# Metode Cepat Identifikasi Dan Mitigasi Malware Ransomware Ketika Terjadi Serangan Siber

Anggrahito
Badan Siber dan Sandi Negara
Email: anggrahito@bssn.go.id

Ramadhan Ibrahim
Badan Siber dan Sandi Negara
Email:
ramadhan.ibrahim@bssn.go.id

Juliadi Satyo Pramudito
Badan Siber dan Sandi Negara
Email: Juliadi.satyo@bssn.go.id

*Abstrak*—**Saat ini perkembangan teknologi siber berkembang pesat. Pada penelitian ini membahas terkait pengamanan siber. Adapun serangan siber yang difokuskan pada penelitian ini adalah terkait malware. Permasalahan saat ini yang dihadapi ketika terdapat malware baru yang menginfeksi suatu sistem, infrastruktur, dan jaringan adalah sulitnya dilakukan identifikasi dan penanggulangan agar tidak tersebar lebih luas. Oleh karena itu penelitian ini ditujukan untuk mengembangan metode cepat pembuatan anti malware yang kemudian dapat digunakan untuk mengidentifikasi dan mitigasi sistem ketika terjadi serangan malware.**

***Kata kunci; malware, reverse engineering, yara, clamAV, bash, python.***

*Abstract*—**At present the development of cyber technology is growing rapidly. In this research, it is related to cyber security. The cyber attack that is focused on this research is related to malware. The current problem faced when there is new malware infecting a system, infrastructure, and network is the difficulty of identification and prevention so that it is not spread more widely. Therefore this research is aimed at developing a rapid method of making anti-malware which can then be used to identify and mitigate the system when a malware attack occurs.**

***Keywords; malware, reverse engineering, yara, clamAV, bash, python.***

## I. PENDAHULUAN

### A. *Latar Belakang*

perkembangan teknologi siber saat ini berkembang dengan pesat baik dari segi pengamanan ataupun serangan. Pesatnya perkembangan teknologi siber memicu munculnya teknologi – teknologi baru yang belum tentu tahan terhadap serangan siber. Pada tahun 2020, tiga dari sepuluh serangan siber yang terjadi merupakan ransomware kemudian disusul dengan Trojan, phishing, fleeceware, spyware, cryptojacking, dan AI attack [1]. Selain itu, terhitung pada tanggal 28 februari tahun 2020 sudah terdapat 22 jenis malware ransomware yang melakukan seranga siber [2]. Permasalahan berikutnya adalah ketika terdapat malware baru yang belum ada anti malwarenya dan kondisi mendesak agar segara mengidentifikasi dan mencegah malware tersebut tersebar lebih luas pada sistem. Berdasarkan permasalahan yang ada, penelitian ini bertujuan untuk mengembangan metode cepat pembuatan anti malware yang kemudian dapat digunakan untuk mengidentifikasi dan mitigasi sistem ketika terjadi serangan malware.

Untuk menjawab permasalahan diatas, maka dibutuhkan metode cepat dalam mengidentifikasi dan mitigasi malware ransomware ketika terjadi serangan siber. Pada penelitian ini, implementasi dilakukan dengan memanfaatkan beberapa tools berbasis Open Source yaitu radare2, yara, clamAV, python, dan bash. Oleh karena itu, dengan diimplementasikannya metode ini dapat mempercepat proses identifikasi dan penanggulangan malware ransomware ketika terjadi serangan siber.

### B. *Ruang Lingkup*

Pada penelitian ini, akan melakukan implementasi dan pengujian dalam bentuk *Proof of Concept* dalam skala lab. Pengujian dilakukan berdasarkan sepuluh sampel malware ransomware untuk kemudian dilakukan scaning dan membuktikan keakuratan hasil *scanning* berdasarkan rule yara yang telah dibuat.

### C. *Metode Penelitian*

Untuk melakukan pembuktian dan pengujian, penelitian ini menggunakan metode eksperimen. Pada TABEL I menunjukkan tahapan yang dilakukan pada penelitian ini [3].

TABEL I. TABEL PENGETAHUAN DAN SAINS

| Aspek | Tahap 1 | Tahap 2 | Tahap 3 |
|---|---|---|---|
| A. Pengetahuan | 1. Keyakinan | 2. Justifikasi | 3. Kebenaran |
| | ↓ | ↓ | ↓ |
| B. Sains | 1. Teori | 2. Eksperimen | 3. Kebenaran sains |
| | ↓ | | |
| | Pemodelan | | |
| | Engineering | | |

Bedasarkan TABEL I, terdapat tiga tahapan yang harus dipenuhi berdasarkan aspek pengetahuan dan sains. Pada penelitian ini pemodelan dilakukan menggunakan flowchart.

## II. LANDASAN TEORI

### A. *Malware*

Definisi malware atau *malicious software* (perangkat lunak berbahaya) adalah program kode yang digunakan penyerang untuk merusak atau menyalahgunakan sistem [4]. Terdapat berbagai manacam jenis malware, diantaranya Infectors, Network worms, Trojan horses, Backdoors, Remote-access Trojans, Information stealers, Ransomware, Scareware, Fakeware, Greyware. Pada

penelitian berfokus pada malware jenis ransomware. Malware jenis ransomware memiliki tujuan untuk menyimpan data atau akses ke sistem atau sumber daya untuk kemudian data tersebut di sandera kecuali korban membayar uang tebusan [5].

### B. *Reverse Engineering*

Definisi *Reverse Engineering* (rekayasa balik) adalah proses di mana objek buatan manusia didekonstruksi untuk mengungkapkan desain, arsitektur, atau untuk mengekstraksi pengetahuan dari objek [6]. Pada penelitian ini, proses *reverse engineering* dilakukan menggunakan tools radare2. Dipilihnya tools radare2 pada penelitian ini karena memiliki kemampuan untuk melakukan *disassemble* suatu *executable* maupun binaries untuk kemudian memunculkan informasi nilai hexa. Adanya nilai hexa tersebut dapat digunakan untuk membuat pattern pendeteksian malware dengan memanfaatkan yara dan clamav.

### C. *Yara*

YARA merupakan *tools* yang ditujukan untuk melakukan mengidentifikasi dan mengklasifikasikan sampel malware. Selain itu, YARA dapat membuat deskripsi keluarga malware berdasarkan pola tekstual atau biner.

Dipilihnya tools yara pada penelitian ini, karena yara memungkinkan pihak manapun untuk menciptakan rule sendiri sesuai kebutuhan. Penulisan rule pada YARA mudah untuk ditulis dan dipahami, dan serta memiliki sintaks yang menyerupai bahasa C [7].

### D. *clamAV*

ClamAV merupakan salah satu engine anti-virus yang berbasis open source (GPL) dan dapat digunakan dalam berbagai situasi termasuk pemindaian email, pemindaian web, dan keamanan pada endpoint. clamAV menyediakan sejumlah utilitas termasuk daemon multi-threaded yang fleksibel dan dapat diskalakan, pemindai berbasis *command*, dan otomatisasi pembaruan basis data [8]. Digunakannya clamAV pada penelitian ini karena medukung *scanning* malware dengan memanfaatkan rule Yara.

### E. *Python*

Python merupakan Bahasa pemrograman yang terlihat mudah namun memiliki kemampuan yang stabil dan tidak memakan memori yang cukup besar ketika dijalankan. Dengan menggunakan Bahasa pemrograman python dapat mempermudah seorang programmer untuk mengatasi masalah yang kompleks [9]. Pengguaan python pada penelitian ini karena memiliki library radare2 sehingga mendukung untuk melakukan parsing hasil ekstrasi nilai hexa pada *executable* atau binaries malware.

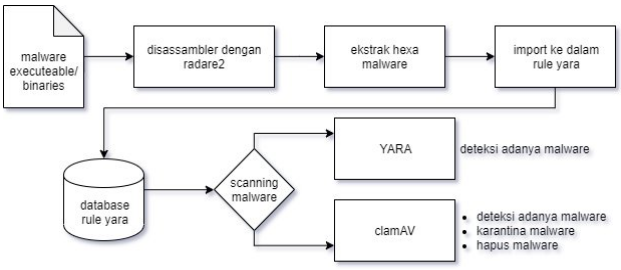
### F. *Bash*

*Shell Bourne-Again* lebih umum disebut bash adalah salah satu program di Proyek GNU yang merupakan shell paling banyak digunakan di lingkungan * NIX. *Bourne-Again shell* adalah pengembangan dari Bourne shell. Nama tersebut diambil dari penciptanya, yaitu Stephen Bourne. bash sendiri kemudian dikembangkan oleh Brian Fox [10]. Pengguaan bash pada penelitian ini adalah untuk melakukan regex processing dan otomatisasi serangkaian command linux.

## III. IMPLEMENTASI

### A. *Pemodelan*

Pada penelitian ini pemodelan menggunakan flowchart agar mudah dipahami. Berikut ini adalah flowchart metode cepat pembuatan anti malware yang akan diimplementasikan:




Gambar 1. flowchart metode cepat pembuatan anti malware

berdasarkan Gambar 1, tahapan yang dilakukan adalah sebagai berikut:

1. mendapatkan/ memiliki malware dalam bentuk *executable* ataupun binaries.
2. Melakukan proses disassamble terhadap malware tersebut sehingga dapat diketahui hexa asemmbly yang berjalan pada malware.
3. Melakukan ekstraksi nilai hexa malware ke dalam bentuk file txt.
4. Melakukan parsing nilai hexa pada file txt kedalam format rule yara.
5. Menyimpan rule yara satu file direktori.
6. Melakukan pembuktikan dan pengujian *scanning*, diantaranya menggunakan:
   a. Yara
   b. clamAV

### B. *Engineering*

Pada bagian ini akan menjelaskan penggunaan script yang sudah dikembangkan untuk memudahkan dalam pembuatan yara rule. Tahap pertama adalah menjalankan scirpt.sh menggunakan command `bash script.sh`. berikut ini adalah output yang ditampilkan ketikan menjalankan script.sh:

```
================================================================
total 15072
-rwxrwxrwx        ...        cerber.exe
-rw-rw-r--        ...        cryptowall.bin
-rw-rw-r--        ...        jigsaw
-rw-rw-r--        ...        Locky
-rw-rw-r--        ...        mamba.exe
-rw-rw-r--        ...        matsnu.exe
-rw-rw-r--        ...        petrwrap.exe
-rw-rw-r--        ...        radamant.exe
-rw-rw-r--        ...        satana.exe
-rw-rw-r--        ...        wannacry.exe

========== generate rule based on hexa====================
```

```
masukkan nama executebale :
wannacry.exe

masukkan nama output rule :
wannacry

generate rule yara
```

Berdasarkan contoh proses di atas, percobaan ini menggunakan ransomware wannacry. *Executable* yang digunakan adalah wannacry.exe dan output nama rule yang diinginkan adalah wannacry. Berdasarkan input yang dimasukkan pada script.sh, output dari hasil proses tersebut adalah wannacry.yar. Berikut ini adalah isi dari rule wannacry.yar:

```
rule wannacry : malaware_test

{
    meta:
        description = "This is just an example"
        threat_level = 3
        in_the_wild = true

    strings:
        $pattern1 = {FF 68 88}
        $pattern2 = {68 88 D4 40 00}
        $pattern3 = {00 68 F4}
            .
            .
            .
        $pattern50 = {00 8B 00 A3 54 F9}

    condition:
        $pattern1 and $pattern2 ...  $pattern50
}
```

Berdasarkan output di atas rule ini dinamakan wannacry. Kemudian meta pada rule merupakan informasi / penjelasan malware tersebut. **Meta** pada rule yara dapat dimodifikasi sesuai dengan kebutuhan. Pada rule yara, **strings** merupakan parameter untuk menentukan sebuah *executable* atau binaries termasuk malware atau bukan. **condition** pada rule yara digunakan untuk memproses strings yang sudah ditentukan.

Pada condition terdapat logika yang digunakan, yaitu **and** dan **or**. Penggunaan logika ini akan mempengaruhi hasil akhir *scanning* menggunakan rule yara. Ketika rule yara menggunakan logika and, maka sebuat *executable* atau binaries baru dapat dikatakan malware ketika memenuhi semua unsur string yang menjadi parameter. Saat menggunakan logika or, maka sebuat *executable* atau binaries baru dapat dikatakan malware ketika memenuhi salah satu unsur string yang menjadi parameter. Hasil dari setiap proses generate rule yara akan disimpan dalam satu folder / direktori **ruleYara**.

### C. Justifikasi

Pada bagian ini akan menjelaskan hipotesis penelitian. Adapun hipotesis yang muncul pada penelitian ini yaitu:

- Rule yara yang memanfaatkan informasi nilai hexa hasil disassamble *executable* dapat digunakan untuk scaning malware.
- Penggunaan logika and pada rule yara dapat meningkatkan akurasi dan mengurangi *false positif* ketika melakukan *scanning* malware.

### D. Eksperimen

Pada penelitian ini Eksperimen dilakukan dengan cara melakukan scannig terhadap sepuluh sample malware. Terdapat dua metode *scanning* yang akan dilakukan pada eksperimen ini, yaitu *scanning* menggunakan yara dan *scanning* menggunakan clamAV.

#### 1) Scanning menggunakan Yara

Berikut ini adalah salah satu contoh *scanning* menggunakan yara, command yang dijalankan adalah:

```
yara -s ruleYARA/wannacry.yar execute/wannacry.exe
```

output yang ditampilkan ketika rule tersebut digunakan untuk *scanning* oleh yara adalah sebagai berikut:

```
wannacry execute/wannacry.exe
0x77be:$pattern1: FF 68 88
0x77bf:$pattern2: 68 88 D4 40 00
0x14af:$pattern3: 00 68 F4
0x1902:$pattern3: 00 68 F4
        .
        .
        .
0x7819:$pattern48: B8 81 40 00 8B
0x781a:$pattern49: 81 40 00 8B 00 A3 54
0x781c:$pattern50: 00 8B 00 A3 54 F9
```

Berdasarkan output yang muncul, menunjukkan bahwa *scanning* berdasarkan wannacry.yar berhasil mendeteksi kecocokan nilai hexa dari setiap address wannacry.exe. untuk menampilkan kesimpulan hasil *scanning* menggunakan yara dapat dilakukan dengan commad berikut:

```
yara -m ruleYARA/wannacry.yar execute/wannacry.exe
```

output kesimpulan yang ditampilkan ketika rule tersebut digunakan untuk *scanning* oleh yara adalah sebagai berikut:

```
wannacry [description="This is just an
example",threat_level=3,in_the_wild=true]
execute/wannacry.exe
```

berdasarkan output yang ditampilkan, mendeskripsikan bahwa wannacry.exe berdasarkan rule wannacry teridentifikasi adalah sebagai ransomware wannacry. Hasil pengujian *scanning* dari 10 sample malware yang dilakukan berdasarkan hasil generate rule menggunakan script.sh ditunjukkan pada tabel di bawah ini:

TABEL II. TABEL *SCANNING* YARA

| No | Nama malware | Nama rule | Hasil *scanning* |
|---|---|---|---|
| 1 | cerber.exe | cerber.yar | Teridentifikasi |
| 2 | cryptowall.bin | cryptowall.yar | Teridentifikasi |
| 3 | jigsaw | jigsaw.yar | Teridentifikasi |
| 4 | Locky | Locky.yar | Teridentifikasi |
| 5 | mamba.exe | mamba.yar | Teridentifikasi |

| No | Nama malware | Nama rule | Hasil *scanning* |
|---|---|---|---|
| 6 | matsnu.exe | matsnu.yar | Teridentifikasi |
| 7 | petrwrap.exe | petrwrap.yar | Teridentifikasi |
| 8 | radamant.exe | radamant.yar | Teridentifikasi |
| 9 | satana.exe | satana.yar | Teridentifikasi |
| 10 | wannacry.exe | wannacry.yar | Teridentifikasi |

Bedasarkan TABEL II, dapat ditarik kesimpulan bahwa dari 10 sample rule yang digenerate berhasil menidentifikasi setiap malware ketika melakukan *scanning* menggunakan yara.

*2) Scanning menggunakan ClamAV*

ClamAV memiliki kelebihan ketika melakukan *scanning*, yaitu mampu melakukan *scanning* berdasarkan sebuah rule yara atau kumpulan rule yara yang sudah terdapat pada satu folder terhadap suatu folder. Berikut ini adalah salah satu contoh *scanning* menggunakan ClamAV berdasarkan satu buah rule yara terhadap sebuah folder, command yang dijalankan adalah:

```
clamscan -d ruleYARA/wannacry.yar execute/
```

output yang ditampilkan ketika rule tersebut digunakan untuk *scanning* oleh ClamAV adalah sebagai berikut:

```
execute/petrwrap.exe: OK
execute/mamba.exe: OK
execute/wannacry.exe: YARA.wannacry.UNOFFICIAL
FOUND
execute/cryptowall.bin: OK
execute/radamant.exe: OK
execute/jigsaw: OK
execute/satana.exe: OK
execute/Locky: OK
execute/matsnu.exe: OK
execute/cerber.exe: OK

----------- SCAN SUMMARY -----------
Known viruses: 1
Engine version: 0.102.2
Scanned directories: 1
Scanned files: 10
Infected files: 1
Data scanned: 21.32 MB
Data read: 7.49 MB (ratio 2.85:1)
Time: 0.330 sec (0 m 0 s)
```

berdasarkan output yang ditampilkan, hasil *scanning* menggunakan ClamAV berdasarkan satu buah rule yaitu rule wannacry.yar membuktikan bahwa dengan menggunakan logika **and** pada rule yara dapat mendeteksi malware secara spesifik pada suatu folder. Berikut adalah tabel hasil pengujian *scanning* menggunakan spesifik rule yara:

TABEL III. TABEL SPESIFIK *SCANNING* CLAMAV

| No | Nama malware | Nama rule | Hasil *scanning* |
|---|---|---|---|
| 1 | cerber.exe | - | Tidak Teridentifikasi |
| 2 | cryptowall.bin | - | Tidak Teridentifikasi |
| 3 | jigsaw | - | Tidak Teridentifikasi |
| 4 | Locky | - | Tidak Teridentifikasi |
| 5 | mamba.exe | - | Tidak Teridentifikasi |
| 6 | matsnu.exe | - | Tidak Teridentifikasi |
| 7 | petrwrap.exe | - | Tidak Teridentifikasi |
| 8 | radamant.exe | - | Tidak Teridentifikasi |
| 9 | satana.exe | - | Tidak Teridentifikasi |
| 10 | wannacry.exe | wannacry.yar | Berhasil Teridentifikasi |

Selanjutnya adalah pengujian *scanning* dari 10 sample malware yang dilakukan berdasarkan hasil generate rule dilakukan menggunakan command dibawah ini:

```
clamscan -d ruleYARA/ execute/
```

output yang ditampilkan ketika rule tersebut digunakan untuk *scanning* oleh ClamAV adalah sebagai berikut:

```
execute/petrwrap.exe: YARA.petrwrap.UNOFFICIAL FOUND
execute/mamba.exe: YARA.mamba.UNOFFICIAL FOUND
execute/wannacry.exe: YARA.wannacry.UNOFFICIAL FOUND
execute/cryptowall.bin: YARA.crypto_wall.UNOFFICIAL
FOUND
execute/radamant.exe: YARA.radamant.UNOFFICIAL FOUND
execute/jigsaw: YARA.jigsaw.UNOFFICIAL FOUND
execute/satana.exe: YARA.satana.UNOFFICIAL FOUND
execute/Locky: YARA.Locky.UNOFFICIAL FOUND
execute/matsnu.exe: YARA.matsnu.UNOFFICIAL FOUND
execute/cerber.exe: YARA.cerber.UNOFFICIAL FOUND

----------- SCAN SUMMARY -----------
Known viruses: 10
Engine version: 0.102.2
Scanned directories: 1
Scanned files: 10
Infected files: 10
Data scanned: 7.49 MB
Data read: 7.49 MB (ratio 1.00:1)
Time: 0.141 sec (0 m 0 s)
```

berdasarkan output yang ditampilkan, hasil *scanning* menggunakan ClamAV berdasarkan kumpulan rule yara yang sudah terdapat pada satu folder terhadap sebuah folder yaitu semua rule yara dapat mendeteksi malware secara spesifik pada suatu folder. Berikut adalah tabel hasil pengujian *scanning* berdasarkan kumpulan rule yara yang sudah terdapat pada satu folder terhadap sebuah folder:

TABEL IV. TABEL FOLDER *SCANNING* CLAMAV

| No | Nama malware | Nama rule | Hasil *scanning* |
|---|---|---|---|
| 1 | cerber.exe | cerber.yar | Teridentifikasi |
| 2 | cryptowall.bin | cryptowall.yar | Teridentifikasi |
| 3 | jigsaw | jigsaw.yar | Teridentifikasi |
| 4 | Locky | Locky.yar | Teridentifikasi |
| 5 | mamba.exe | mamba.yar | Teridentifikasi |
| 6 | matsnu.exe | matsnu.yar | Teridentifikasi |
| 7 | petrwrap.exe | petrwrap.yar | Teridentifikasi |
| 8 | radamant.exe | radamant.yar | Teridentifikasi |

| No | Nama malware | Nama rule | Hasil *scanning* |
|---|---|---|---|
| 9 | satana.exe | satana.yar | Teridentifikasi |
| 10 | wannacry.exe | wannacry.yar | Teridentifikasi |

## IV. KESIMPULAN DAN SARAN

Hasil dari *eksperiment* pada penelitian ini digunakan untuk membuktikan hipotesis yang terdapat pada bagian justifikasi.

Hipotesis pertama yaitu : Rule yara yang memanfaatkan informasi nilai hexa hasil disassamble *executable* dapat digunakan untuk scaning malware Terbukti dapat dilakukan berdasarkan hasil eksperimen *scanning* menggunakan Yara dan ClamAV.

Hipotesis kedua yaitu: Penggunaan logika and pada rule yara dapat meningkatkan akurasi dan mengurangi false positif ketika melakukan *scanning* malware Terbukti dapat dilakukan berdasarkan hasil eksperimen *scanning* menggunakan Yara dan ClamAV yang dijelaskan pada TABEL II dan TABEL III pada bagian eksperimen.

Proses mitigasi dapat dilakukan agar tidak tersebarnya malware ke sistem lebih luas, karena anti *malware* dapat segera dibuat dengan metode ini dan mudah untuk diimplementasikan pada sistem yang sudah ada.

Berdasarkan TABEL III, dari sepuluh sample *malware* yang diujikan, setiap *rule* YARA yang tergenerate hanya terfokus pada jenis *malware* yang diperuntukkan secara spesifik berdasarkan berdasarkan kecocokan nilai hexa dari setiap address dan tidak beririsan dengan *malware* yang lain, sehingga memungkinkan kecilnya terjadi *false positif* pada proses pendeteksian malware.

Dengan terbuktinya hipotesis pertama dan kedua membuktikan bahwa metode cepat dalam mengidentifikasi dan mitigasi malware ransomware ketika terjadi serangan siber dapat dilakukan dan output yang dihasilkan sesuai dengan yang diharapkan.

## REFERENCES


[1] Anderson. Sophie, “10 Latest (MOST DANGEROUS) Virus & Malware Threats in 2020”. https://www.safetydetectives.com/blog/most-dangerous-new-malware-and-security-threats/.2020.

[2] Abrams. Lawrence, “The Week in Ransomware - February 28th 2020 - Data Leaks Everywhere”. https://www.bleepingcomputer.com/news/security/the-week-in-ransomware-february-28th-2020-data-leaks-everywhere/. 2020.

[3] Frank A. Coutelieris, Kanavouras. Antonios, “Experimentation Methodology for Engineers”. Springer. 2018.

[4] Namanya. Anitta Patience, et all. “TheWorld of Malware: An Overview”. IEEE 6th International Conference on Future Internet of Things and Cloud. 2018.

[5] Christopher C. Elisan, “Advanced Malware Analysis”, McGraw-Hill Education. 2015.

[6] Eilam, Eldad, “Reversing: secrets of reverseengineering”. John Wiley & Sons. 2005.

[7] Victor M. Alvarez, “yara Documentation Release 3.11.0”. Yara. 2020.

[8] ClamAV Team, “ClamAV® is an open source antivirus engine for detecting trojans, viruses, malware & other malicious threats.”, https://www.clamav.net/. 2020.

[9] Richard L. Halterman. “Fundamentals Python Programming”. Southern Adventist University.2018.

[10] Chet Ramey, Brian Fox, “Reference Documentation for Bash Edition 5.0, for Bash Version 5.0.”, Free Software Foundation, Inc, 2019.