Terbit *online* pada laman web jurnal : http://journal.unbara.ac.id/index.php/INTECH

INFORMATIKA DAN TEKNOLOGI (INTECH)

ISSN (Online) : 2722-7367

Analisa Kerentanan Sistem Dengan Menerapkan Open Vulnerability Assessment System Menggunakan Greenbone Vulnerability Management (GVM)

MUH. AHSAN [1] , Dwi Anindyani Rochmah [2]

Universitas Mercu Buana, Jl Meruya Selatan No. 1, Kembangan, Jakarta Barat 11650, Indonesia

[1]41518010169@mercubuana.ac.id, [2]dwi.anindya@mercubuana.ac.id

INFORMASI ARTIKEL

Diterima Redaksi: 18 Agustus 2022

Revisi Akhir: 30 Oktober 2022

Diterbitkan *Online*: 30 November 2022

KATA KUNCI

Network Scanning, Vulnerability Assessment, GVM

ABSTRACT

A computer network is a network telecommunication that connects one or more computers to exchange data and information with each other. Such huge benefit will certainly be reduced by the presence of interference that arises in the network, when the network only involves local devices or in other words, is not connected to the internet network then interference may be less calculated. However, when the local network is connected to the internet network, a security system will be something that must be considered. Every system and network will undoubtedly have vulnerabilities and can cause damage to the system and even data to cause losses. Tracing activities and identifying system vulnerabilities are effective ways to minimize the risk of continuous vulnerability, therefore this study aims to conduct network analysis to determine vulnerabilities system by applying the open vulnerability assessment system method using Greenbone Vulnerability Management (GVM) as a platform of network scanning and vulnerability management system. The results are achieved by doing the vulnerability analysis to evaluate the results of vulnerability findings and then categorized according to the level of risk, namely high, medium and low. Conducting a vulnerability assessment will improve information security and avoid bad risks that can cause losses. The results achieved doing the vulnerability assessment activity are to provide an evaluation of the risk of vulnerability found and the impact that can be generated and as a preventive measure to increase security system awareness.

1. PENDAHULUAN

Pada setiap sistem dan jaringan tentu akan mempunyai kerentanan dan dapat mengakibatkan kerusakan bahkan kehilangan data sehingga menimbulkan kerugian. Sangat diperlukan aktifitas penelusuran dan identifikasi kerentanan sistem untuk menanggulangi dampak kerusakan karena akibat adanya serangan pihak yang tidak bertanggung jawab. Hal ini menjadi dasar untuk meningkatkan kesadaran dan melakukan langkah awal untuk mendeteksi, mengidentifikasi dan mempelajari kelemahan yang dimiliki dari suatu sistem. Berdasarkan kasus tersebut maka sangat penting untuk menerapkan *vulnerability assessment* yang Dilakukan dengan menggunakan Greenbone Vulnerability Management (GVM). GVM sebagai wadah *network scanning* dan *vulnerability management system* yang mampu menjadi salah satu solusi untuk memberikan gambaran dari sebuah penelusuran celah keamanan. Hasil yang dicapai dalam analisa *vulnerability* adalah mengevaluasi hasil temuan kerentanan kemudian dikategorikan sesuai dengan tingkat resikonya yaitu tingkat tinggi, tingkat sedang dan tingkat rendah. Melakukan *vulnerability assessment* akan meningkatkan keamanan informasi serta terhindar dari resiko buruk yang dapat menimbulkan kerugian. Hasil yang dicapai dalam kegiatan *vulnerability assessment* adalah

memberikan evaluasi terhadap resiko kerentanan yang ditemukan dan dampak yang dapat ditimbulkan serta sebagai langkah preventif untuk meningkatkan kesadaran keamanan sistem.

2. TINJAUAN PUSTAKA

2.1 Virtual Machine

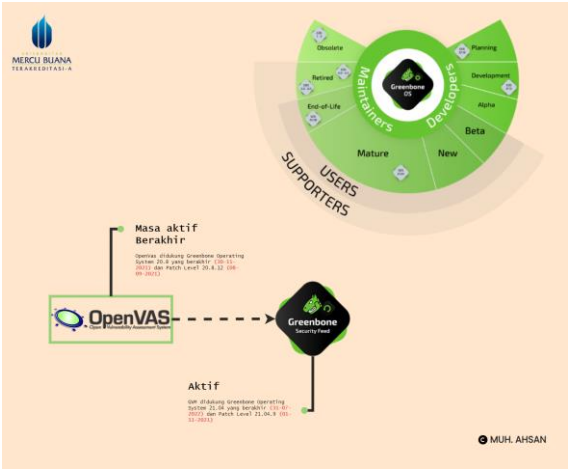
Virtual Machine adalah rekaya perangkat lunak yang memiliki fungsi hampir sama seperti komputer fisik yang dapat digunakan untuk menambah sistem operasi didalam sistem operasi utama atau dapat dikatakan sebagai tempat untuk melakukan simulasi sistem operasi.

2.2 Nmap

Netwok Mapper merupakan *opensource tool* untuk eksplorasi dan audit keamanan jaringan. Dirancang untuk memeriksa jaringan besar secara cepat, meskipun dapat pula bekerja terhadap single host. Nmap menggunakan paket IP raw untuk menentukan host mana saja yang tersedia pada jaringan, layanan nama aplikasi dan versi apa yang diberikan, sistem operasi dan versi apa yang digunakan, apa jenis *firewall* atau *filter* paket yang digunakan, dan sejumlah karakteristik lainnya [1]. Meskipun Nmap umumnya digunakan untuk audit keamanan, namun banyak administrator sistem dan jaringan menganggapnya berguna untuk tugas rutin seperti inventori jaringan, mengelola jadwal upgrade layanan, dan melakukan *monitoring up time host* atau layanan.

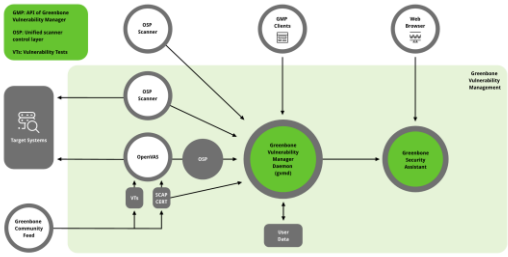
2.3 Greenbone Vulnerability Management (GVM)

Greenbone Vulnerability Management awalnya dibangun sebagai proyek komunitas yang bernama OpenVAS. GVM diteruskan dan dikembangkan oleh Greenbone Network [2] [3] [4]. Gambar berikut merupakan proses OpenVAS 9 menjadi GVM 11.



Gambar 1 OpenVAS 9 menjadi GVM 11

Gambar berikut merupakan arsitektur *patch level* GVM 21.04 secara umum.



Gambar 2 Arsitektur GVM 21.04

GMV dikelompokkan menjadi tiga bagian utama:

a. Aplikasi *scan* dapat melakukan eksekusi dengan menjalankan Vulnerability Test (VT) terhadap sistem target.
b. Greenbone Vulnerability Manager Daemon (gvmd)
c. Greenbone Security Assistant (GSA) dengan the Greenbone Security Assistant Daemon (gsad)

GVM memiliki komponen yang saling terkait satu dengan yang lain [5]. Komponen tersebut antara lain:

a. Greenbone Vulnerability Manager Daemon (gvmd)

Greenbone Vulnerability Manager Daemon (gvmd) adalah pusat layanan yang mengkonsolidasikan pemindaian kerentanan menjadi solusi manajemen kerentanan penuh. gvmd mengontrol OpenVAS Scanner melalui Open Scanner Protocol (OSP). Layanan itu sendiri menawarkan Greenbone Management Protocol (GMP) berbasis XML. gvmd juga mengontrol database SQL (PostgreSQL) tempat semua konfigurasi dan data hasil pemindaian disimpan secara terpusat. Selanjutnya, gvmd juga menangani manajemen pengguna termasuk kontrol izin dengan grup dan peran. Dan terakhir, layanan memiliki sistem runtime internal untuk tugas terjadwal dan acara lainnya.

b. Greenbone Security Assistant (GSA)

Greenbone Security Assistant (GSA) adalah web interface dari GVM yang mana pengguna mengatur scan dan mengakses informasi kerentanan secara bersama. Ini adalah titik kontak utama bagi pengguna dengan GVM. Terhubung ke gvmd melalui server web Greenbone Security Assistant Daemon (gsad) untuk menyediakan aplikasi web berfitur lengkap untuk manajemen kerentanan. Komunikasi terjadi menggunakan Greenbone Management Protocol (GMP) dimana pengguna juga dapat berkomunikasi secara langsung dengan menggunakan alat yang berbeda.

c. OpenVAS Scanner

OpenVAS Scanner adalah mesin pemindai berfitur lengkap yang menjalankan uji kerentanan (VT) terhadap sistem target. Untuk ini, ia menggunakan feed harian yang diperbarui dan komprehensif: Greenbone Security Feed (GSF) komersial berfitur lengkap, ekstensif, atau Greenbone Community Feed (GCF) yang tersedia gratis. OpenVAS Scanner terdiri dari komponen ospd-openvas dan openvas-scanner Pemindai OpenVAS dikendalikan melalui OSP. OSP Daemon untuk OpenVAS Scanner (ospd-openvas) berkomunikasi dengan gvmd melalui OSP: Data VT dikumpulkan, pemindaian dimulai dan dihentikan, dan hasil pemindaian ditransfer ke gvmd melalui ospd.

2.4 Vulnerability Assessment

Vulnerability Assessment melakukan identifikasi kerentanan dari suatu aplikasi, sistem operasi dan infrastruktur jaringan. *Vulnerability Assessment* tidak melakukan eksploitasi celah atau kelemahan dari suatu sistem lebih fokus untuk menemukan beragam public vulnerability pada seluruh sistem komputer dalam jaringan target dan tidak menuju ke proses eksploitasi namun memiliki potensi untuk di eksploitasi sehingga harus ditutup kerentanan yang ditemukan. [6].

2.5 Vulnerability Management

1. *Vulnerability Identification*

Tujuan dari langkah ini adalah untuk menyusun daftar lengkap kerentanan aplikasi. Analis keamanan menguji kesehatan keamanan aplikasi, server, atau sistem lain dengan memindainya dengan alat otomatis, atau menguji dan mengevaluasinya secara manual. Analisa ini juga mengandalkan database kerentanan, pengumuman kerentanan vendor, sistem manajemen aset, dan umpan intelijen ancaman untuk mengidentifikasi kelemahan keamanan.

2. *Vulnerability Analysis* (*Vulnerability Scanning*)

Tujuan dari langkah ini adalah untuk mengidentifikasi sumber dan akar penyebab kerentanan yang diidentifikasi pada langkah pertama.

Ini melibatkan identifikasi komponen sistem yang bertanggung jawab untuk setiap kerentanan, dan akar penyebab kerentanan.

3. *Risk Assessment* (*Vulnerability Assessment*)

Tujuan dari langkah ini adalah memprioritaskan kerentanan. Ini melibatkan analis keamanan yang menetapkan peringkat atau skor keparahan untuk setiap kerentanan, berdasarkan faktor-faktor seperti:

a. Sistem mana yang terpengaruh.
b. Data apa yang berisiko.
c. Fungsi bisnis mana yang berisiko.
d. Kemudahan menyerang atau kompromi.
e. Tingkat keparahan serangan.
f. Potensi kerusakan sebagai akibat dari kerentanan.

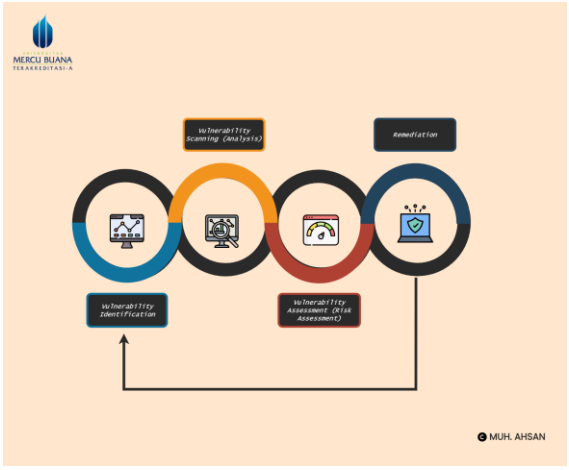
4. *Remediation*

Tujuan dari langkah ini adalah untuk menutup celah keamanan. Ini biasanya merupakan upaya bersama oleh staf keamanan, tim pengembangan dan operasi, yang menentukan jalur paling efektif untuk remediasi atau mitigasi setiap kerentanan.

Langkah-langkah perbaikan khusus mungkin termasuk:

a. Pengenalan prosedur, tindakan, atau alat keamanan baru.
b. Pembaruan perubahan operasional atau konfigurasi.
c. Pengembangan dan implementasi patch kerentanan.

Penilaian kerentanan tidak bisa menjadi kegiatan satu kali saja. Agar efektif, organisasi harus mengoperasionalkan proses ini dan mengulanginya secara berkala. Penting juga untuk mendorong kerja sama antara tim keamanan, operasi, dan pengembangan-sebuah proses yang dikenal sebagai DevSecOps



Gambar 3 Proses *Vulnerability Management*

3. METODE PENELITIAN

3.1 Jenis Penelitian

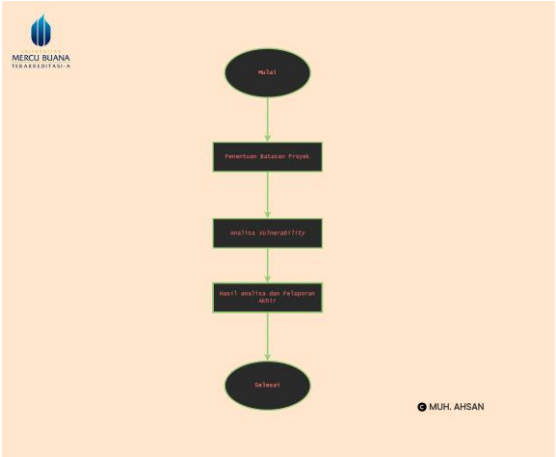
Metode yang digunakan penulis dalam penelitian ini adalah metode penelitian terapan yang berfokus pada analisis hasil evaluasi sehingga diharapkan dapat menghasilkan berupa informasi yang dijadikan masukan atau pengambilan keputusan tertentu sesuai urgensi sasaran.

3.2 Metode Pengumpulan Data

Metode pengumpulan data yang digunakan pada penelitian ini adalah observasi yaitu melakukan analisa terhadap kerentanan sistem dan penelitian tindakan yang mengimplementasi *Open Vulnerability Assessment System* menggunakan GVM berdasarkan topologi yang sedang berjalan.

3.3 Tahap Penelitian

Secara teknis penelitian ini akan dilaksanakan menggunakan 3 tahapan inti dari proses analisa *vulnerability*. Tahapan tersebut seperti yang terlihat pada Gambar 4.



Gambar. 4 *Flowchart* Analisa *Vulnerability*

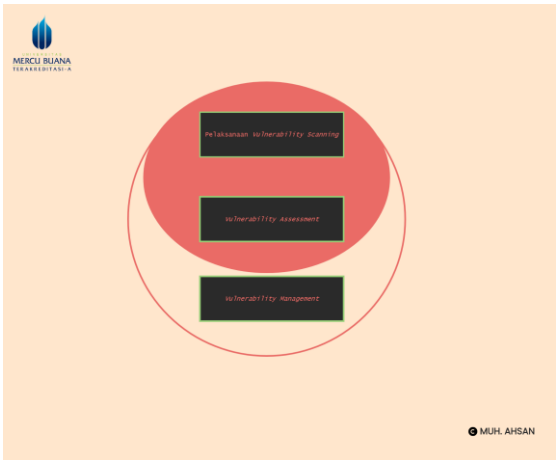
Dapat dilihat pada Gambar 4 diagram alir / *flowchart* dari penelitian yang dilaksanakan.

Pertama, Batasan proyek diperlukan agar analisa vulnerability tidak terlalu luas, sehingga melibatkan hal-hal lain yang tidak perlu dan tidak terlampau sempit sehingga

melewatkan hal-hal yang penting. Untuk menentukan batasan sistem, ada 2 hal yang perlu dijadikan sebagai pertimbangan yaitu pemahaman terhadap proses sistem yang akan diuji dan pemahaman kompleksitas sistem.

Kedua, Analisa vulnerability yang mengacuh pada utilisasi GVM

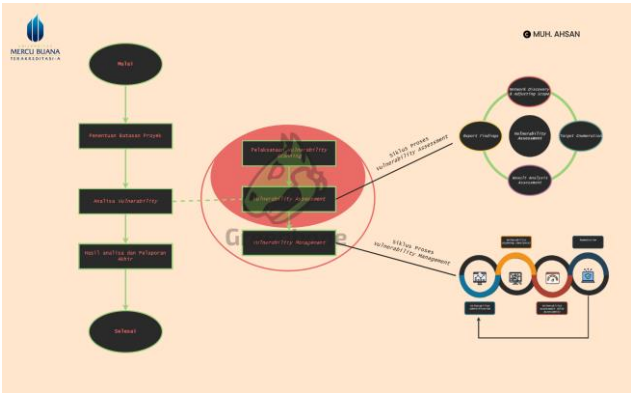
Ketiga, Hasil analisa dan pelaporan akhir. Dalam tahap ini adalah tahap yang terakhir dari proses mekanisme analisa vulnerability. Tahap pelaporan adalah tahap yang paling penting, fase ini adalah memberikan rekomendasi tentang temua hasil identifikasi kerentanan. Fase reporting merupakan kegiatan memetakan hasil identifikasi sehingga kerentanan yang ditemukan dapat dikategorikan dengan baik serta dapat dilakukan tindakan mitigasi untuk memberikan rekomendasi kepada pihak target tentang kerentanan sistem yang dimilikinya.



Gambar. 5 Siklus Analisa *Vulnerability*

Pada gambar 5 penulis mengambarkan skema analisa yang terjadi dibagian urutan ke-2 gambar 7 terkait analisa vulnerability yang memiliki cakupan 3 unsur yaitu pelaksanaan *Vulnerability Scanning*, *Vulnerability Assessment*, *Vulnerability Management*.

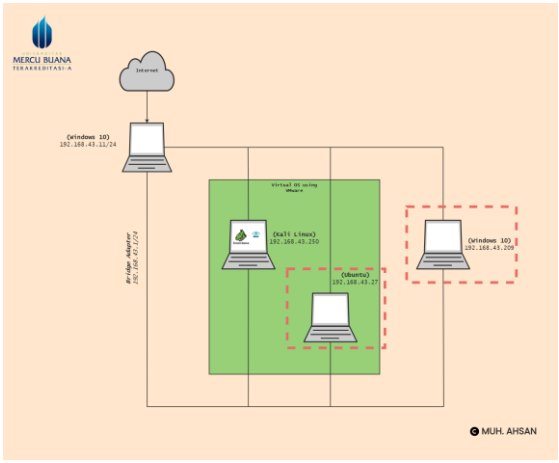
1. *Vulnerabiliy Scanning*: Melakukan scanning pada sistem.
2. *Vulnerability Assessment*: Identifikasi vulnerability pada sistem.
3. *Vulnerability Management*: Pengelolahan analisa vulnerability pada sistem.



Gambar. 6 Proses Analisa *Vulnerability*

Skema umum kompleks yang terjadi pada proses analisa *vulnerability* yang penulis lakukan dapat dilihat pada gambar 6.

Berikut gambaran umum sistem yang didesain menjadi topologi jaringan yang digunakan untuk analisa penggunaan GVM sebagai mekanisme *vulnerability assessment*.



Gambar. 7 Desain Topologi Jaringan

Jaringan yang digunakan adalah jaringan lokal. Topologi yang digunakan untuk pengujian ini menggunakan satu buah server dan dua client yang terhubung dalam jaringan virtual yang dipasang pada VMware Workstation. Di sisi linux terdapat sistem GVM yang dipasang sebagai alat bantu pengujian, kemudian sebagai sisi target terdapat windows 10 dan Ubuntu Linux yang memiliki masing-masing ip *address* dhcp. Pada sisi client yaitu kali linux yang sudah terpasang sistem GVM sebagai alat bantu pengujian. Didalam topologi terdapat satu buah server GVM yang terpasang dalam sistem kali linux dan dua buah client yang terhubung melalui jaringan virtual menggunakan Vmwork Workstation dengan network bridge mode. Maksudnya ketika virtual machine ini salah satu adapter network-nya menggunakan bridge maka virtual machine tersebut akan bisa terhubung dengan jaringan Wi-Fi, PC lain maupun LAN. IP *Address* yang diterima VM akan satu segment dengan *real network* dan bisa saling berkomunikasi antar keduanya.

Berikut alokasi Alamat IP pada tabel 2

Tabel 1 Alokasi Alamat IP

Nama Perangkat	Alamat IP
Bridge Network	192.168.43.1/24
Main OS	192.168.43.11/24
Kali Linux	192.168.43.250/24
Ubuntu Linux	192.168.43.27/24
Windows 10	192.168.43.209/24

Pada gambar 8 penulis mencari dan memastikan bawah traget ip *address* tersedia dengan melakukan explorasi ip *address* menggunakan nmap *tool*.

Gambar. 8 Eksplorasi IP *Address*

Hasil dari explorasi ip *address* pada gambar 8 terlihat bahwa terdapat target ip *address* 192.168.43.27 dan 192.168.43.209 yang penulis inginkan.

```
┌──(kali㉿kali)-[~/Documents/Penelitian Latest Updated/IP Target]
└─$ cat "IP Target.txt"
Target IP Address Untuk Bahan Penelitian Sebagai Berikut:

192.168.43.27

192.168.43.209

#41518010169
#MUH. AHSAN

┌──(kali㉿kali)-[~/Documents/Penelitian Latest Updated/IP Target]
└─$
```

Gambar. 9 IP *Address* Target

Pada gambar 9 penulis membuat direktori serta *file* baru untuk menyimpan ip *address* target yang akan digunakan untuk melakukan analisa *vulnerability*.

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -v -sS 192.168.43.27
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-22 12:00 WIB
Initiating ARP Ping Scan at 12:00
Scanning 192.168.43.27 [1 port]
Completed ARP Ping Scan at 12:00, 0.06s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 12:00
Completed Parallel DNS resolution of 1 host. at 12:00, 13.00s elapsed
Initiating SYN Stealth Scan at 12:00
Scanning 192.168.43.27 [1000 ports]
Completed SYN Stealth Scan at 12:00, 0.10s elapsed (1000 total ports)
Nmap scan report for 192.168.43.27
Host is up (0.00019s latency).
All 1000 scanned ports on 192.168.43.27 are in ignored states.
Not shown: 1000 closed tcp ports (reset)
MAC Address: [redacted] (VMware)

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 13.33 seconds
           Raw packets sent: 1001 (44.028KB) | Rcvd: 1001 (40.028KB)

┌──(kali㉿kali)-[~]
└─$
```

```
┌──(kali㉿kali)-[~/Documents/Penelitian (Latest Updated)/Port Target]
└─$ cat 192.168.43.27.txt
All 1000 scanned ports on 192.168.43.27 are in ignored states.

Not shown: 1000 closed tcp ports (reset)

MAC Address: [redacted] (VMware)

#41518010169
#MUH. AHSAN

┌──(kali㉿kali)-[~/Documents/Penelitian (Latest Updated)/Port Target]
└─$
```

Gambar. 10 *Port* 192.168.43.27

Hasil dari ekplorasi *port* ip *address* 192.168.43.27 pada gambar 10 terlihat bahwa tidak ada *port* yang terbuka dari 1000 tcp *ports* yang dieksplorasi oleh npm *tool* sehingga ip *address* 192.168.43.27 tidak digunakan oleh penulis untuk melakukan analisa *vulnerability*.

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -v -sS 192.168.43.209
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-22 11:14 WIB
Initiating ARP Ping Scan at 11:14
Scanning 192.168.43.209 [1 port]
Completed ARP Ping Scan at 11:14, 0.08s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 11:14
Completed Parallel DNS resolution of 1 host. at 11:14, 13.00s elapsed
Initiating SYN Stealth Scan at 11:14
Scanning 192.168.43.209 [1000 ports]
Discovered open port 443/tcp on 192.168.43.209
Discovered open port 445/tcp on 192.168.43.209
Discovered open port 80/tcp on 192.168.43.209
Discovered open port 135/tcp on 192.168.43.209
Discovered open port 3306/tcp on 192.168.43.209
Discovered open port 139/tcp on 192.168.43.209
Discovered open port 5357/tcp on 192.168.43.209
Completed SYN Stealth Scan at 11:14, 6.58s elapsed (1000 total ports)
Nmap scan report for 192.168.43.209
Host is up (0.028s latency).
Not shown: 993 filtered tcp ports (no-response)
PORT     STATE SERVICE
80/tcp   open  http
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
443/tcp  open  https
445/tcp  open  microsoft-ds
3306/tcp open  mysql
5357/tcp open  wsdapi
MAC Address: [redacted] (CyberTAN Technology)

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 19.79 seconds
           Raw packets sent: 1998 (87.896KB) | Rcvd: 12 (512B)

┌──(kali㉿kali)-[~]
└─$
```

```
┌──(kali㉿kali)-[~/Documents/Penelitian (Latest Updated)/Port Target]
└─$ cat 192.168.43.209.txt
PORT     STATE SERVICE
80/tcp   open  http
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
443/tcp  open  https
445/tcp  open  microsoft-ds
3306/tcp open  mysql
5357/tcp open  wsdapi

#41518010169
#MUH. AHSAN

┌──(kali㉿kali)-[~/Documents/Penelitian (Latest Updated)/Port Target]
└─$
```

Gambar. 11 *Port* 192.168.43.209

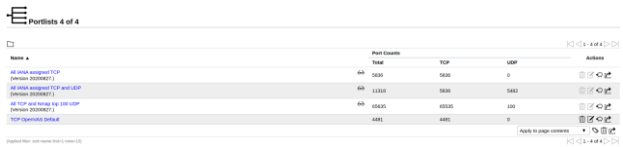
Hasil dari ekplorasi *port* ip *address* 192.168.43.209 pada gambar 11 terlihat bahwa adanya *port* yang terbuka dari 1000 tcp *ports* yang dieksplorasi oleh npm *tool* sehingga ip *address* 192.168.43.27 dapat digunakan oleh penulis untuk melakukan analisa *vulnerability*.

```
└─$ sudo gvm-check-setup
[sudo] password for kali:
gvm-check-setup 21.4.3
  Test completeness and readiness of GVM-21.4.3
Step 1: Checking OpenVAS (Scanner) ...
        OK: OpenVAS Scanner is present in version 21.4.4.
        OK: Server CA Certificate is present as /var/lib/gvm/CA/servercert.pem.
Checking permissions of /var/lib/openvas/gnupg/*
        OK: _gvm owns all files in /var/lib/openvas/gnupg
        OK: redis-server is present.
        OK: scanner (db_address setting) is configured properly using the redis-server socket: /var/run/redis-openvas/redis-server.sock
        OK: redis-server is running and listening on socket: /var/run/redis-openvas/redis-server.sock.
        OK: redis-server configuration is OK and redis-server is running.
        OK: _gvm owns all files in /var/lib/openvas/plugins
        OK: NVT collection in /var/lib/openvas/plugins contains 97004 NVTs.
Checking that the obsolete redis database has been removed
        OK: No old Redis DB
        OK: ospd-OpenVAS is present in version 21.4.4.
Step 2: Checking GVMD Manager ...
        OK: GVM Manager (gvmd) is present in version 21.4.5.
Step 3: Checking Certificates ...
        OK: GVM client certificate is valid and present as /var/lib/gvm/CA/clientcert.pem.
        OK: Your GVM certificate infrastructure passed validation.
Step 4: Checking data ...
        OK: SCAP data found in /var/lib/gvm/scap-data.
        OK: CERT data found in /var/lib/gvm/cert-data.
Step 5: Checking Postgresql DB and user ...
        OK: Postgresql version and default port are OK.
 gvmd | _gvm       | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
        OK: At least one user exists.
Step 6: Checking Greenbone Security Assistant (GSA) ...
Oops, secure memory pool already initialized
        OK: Greenbone Security Assistant is present in version 21.4.4.
Step 7: Checking if GVM services are up and running ...
        OK: ospd-openvas service is active.
        OK: gvmd service is active.
        OK: gsad service is active.
Step 8: Checking few other requirements...
        OK: nmap is present in version 21.4.4.
        OK: ssh-keygen found, LSC credential generation for GNU/Linux targets is likely to work.
        WARNING: Could not find makensis binary, LSC credential package generation for Microsoft Windows targets will not work.
        SUGGEST: Install nsis.
        OK: xsltproc found.
        WARNING: Your password policy is empty.
        SUGGEST: Edit the /etc/gvm/pwpolicy.conf file to set a password policy.

It seems like your GVM-21.4.3 installation is OK.
```

Gambar. 12 GVM *Check Setup*

Pada gambar 12 penulis melakukan pengecekan GVM *tool* secara berkala yang sudah penulis instalasi di server kali linux. Hal ini sangat penting untuk mengetahui apakah *tool* dapat digunakan tanpa adanya *error* sebelum dijalankan dengan memanggil perintah gvm-start.



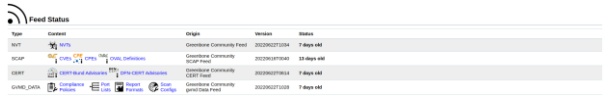
Gambar. 13 GVM *Check Setup*

Pada gambar 13 terdapat bahwa ada beberapa daftar *port* secara bawahaan pada *portlists menu*. Penulis menggunakan All TCP an Nmap top 100 UDP yang memiliki *Port Counts* untuk TCP 65535.

Gambar. 14 GVM *Targets Menu*

Pada gambar 14 terdapat target yang dibuat oleh si penulis dengan rincian sebagai berikut:

a. *Name*: Penelitian MPTI - TA 2021
b. *Host*: 192.168.43.209
c. *Port Type List*: All TCP and Nmap top 100 UDP

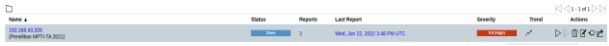


Gambar. 15 GVM *Feed Status Menu*

Pada gambar 15 terdapat daftar feed status NVT, SCAP, CERT, GVMD_DATA. Perbaruhan *feed* status secara berkala sangat diperlukan untuk melakukan analisa vulnerability yang mana nantinya secara otomatis sumber konten komponen utama dari GVM akan *up to date*.

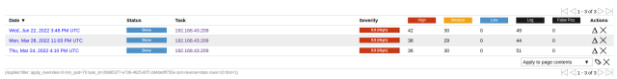
4. HASIL DAN PEMBAHASAN

Proses identifikasi kerentanan diwadahi dengan GVM dan secara umum menggunakan *open vulnerability assessment tool* dengan menargetkan 1 *host* yang sebelumnya di hasilkan dari proses tahapan eksperimen. Berikut hasil eksperimen analisa vulnerability.



Gambar. 16 Hasil Eksperimen Akhir

Tanggal 22 Juni 2022 merupakan hasil akhir dari analisa kerentanan yang menghasilkan *severity* atau tingkat kerentanan 9.9 (*High*) dengan diikuti *trend level* yaitu *up*.



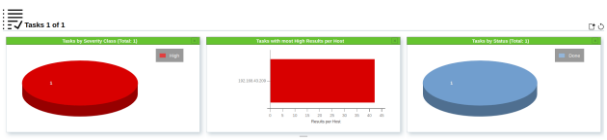
Gambar. 17 Priode Hasil Ekperimen

Terdapat 3 priode hasil eksperimen pada gambar 17 sebagai berikut:

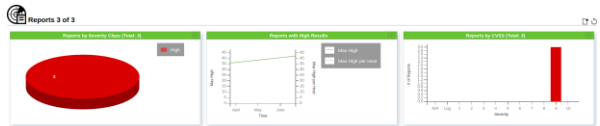
Pertama, pada Kamis, 24 Maret 2022 menghasilkan severity 9.9 (*High*). Dengan 5 parameter yaitu heigh sebanyak 36, *medium* 30, *low* 0, *log* 51, *False Pos*. 0.

Kedua, pada Kamis, 29 Maret 2022 menghasilkan severity 9.9 (*High*). Dengan 5 parameter yaitu *heigh* sebanyak 36, *medium* 30, *low* 0, *log* 44, *False Pos*. 0.

Ketiga, pada Rabu, 22 Juni 2022 menghasilkan severity 9.9 (*High*). Dengan 5 parameter yaitu *heigh* 42, *medium* 29, *low* 0, *log* 49, *False Pos*. 0.

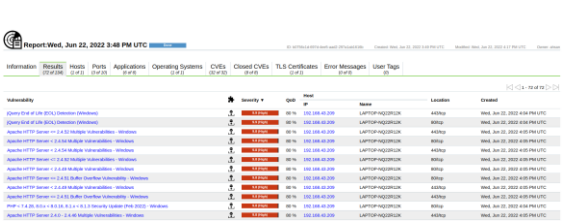


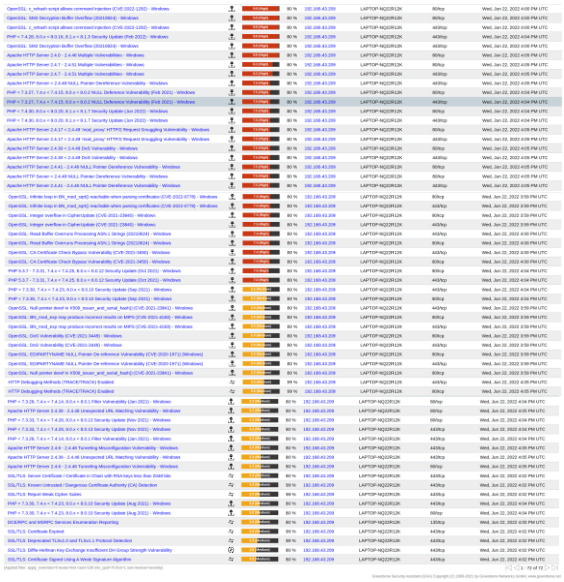
Gambar. 18 Hasil Eksperimen pada *Teks Section*



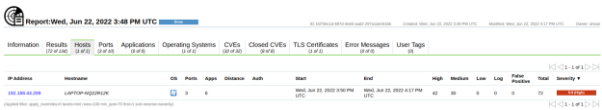
Gambar. 19 Hasil *Main Report Section*

Gambar. 20 Hasil informasi



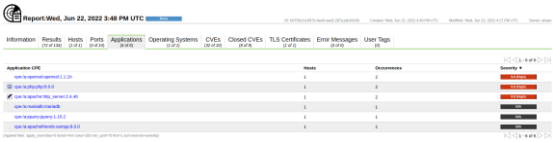


Gambar. 21 Hasil Kerentanan



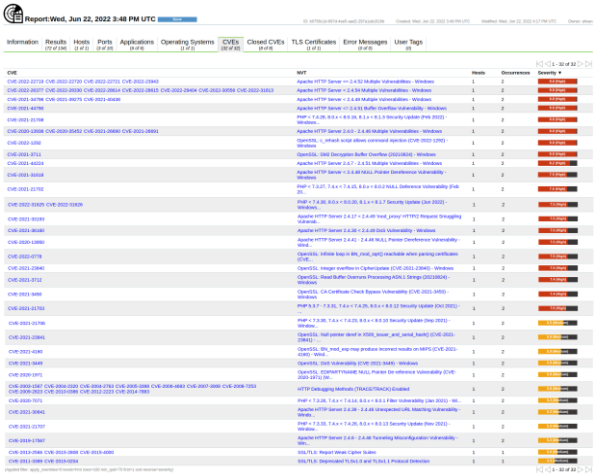
Gambar. 22 Hasil *Host*

Gambar. 23 Hasil *Port*

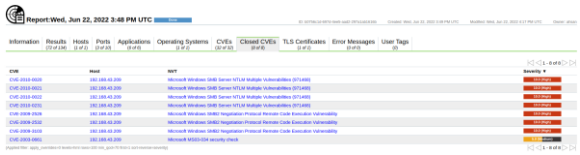


Gambar. 24 Hasil Aplikasi

Gambar. 25 Hasil Sistem Operasi



Gambar. 26 Hasil CVEs



Gambar. 27 Hasil *Closed* CVEs

Gambar. 28 Hasil TLS *Certification*

Tanggal 24 maret - 22 juni 2022 adanya kenaikan *trend* kerentanan selama melakukan analisa *vulnerability* dengan target *single host* 192.168.43.209 seperti yang terlihat pada gambar 17.

Untuk hasil akhir penelitihan pada tanggal 22 juni terdapat 42 kerentan tinggat *high* dan 29 medium seperti yang terlihat pada gambar 16.

Terdapat 3 ports dengan veriety kerentanan yaitu 443/tcp dan 80/tcp memiliki nilai 9.9 (*heigh*) serta 135/tcp memiliki nilai 5.0 (*medium*). Dapat dilihat pada gambar 23.

Hasil 42 kerentan dengan tingkat heigh memiliki tipe solusi vendor fix. Tipe solusi vendor fix yaitu dari setiap kerentan yang didapatkan ada tersedia informasi tentang perbaikan resmi yang dikeluarkan oleh pembuat asli produk. Dapat dilihat pada gambar 21.

Hasil 32 kerentanan dengan tingkat *medium* memiliki tipe solusi *workaround* dan *mitigation*. Tipe solusi *workaround* terdapat informasi tentang skenario spesifik konfigurasi atau pengembangan untuk menghindari adanya serangan ancaman dan ini merupakan langkah awal untuk *defence* terhadap *vulnerability*. Kemudian, untuk tipe solusi mitigation terdapat informasi sebuah skenario pengembangan dan konfigurasi untuk membantu dan meminimalisir tingkat resiko ancaman dari kerentanan akan tetapi solusi ini tidak menyelesaikan dampak kerentanan pada produk tersebut. Dapat dilihat pada gambar 21.

5. KESIMPULAN

Berdasarkan hasil pengematan pada penggunaan Greenbone Vulnerability Management yang mana telah berhasil mengidentifikasi kerentanan sistem dengan menemukan beragam variasi dan tingkat kerentanan.

Hal tersebut merupakan suatu bentuk tindakan preventif dan meningkatkan kesadaran akan pentingnya keamanan sistem yang dilakukan oleh penulis melalui penerapan *vulnerability assessment* yang di wadai Greenbone Vulnerability Management (GVM).

DAFTAR PUSTAKA

[1] R. S. Lumbu, "PRAKTTIKUM KEAMANAN JARINGAN," 2018.

[2] A. Analysis and U. Manual, "User Manual Greenbone Security Manager," *Data Base*, vol. 3304, no. January, pp. 1–148, 2012.

[3] B. Schneider and P. M. Asprion, "Testing a Vulnerability Scanner Greenbone Community Edition," no. August, 2020.

[4] "LIFECYCLE, GREENBONE OPERATING SYSTEM: ROADMAP &," *greenbone.net*, 2021, [Online]. Available: https://www.greenbone.net/en/roadmap-lifecycle/.

[5] "GVM Architecture," *greenbone.github.io*, 2021, [Online]. Available: https://greenbone.github.io/docs/architecture.html.

[6] "VULNERABILITY ASSESSMENT," *solvit.rs*. https://solvit.rs/vulnerability-assessment/.