Pengembangan *Custom Ruleset* IDS Berbasis Skenario Serangan Multi-Tahap pada Kompetisi CTF, dengan Evaluasi *Detection Rate* dan *False Positive*

Surya Shinta Widya*, Maidel Fani *

* Politeknik Negeri Batam
Rekayasa Keamanan Siber
Jl. Ahmad Yani, Batam Centre, Batam 29461, Indonesia
E-mail: surya.shinta1804@gmail.com
Politeknik Negeri Batam
Rekayasa Keamanan Siber
Jl. Ahmad Yani, Batam Centre, Batam 29461, Indonesia
E-mail: maidelfani@polibatam.ac.id

Abstrak

Suricata sebagai *Network Intrusion Detection System* (NIDS) *open-source* umumnya mengandalkan *Emerging Threats Open* (ET Open) ruleset yang bersifat generik, sehingga berpotensi memiliki celah deteksi terhadap tahapan serangan yang terstruktur dalam sebuah *attack chain*. Penelitian ini bertujuan mengidentifikasi celah deteksi ET Open terhadap *attack chain* kompetisi CTF KMIPN 2024, merancang *Custom rules* untuk menutup celah tersebut, serta mengevaluasi *detection rate* dan *false positive rate* dari *Custom rules* yang dikembangkan. Penelitian menggunakan metode eksperimental dengan desain before-after pada lingkungan lab terkontrol, di mana Suricata berperan sebagai *sniffer* pasif dalam mode promiscuous dan AF_PACKET. *Custom rules* dirancang berdasarkan analisis *packet capture* pada setiap tahapan serangan yang tidak terdeteksi oleh ET Open. Hasil penelitian menunjukkan peningkatan *detection rate* setelah penambahan *custom rules*, dengan visibilitas serangan yang dipetakan ke framework MITRE ATT&CK dalam bentuk coverage map. Penelitian ini memberikan kontribusi berupa *custom ruleset* yang dapat meningkatkan visibilitas Suricata terhadap *attack chain* terstruktur.

Kata kunci: Suricata, *Intrusion Detection System, custom rules, attack chain,* MITRE ATT&CK

Abstract

Suricata, as an open-source Network Intrusion Detection System (NIDS), commonly relies on the Emerging Threats Open (ET Open) ruleset, which is generic in nature and may therefore have detection gaps against the structured stages of an attack chain. This study aims to identify ET Open's detection gaps against the attack chain in the KMIPN 2024 CTF competition, design Custom rules to close those gaps, and evaluate the detection rate and false positive rate of the developed custom rules. An experimental method with a before-after design was applied in a controlled VMware-based lab environment, in which Suricata operated as a passive sniffer in promiscuous and AF_PACKET mode. Custom rules are developed based on packet capture analysis of attack stages that are not detected by ET Open. The results show an improved detection rate following the addition of custom rules, with attack visibility mapped to the MITRE ATT&CK framework in the form of a coverage map. This research contributes a custom ruleset capable of improving Suricata's visibility against structured attack chains.

Keywords : Suricata, Intrusion Detection System, custom rules, attack chain, MITRE ATT&CK

1. Pendahuluan

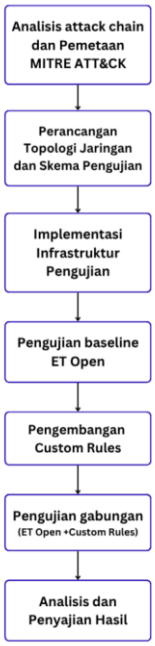
Suricata sebagai *Network Intrusion Detection System* (NIDS) umumnya digunakan dengan mengandalkan *Emerging Threats Open* (ET Open) *ruleset* sebagai sumber *signature default* [1]. *Ruleset* ini bersifat generik karena dirancang untuk mencakup berbagai jenis ancaman secara luas berdasarkan payload dan pola eksploitasi yang sudah dikenal, sehingga tidak disusun berdasarkan skenario serangan atau topologi jaringan tertentu. Akibatnya, ET Open berpotensi

memiliki celah deteksi *(detection gap)* terhadap aktivitas yang menggunakan port *non-standar* atau yang secara karakteristik *traffic* menyerupai aktivitas normal, seperti enumerasi layanan melalui port RPC/SMB yang tidak *default*, *login* FTP anonymous, maupun directory dan *credential Enumeration* berbasis tools umum (gobuster, hydra) yang *traffic*-nya menyerupai permintaan HTTP biasa. Permasalahan generiknya *ruleset default* ini diperkuat oleh temuan StealthCup, sebuah evaluasi IDS berbasis kompetisi CTF multi-tahap, yang menunjukkan bahwa dari 32 teknik serangan yang diujikan, 11 teknik tidak terdeteksi oleh konfigurasi IDS apa pun yang diuji, termasuk Suricata, sehingga ketergantungan pada *ruleset default* saja belum cukup untuk memberikan visibilitas yang memadai terhadap rangkaian serangan terstruktur [2]. Permasalahan generiknya *ruleset default* ini menjadi relevan ketika IDS dihadapkan dengan *attack chain*, yaitu rangkaian tahapan serangan yang saling berurutan mulai dari reconnaissance hingga *privilege escalation*, sebagaimana dipetakan dalam *framework* MITRE ATT&CK [3]. Beberapa penelitian sebelumnya telah membahas evaluasi kemampuan deteksi IDS berbasis Suricata, namun sebagian besar berfokus pada perbandingan performa antar IDS atau pengujian terhadap jenis serangan tunggal, belum banyak yang secara khusus mengevaluasi visibilitas IDS terhadap rangkaian tahapan serangan yang terstruktur dan saling berurutan dalam satu attack chain yang utuh. *ruleset default* seperti ET Open umumnya bersifat generik dan tidak dirancang untuk konfigurasi maupun teknik serangan yang spesifik pada suatu lingkungan, sehingga berpotensi menyisakan celah deteksi pada sebagian tahapan *attack chain*. Untuk menutup celah tersebut, *Custom rules* perlu dirancang berdasarkan analisis *packet capture* pada tiap tahapan *attack chain* yang tidak ter-*trigger* oleh ET Open. Akan tetapi, sebagian tahapan serangan dalam *attack chain* pada dasarnya tidak memiliki *payload* atau pola eksploitasi yang unik, melainkan berupa aktivitas yang secara *native* menyerupai *traffic* normal, misalnya proses *login* SSH menggunakan kredensial yang valid, koneksi ke *share* SMB, atau permintaan HTTP biasa ke direktori tertentu. *Custom rules* untuk tahapan semacam ini hanya dapat dirancang secara generik, misalnya berbasis koneksi ke port tertentu atau pola koneksi yang sederhana, tanpa dapat menyertakan *signature payload* yang spesifik [4]. Pendekatan generik ini berisiko menghasilkan *false positive* ketika *rule* diterapkan pada *traffic* jaringan yang lebih luas, karena pola yang dideteksi tidak secara eksklusif menjadi indikator serangan, sehingga setiap *custom rule* yang dikembangkan perlu diuji *false positive* rate-nya agar tetap layak digunakan tanpa menimbulkan *Alert* yang berlebihan.

Untuk menguji permasalahan *detection rate* dan *false positive rate* tersebut secara empiris pada konteks *attack chain* yang utuh, dibutuhkan *attack chain* yang terstruktur, terdokumentasi, dan dapat direplikasi pada lingkungan terkontrol. Kompetisi Mahasiswa Indonesia Politeknik Nasional (KMIPN) 2024 menyediakan kondisi ini melalui salah satu CTF-nya, yang menyajikan *attack chain* dengan tahapan serangan yang dapat dipetakan ke dalam teknik-teknik MITRE ATT&CK. Karena *walkthrough* dan tahapan serangannya telah terdokumentasi resmi, *attack chain* Amethysts dapat dijadikan objek pengujian yang representatif untuk mengevaluasi sejauh mana ET Open *ruleset* mampu mendeteksi setiap tahapannya, serta untuk merancang dan mengevaluasi *Custom rules* pada tahapan yang tidak terdeteksi. Berdasarkan permasalahan tersebut, penelitian ini bertujuan untuk mengidentifikasi celah deteksi pada *ruleset* ET Open terhadap *attack chain* CTF KMIPN 2024, merancang *Custom rules* Suricata yang spesifik untuk tahapan yang tidak terdeteksi, serta mengevaluasi *detection rate* gabungan antara ET Open dan *custom ruleset* beserta *false positive rate* dari *Custom rules* yang dikembangkan. Penelitian ini mereplikasi *attack chain* pada mesin Amethysts dalam lingkungan lab terkontrol menggunakan tiga *virtual machine*, dengan Suricata diimplementasikan sebagai NIDS dalam mode *passive* AF_PACKET pada *virtual machine* terpisah tanpa alamat IP, yang terhubung pada *virtual switch* yang sama dengan *virtual machine attacker* dan *target* serta dikonfigurasi dalam *mode promiscuous* agar dapat menyadap seluruh *traffic* yang melintas pada segmen jaringan tersebut. Hasil evaluasi divisualisasikan dalam bentuk *coverage map* yang dipetakan ke *framework* MITRE ATT&CK, sehingga penelitian ini diharapkan dapat memberikan kontribusi berupa gambaran keterbatasan *ruleset default* serta *custom ruleset* yang dapat digunakan untuk meningkatkan visibilitas Suricata terhadap *attack chain* yang terstruktur.

2. Metode

Penelitian ini merupakan penelitian terapan (*applied research*) dengan pendekatan eksperimental desain komparatif *before-after*. Variabel bebas yang dimanipulasi adalah konfigurasi *ruleset Intrusion Detection System* (IDS), sedangkan variabel terikat yang diamati adalah *detection rate* dan *false positive rate* dalam lingkungan yang terkontrol. Secara keseluruhan, penelitian ini terdiri dari tujuh tahapan utama yang disajikan pada Gambar 1.



Gambar 1 Alur Penelitian

2.1 Analisis Attack chain dan Pemetaan MITRE ATT&CK

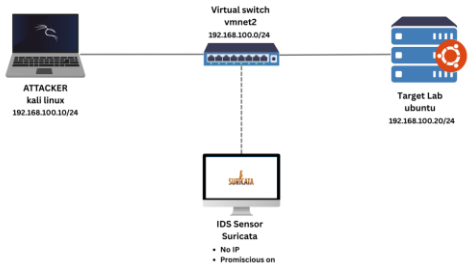
Attack chain pada mesin Amethysts CTF KMIPN 2024 terdiri dari 11 tahapan yang diidentifikasi berdasarkan *walkthrough* resmi kompetisi. Tahap *Upload Shell* dan *Credentials Leakage* masing-masing dijalankan dalam dua skenario: skenario (a) menggunakan *generic* PHP shell dengan parameter ?cmd= sebagai pembanding komparatif terhadap ET Open, dan skenario (b) menggunakan p0wny shell dengan parameter ?feature=shell sesuai walkthrough resmi sebagai skenario utama penelitian. Setiap tahapan dipetakan ke *framework* MITRE ATT&CK untuk memberikan konteks teknis terhadap jenis aktivitas yang dilakukan. Pemetaan ini menjadi acuan utama dalam menentukan target deteksi pada pengujian *baseline* maupun pada perancangan *custom rules*. Hasil pemetaan disajikan pada Tabel I.

Tabel I Pemetaan Tahapan *Attack chain* ke MITRE ATT&CK

No	Tahapan	Deskripsi	Teknik MITRE ATT&CK
1	*Port Scanning*	Pemindaian port dengan Nmap	T1046 - Network Service Discovery
2	SMB *Enumeration*	Enumerasi *share* SMB port *non-standar* 1445	T1135 - Network Share Discovery
3	RPC *Enumeration*	Enumerasi *username via* rpcclient port 1139	T1087.001 - Account Discovery: Local Account
4	FTP *Enumeration*	Akses & unduh berkas *via* FTP *anonymous*	T1078.001 - Valid Accounts: Default Accounts
5	*ZIP & Hash Cracking*	Brute force *Password* ZIP dan hash MD5	T1110.002 - Brute Force: Password Cracking
6	HTTP *Enumeration*	*Directory Enumeration* dengan Gobuster	T1595.003 - Active Scanning: Wordlist Scanning
7	*Bruteforcing*	Bruteforce *login* admin dengan Hydra	T1110.001 - Brute Force: Password Guessing
8a	*Upload Shell*(cmd)	Eksploitasi file *upload* RiteCMS dengan *Upload generic* PHP shell — skenario komparatif	T1190 - Exploit Public-Facing Application
8b	*Upload Shell*(p0wny)	Eksploitasi file *upload* RiteCMS dengan *Upload* p0wny shell — *walkthrough* resmi	T1190 - Exploit Public-Facing Application
9a	*Credentials Leakage* (cmd)	Akses .env *via* URL ?cmd= — skenario komparatif	T1552.001 - Unsecured Credentials: Credentials In Files
9b	*Credentials Leakage* (p0wny)	Akses .env *via* p0wny POST *body* — *walkthrough* resmi	T1552.001 - Unsecured Credentials: Credentials In Files
10	SSH *Login*	*Login* SSH menggunakan kredensial ditemukan	T1078.003 - Valid Accounts: Local Accounts
11	*Privilege escalation*	Eksploitasi Linux capabilities pada vim	T1548.001 - Abuse Elevation Control Mechanism: Setuid and Setgid

2.2 Perancangan Topologi dan Skema Pengujian

Lingkungan pengujian menggunakan tiga *virtual machine* pada VMware Workstation, yaitu VM *attacker* (Kali Linux), VM target (Ubuntu Server), dan VM IDS sensor (Suricata), yang seluruhnya terhubung pada *virtual switch* vmnet2 (192.168.100.0/24). Suricata dikonfigurasi tanpa alamat IP dalam *mode promiscuous* sebagai *sniffer pasif* NIDS . Topologi IDS dapat dilihat pada Gambar 2.



Gambar 2 Topologi IDS

Pengujian dirancang dalam dua kondisi: (1) kondisi *baseline* menggunakan ET Open *ruleset* saja, dan (2) kondisi gabungan menggunakan ET Open ditambah *custom rules*. Pada setiap kondisi, seluruh 11 tahapan *attack chain* dijalankan secara berurutan. Suatu tahapan dinyatakan terdeteksi apabila Suricata menghasilkan *Alert* yang sesuai selama rentang waktu eksekusi tahapan tersebut. Pengujian *false positive* dilakukan khusus terhadap *Custom rules* yang dibuat.

Tabel II Rancangan skenario pengujian

No	Tahapan	Tool	Kriteria Keberhasilan
1	*Port Scanning*	nmap	*Alert scan/sweep* muncul pada rentang waktu eksekusi nmap
2	*SMB Enumeration*	smbclient	*Alert* koneksi SMB port 1445 / akses *share* backup
3	*RPC Enumeration*	rpcclient	*Alert* koneksi RPC port *non-standar* 1139
4	*FTP Enumeration*	wget -r	*Alert login anonymous* dan unduh berkas via RETR
5	*ZIP & Hash Cracking*	zip2john +john dan crackstation	Tidak diuji pada *level* jaringan, dilakukan sepenuhnya *offline* di sisi *attacker* menggunakan *tools* lokal, tidak menghasilkan *traffic* yang dapat diinspeksi Suricata
6	HTTP *Enumeration*	gobuster	*Alert* User-Agent gobuster pada HTTP request
7	*Bruteforcing*	hydra	*Alert* User-Agent Hydra / POST volume tinggi
8a	*Upload Shel l*cmd (komparatif)	*Browser/curl (generic PHP web shell)*	*Alert* POST *body* mengandung <?php + exec (muncul saat upload file php)
8b	*Upload Shell* p0wny (*walkthrough* resmi)	*RiteCMS Admin – File Manager (upload p0wny shell)*	*Alert* URI mengandung feature=shell muncul saat akses p0wny shell
9a	*Credentials Leakage* cmd (komparatif)	*Browser/curl (parameter ? cmd=cat)*	*Alert* URI /.env + pattern cmd=cat muncul saat akses .env via URL
9b	*Credentials Leakage* p0wny (*walkthrough* resmi)	*p0wny shell (POST body request)*	*Alert* URI feature=shell + POST *body* mengandung .env
10	SSH *Login*	ssh	*Alert* koneksi TCP port 22 ke target
11	*Privilege escalation*	vim (python3 integration), getcap	Tidak diuji pada *level* jaringan, seluruh aktivitas dilakukan di dalam sesi SSH terenkripsi sehingga *command* yang dieksekusi tidak dapat diinspeksi oleh Suricata

Tahapan *ZIP Password & MD5 Hash Cracking* dilakukan sepenuhnya secara *offline* di sisi *attacker*, sehingga seluruh prosesnya tidak pernah melewati jaringan dan tidak menghasilkan *traffic* apapun yang dapat diinspeksi oleh Suricata. Sementara itu, tahap *Privilege escalation* menggunakan Linux capabilities pada vim berlangsung di dalam sesi SSH yang sudah terenkripsi, sehingga Suricata hanya dapat mengamati bahwa ada koneksi SSH yang aktif tanpa dapat membaca *command* yang dijalankan di dalamnya. Dengan demikian, kedua tahapan ini pada dasarnya berada di luar kemampuan deteksi *Network Intrusion Detection System* (NIDS).

2.3 Implementasi Infrastruktur Pengujian

Membangun dan mengonfigurasi lingkungan pengujian sesuai rancangan, serta melakukan instalasi dan konfigurasi Suricata sebagai sensor IDS.

2.4 Pengujian *Baseline* dengan ET Open

Seluruh tahapan *attack chain* dijalankan menggunakan *ruleset* ET Open sebagai konfigurasi awal. Hasil pengujian digunakan untuk mengetahui kemampuan deteksi bawaan Suricata terhadap skenario serangan yang diuji.

2.5 Pengembangan *Custom rules*

Pengembangan *Custom rules* pada penelitian ini dilakukan berdasarkan analisis *packet capture* pada tahapan *attack chain* yang tidak terdeteksi ET Open, dengan pendekatan deteksi *payload -specific* maupun *generic-connection/port* sesuai karakteristik *traffic* masing-masing tahapan.

2.6 Pengujian Gabungan ET Open dan Custom Rules

Setelah *Custom rules* selesai dibuat, seluruh skenario serangan dijalankan kembali menggunakan kombinasi ET Open dan *custom rules*. Pengujian ini bertujuan untuk mengevaluasi peningkatan kemampuan deteksi yang diperoleh.

2.7 Analisis dan Penyajian Hasil

Hasil pengujian dianalisis menggunakan dua pendekatan yaitu Matriks Visibilitas atau Cakupan (*Coverage/Visibility Map*) untuk mengevaluasi capaian deteksi terhadap *attack chain*, dan *false positive rate* untuk mengevaluasi *Custom rules* yang dikembangkan. *Coverage Map* memetakan setiap tahapan attack chain ke teknik MITRE ATT&CK yang bersesuaian, beserta status visibilitasnya pada level jaringan dan status deteksinya pada kondisi *baseline* (ET Open) maupun kondisi gabungan (ET Open + *Custom rules*). Pendekatan ini memberikan gambaran yang lebih objektif, karena turut menunjukkan teknik mana yang pada dasarnya berada di luar jangkauan visibilitas NIDS akibat aktivitasnya yang lokal pada host atau berlangsung dalam sesi yang terenkripsi. False positive rate pada penelitian ini dihitung menggunakan Persamaan [5]:

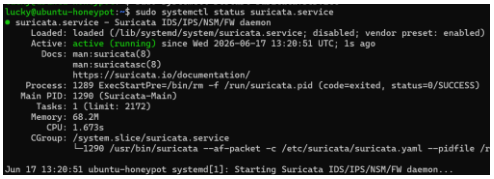
$$FPR = \frac{FP}{(FP + TN)} \quad (1)$$

dengan FP (*False Positive*) adalah jumlah *traffic* normal yang keliru menghasilkan *Alert*, dan TN (*True Negative*) adalah jumlah *traffic* normal yang secara benar tidak menghasilkan *Alert*, sehingga FP+TN merepresentasikan total *traffic* normal yang diuji untuk masing-masing *Custom rule*. Suatu *Custom rule* dinyatakan menghasilkan *false positive* apabila *rule* tersebut men-*trigger Alert* ketika dihadapkan pada *traffic* normal yang tidak berkaitan dengan aktivitas serangan.

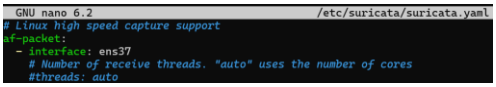
3. Hasil dan Pembahasan

3.1 Implementasi Infrastruktur Pengujian

Suricata berhasil diimplementasikan sesuai rancangan topologi. Suricata dikonfigurasi pada *file* suricata.yaml dengan *interface* jaringan dalam *mode* AF_PACKET dan *promiscuous mode* diaktifkan. Verifikasi awal dilakukan melalui validasi konfigurasi (suricata -T) dan pemantauan berkas log eve.json. Hasil konfigurasi ditunjukkan pada Gambar 3 dan 4.



Gambar 3 *Status Service* Suricata



Gambar 4 Konfigurasi AF-PACKET pada suricata.yaml

3.2 Hasil Pengujian *Baseline* (ET Open Ruleset)

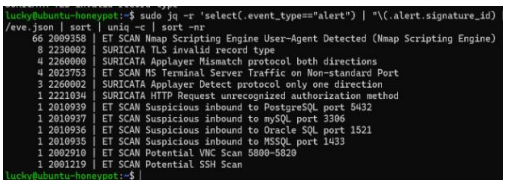
Pengujian *baseline* dilakukan dengan menjalankan seluruh 11 tahapan *attack chain* secara berurutan dengan Suricata hanya menggunakan ET Open *ruleset*. *Alert* yang dihasilkan dicatat berdasarkan rentang waktu eksekusi masing-masing tahapan. Rekapitulasi hasil deteksi ET Open pada seluruh tahapan *attack chain* disajikan pada Tabel III.

Tabel III Hasil Deteksi Baseline ET Open per Tahapan Attack chain

Tahapan	Status ET Open	SID *Alert*	Keterangan
Port Scanning	✓ Terdeteksi	2001219, 2002910, 2010935, 2010936,	ET Open memiliki *rule* deteksi Nmap SYN scan / port sweep.

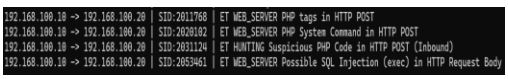
		2010937, 2010939, 2023753, 2009358.	
SMB Enumeration	✗ Tidak Terdeteksi	-	Port *non-standar* 1445 tidak tercakup *rule* ET Open
RPC Enumeration	✗ Tidak Terdeteksi	-	Port *non-standar* 1139 tidak tercakup *rule* ET Open
FTP Enumeration	✗ Tidak Terdeteksi	-	FTP anonymous pada port standar 21 tidak menghasilkan *alert*
ZIP & Hash Cracking	✗ Tidak Terdeteksi	-	dilakukan sepenuhnya *offline* di sisi *attacker* menggunakan *tools* lokal, tidak menghasilkan *traffic* yang dapat diinspeksi Suricata.
HTTP *Enumeration*	✗ Tidak Terdeteksi	-	*User-Agent* gobuster tidak tercakup *rule* ET Open *baseline.*
Bruteforcing	✗ Tidak Terdeteksi	-	*HTTP POST flood* tidak menghasilkan *alert*.
Upload Shell cmd (komparatif)	✓ Terdeteksi	2011768, 2020102, 2010920.	ET Open mendeteksi aktivitas *upload web shell* PHP serta upaya eksekusi perintah sistem pada aplikasi web melalui fungsi PHP dan *parameter command injection* (cmd=)
Upload Shell p0wny (*walkthrough* resmi)	✓ Terdeteksi	2011768, 2020102, 2031124, 2053461.	ET Open mendeteksi aktivitas *upload web shell* PHP dan upaya eksekusi perintah pada server web melalui fungsi PHP berbahaya seperti system() dan exec() yang dikirim melalui HTTP POST.
Credentials Leakage cmd (komparatif)	✓ Terdeteksi	2031502	ET Open memiliki *rule* deteksi akses file .env
Credentials Leakage p0wny (*walkthrough* resmi)	✗ Tidak Terdeteksi	-	ET Open tidak mendeteksi aktivitas eksekusi di dalam *upload web* PHP p0wny
SSH *Login*	✗ Tidak Terdeteksi	-	Koneksi SSH normal tidak dideteksi secara *default.*
Privilege escalation	✗ Tidak Terdeteksi	-	seluruh aktivitas dilakukan di dalam sesi SSH terenkripsi sehingga *command* yang dieksekusi tidak dapat diinspeksi oleh Suricata

** Data SID ET Open pada kolom 'SID Alert' diisi berdasarkan output eve.json yang dihasilkan selama pengujian berlangsung.*



Gambar 5 *Alert* ET Open pada *Port Scanning*

Gambar 6 *Alert* ET Open pada *Upload Shell*



Gambar 7 *Alert* ET Open pada *Upload* Shell p0wny

Gambar 8 *Alert* ET Open pada *Credentials Leakage*

Dari seluruh rangkaian *attack chain* yang diuji, ET Open hanya berhasil mendeteksi tahapan *Port Scanning, Upload Shell* pada kedua variasi *payload* (cmd dan p0wny), serta *Credentials Leakage* pada skenario komparatif yang menggunakan *parameter ?cmd=*. Gambar 5, 6, 7, dan 8 menampilkan *Alert* yang dihasilkan ET Open pada masing-masing skenario tersebut. Kegagalan deteksi ET Open pada skenario lainnya, termasuk *Credentials Leakage* melalui p0wny shell, disebabkan oleh tiga faktor utama. Pertama, penggunaan *port non-standar*: ET Open mengasumsikan layanan SMB berjalan pada port 445 dan RPC pada port 139, sehingga penggunaan port 1445 dan 1139 pada mesin target tidak tercakup oleh *rule* yang tersedia. Kedua, *traffic* yang menyerupai protokol *legitimate*: FTP anonymous menggunakan perintah standar protokol FTP sehingga tidak mengandung *pattern* yang dianggap berbahaya oleh ET Open. Ketiga, *tool* modern dengan *signature* yang belum dikenal: akses *Credentials Leakage* melalui p0wny shell menggunakan *parameter ?feature=shell*, berbeda dari *parameter ?cmd=* yang telah dikenal ET Open, sehingga meskipun aktivitas *upload web shell p0wny* itu sendiri terdeteksi (melalui *rule generik web shell PHP*), aktivitas pembacaan file .env yang dilakukan setelahnya melalui parameter tersebut tidak dapat terdeteksi. Celah-celah inilah yang menjadi dasar pengembangan *Custom rules* pada tahap selanjutnya.

3.3 Pengembangan Custom rules

3.3.1 Struktur Penulisan Rule Suricata

Berdasarkan dokumentasi resmi Suricata, setiap rule terdiri dari tiga komponen utama: action, header, dan rule options [6]. Berikut adalah format umum penulisan rule Suricata:

```
action protokol ip_sumber port_sumber -> ip_tujuan port_tujuan (rule options;)
```

berikut contoh rule dari dokumentasi resmi Suricata [6]:

```
Alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"HTTP GET Request Containing Rule in URI"; flow:established,to_server; http.method; content:"GET"; http.uri; content:"rule"; fast_pattern; classtype:bad-unknown; sid:123; rev:1;)
```

Action menentukan apa yang dilakukan Suricata ketika rule cocok. Seluruh *custom rule* pada penelitian ini menggunakan *action Alert* karena Suricata beroperasi dalam *mode NIDS pasif*, hanya mendeteksi dan mencatat, tidak memblokir *traffic*. *Header* mendefinisikan protokol, IP sumber, port sumber, arah *traffic*, IP tujuan, dan port tujuan. *Rule options* berisi seluruh logika deteksi, yang diapit tanda kurung dan diakhiri tanda titik koma.

3.3.2 Analisis *Packet Capture*

Analisis *packet capture* menggunakan Wireshark dilakukan pada setiap tahapan yang tidak terdeteksi ET Open. Hasil analisis mengidentifikasi karakteristik *traffic* unik pada masing-masing tahapan sebagai dasar pembuatan *rule*.

3.3.3 *Custom rules* Final

Berdasarkan hasil analisis pcap, 10 *Custom rules* dikembangkan dan didokumentasikan. Setiap *draft rule* divalidasi sintaksnya dengan suricata -T dan diuji sebelum diterapkan ke *environment* utama. Berikut ringkasan *rule final*, beserta penjelasan penulisan setiap *rule*.

a) SID 9000001 — *SMB Enumeration*

```
alert smb any any -> $HOME_NET 1445 (msg:"AMETHYSTS [SMB] Backup"; flow:to_server,established; smb.share; content:"backup"; nocase; endswith; threshold:type limit, track by_src, count 1, seconds 60; classtype:attempted-recon; sid:9000001;)
```

Action Alert dipilih karena Suricata beroperasi pasif. *Header* menggunakan protokol smb dengan port tujuan 1445 untuk menangkap *traffic* spesifik SMB *non-standar*. *Sticky buffer smb.share* membatasi pencarian *content* hanya pada *field* nama *share SMB*, sehingga *content:"backup"* hanya cocok jika nama *share* yang diakses mengandung *string 'backup'*.

b) SID 9000002 — *RPC Enumeration*

```
alert tcp any any -> $HOME_NET 1139 (msg:"[SMB] Connection to Non-Standard RPC Port 1139"; flow:to_server,established; threshold:type threshold, track by_src, count 1, seconds 60; classtype:attempted-recon; sid:9000002;)
```

Protokol tcp digunakan karena RPC via rpcclient berjalan di atas TCP. Port tujuan 1139 adalah port *non-standar* RPC yang digunakan mesin Amethysts. *Rule* ini tidak memerlukan *content matching* karena karakteristik utama yang dideteksi adalah koneksi ke port *non-standar* itu sendiri, bukan *payload*

spesifik.

c) SID 9000003, 9000004, 9000005 — *FTP Enumeration*

```
# SID 9000003 — Anonymous FTP Login
alert ftp any any -> $HOME_NET 21 (msg:"AMETHYSTS [FTP] Anonymous Login Detected"; flow:to_server,established; ftp.command; content:"USER"; ftp.command_data; content:"anonymous"; nocase; threshold:type limit, track by_src, count 1, seconds 60; classtype:attempted-recon; sid:9000003;)

#SID 9000004 — FTP Recursive Download
alert ftp any any -> $HOME_NET 21 (msg:"AMETHYSTS [FTP] Possible Recursive File Download (RETR)"; flow:to_server,established; ftp.command; content:"RETR"; classtype:attempted-recon; sid:9000004;)

# SID 9000005 — Deteksi download file backup
alert ftp-data any any -> any any (msg:"AMETHYSTS [FTP] Backup File Transfer Detected"; flow:to_client,established; file.name; content:"backup"; nocase; classtype:policy-violation; sid:9000005;)
```

SID 9000003 menggunakan dua *sticky buffer* secara berurutan: *ftp.command* untuk mencocokkan perintah FTP (USER), dan *ftp.command_data* untuk mencocokkan argumen perintah tersebut (*anonymous*). Kombinasi keduanya memastikan *rule* hanya ter-*trigger* pada *login FTP anonymous*, bukan *login* dengan *username* lain. SID 9000004 hanya memantau keberadaan perintah RETR pada sesi FTP. SID 9000005 menggunakan protokol ftp-data (bukan ftp) dengan arah to_client karena *transfer file FTP* terjadi pada koneksi data terpisah dari koneksi kontrol, dan arah *traffic* data adalah dari server ke *client. Sticky buffer file.name* memungkinkan pencocokan pada nama *file* yang ditransfer.

d) SID 9000006 — *HTTP Enumeration* / Gobuster

```
alert http any any -> $HOME_NET any (msg:"AMETHYSTS [HTTP] Directory Enumeration - Gobuster Detected"; flow:to_server,established; http.user_agent; content:"gobuster"; nocase; threshold:type limit, track by_src, count 1, seconds 60; classtype:web-application-attack; sid:9000006;)
```

Sticky buffer http.user_agent membatasi pencarian *content* hanya pada *header User-Agent HTTP request*. *String* 'gobuster' sangat spesifik terhadap *tool* Gobuster sehingga risiko *false positive* sangat rendah.

e) SID 9000007— *Bruteforcing*

```
alert http any any -> $HOME_NET any (msg:"AMETHYSTS [HTTP] Brute Force Tool - Hydra Detected"; flow:to_server,established; http.user_agent; content:"Hydra"; nocase; threshold:type limit, track by_src, count 3, seconds 30; classtype:web-application-attack; sid:9000007;)
```

SID 9000007 mendeteksi Hydra melalui *User-Agent* yang mengandung 'Hydra'.

f) SID 9000008, 9000009 — *Upload Shell& Credentials Leakage via p0wny*

```
# SID 9000008 — P0wny Shell Access Detection
alert http any any -> $HOME_NET any (msg:"AMETHYSTS [HTTP] P0wny Shell Command Execution Detected"; flow:to_server,established; http.uri; content:"feature=shell"; nocase; threshold:type limit, track by_src, count 2, seconds 20; classtype:web-application-attack; sid:9000008;)

# SID 9000009 — Credentials Leakage via P0wny (.env)
alert http any any -> $HOME_NET any (msg:"AMETHYSTS [HTTP] Sensitive File .env Access via Shell"; flow:to_server,established; http.uri; content:"feature=shell"; nocase; http.request_body; content:".env"; nocase; threshold:type limit, track by_src, count 1, seconds 60; classtype:credential-theft; sid:9000009;)
```

SID 9000008 mendeteksi akses ke *p0wny shell* melalui parameter *feature=shell* pada URI. SID 9000009 menggunakan kombinasi dua *sticky buffer*: http.uri untuk memastikan *request* berasal dari konteks *p0wny shell*, dan *http.request_body* untuk mendeteksi string '.env' pada *body POST* yang dikirimkan saat *attacker* mengeksekusi perintah *cat* terhadap *file .env*.

g) SID 9000010 — SSH *Login*

```
alert ssh any any -> $HOME_NET 22 (msg:"AMETHYSTS [SSH] Inbound SSH Connection"; flow:to_server,established; threshold:type limit, track by_src, count 1, seconds 60; classtype:attempted-recon; sid:9000010;)
```

Protokol ssh digunakan agar Suricata memproses *rule* ini khusus pada *layer* SSH. Port tujuan 22 adalah port SSH *standar*. *Rule* ini tidak memiliki *content matching* karena koneksi SSH terenkripsi sehingga *payload* tidak dapat diinspeksi; deteksi hanya berdasarkan keberadaan koneksi SSH itu sendiri. *Threshold type limit, count 1,*

seconds 60 membatasi 1 *Alert* per IP sumber per 60 detik.

3.4 Hasil Pengujian Gabungan (ET Open + *Custom Rule*s)

Setelah seluruh Custom rules diimplementasikan, pengujian ulang dilakukan terhadap seluruh tahapan attack chain. Rekapitulasi hasil pengujian gabungan disajikan pada Tabel IV.

Tabel IV Hasil Deteksi Gabungan ET Open + Custom Rules

Tahapan	**Status ET Open + Custom Rules**	**Sumber Deteksi**
Port Scanning	✓ Terdeteksi	ET Open
SMB Enumeration	✓ Terdeteksi	Custom Rule SID 9000001
RPC Enumeration	✓ Terdeteksi	Custom Rule SID 9000002
FTP Enumeration	✓ Terdeteksi	Custom Rule SID 9000003, 9000004, 9000005
ZIP & Hash Cracking	Tidak diuji pada level jaringan	–
HTTP *Enumeration*	✓ Terdeteksi	Custom Rule SID 9000006
Bruteforcing	✓ Terdeteksi	Custom Rule SID 9000007
Upload Shell cmd (komparatif)	✓ Terdeteksi	ET Open
Upload Shell p0wny (*walkthrough* resmi)	✓ Terdeteksi	Custom Rule SID 9000008
Credentials Leakage cmd (komparatif)	✓ Terdeteksi	ET Open
Credentials Leakage p0wny (*walkthrough* resmi)	✓ Terdeteksi	Custom Rule SID 9000009
SSH *Login*	✓ Terdeteksi	Custom Rule SID 9000010
Privilege escalation	Tidak diuji pada level jaringan	–

Berdasarkan Tabel IV, seluruh *attack chain* yang dapat diamati pada level jaringan berhasil terdeteksi setelah kombinasi ET Open dan *Custom rules* diterapkan. Dua tahapan lainnya (ZIP & Hash Cracking dan Privilege escalation) tetap berada di luar jangkauan visibilitas NIDS sebagaimana dijelaskan pada Tabel II.

3.5 Analisis dan Penyajian Hasil

3.5.1 Analisis *detection rate* pada pengujian *Rules* ET Open dan *Rules* gabungan ET Open dengan *Custom Rules*.

Kemampuan deteksi Suricata dievaluasi menggunakan Matriks Visibilitas/Cakupan (*Coverage Map*) sebagaimana ditunjukkan pada Tabel V. Matriks ini memetakan status deteksi setiap tahapan serangan (hasil pengujian pada Tabel III dan Tabel IV) terhadap teknik MITRE ATT&CK yang bersesuaian.

Tabel V Matriks Visibilitas/Cakupan Deteksi terhadap MITRE ATT&CK

Tahapan	**Teknik MITRE ATT&CK**	**ET Open**	**Gabungan (ET Open + Custom Rules)**	**Status Visibilitas**
Port Scanning	T1046	✓	✓	Terdeteksi ET Open
SMB Enumeration	T1135	✗	✓	Terdeteksi setelah Custom Rules
RPC Enumeration	T1087.001	✗	✓	Terdeteksi setelah Custom Rules
FTP Enumeration	T1078.001	✗	✓	Terdeteksi setelah Custom Rules
ZIP & Hash Cracking	T1110.002	–	–	Di luar jangkauan NIDS
HTTP *Enumeration*	T1595.003	✗	✓	Terdeteksi setelah Custom Rules
Bruteforcing	T1110.001	✗	✓	Terdeteksi setelah Custom Rules
*Upload Shell*cmd (komparatif)	T1190	✓	✓	Terdeteksi ET Open
Upload Shell p0wny	T1190	✓	✓	Terdeteksi ET Open

(*walkthrough* resmi)				
Credentials Leakage cmd (komparatif)	T1552.001	✓	✓	Terdeteksi ET Open
Credentials Leakage p0wny (*walkthrough* resmi)	T1552.001	✗	✓	Terdeteksi setelah Custom Rules
SSH *Login*	T1078.003	✗	✓	Terdeteksi setelah Custom Rules
Privilege escalation	T1548.001	–	–	Di luar jangkauan NIDS

Berdasarkan Tabel V, tahapan *ZIP & Hash Cracking dan Privilege escalation* tidak dapat diamati pada level jaringan. Pada tahapan-tahapan yang dapat diamati pada level jaringan, konfigurasi *baseline* ET Open hanya mencakup *Port Scanning, Upload Shell cmd dan p0wny, serta Credentials Leakage cmd*, pola serangan generik yang telah dikenali secara luas oleh *signature* bawaan ET Open. Pengelompokan ini menunjukkan bahwa keterbatasan ET Open bersifat sistematis, yaitu terpusat pada tahapan-tahapan yang menggunakan *port non-standar* atau menggunakan *tools* dengan *signature* belum dikenal. *Custom rules* yang dikembangkan berhasil menutup seluruh celah pada kelompok tahapan tersebut, sehingga seluruh tahapan yang dapat diamati pada level jaringan berhasil dideteksi.

3.5.2 Pengujian *False positive Custom Rules*.

Pengujian *false positive* dilakukan pada setiap *Custom rule* terhadap *traffic normal.* Dengan total 2000 percobaan *traffic* normal per *rule* (mengacu pada rumus ukuran sampel Cochran (1977) dengan *margin error* ≈2% untuk memastikan hasil *false positive rate* yang diperoleh representatif secara *statistic* [7]. Hasil pengujian per *Custom rule* disajikan pada Tabel VI.

Tabel VI Hasil Pengujian *False Positive* per *Custom Rule*

SID	Tahapan	FPR
9000001	SMB Enumeration	27,59%
9000002	RPC Enumeration	100%
9000003	FTP Anonymous Login	0%
9000004	FTP RETR Generik	100%
9000005	FTP Backup Filename	38,04%
9000006	HTTP Enumeration / Gobuster	0%
9000007	Bruteforce Hydra User-Agent	0%
9000008	P0wny Shell Command Execution	9,90%
9000009	Credential Leakage .(.env via p0wny)	12,55%
9000010	*SSH Login*	100%

Berdasarkan Tabel VI, hasil pengujian dapat dikelompokkan menjadi tiga kategori berdasarkan tingkat spesifisitas *rule*: Pertama, *rule* tanpa *content matching* sama sekali, hanya mensyaratkan koneksi/*command* ke port tertentu (SID 9000002 RPC, SID 9000004 FTP RETR, SID 9000010 SSH) secara konsisten menghasilkan FPR 100%, karena secara struktural tidak dapat membedakan *traffic* normal dari *traffic* serangan. Kedua, *rule* dengan *content matching* pada *string* unik yang jarang muncul pada *traffic* normal (SID 9000003 *anonymous login*, SID 9000006 *User-Agent gobuster*, SID 9000007 *User-Agent Hydra*) menghasilkan FPR 0%, karena pola yang dicocokkan bersifat sangat spesifik terhadap *tools* serangan dan tidak tumpang tindih dengan pola *traffic* normal apapun.

Ketiga, *rule* dengan *content matching* yang secara sintaksis benar namun tidak dibatasi posisinya (SID 9000001 SMB tanpa proteksi *endswith* yang cukup, SID 9000005 FTP filename, SID 9000008 URI *feature=shell* tanpa *endswith*, serta SID 9000009 yang mensyaratkan kombinasi URI *feature=shell* dan *request body* mengandung .env) menghasilkan FPR pada rentang 9,90%–38,05%, karena pola yang dicocokkan juga muncul pada penamaan file/folder/fitur yang digunakan pada *traffic* normal. Khusus SID 9000010, FPR 100% juga berkaitan dengan interaksi *rule* tersebut dengan tahap *Port Scanning*: pemindaian *service version* menggunakan nmap turut menyelesaikan *handshake* SSH pada port 22,

sehingga *rule* berbasis koneksi murni ini tidak hanya rentan terhadap *traffic* normal, tetapi juga tidak dapat membedakan tahapan *attack chain* yang berbeda-beda, karena satu-satunya sinyal yang digunakan adalah keberadaan koneksi ke port tersebut.

3.5.3 Pembahasan Hasil

a). Efektivitas *Rule* Berdasarkan Spesifisitas *Content Matching*

Hasil pengujian *false positive rate* pada Tabel VI menunjukkan bahwa efektivitas *Custom rules* dipengaruhi langsung oleh tingkat spesifisitas *content matching* yang diterapkan. *Rule* dengan *content matching* pada *string* unik terhadap *tools* serangan (SID 9000003, 9000006, 9000007) mencapai FPR 0%, sedangkan *rule* berbasis koneksi murni tanpa *content matching* (SID 9000002, 9000004, 9000010) secara konsisten menghasilkan FPR 100%. Hal ini sesuai dengan prinsip *best practice* penulisan *signature* IDS yang menekankan spesifisitas konten sebagai penentu utama akurasi *rule* [8][9]. *Rule* generik tetap diperlukan untuk mendeteksi tahapan yang tidak memiliki artefak jaringan unik, namun idealnya disertai mekanisme tambahan seperti *whitelist IP* atau *threshold* yang lebih ketat untuk meminimalkan *false positive* pada implementasi produksi.

b.) Keterbatasan NIDS Berbasis *Signature*

Dua tahapan yang tidak dapat dideteksi (*Hash Cracking dan Privilege escalation*) menggarisbawahi batas fundamental NIDS: aktivitas yang tidak menghasilkan *traffic* jaringan tidak dapat dideteksi melalui pendekatan ini [10]. Untuk menutup celah ini diperlukan pendekatan komplementer seperti HIDS, *audit log*, atau EDR yang mampu menganalisis aktivitas sistem operasi secara langsung [11].

4. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, diperoleh kesimpulan utama sebagai berikut:

1. ET Open *ruleset* dalam konfigurasi *default* memiliki celah deteksi attack chain mesin Amethysts CTF KMIPN 2024. Pada tahapan yang dapat diamati pada level jaringan, ET Open hanya mampu mendeteksi *Port Scanning, Upload Shell, dan Credentials Leakage cmd* yang merupakan pola serangan generik dan telah dikenali *signature* bawaannya. Tahapan lain tidak terdeteksi karena tiga karakteristik yang berbeda: SMB Enumeration dan RPC Enumeration berjalan pada port layanan yang *non-standar*; *FTP Enumeration (Anonymous Login)* dan SSH *Login* menggunakan protokol secara sah tanpa pola eksploitasi yang dapat dibedakan dari aktivitas normal; sedangkan *HTTP Enumeration, Bruteforcing, serta Credentials Leakage varian p0wny* menggunakan *tools* yang *signature*-nya belum dikenali ET Open.
2. *Custom rules* yang dikembangkan berhasil menutup seluruh celah deteksi pada tahapan-tahapan tersebut, sehingga seluruh tahapan yang dapat diamati pada level jaringan berhasil tercakup. Sementara itu, *ZIP Hash Cracking dan Privilege escalation* tetap berada di luar jangkauan deteksi NIDS, karena aktivitasnya bersifat lokal pada host atau berlangsung dalam sesi yang terenkripsi.
3. Pengujian false positive terhadap traffic normal menunjukkan bahwa efektivitas Custom rules bergantung pada spesifisitas content matching yang diterapkan.

Untuk penelitian selanjutnya, disarankan untuk mengintegrasikan pendekatan NIDS dengan *Host-based IDS (HIDS)* guna menutup celah deteksi pada aktivitas lokal dan terenkripsi, menerapkan perbaikan presisi *rule* pada rule yang teridentifikasi kurang spesifik, serta melakukan validasi lebih lanjut pada *traffic* produksi nyata untuk mengonfirmasi konsistensi *false positive rate* pada skala yang lebih besar.

Ucapan Terima Kasih

Penulis mengucapkan terima kasih kepada kedua orang tua atas doa dan dukungan moral yang senantiasa diberikan selama proses penelitian berlangsung. Ucapan terima kasih juga disampaikan kepada Ibu Maidel Fani, S.Pd., M.Kom., selaku dosen

pembimbing, atas arahan, bimbingan, dan masukan yang sangat berharga dalam penyusunan penelitian ini. Penghargaan yang sama diberikan kepada seluruh dosen Program Studi Rekayasa Keamanan Siber Politeknik Negeri Batam atas ilmu yang telah diberikan.

Referensi

[1] Proofpoint. (2024). Emerging Threats Open *Ruleset*. https://*rules*.emergingthreats.net

[2] Kern, M. et al. (2025). *StealthCup*: Realistic, Multi-Stage, Evasion-Focused CTF for Benchmarking IDS. arXiv:2511.17761v1 [cs.CR]. https://doi.org/10.48550/arXiv.2511.17761

[3] MITRE Corporation. (2023). MITRE ATT&CK *Framework* . https://attack.mitre.org

[4] Teuwen, K. T. W., Mulders, T., Zambon, E., & Allodi, L. (2025). Ruling the Unruly: Designing Effective, Low-Noise Network Intrusion Detection Rules for Security Operations Centers. Proceedings of the ACM Asia Conference on Computer and Communications Security (ASIA CCS '25). https://doi.org/10.1145/3708821.3710823

[5] Rose, J., Swann, M., Bendiab, G., Shiaeles, S., & Kolokotronis, N. (2021). Intrusion Detection using Network Traffic Profiling and Machine Learning for IoT. 2021 IEEE 7th International Conference on Network Softwarization (NetSoft), 409–415. https://ieeexplore.ieee.org/document/9492685/

[6] Suricata Foundation. (2024). Suricata *Rule* Writing Documentation. https://docs.suricata.io/en/latest/*rules*/index.html

[7] Cochran, W. G. (1977). Sampling Techniques (3rd ed.). John Wiley & Sons.

[8] GBHackers. (2025). Writing Effective Detection *Rules* With Sigma, YARA, and Suricata. https://gbhackers.com/writing-effective-detection-*rules*-with-sigma-yara-and-suricata/

[9] Kim, Y., Lee, C., & Yoon, Y. (2025). *Payload* -Aware Intrusion Detection with CMAE and Large Language *Model*s. ACM Trans*action*s on Privacy and Security. https://doi.org/10.1145/3769682

[10] Chismon, D., & Ruks, M. (2015). *Intrusion Detection System*s. GIAC Security Essentials. https://www.giac.org/paper/gsec/235/limitations-network-intrusion-detection/100739

[11] Chen, Z., Simsek, M., Kantarci, B., Bagheri, M., & Djukic, P. (2023). Host-Based Network Intrusion Detection via Feature Flattening and Two-stage Collaborative Classifier. arXiv:2306.09451. https://doi.org/10.48550/arXiv.2306.09451