PEMANFAATAN SKRIP ADAPTIF BERBASIS PYTHON DAN SCAPY UNTUK MENYAMARKAN AKTIVITAS *PORT SCANNING* TERHADAP IDS SURICATA

Junior Silambi[a], Dian W. Chandra[b]

[a,b] *Program Studi Teknik Informatika, Universitas Kristen Satya Wacana, Jawa Tengah*

[a] 672019261@student.uksw.edu, [b] dian.chandra@uksw.edu

ABSTRAK

Sistem Deteksi Intrusi (IDS) berperan penting dalam mendeteksi aktivitas berbahaya pada jaringan, namun efektivitasnya sering kali terganggu oleh teknik pemindaian yang mampu menyamarkan lalu lintas berbahaya. Penelitian ini mengembangkan skrip pemindaian adaptif berbasis Python dengan pustaka Scapy untuk menguji ketahanan IDS Suricata terhadap aktivitas *port scanning*. Pengujian dilakukan dalam lingkungan virtual dengan membandingkan pemindaian konvensional menggunakan Nmap dan skrip adaptif yang dikembangkan. Hasil menunjukkan bahwa skrip adaptif memiliki akurasi yang sama dengan Nmap dalam mengidentifikasi port terbuka, namun tidak memicu *alert* pada Suricata. Temuan ini menunjukkan bahwa pendekatan adaptif sederhana dapat secara efektif menghindari deteksi IDS berbasis *signature*. Penelitian ini menawarkan pendekatan alternatif yang ringan dan efisien untuk menguji keandalan sistem deteksi intrusi modern.

Kata kunci *: Keamanan jaringan, Pemindaian adaptif, Pemindaian port, Sistem Deteksi Intrusi, Suricata, Stealth scan.*

ABSTRACT

Intrusion Detection Systems (IDS) are crucial for identifying malicious network activities, yet their effectiveness is often challenged by stealth scanning techniques. This study develops an adaptive port scanning script using Python and the Scapy library to evaluate the resilience of the Suricata IDS against scanning activities. Experiments were conducted in a virtualized environment comparing conventional Nmap scanning and the proposed adaptive script. Results show that the adaptive script achieved equal accuracy in detecting open ports while generating no alerts on Suricata. These findings demonstrate that a simple adaptive approach can effectively evade signature-based IDS detection. This research contributes a lightweight and efficient alternative for testing the robustness of modern intrusion detection systems.

Keywords: *Network security, Intrusion Detection System, Suricata, Port scanning, Adaptive scanning, Stealth scan.*

Silambi, J., & Chandra, D. W. (2025). PEMANFAATAN SKRIP ADAPTIF BERBASIS PYTHON DAN SCAPY UNTUK MENYAMARKAN AKTIVITAS PORT SCANNING TERHADAP IDS SURICATA. Jurnal Pendidikan Teknologi Informasi (JUKANTI), 8(2), 271–280. https://doi.org/10.37792/jukanti.v8i2.1894

1. PENDAHULUAN

Perkembangan teknologi informasi yang pesat telah mendorong transformasi di berbagai sektor, mulai dari pendidikan, bisnis, hingga layanan publik. Namun, kemajuan ini juga disertai meningkatnya ancaman keamanan siber yang menyasar infrastruktur digital. Laporan Lanskap Keamanan Siber Indonesia 2024 oleh BSSN [1] menegaskan bahwa serangan terhadap sistem dan layanan digital di Indonesia tidak hanya meningkat secara kuantitas, tetapi juga semakin kompleks dari sisi teknik. Kondisi ini menuntut mekanisme pertahanan adaptif guna menjaga ketersediaan, kerahasiaan, dan integritas sistem.

Salah satu teknik dasar dalam serangan siber adalah *port scanning*. Aktivitas ini digunakan untuk memetakan layanan terbuka pada sistem target, yang dapat dimanfaatkan penyerang sebagai titik awal eksploitasi [2]. Sebaliknya, bagi administrator jaringan, *port scanning* berguna sebagai bagian dari uji penetrasi untuk mengidentifikasi kerentanan. Permasalahan muncul karena metode pemindaian konvensional mudah terdeteksi oleh sistem deteksi intrusi (IDS) berbasis *signature* seperti Suricata [3]. Hal ini menimbulkan kebutuhan akan teknik pemindaian yang akurat sekaligus tersamarkan, agar efektivitas IDS modern dapat diuji secara lebih kritis.

Sejumlah penelitian telah berusaha menjawab tantangan ini melalui pendekatan berbeda. Xu et al. [4] memperkenalkan TMorph, *framework* manipulasi trafik untuk menguji ketahanan IDS terhadap serangan adversarial. Pan et al. [5] mengembangkan Scorpio, alat uji penetrasi otomatis yang terintegrasi dengan *cyber range*, sementara Yadav et al. [6] menghadirkan IoT-PEN, kerangka uji penetrasi *end-to-end* khusus untuk *IoT*.

Penelitian lain lebih menekankan pada integrasi IDS dengan sistem manajemen log dan peningkatan akurasi deteksi. Zain et al. [3] mengimplementasikan Suricata bersama ELK Stack, sedangkan Sufardy & Widiasari [7] memanfaatkan PFsense dengan Suricata untuk deteksi serangan jaringan. Emmanuel et al. [8] menyoroti teknik *bypass firewall* dalam konteks *stealth scanning*, dan Smith et al. [9] mendokumentasikan praktik industri melalui studi kasus tim Red Team di Microsoft. Koroniotis et al. [10] mengusulkan kerangka berbasis *deep learning* untuk identifikasi kerentanan, sementara Ghanem et al. [11] mengembangkan pendekatan *reinforcement learning* untuk penetrasi otomatis berskala besar. Di sisi lain, Lyon [2] melalui karyanya tentang Nmap tetap menjadi rujukan utama dalam studi pemindaian port.

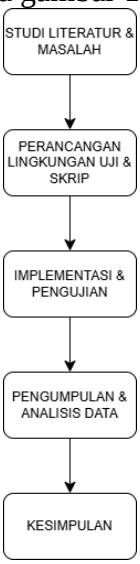
Penelitian ini diposisikan berbeda secara fundamental dari *framework* berskala besar yang ada. TMorph [4] dirancang sebagai alat *traffic morphing* komprehensif, berfokus pada penerapan beragam transformasi ofuskasi (enkripsi, *encoding*, *tunneling*) untuk manipulasi *payload* dan protokol secara umum. Sementara itu, Scorpio [5] merupakan alat otomasi uji penetrasi penuh untuk lingkungan *cyber range*, yang mengotomatisasi seluruh rantai serangan multi-tahap dari pemindaian hingga eksploitasi. Berbeda dari kedua pendekatan tersebut, penelitian ini tidak bertujuan membangun *framework* transformasi umum atau platform otomasi serangan penuh.

Kekosongan penelitian (*research gap*) yang ditangani terletak pada evaluasi spesifik terhadap IDS *signature-based* menggunakan strategi penyamaran yang sederhana namun adaptif. Studi yang berfokus pada efektivitas skrip ringan (*lightweight script*) yang secara dinamis mengombinasikan teknik non-konvensional (seperti *FIN/ACK* dan *Slow SYN Scan*) untuk menguji *signature* Suricata secara spesifik masih terbatas. Oleh karena itu, kontribusi utama penelitian ini adalah pengembangan dan analisis skrip pemindaian port adaptif berbasis Python dan Scapy. Skrip ini dirancang untuk mengombinasikan teknik *FIN/ACK Scan*, *Slow SYN Scan*, dan *SYN Scan* standar, dengan tujuan menguji secara empiris kemampuannya dalam

mengidentifikasi port terbuka secara akurat sekaligus meminimalkan deteksi oleh aturan-aturan spesifik pada IDS Suricata.

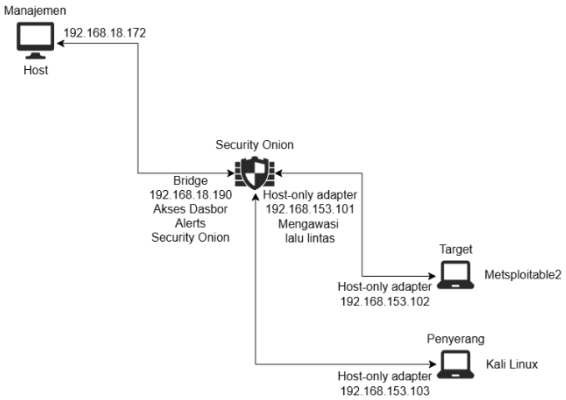
2. METODE PENELITIAN

Penelitian ini dilaksanakan melalui serangkaian tahapan sistematis, mulai dari studi literatur dan menemukan masalah, perancangan metodologi, penyusunan topologi jaringan virtual, penentuan spesifikasi perangkat, pelaksanaan pengujian, hingga analisis hasil, serta penarikan kesimpulan. Keseluruhan alur ditunjukkan pada gambar 1.



Gambar 1. Tahapan Penelitian

Lingkungan pengujian virtual dirancang untuk mensimulasikan skenario pemindaian dalam jaringan yang dipantau. Topologi jaringan terdiri atas empat komponen utama: *Virtual Machine* (VM) Penyerang, VM Target, VM Security Onion, dan komputer *host*. VM Penyerang (Kali Linux) berfungsi sebagai titik asal pemindaian. VM Target (Metasploitable2) berperan sebagai sistem rentan yang menjadi sasaran pemindaian. VM Security Onion 2.4.170 dikonfigurasi sebagai *Intrusion Detection System* (IDS), yang memantau seluruh lalu lintas jaringan melalui dua antarmuka: satu antarmuka *host-only* (192.168.153.101) untuk memonitor komunikasi antar-VM dan satu antar muka *bridged* (192.168.18.190) untuk akses antarmuka web Security Onion. Komputer *host* dengan alamat IP 192.168.18.172 digunakan untuk mengelola lingkungan virtual. Konfigurasi ini memastikan seluruh lalu lintas antara VM Penyerang dan VM Target dapat dipantau oleh Security Onion, sebagaimana dirinci dalam gambar 2 dan tabel 1.



Gambar 2. Topologi Penelitian

Tabel 1. Konfigurasi IP VM

Nama VM	Jenis Adapter	Alamat IP
Attacker	Host-only adapter	192.168.153.103
Target	Host-only adapter	192.168.153.102
Security Onion	Host-only adapter	192.168.153.101
	Bridged adapter	192.168.18.190
Host	Bridged adapter	192.168.18.172

Eksperimen dijalankan menggunakan VMware Workstation 17 Pro pada komputer *host* Windows 11 Pro dengan prosesor AMD Ryzen 3 5300U dan alokasi RAM 20 GB. Alokasi sumber daya virtual mencakup 4 CPU dan 2 GB RAM untuk VM Penyerang, 1 CPU dan 512 MB RAM untuk VM Target, serta 4 CPU dan 8 GB RAM untuk VM Security Onion. Perangkat lunak yang digunakan dalam pengujian meliputi IDS Suricata 7.0.11 (bawaan Security Onion), Nmap 7.95 sebagai alat pemindai konvensional, serta skrip adaptif yang

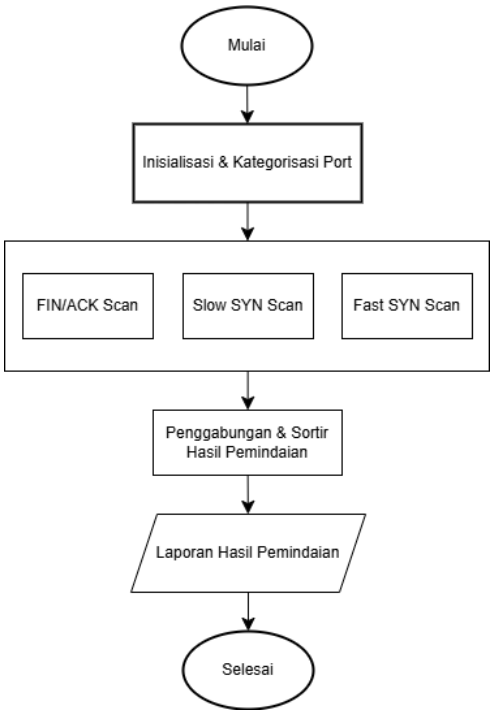
dikembangkan menggunakan Python 3.13.3 dan pustaka Scapy 2.6.1.

Penelitian ini mengadopsi desain eksperimental komparatif untuk mengevaluasi efektivitas dan tingkat kesenyapan dari dua metode pemindaian port terhadap deteksi IDS. Skenario pertama, yang berfungsi sebagai *baseline*, menggunakan metode pemindaian konvensional Nmap dengan perintah `nmap -sS [IP_Target] -p-`. Perintah ini menjalankan *SYN* Scan standar pada seluruh rentang port (1-65535) dan merepresentasikan metode yang umumnya mudah terdeteksi. Skenario kedua menggunakan skrip adaptif berbasis Python-Scapy yang dirancang untuk memodifikasi parameter pemindaian guna menghindari deteksi IDS Suricata. Perbandingan kedua skenario didasarkan pada parameter kunci yang dirangkum dalam tabel 2.

Tabel 2. Parameter Konfigurasi Eksperimen

Parameter	Skenario 1 (Nmap)	Skenario 2 (Skrip Adaptif)
Perintah	`nmap -sS [IP Target] -p-`	`python3 scanner.py [IP Target] 1 65535`
Metode Pemindaian	*SYN Scan*	Hibrida (*Fast SYN*, *Slow SYN*, *FIN /ACK Scan*)
Jumlah *Thread*	Default Nmap	3 *Thread*
Pola Paket	*SYN*	*SYN*, *FIN*, *ACK*
Waktu Jeda (*Delay*)	Default Nmap	15 detik (untuk *Slow SYN Scan*)

Skrip adaptif yang digunakan dalam skenario kedua dirancang untuk mengatasi deteksi IDS Suricata dengan memodifikasi pola pemindaian standar. Logika operasional skrip diatur oleh empat algoritma utama yang dieksekusi secara terkoordinasi (gambar 3).



Gambar 3. Flowchart Skrip

Algoritma 1. Fungsi Utama

```
Input:  target_ip, rentang_port
Output: daftar_port_terbuka
1:BEGIN
2: ports_db ← {Port basis data dalam rentang_port}
3: ports_vnc ← {Port VNC dalam rentang_port}
4: ports_khusus ← ports_db ∪ ports_vnc
5: ports_lainnya ← {Semua port di rentang_port}\ports_khusus
6: antrian_hasil ← Buat Queue untuk hasil paralel
7: // Inisialisasi & Eksekusi Thread Paralel
8: thread_1 ← EXECUTE_IN_PARALLEL(perform_fin_ack_scan,
ports_db)
9: thread_2 ← EXECUTE_IN_PARALLEL(perform_slow_syn_scan,
ports_vnc)
10: thread_3 ← EXECUTE_IN_PARALLEL(perform_fast_syn_scan,
ports_lainnya)
11: // Tunggu semua thread selesai
12: WAIT_FOR_ALL_THREADS_TO_COMPLETE
13: daftar_port_terbuka ← Kumpulkan hasil dari antrian_hasil
14: RETURN daftar_port_terbuka
15:END
```

Algoritma 1 (Fungsi Utama) berfungsi sebagai koordinator. Algoritma ini pertama-tama melakukan inisialisasi dan kategorisasi seluruh rentang port target (1-65535). Port dibagi menjadi tiga kategori: port untuk *FIN/ACK Scan* (1433, 1521, 3306, 4333, 5432), port untuk *Slow SYN Scan* (5800-5820) dan port sisanya untuk *Fast SYN Scan*.

Algoritma kemudian meluncurkan tiga *thread* pemrosesan secara paralel, di mana setiap *thread* mengeksekusi satu dari tiga fungsi pemindaian (Algoritma 2, 3, atau 4) secara simultan terhadap kategori port yang relevan.

Algoritma 2. Fungsi Fast SYN Scan

```
function perform_syn_scan(target, ports):
    for port in ports:
        send TCP packet with SYN flag
        if response has SYN-ACK:
            mark port as OPEN
        elif response has RST:
            mark port as CLOSED
        else:
            mark port as FILTERED
```

Algoritma 2 (Fungsi Fast SYN Scan) mengimplementasikan *SYN Scan* konvensional. Fungsi ini mengirimkan paket *SYN* secara massal ke rentang port yang ditetapkan. Port diidentifikasi sebagai terbuka jika menerima balasan *SYN-ACK*, tertutup jika menerima *RST*, atau *filtered* jika tidak ada respons dari VM Target.

Algoritma 3. Fungsi FIN/ACK Scan

```
function perform_fin_ack_scan(target, ports):
    for port in ports:
        send TCP packet with FIN flag
        if no response:
            mark port as OPEN|FILTERED
            send TCP packet with ACK flag
            if response has RST:
                mark port as OPEN
            else:
                mark port as FILTERED
        elif response has RST:
            mark port as CLOSED
```

Algoritma 3 (Fungsi FIN/ACK Scan) dirancang untuk memvalidasi port yang dilindungi *firewall*. Teknik ini secara fundamental mengeksploitasi standar TCP, sebagaimana telah diatur dalam RFC 9293[12], di mana port tertutup diwajibkan merespons paket FIN dengan RST, sementara port terbuka umumnya akan mengabaikannya. Untuk pengiriman paket ACK sendiri digunakan untuk mengecek apakah ada *firewall* yang menghalangi akses ke port tujuan. Proses dimulai dengan pengiriman paket *FIN*. Jika tidak ada respons, port dianggap berstatus *open|filtered*. Untuk membedakan status tersebut, paket *ACK* kemudian dikirim ke port yang sama. Jika target merespons dengan paket *RST*, port tersebut divalidasi sebagai terbuka (*open*), jika tidak ada respons maka port dianggap *filtered* oleh *firewall*.

Algoritma 4. Fungsi Slow SYN Scan

```
function perform_slow_syn_scan(target, ports, delay):
    for port in ports:
        send TCP packet with SYN flag
        wait for 'delay' interval before next packet
        if response has SYN-ACK:
            mark port as OPEN
        elif response has RST:
            mark port as CLOSED
        else:
            mark port as FILTERED
```

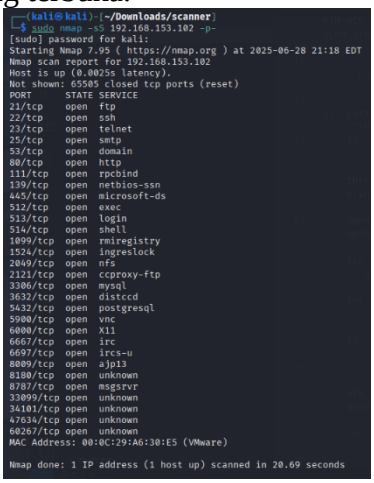
Algoritma 4 (Fungsi Slow SYN Scan) memiliki logika yang identik dengan Algoritma 2 dalam mendeteksi balasan *SYN-ACK*, namun dengan modifikasi temporal yang signifikan. Fungsi ini memasukkan jeda waktu (*delay*) selama 15 detik di antara pengiriman setiap paket *SYN*. Pendekatan ini bertujuan untuk mengalahkan *signature* IDS berbasis waktu (*time-based*) yang melacak pemindaian berkecepatan tinggi.

Prosedur pengujian dilaksanakan secara berurutan. Tahap pertama berupa pemindaian menggunakan Nmap (Skenario 1), diikuti pencatatan *alert* yang muncul pada antarmuka web Security Onion. *Alert* yang terpicu kemudian dianalisis berdasarkan aturan deteksi *ET Open Ruleset* [13]. Hasil analisis ini menjadi dasar penyusunan skrip pemindaian adaptif (Skenario 2), yang kemudian diuji dengan prosedur pencatatan *alert* yang sama. Untuk menjamin validitas hasil, sebuah prosedur verifikasi akurasi diterapkan dengan

mengeksekusi perintah `netstat -tlpn` langsung pada VM Target (Metasploitable2) untuk memperoleh daftar port yang aktif. Evaluasi kinerja dilakukan menggunakan dua metrik utama. Metrik pertama adalah akurasi identifikasi port, yang didefinisikan sebagai kesesuaian 100% antara port terbuka yang dilaporkan pemindai dengan daftar port terbuka pada VM Target. Metrik kedua adalah sensitivitas deteksi IDS, yang diukur berdasarkan jumlah total *alert* yang dihasilkan Suricata. Keseimbangan antara akurasi tinggi dan jumlah *alert* rendah menjadi indikator keberhasilan skrip adaptif dalam menghindari deteksi IDS.

3. HASIL DAN PEMBAHASAN

Pengujian skenario pertama dilakukan menggunakan pemindaian Nmap 7.95 dengan teknik *SYN Scan* (-sS) terhadap seluruh rentang port (1-65535) pada VM target. Hasil pemindaian ini berhasil mengidentifikasi 30 port yang terbuka, termasuk layanan seperti SSH, HTTP, dan MySQL, seperti yang ditunjukkan pada gambar 4. Akurasi jumlah port ini telah divalidasi silang (*cross-check*) menggunakan perintah `netstat -tlpn` langsung pada VM Target, yang mengonfirmasi bahwa 30 port tersebut memang terbuka.



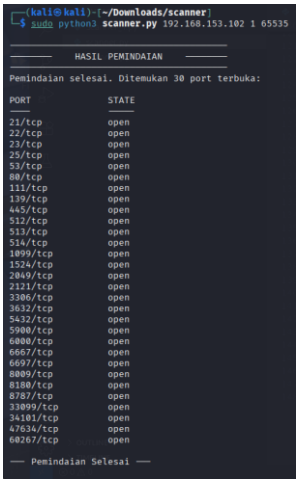
Gambar 4. Hasil Pemindaian Nmap

Akan tetapi, aktivitas pemindaian Nmap ini secara signifikan memicu enam *alert* pada IDS Suricata, sebagaimana divisualisasikan pada gambar 5. Temuan ini mengonfirmasi bahwa pemindaian *SYN Scan* konvensional menghasilkan pola lalu lintas yang sangat mudah dikenali oleh sistem berbasis *signature*.

Count	rule.name
1	ET SCAN Potential VNC Scan 5800-5820
1	ET SCAN Suspicious inbound to MSSQL port 1433
1	ET SCAN Suspicious inbound to Oracle SQL port 1521
1	ET SCAN Suspicious inbound to PostgreSQL port 5432
1	ET SCAN Suspicious inbound to mSQL port 4333
1	ET SCAN Suspicious inbound to mySQL port 3306

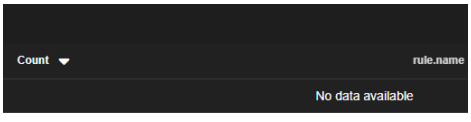
Gambar 5. Alert Suricata Akibat Pemindaian Nmap

Berbeda dengan skenario pertama, skenario kedua dilakukan menggunakan skrip adaptif yang dikembangkan menunjukkan hasil yang kontras secara signifikan. Skrip adaptif juga berhasil mengidentifikasi 30 port terbuka yang sama (gambar 6), mencapai akurasi 100% terhadap verifikasi netstat.



Gambar 6. Hasil Pemindaian Skrip

Namun, pemindaian ini tidak menghasilkan *alert* sama sekali pada *dashboard* Suricata, seperti yang terlihat pada gambar 7.



Gambar 7. Alert Suricata Akibat Pemindaian Skrip

Perbandingan kuantitatif hasil kedua skenario pengujian dirangkum dalam tabel 3.

Tabel 3. Perbandingan Hasil Pemindaian Nmap dan Skrip

Aspek Pengujian	**Skenario 1 (Nmap)**	**Skenario 2 (Skrip Adaptif)**
Port Terbuka Terdeteksi	30	30
Jumlah *Alert* Suricata	6	0
Port Pemicu *Alert*	1433, 1521, 3306, 4333, 5432, 5800-5820	(Tidak ada)
Metode Evasi Skrip	(Tidak ada)	*FIN/ACK Scan* (Port DB), *Slow SYN Scan* (Port VNC)

Keberhasilan skrip adaptif dalam menghindari deteksi secara total bukanlah akibat dari kombinasi teknik yang acak. Analisis mendalam terhadap *signature rule ET Open Ruleset* yang terpicu oleh Nmap dan kegagalannya mendeteksi skrip adaptif mengungkap dua mekanisme penghindaran yang berbeda namun saling melengkapi.

Mekanisme pertama adalah penghindaran *signature* berbasis *flag*. Kelima *alert* Nmap pada port basis data (1433, 1521, 3306, 4333, 5432) terpicu oleh *rule* seperti "*ET SCAN Suspicious inbound to mySQL port 3306*". Logika deteksi fundamental untuk semua *rule* ini adalah pencocokan paket yang memiliki `flags:S`. Parameter `threshold: type limit` yang digunakan pada *rule* ini berfungsi sebagai pembatas pencatatan (*alert suppression*), yang berarti pemicu deteksi sesungguhnya adalah kecocokan murni pada paket *SYN*. Hal ini menjelaskan mengapa skrip adaptif, yang secara proaktif menggunakan *FIN/ACK Scan* pada port-port tersebut berhasil lolos. Sesuai standar TCP yang diatur dalam RFC 9293[12], paket FIN dan ACK merupakan paket legal dalam protokol, namun memiliki struktur *flag* yang secara fundamental berbeda dari paket SYN. Perbedaan inilah yang membuatnya tidak cocok dengan *signature* `flags:S` yang dicari oleh IDS.

Mekanisme kedua adalah penghindaran *threshold* berbasis laju (*rate evasion*). *Alert* Nmap "*ET SCAN Potential VNC Scan 5800-5820*", menggunakan logika deteksi yang berbeda yaitu `threshold: type both`. Ini berarti *alert* hanya akan terpicu jika dua kondisi terpenuhi: (1) paket cocok dengan `flags:S`, dan (2) laju paket dari sumber yang sama (`track by_src`) melebihi ambang batas `count 5, seconds 60`. Skrip adaptif dirancang untuk mengeksploitasi kondisi kedua ini. Dengan mengategorikan rentang port VNC (5800-5820) ke Fungsi Slow SYN Scan dan menerapkan jeda waktu (*delay*) selama 15 detik antar paket, laju pengiriman (4 paket per 60 detik) berada di bawah ambang batas `count 5` yang disyaratkan *rule* VNC. Meskipun paket *SYN* digunakan, kegagalan memenuhi *threshold* laju paket membuat aktivitas ini tidak terdeteksi sebagai pemindaian port.

Temuan ini konsisten dengan penelitian Emmanuel et al. [8], yang menyoroti teknik *bypass* berbasis modifikasi paket, serta studi Xu et al. [4] dan Pan et al. [5] yang menekankan pentingnya variasi trafik dalam pengujian IDS. Secara spesifik, penelitian ini menegaskan kontribusi dalam konteks pengembangan alat uji penetrasi. Sementara *framework* komprehensif seperti TMorph [4] berfokus pada penghindaran deteksi

melalui modifikasi *payload* yang kompleks (seperti enkripsi), penelitian ini menunjukkan bahwa penghindaran deteksi yang efektif terhadap IDS berbasis *signature* dapat dicapai melalui strategi yang lebih sederhana, yaitu manipulasi *header* protokol (TCP *flags*) dan modifikasi laju lalu lintas (*traffic rate*). Pendekatan ini sejalan dengan pengembangan alat Scorpio [5], yang juga mengidentifikasi pemindaian *stealth* (misalnya, *TCP FIN scan*) sebagai metode krusial untuk mengurangi deteksi. Kelebihan skrip yang dikembangkan dalam penelitian ini terletak pada implementasi model hibrida paralel, yang secara proaktif mengategorikan dan menerapkan teknik pemindaian paling efisien (baik dari segi kecepatan maupun kesenyapan) secara simultan berdasarkan sensitivitas target yang telah diketahui.

Lebih lanjut, temuan ini menawarkan kontras yang jelas terhadap pendekatan pertahanan berbasis *machine learning* (ML) (misalnya [10], [11], [14]) yang dirancang untuk mendeteksi pola non-konvensional. Penelitian ini justru menunjukkan bahwa IDS berbasis *signature* yang umum digunakan saat ini masih dapat dihindari secara efektif menggunakan strategi adaptif sederhana yang menggabungkan manipulasi *flag* dan waktu.

Meskipun hasilnya efektif dalam lingkungan terkontrol terhadap Suricata 7.0.11, penelitian ini memiliki keterbatasan. Pengujian hanya dilakukan pada satu target (Metasploitable2) dan belum diuji terhadap IDS berbasis anomali atau *machine learning*, yang mungkin dapat mendeteksi pola non-konvensional seperti *Slow SYN Scan* atau *FIN/ACK Scan*. Penelitian lanjutan dapat memperluas evaluasi ini pada jaringan produksi, melibatkan IDS berbasis kecerdasan buatan, dan menggunakan sistem operasi target yang modern untuk menguji ketahanan sistem keamanan secara lebih komprehensif.

4. KESIMPULAN

Penelitian ini membahas pengembangan skrip pemindaian adaptif berbasis Python dan pustaka Scapy untuk menguji efektivitas IDS Suricata terhadap aktivitas pemindaian port. Hasil pengujian menunjukkan bahwa baik Nmap maupun skrip adaptif memiliki akurasi yang sama dalam mengidentifikasi port terbuka pada target Metasploitable2. Namun, perbedaan signifikan terlihat dari sisi deteksi oleh IDS. Pemindaian dengan Nmap memicu enam *alert* Suricata, sedangkan pemindaian dengan skrip adaptif tidak menimbulkan *alert* sama sekali. Hal ini menunjukkan bahwa pemindaian adaptif yang menggabungkan manipulasi *flag* TCP dan modifikasi laju temporal mampu menghasilkan pola lalu lintas yang lebih tersamarkan dibandingkan pendekatan konvensional.

Dengan demikian, penelitian ini memberikan kontribusi empiris terhadap pengembangan metode pengujian IDS berbasis *signature* dengan pendekatan adaptif yang lebih realistis. Secara praktis, skrip ini dapat dimanfaatkan dalam skenario audit keamanan jaringan atau sebagai alat *benchmark* untuk menguji efektivitas dan konfigurasi *ruleset* IDS secara efisien dan fleksibel.

Namun, penelitian ini memiliki keterbatasan karena hanya diuji pada satu target (Metasploitable2) dan satu IDS (Suricata). Rencana penelitian lanjutan akan berfokus pada pengujian skrip terhadap IDS berbasis kecerdasan buatan (AI-based IDS), yang mungkin mampu mendeteksi pola non-konvensional. Rencana pengembangan lanjutan juga mencakup integrasi skrip ke dalam *framework* otomatisasi pengujian yang lebih besar serta eksplorasi teknik penyamaran yang lebih kompleks seperti *packet morphing* untuk memperkaya evaluasi ketahanan sistem keamanan jaringan.

DAFTAR PUSTAKA

[1] Badan Siber dan Sandi Negara, "Lanskap Keamanan Siber Indonesia 2024," Jakarta: Id-SIRTII/CC - BSSN, 2024.

[2] G. Lyon, *Nmap network scanning: official Nmap project guide to network discovery and security scanning*, Zero-day Release: May 2008. Sunnyvale, CA: Insecure.Com LLC, 2010.

[3] A. R. Zain, P. Oktivasari, N. Fauzi Soelaiman, and F. Watsiqul Umam, "Implementasi Intrusion Detection System (IDS) Suricata Dan Management Log Elk Stack Untuk Pendeteksian Kegiatan Mining," *J. Poli-Teknol.*, vol. 22, no. 1, pp. 23–29, Jan. 2023, doi: 10.32722/pt.v22i1.4974.

[4] Z. Xu, H. Khan, and R. Muresan, "TMorph: A Traffic Morphing Framework to Test Network Defenses Against Adversarial Attacks," in *2022 International Conference on Information Networking (ICOIN)*, Jeju-si, Korea, Republic of: IEEE, Jan. 2022, pp. 18–23. doi: 10.1109/ICOIN53446.2022.9687165.

[5] W. Pan, X. Liu, J. Han, W. Zheng, and M. Yin, "Scorpio: an Automated Penetration Testing Tool and Its Integration with a Cyber Range," in *2021 2nd International Conference on Electronics, Communications and Information Technology (CECIT)*, Sanya, China: IEEE, Dec. 2021, pp. 1113–1119. doi: 10.1109/CECIT53797.2021.00197.

[6] G. Yadav, K. Paul, A. Allakany, and K. Okamura, "IoT-PEN: An E2E Penetration Testing Framework for IoT," *J. Inf. Process.*, vol. 28, no. 0, pp. 633–642, 2020, doi: 10.2197/ipsjjip.28.633.

[7] D. B. Sufardy and I. R. Widiasari, "The Use of PFSense and Suricata as a Network Security Attack Detection and Prevention Tool on Web servers," *INOVTEK Polbeng - Seri Inform.*, vol. 9, no. 2, pp. 765–777, Oct. 2024, doi: 10.35314/shxy2045.

[8] O. I. Emmanuel, A. A. Ayodele, A. M. Adebiyi, and B. F. Osang, "Windows Firewall Bypassing Techniques: An Overview of HTTP Tunneling and Nmap Evasion," in *Computational Science and Its Applications – ICCSA 2021*, vol. 12957, O. Gervasi, B. Murgante, S. Misra, C. Garau, I. Blečić, D. Taniar, B. O. Apduhan, A. M. A. C. Rocha, E. Tarantino, and C. M. Torre, Eds., in Lecture Notes in Computer Science, vol. 12957. , Cham: Springer International Publishing, 2021, pp. 546–556. doi: 10.1007/978-3-030-87013-3_41.

[9] J. Smith, C. Theisen, and T. Barik, "A Case Study of Software Security Red Teams at Microsoft," in *2020 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, Dunedin, New Zealand: IEEE, Aug. 2020, pp. 1–10. doi: 10.1109/VL/HCC50065.2020.9127203.

[10] N. Koroniotis, N. Moustafa, B. Turnbull, F. Schiliro, P. Gauravaram, and H. Janicke, "A Deep Learning-based Penetration Testing Framework for Vulnerability Identification in Internet of Things Environments," in *2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, Shenyang, China: IEEE, Oct. 2021, pp. 887–894. doi: 10.1109/TrustCom53373.2021.00125.

[11] M. C. Ghanem, T. M. Chen, and E. G. Nepomuceno, "Hierarchical reinforcement learning for efficient and effective automated penetration testing of large networks," *J. Intell. Inf. Syst.*, vol. 60, no. 2, pp. 281–303, Apr. 2023, doi: 10.1007/s10844-022-00738-0.

[12] W. Eddy, "Transmission Control Protocol (TCP)," RFC Editor,

RFC9293, Aug. 2022. doi: 10.17487/RFC9293.

[13] "Proofpoint Emerging Threats Rules." Accessed: Jul. 01, 2025. [Online]. Available: https://rules.emergingthreats.net/open/suricata-7.0.3/rules/

[14] A. A. Mohamed, A. Al-Saleh, S. K. Sharma, and G. G. Tejani, "Zero-day exploits detection with adaptive WavePCA-Autoencoder (AWPA) adaptive hybrid exploit detection network (AHEDNet)," *Sci. Rep.*, vol. 15, no. 1, p. 4036, Feb. 2025, doi: 10.1038/s41598-025-87615-2.