *phishing* memiliki fungsi *tag* <a *href*> yang merujuk ke domain lainya dan *tag* ini tidak hanya terdapat 1 dalam penelitian kali ini peneliti membuat *tracehold* presentasi untuk *tag* <a *href*> sebanyak 31% karena jika melebihi itu maka di indikasikan situs *phishing*. Berikut salah satu program fitur SSL yang menduduki peringkat pertama pada Gambar 8.

```
try:
    if(re.search('^HTTPS',url)):
        usehttps = 1
    else:
        usehttps = 0
    domain = re.findall(r"://([^/]+)/?", url)[0]
    if re.match(r"^www.",domain):
        domain = domain.replace("www.","")
    context = ssl.create_default_context()
    sct = context.wrap_socket(socket.socket(), server_hostname = domain)
    sct.settimeout(10)
    sct.connect((domain, 443))
    certificate = sct.getpeercert()
    issuer = dict(x[0] for x in certificate['issuer'])
    certificate_Auth = str(issuer['commonName'])
    certificate_Auth = certificate_Auth.split()
    if((certificate_Auth[0] == "Network" or certificate_Auth == "Deutsche"):
        certificate_Auth = certificate_Auth[0] + " " + certificate_Auth[1]
    else:
        certificate_Auth = certificate_Auth[0]
    trusted_Auth = ['DigiCert','Lets','Encrypt','LetsEncrypt','Comodo','Google','RapidSSL','CloudFlare','Sectigo','Comodo','Symantec','Go
    startingDate = str(certificate['notBefore'])
    endingDate = str(certificate['notAfter'])
    startingYear = int(startingDate.split()[3])
    endingYear = int(endingDate.split()[3])
    age_of_certificate = endingYear-startingYear
    if((usehttps == 1) and (certificate_Auth in trusted_Auth) and (age_of_certificate> 1) ):
        data_set.append (1) #legitimate
        logging.debug("%% : 1")
    elif((usehttps==1) and (certificate_Auth not in trusted_Auth)):
        data_set.append (0) #suspicious
        logging.debug("SSL : 0")
    else:
```

Gambar 8. Fitur SSL Final State

Cara kerja dari *fhishing* URL *detection* membedakan situs *phishing* dan sah seperti yang dijelaskan pada nilai fitur jika URL pada fitur SSL mendapatkan nilai 0,30. Diberikan kondisi juga jika SSL terindikasi 1 maka menambahkan *point* sebanyak +0,30. Tetapi jika SSL mendapatkan nilai -1 maka mendapatkan nilai -0,30. Lalu setelah mengetahui semua nilai fitur *dataset* maka dijumlahkan, jika hasil semua total berjumlah positif maka situs terindikasi URL aman.

f. Menghitung *score* akurasi dari satu URL yang dideteksi. Tahap ini setelah mengimplementasikan model algoritma *random forest* dan juga mencoba fitur deteksi maka data dievaluasi untuk mengetahui hasil *score* akurasi dari data yang diuji berikut *code* pada Gambar 9.

```
score = classifier.score(X_test, y_test) * 100
logging.info(f"Model accuracy: {score}%")
result_of_analysis = score
```

Gambar 9. Evaluasi score random forest

3.3 Pengujian

Pada Tabel 3 hasil pengecekan 50 URL menghasilkan hasil *true* positif URL *phishing* sebanyak 24 *true positive* URL tetapi *website* mendeteksi juga 1 *False positive* situs *phishing* yang di deteksi aman. Lalu mendeteksi 23 *true negative* URL aman yang dideteksi aman, dan mendeteksi 2 URL *false negative*.

Tabel 2. Tabel uji coba *confusion matrix*

Prediksi Keadaan Data	**Keadaan Data Sebenarnya**	
	Phishing	**URL aman**
Phishing	24	1
URL aman	2	23
Jumlah	26	25

Berikut perhitungan untuk menentukan prediksi, *recall*, f-*score*, akurasi.

$$Prediksi = \frac{TP}{\text{TP+FP}} \text{ x } 100\%$$
$$= \frac{24}{24+1} \text{ x } 100\%$$
$$= 0.96 \text{ atau } 96\%$$

$$Recall = \frac{TP}{\text{TP+FN}} \text{ x } 100\%$$
$$= \frac{24}{24+2} \text{ x } 100\%$$
$$= 0.92 \text{ atau } 92\%$$

$$Akurasi = \frac{(TP+TN)}{(\text{TP + FP + FN +TN})} \text{ x } 100\%$$
$$= \frac{(24+23)}{(24+1+2+23)} \text{ x } 100\%$$
$$= 0.94 \text{ atau } 94\%$$

$$F-SCORE = \frac{2\,X\,P\,X\,R}{(\text{P + R})} \text{ x } 100\%$$
$$= \frac{2\,X\,P\,X\,R}{(0.96 + 0.92)} \text{ x } 100\%$$
$$= 93.95 \text{ atau } 93\%$$

Hasil pengujian di sini dapat disimpulkan bahwa *website* deteksi *phishing* menggunakan algoritma *random forest* dan penambahan beberapa fitur deteksi

dapat menyelesaikan masalah dengan menunjukkan hasil akurasi yang cukup tinggi untuk memprediksi *website phishing*.

4. KESIMPULAN

Kesimpulan yang dapat diambil berdasarkan pengujian, *website fhishing* URL *detection* yang dibuat dapat berguna dengan baik walaupun terdapat beberapa URL yang terdeteksi *false negative* tetapi angka akurasinya masih menunjukkan cukup tinggi dan dinilai dapat menyelesaikan masalah. Penggunaan algoritma *random forest* sangat cocok untuk *website* ini karena mencapai tingkat akurasi 94%, prediksi 96%, *Recall* 92%, dan f-*score* 93%. Dengan penggunaan fitur deteksi sebanyak 30 fitur. Beberapa saran yang dapat dilakukan pada penelitian berikutnya ialah Menambahkan data set menggunakan URL terbaru baik URL *phishing* maupun *non-phishing*, Menambahkan fitur deteksi lainya seperti melalui email DLL, Melakukan *update* fitur deteksi terbaru untuk mengecek URL *phishing*.

DAFTAR PUSTAKA

[1] Saroji Ahmad, Harmini Triana, and Taqiyuddin Muhammad, 'Internet Evolution: A Historical View (SEJARAH EVOLUSI GENERASI INTERNET)', 2021, doi: 10.30598/Lanivol2iss2page65-75.

[2] Acsany Philipp, 'Build a Scalable Flask Web Project From Scratch', https://realpython.com/flask-project/. Accessed: Jun. 07, 2024. [Online]. Available: https://realpython.com/flask-project/

[3] Nurfitrianti Fifi, 'Apa Itu Phishing, Smishing, dan Vhishing?', https://www.jenius.com/highlight/detail/apa-itu-phishing-smishing-dan-vhishing. Accessed: Jun. 07, 2024. [Online]. Available: https://www.jenius.com/highlight/detail/apa-itu-phishing-smishing-dan-vhishing

[4] A. Aljofey *et al.*, 'An effective detection approach for phishing websites using URL and HTML features', *Sci Rep*, vol. 12, no. 1, Dec. 2022, doi: 10.1038/s41598-022-10841-5.

[5] Badan Siber dan Sandi Negara, 'LANSKAP KEAMANAN SIBER INDONESIA', https://www.bssn.go.id/wp-content/uploads/2024/03/Lanskap-Keamanan-Siber-Indonesia-2023.pdf. Accessed: Jun. 07, 2024. [Online]. Available: https://www.bssn.go.id/wp-content/uploads/2024/03/Lanskap-Keamanan-Siber-Indonesia-2023.pdf

[6] A. M. Veach and M. Abualkibash, 'Phishing Website Detection Using Several Machine Learning Algorithms: A Review Paper', *International Journal of Informatics, Information System and Computer Engineering (INJIISCOM)*, vol. 3, no. 2, pp. 219–230, Dec. 2022, doi: 10.34010/injiiscom.v3i2.8805.

[7] I. Yurita, M. Kevin Ramadhan, and M. Candra, 'Pengaruh Kemajuan Teknologi Terhadap Perkembangan Tindak Pidana Cybercrime (Studi Kasus Phising Sebagai Ancaman Keamanan Digital)', 2023.

[8] R. Zieni, L. Massari, and M. C. Calzarossa, 'Phishing or Not Phishing? A Survey on the Detection of Phishing Websites', *IEEE Access*, vol. 11, pp. 18499–18519, 2023, doi: 10.1109/ACCESS.2023.3247135.

[9] W. Ali and S. Malebary, 'Particle Swarm Optimization-Based Feature Weighting for Improving Intelligent Phishing Website Detection', *IEEE Access*, vol. 8, pp. 116766–116780, 2020, doi: 10.1109/ACCESS.2020.3003569.

[10] G. (Gareth M. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning : with applications in R*. 2013.