WEB SECURITY COMPLIANCE TO OWASP AND SANS STANDARD

Disusun Oleh:

N a m a : Bella Tasya Kumala Dewi

NIM : 18523305

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2022

HALAMAN PENGESAHAN DOSEN PEMBIMBING

WEB SECURITY COMPLIANCE TO OWASP AND SANS STANDARD

TUGAS AKHIR

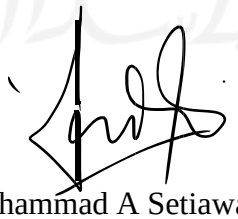
Disusun Oleh:

N a m a : Bella Tasya Kumala Dewi

NIM : 18523305

Yogyakarta, 5 Januari 2022

Pembimbing,



(Dr. Mukhammad A Setiawan, S.T., M.Sc.)

HALAMAN PENGESAHAN DOSEN PENGUJI

WEB SECURITY COMPLIANCE TO OWASP AND SANS STANDARD

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 5 Januari 2022

Tim Penguji

Dr. Mukhammad A Setiawan, S.T., M.Sc.

Anggota 1

Kurniawan Dwi Irianto, S.T., M.Sc.

Anggota 2

Moh. Idris, S.Kom., M.Kom.

Mengetahui,

Ketua Program Studi Informatika – Program Sarjana

Fakultas Teknologi Industri

Universitas Islam Indonesia

(Dr. Raden Teduh Dirgahayu, S.T., M.Sc.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Bella Tasya Kumala Dewi

NIM : 18523305

Tugas akhir dengan judul:

WEB SECURITY COMPLIANCE TO OWASP AND SANS STANDARD

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila dikemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung resiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 5 Januari 2022

(Bella Tasya Kumala Dewi)

HALAMAN PERSEMBAHAN

Puji syukur atas ridho dan rahmat yang Allah Subhana Wa Ta'ala berikan. Tugas Akhir ini dibuat dan dipersembahkan kepada seluruh keluarga yang tiada henti mamberi dukungannya, kepada teman, sahabat, dan orang istimewa yang selalu menemani dan memberikan semangat, kepada bapak/ibu dosen terutama pembimbing yang selalu memberikan arahan sehingga tugas akhir ini dapat terlaksana dengan baik dan selesai tepat waktu. –

HALAMAN MOTO

“Do the best today for tomorrow’- Bellatasya

“Yesterday is history, Tomorrow is a mystery, But today is a Gift” - Bil keane

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Allah SWT yang telah memberikan ridho dan rahmat-Nya sehingga penelitian Tugas Akhir ini dapat terselesaikan dengan baik. Tugas Akhir dengan judul "WEB SECURITY COMPLIANCE TO OWASP AND SANS STANDARD" ini digunakan sebagai syarat kelulusan dalam meraih gelar Sarjana Program Studi Informatika pada Fakultas Teknologi Industri, Universitas Islam Indonesia.

Peneltian Tugas Akhir ini tidak lepas dari bimbingan dan dukungan dari bapak/ibu dosen, berbagai rintangan penulis temui selama proses penelitian, namun dengan adanya rintangan penulis lebih bersemangat dan menganggapnya sebagai tantangan. Terima Kasih atas bimbingan, motivasi, bantuan, dan segala do'a yang telah dipanjatkan, dengan penuh kesadaran dan kerendahan hati penulis ucapkan terimakasih dan penghargaan sebesar-besarnya kepada:

1. Kedua orang tua dan seluruh keluarga yang selalu memberi dukungan kepada penulis.
2. Bapak Hendrik, S.T., M.Eng, selaku Ketua Jurusan Teknik Informatika Fakultas Teknologi Industri Universitas Islam Indonesia.
3. Bapak Ahmad Munasir Raf'ie Pratama, S.T., MIT., Ph.D. selaku Sekretaris Jurusan Informatika
4. Bapak Raden Teduh Dirgahayu, S.T., M.Sc. selaku Ketua Program Studi (Program Sarjana Informatika).
5. Bapak Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D. selaku Sekretaris Program Studi (Program Sarjana Informatika).
6. Bapak Mukhammad Andri Setiawan, S.T., M.Sc., Ph.D. selaku Dosen Pembimbing Tugas Akhir yang selalu memberikan bimbingan dan arahan kepada penulis sehingga penelitian ini dapat terselesaikan dengan baik.
7. Bapak Irving Vitra Paputungan, S.T., M.Sc., Ph.D. selaku Dosen Pembimbing Akademik yang selalu memberikan dukungan dan arahan kepada penulis.
8. Ibu Chanifah Indah Ratnasari, S.Kom., M.Kom. selaku Dosen Program Studi Informatika sekaligus admin website Cek-ejaan.com yang telah memberikan izin dalam menggunakan website sebagai objek penelitian.
9. Teman seperjuangan dan kakak tingkat Program Studi Informatika yang telah memberikan pengetahuan dan pengalaman kepada penulis.

Semoga dengan selesainya penelitian tugas akhir ini dapat memberikan manfaat, pengetahuan, dan pengalaman kepada semua pihak. Semoga semua pihak yang telah membantu

penulis dengan tulus dapat memperoleh balasan dari Allah SWT dan senantiasa diberikan kesehatan. Penulis memohon maaf jika terdapat kekeliruan maupun kekurangan selama penelitian ini berlangsung.

Yogyakarta, 5 Januari 2022

(Bella Tasya Kumala Dewi)

SARI

Perkembangan ilmu pengetahuan dan teknologi begitu pesat membawa banyak perubahan di seluruh dunia. Hal ini ditandai dengan akses informasi yang semakin mudah dan hampir seluruh aktivitas dapat dilakukan menggunakan teknologi melalui media internet. Namun, kejahatan dunia maya ikut meningkat seiring dengan naiknya penggunaan teknologi, banyak informasi penting mengenai pribadi maupun organisasi yang harus dijaga agar tidak menimbulkan hal-hal yang tidak diinginkan. Tidak hanya itu, *website* sebagai salah satu sarana dalam mendapatkan informasi juga harus dijaga keamanannya agar tidak mudah diserang oleh orang yang tidak bertanggung jawab.

Oleh karena itu, diperlukannya analisis celah keamanan suatu *website* yang dapat dilakukan melalui proses pengujian celah keamanan atau biasa disebut dengan *penetration testing*, teknik *penetration testing* ini dapat dilakukan oleh penguji dengan menggunakan standar keamanan yang ada dengan melakukan simulasi sebagai pihak luar atau penyusup yang hendak masuk kedalam suatu jaringan atau sistem. Penulis menerapkan metode pengujian *OWASP* Versi 4.2 sebagai *framework* karena metode ini menenrapkan tata cara pengujian secara terperinci. Selain itu, penulis menerapkan standar keamanan *CWE/SANS Top 25* sebagai daftar acuan celah keamanan terkini yang dilengkapi dengan *OWASP Top 10* yang berisi daftar serupa mengenai kerentanan keamanan. Penulis menggunakan *CWE/ SANS Top 25* dan *OWASP Top 10* karena daftar kerentanan tersebut selalu diperbaharui seiring dengan perkembangan teknologi.

Selama penelitian berlangsung ditemukan hasil kemungkinan celah keamanan pada beberapa pengujian, meliputi: *testing for weak lock out mechanism (WSTG-ATHN-03), test remember password functionality (WSTG-ATHN-05), testing for weak password change or reset functionalities (WSTG-ATHN-09), testing directory traversal/file include (WSTG-ATHZ-01), testing for bypassing authorization schema (WSTG-ATHZ-02), testing for bypassing session management schema (WSTG-SESS-01), testing for cookies attributes (WSTG-SESS-02), testing for session fixation (WSTG-SESS-03), testing for cross-side request forgery (WSTG-SESS-05), testing for logout functionality (WSTG-SESS-06), test session timeout (WSTG-SESS-07), testing for session puzzling (WSTG-SESS-08),* dan *testing for session hijacking (WSTG-SESS-09).* Beberapa kemungkinan celah tersebut dianalisis berdasarkan daftar celah keamanan *OWASP Top* 10 dan *SANS/CWE Top* 25 dengan hasil yang diperoleh *website* cek-ejaan.com memiliki kerentanan berupa *broken access control, cryptographic failures, insecure design, identification and authentication failures.* Berdasarkan hasil yang

diperoleh *website* cek-ejaan.com kurang menerapkan prinsip keamanan informasi yang terdiri dari *confidentiality*, *integrity*, dan *availability*. Jika dilihat dari beberapa fitur dan *activity* yang dilakukan *website* cek-ejaan.com tergolong cukup aman karena tidak mengandung informasi sensitif. Selain itu *website* cek-ejaan.com memiliki *firewall* yang cukup tangguh dalam memblokir serangan. Metode *OWASP* versi 4.2 yang digunakan dalam menguji *website* cek-ejaan.com sangat cocok digabungkan dengan daftar celah keamanan *OWASP Top 10* dan *SANS/CWE Top 25. OWASP ZAP* sebagai alat *scan* otomatis sangat baik menampilkan jenis kerentanan secara detail.

Kata kunci: Keamanan informasi, *website*, *OWASP*, *SANS*, *Kali Linux*.

GLOSARIUM

Attacker	Seseorang yang mencoba untuk menyerang dan menyusupi sebuah sistem.
CLI	Program yang digunakan untuk memerintahkan komputer melakukan tugas tertentu dengan mengetikkan perintah berupa teks.
Firewall	Sistem keamanan yang melindungi komputer melalui jaringan internet.
Footprinting	Langkah awal sebelum dilakukan penyerangan oleh *attacker* dengan mengumpulkan informasi target.
Host	Sistem dalam server yang saling terhubung secara langsung dengan server yang lain.
IP Address	Identitas berupa angka yang digunakan untuk menghubungkan semua komputer dalam jaringan internet.
JavaScript	Bahasa premrograman yang digunakan dalam pembuatan dan pengembangan website
Kali Linux	Sebuah sistem operasi open source berbasis debian yang berisi beberapa ratus *tools* yang digunakan untuk pengujian keamanan informasi
Penetration Testing	Proses pengujian celah keamanan yang dilakukan oleh seorang penguji dengan menjadikannya sebagai pihak luar yang melakukan penyerangan tehadap suatu objek dapat berupa sistem atau jaringan.
Port	Meksnisme dalam jaringan yang bertujuan untuk mendukung beberapa sesi koneksi antara sebuah komputer dengan komputer lain dan program.
Scanning	Proses memindai sistem atau target yang hendak diuji.
Server	Sebuah sistem komputer sebagai penyedia jenis layanan tertentu pada jaringan komputer
Threats	Ancaman keamanan berupa aksi yang berasal dari dalam maupun luar suatu sistem.
URL	Rangkaian karakter tertentu yang digunakan menuju ke sebuah situs web.

Vulnerability	Celah keamanan suatu sistem berupa kelemahan yang dapat mengakibatkan pihak luar yang tidak berwenang dapat mengakses sistem.
Web	Sebuah situs.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

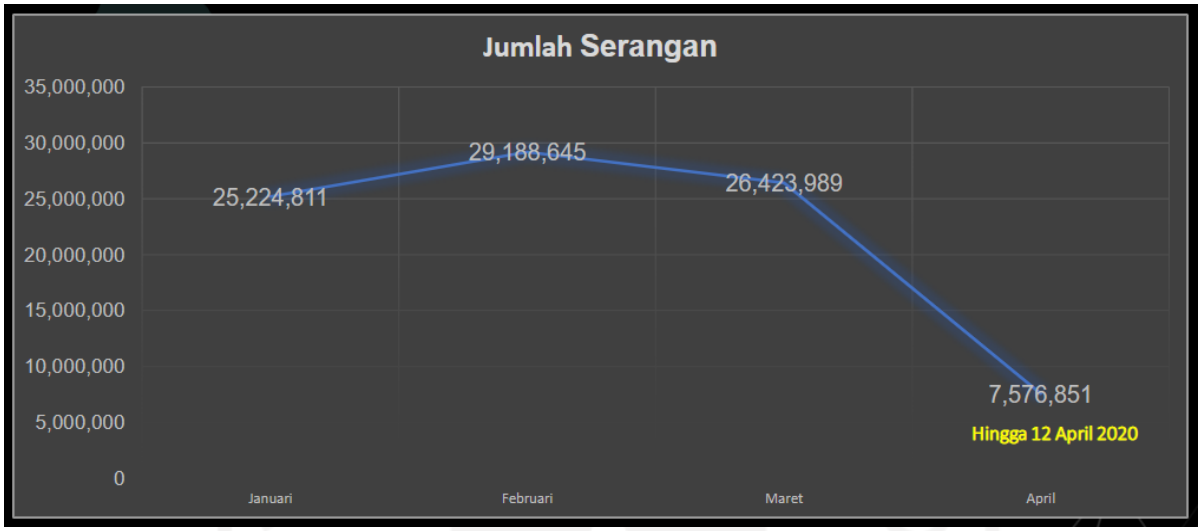
BAB I
PENDAHULUAN

1.1 Latar Belakang

Internet merupakan salah satu hasil dari kemajuan teknologi yang berkembang pesat pada era sekarang ini. Perkembangan teknologi informasi dan komunikasi telah membuat perubahan signifikan pada seluruh bidang ilmu pengetahuan. Internet *(interconnection networking)* dapat diartikan sebagai jaringan komputer yang dapat menghubungkan seluruh orang dalam bertukar informasi atau data melalui *TCP/IP (Transmission Control Protocol / Internet Protocol)* serta menyediakan berbagai sumber informasi yang terhubung dengan jaringan komputer (Putri, 2020). Akses informasi semakin mudah dengan adanya internet, namun tidak semua informasi dapat diakses dan diketahui secara bebas oleh semua pihak. Keamanan informasi erat kaitannya dengan *CIA (Confidentiality, Integrity, and Availability)* yang dijadikan sebagai acuan dalam sebuah *website* sebagai salah satu parameter dalam melakukan analisis celah keamanan (Guntoro et al., 2020). Informasi yang memuat data pribadi seseorang sebagai salah satu contoh informasi yang bersifat konfidensial, dapat diartikan bahwa hanya pihak yang berwenang yang dapat mengaksesnya sehingga tidak semua orang dapat mengakses dan mengetahuinya. Sebuah perusahaan atau instansi pada umumnya memiliki data yang bersifat konfidensial. Data tersebut digunakan untuk menunjang proses bisnis pada sebuah perusahaan atau instansi terkait yang tersimpan pada ruang penyimpanan dan dilengkapi dengan sistem keamanan. Sistem keamanan tidak hanya diterapkan pada suatu hal yang bersifat konfidensial atau memuat data pribadi seseorang, *website* sebagai salah satu contoh sarana memperoleh informasi yang dapat diakses oleh publik juga harus dilengkapi dengan sistem keamanan. Menjaga keamanan *website* merupakan upaya perlindungan dan pencegahan dari ancaman serangan keamanan yang dapat terkoneksi melalui jaringan. Tingkat keamanan ini harus dianalisis apakah sudah sesuai dengan standar keamanan yang tepat untuk menghindari risiko penyerangan keamanan terhadap *website* yang terjadi pada masa mendatang.

Pola kehidupan bergantung pada internet pada era pandemi sekarang ini. Hal ini berdampak pada kenaikan jumlah kasus serangan siber. Pusat Operasi Keamanan Siber Nasional (Pusopskamsinas) Badan Siber dan Sandi Negara (BSSN) telah mencatat serangan siber yang terjadi di Indonesia pada tanggal 1 Januari sampai 12 April 2020 sebanyak 88.414.296 serangan, dengan data pada bulan Januari sebanyak 25.224.811 serangan, pada bulan Februari sebanyak 29.188.645 serangan, pada bulan Maret terjadi 26.423.989 serangan,

dan sampai dengan tanggal 12 April 2020 telah tercatat sebanyak 7.576.851 serangan (Victor Tobing, 2020).



Gambar 1.1 Jumlah Serangan Siber di Indonesia pada Januari – April (Victor Tobing, 2020)

Berdasarkan hasil tinjauan pustaka yang telah dilakukan, penulis menyimpulkan untuk melakukan penelitian berupa analisis kepatuhan keamanan *website* cek-ejaan.com dengan standar keamanan *SANS* dan *OWASP* Versi 4.2 yang berfokus pada pengujian *authentication, authorization, session management* untuk mengetahui apakah terdapat *kerentanan* yang dapat memberikan dampak negatif terhadap *website* cek-ejaan.com. Website cek-ejaan.com merupakan sebuah situs yang digunakan untuk melakukan deteksi ejaan yang tidak sesuai dengan PUEBI (Pedoman Umum Ejaan Bahasa Indonesia).

1.2 **Identifikasi Masalah**

Berdasarkan pemaparan latar belakang masalah, maka diperoleh masalah sebagai berikut:

a. Apakah terdapat celah keamanan pada *website* cek-ejaan.com.
b. Bagaimana cara mengetahui celah keamanan yang terdapat pada *website* cek-ejaan.com.
c. Metode apakah yang digunakan dalam mencari celah keamanan *website* cek-ejaan.com.
d. Bagaimana hasil yang diperoleh dari pengujian celah keamanan *website* cek-ejaan.com menggunakan metode *OWASP* dan *SANS*.

1.3 Tujuan dan Manfaat Penelitian

Tujuan dilakukannya penelitian tugas akhir ini adalah:

a. Melakukan analisis celah keamanan *website* cek-ejaan.com menggunakan standar pengujian keamanan *OWASP* dan *SANS*.
b. Mengetahui celah keamanan pada *website* cek-ejaan.com sehingga dapat dijadikan acuan bagi stakeholder *website* dalam memperbaiki celah keamanan pada *website* cek-ejaan.com.
c. Mengetahui kerentanan *website* cek-ejaan.com dari hasil analisis pengujian celah keamanan menggunakan metode *OWASP* dan *SANS.*

1.4 Batasan Penelitian

Batasan yang digunakan dalam penelitian tugas akhir ini adalah:

a. Penelitian ini menggunakan studi kasus *website* cek-ejaan.com.
b. Jenis pengujian yang digunakan pada *website* cek-ejaan.com adalah *Black Box Testing.*
c. Pengujian dilakukan berdasarkan standar keamanan *SANS* dan panduan pengujian *OWASP* versi 4.2 dengan kategori pengujian autentikasi, pengujian otorisasi, dan pengujian manajemen sesi.
d. Tidak melakukan perbaikan pada celah keamanan *website* cek-ejaan.com.

1.5 Metodologi Penelitian

Metodologi penelitian dalam tugas akhir ini dilakukan untuk membuat pengujian menjadi lebih sistematis agar tujuan dapat tercapai sesuai dengan yang diharapkan. Metodologi penelitian yang digunakan dalam penelitian tugas akhir ini, meliputi:

a. Analisis Pertanyaan

Analisis pertanyaan dalam penelitian ini digunakan untuk memetakan latar belakang masalah yang hendak dipecahkan sebagai bentuk analisis sistem dan penelitian yang hendak dilakukan dengan membuat pertanyaan yang terdiri dari 5W *(What, Why, Who, Where, When)* + 1H *(How)* dan disertai dengan jawaban.

b. Studi Literatur

Studi Literatur digunakan untuk memetakan kajian pustaka yang dijadikan pendukung penelitian dapat berupa teori yang didapatkan melalui buku, jurnal, laporan penelitian terdahulu, artikel maupun situs di internet yang terkait dengan penelitian.

c. Pengumpulan Data dan Identifikasi Sistem

Pengumpulan data (*footprinting*) dalam penelitian ini dilakukan dengan mencari data atau informasi target yang hendak diuji yaitu *website* cek-ejaan.com, kemudian dilakukan identifikasi sistem dengan cara *scanning website* target untuk mengetahui celah keamanannya.

d. Pengujian Penetrasi

Pengujian penetrasi terbagi menjadi dua tahap yaitu persiapan dan pengujian yang dilakukan pada *website* target dengan standar keamanan yang digunakan yaitu metode *OWASP* dengan versi 4.2 menggunakan pengujian yang berfokus pada pengujian autentikasi, pengujian otorisasi, dan pengujian manajemen sesi, kemudian dilakukan analisis dari hasil pengujian yang diperoleh berdasarkan daftar kerentanan *CWE/SANS Top* 25 dan *OWASP Top* 10.

e. Analisis Hasil dan Penyusunan Dokumentasi Laporan

Pada tahapan ini dilakukan analisis dan penjelasan terhadap hasil pengujian yang diperoleh, kemudian disusun dalam laporan penelitian tugas akhir ini.

1.6 Sistematika Penulisan

Sistematika penulisan dalam laporan ini digunakan sebagai gambaran mengenai masalah yang hendak dibahas dalam laporan tugas akhir ini yang terbagi menjadi 5 bab, diantaranya:

BAB I PENDAHULUAN

Bab pendahuluan ini terdiri dari enam sub bab yang berisi tentang latar belakang, identifikasi masalah, tujuan dan manfaat penelitian, batasan penelitian, metodologi penelitian, dan sistematika penulisan laporan tugas akhir dengan judul "Web Security Compliance to OWASP and SANS Standard" studi kasus *website* cek-ejaan.com.

BAB II LANDASAN TEORI

Bab ini berisi penelitian terdahulu yang menjelaskan tentang metode dan *tools* yang digunakan dalam pengujian, selain itu terdapat dasar teori yang digunakan sebagai acuan dalam penelitian serta pengujian terkait dengan standar keamanan *OWASP* dan *SANS*.

BAB III METODOLOGI PENELITIAN

Bab ini membahas metode yang dilakukan dalam penelitian yang terbagi menjadi menjadi 5 tahapan yang diawali dengan analisis pertanyaan penelitian, studi pustaka, pengumpulan data dan identifikasi sistem, *pentest (penetration testing)* dengan metode *OWASP* versi 4.2, serta analisis hasil dan penyusunan dokumentasi laporan.

BAB IV HASIL DAN PEMBAHASAN

Bab hasil dan pembahasan berisi tentang tahapan proses yang dilakukan dan hasil yang diperoleh selama pengujian menggunakan standar keamanan *OWASP* versi 4.2 dan berdasarkan daftar kerentanan yang ada dengan studi kasus *website* cek-ejaan.com.

BAB V KESIMPULAN DAN SARAN

Bab ini berisi penutup yang membahas tentang kesimpulan dan saran dari hasil pengujian yang diperoleh melalui analisis hasil pengujian.

BAB II
LANDASAN TEORI

2.1 Dasar Teori

2.1.1 Keamanan Informasi

Pengertian keamanan informasi menurut ISO/IEC 17799:2005 yaitu upaya perlindungan atau tindakan untuk mendeteksi dan mencegah akses tidak sah yang dapat mengakibatkan pencurian informasi, kerusakan pada sistem informasi sehingga dapat menimbulkan kerugian pada proses bisnis yang dikelola (Dewanto, 2018). Terdapat dua hal yang menjadi masalah utama dalam keamanan informasi yaitu *threats* dan *vulnerability* (Guntoro et al., 2020).

Keamanan bukan bergantung pada suatu sistem seperti *firewall* atau *intruder detection* yang terdapat pada hardware atau software yang digunakan (Nazwita, 2017). *Intrusion Detection (ID)* merupakan usaha dalam mendeteksi jika ada penyusup yang mencoba memasuki sistem secara paksa tanpa dilakukan otorisasi. *Intrusion Detection System (IDS)* merupakan sebuah sistem pendeteksi penyusupan, IDS mencoba mendeteksi hal-hal yang dianggap mencurigakan atau ilegal kemudian memunculkan notifikasi. Namun, IDS tidak dapat melakukan pencegahan terhadap penyusupan (Elanda & Tjahjadi, 2018).

Seiring dengan perkembangan teknologi *website* sebagai wadah informasi semakin banyak dan berkembang, maka semakin meningkat pula angka penyerangan terhadap *website*. Oleh karena itu, diperlukan upaya menjaga keamanan website untuk mengurangi risiko celah keamanan. Pentingnya melakukan pengecekan atau pengujian celah keamanan agar dapat dilakukan perbaikan sebelum terjadinya serangan keamanan dari *attacker* yang tidak bertanggung jawab sehingga menyebabkan kerusakan maupun kerugian pada *website*.

2.1.2 *Website*

Website merupakan sekumpulan halaman web yang saling berhubungan, umumnya berisikan informasi yang terdiri dari data dapat berupa gambar, audio, video, teks, animasi, maupun gabungan dari semuanya dapat diakses melalui *browser* dan jaringan internet (Guntoro et al., 2020). Perlunya penerapan metode dalam melakukan analisis risiko celah keamanan *website* agar pengujian keamanan yang dilakukan lebih sistematis dan sesuai dengan standar yang ada.

2.1.3 Metode Analisis Keamanan *Website*

Analisis risiko celah keamanan *website* dapat dilakukan dengan beberapa metode, diantaranya :

1. *DREAD*

   DREAD (Damage Potential, Reproducibility, Exploitability, Affected User, Discoverability) merupakan kerangka kerja dalam menganalisis dan mengidentifikasi kerentanan suatu *website*, selain itu dapat digunakan dalam perhitungan risiko dan pemeringkatan suatu ancaman (Saputra et al., 2017).

2. *Ethical Hacking*.

   Ethical Hacking adalah suatu metode yang digunakan untuk mengidentifikasi kelemahan yang mengancam nilai *integrity, confidentiality*, dan *availability* (*vulnerability)* yang meliputi penggunaan, trik-trik dan teknik aplikasi *hacking* guna memastikan keamanannya (Alwi et al., 2020).

3. *PTES*

   Penetration Testing Execution Standards (PTES) merupakan salah satu *framework* yang digunakan untuk melakukan *penetration testing* dengan menyediakan panduan dan memberikan acuan untuk melakukan pengujian secara terstruktur dan mendetail (Utoro et al., 2020).

4. *ISSAF*

   ISSAF atau *Information System Security Assessment Framework* merupakan suatu metode yang dapat digunakan dalam melakukan evaluasi sebuah aplikasi berupa jaringan maupun sistem yang terdiri dari tiga fase pendekatan serta sembilan langkah dalam melakukan penilaian (Guntoro et al., 2020).

5. *Vulnerability Assessment*

   Vulnerability Assessment adalah sebuah metode yang digunakan untuk mengidentifikasi celah kerentanan dan potensi ancaman keamanan yang ada pada setiap sumber daya sebuah *website* (Tania et al., 2018).

6. *OCTAVE*

   OCTAVE (Operationally Critical Threat, Asset and Vulnerability Evaluation) merupakan kerangka kerja yang digunakan dalam identifikasi, analisis, serta melakukan pengawasan dan pengelolaan risiko berdasrarkan hasil identifikasi yang dilakukan.

7. *SANS Standard*

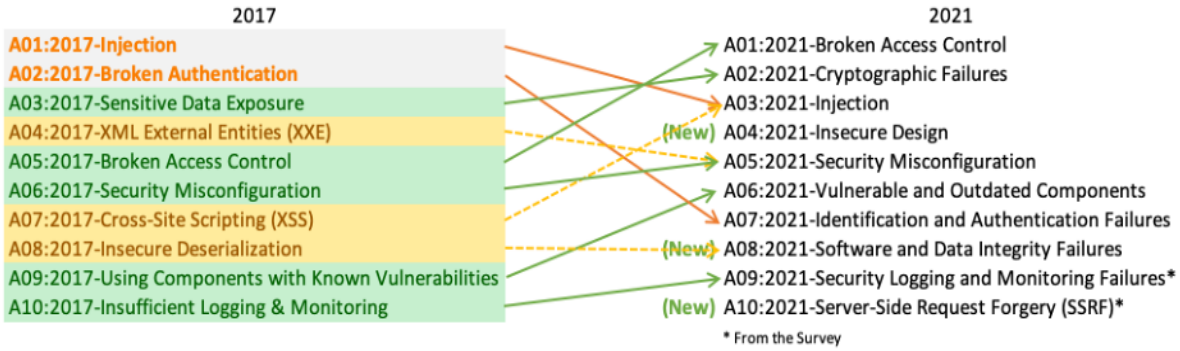
SANS (SysAdmin, Audit, Network, and Security) adalah sebuah perusahaan yang berfokus pada keamanan informasi, menawarkan pelatihan keamanan siber, serta sertifikasi melalui *GIAC*. Menurut *SANS*, area fokus operasi keamanan siber yang efektif bergantung pada lapisan pengujian ofensif, arsitektur dan pemantauan defensif, respon forensik dan insiden, keamanan cloud, dan kepemimpinan.

8. *OWASP*

Open Web Application Security Project atau *OWASP* adalah sebuah *framework* yang bersifat *open source*, dibangun oleh suatu organisasi dalam menentukan letak celah keamanan pada suatu aplikasi berbasis web dan memperbaikinya. Terdapat sebelas panduan yang digunakan untuk menguji keamanan sebuah *website* menurut standar yang dikeluarkan oleh OWASP versi 4, yaitu: *Information Gathering, Configuration and Deploy Management Testing, Identity Management Testing, Authentication Testing Authorization Testing, Session Management Testing, Data Validation Testing, Error Handling, Cryptography, Business Logic Testing, Client Side Testing* (OWASP, 2014). *OWASP* merilis daftar yang berisi sepuluh data kerentanan teratas yang disebut *OWASP Top* 10, daftar tersebut dapat berubah seiring dengan perkembangan teknologi.

2.1.4 OWASP TOP 10

OWASP Top 10 merupakan sebuah daftar yang berisikan sepuluh risiko celah keamanan paling kritis yang dapat mengancam keamanan *website.* Menurut OWASP Top 10 - 2021, terdapat sepuluh risiko keamanan aplikasi web yang paling kritis, meliputi: *Broken Access Control, Cryptographic Failures, Injection, Insecure Design, Security Misconfiguration, Vulnerable and Outdated Components, Identification and Authentication Failures, Software and Data Integrity Failures, Security Logging and Monitoring Failures, Server-Side Request Forgery (SSRF).*, seperti yang tertuang pada Gambar 2.1.



Gambar 2.1 Perubahan OWASP Top 10 tahun 2017 ke tahun 2021 (OWASP, 2021)

Berdasarkan Gambar 2.1 dapat dilihat bahwa daftar risiko celah keamanan yang dirilis oleh *OWASP* dapat berubah dan terus berkembang mengikuti perkembangan teknologi. Selain *OWASP Top* 10 terdapat pula daftar kerentanan teratas perangkat lunak yang terdiri dari 25 jenis kerentanan, daftar tersebut disebut sebagai *SANS/CWE Top* 25 ysng dirilis oleh SANS dan CWE.

2.1.5 SANS/CWE Top 25

SANS Institute bersama dengan *Common Weakness Enumeration (CWE)* mengembangkan daftar kerentanan perangkat lunak paling berbahaya yang disebut dengan *CWE/SANS Top 25*. Kerentanan ini berbahaya karena seringkali mudah ditemukan, dieksploitasi, dan dapat memungkinkan *attacker* mengambil alih sistem sepenuhnya, mencuri data, atau mencegah aplikasi bekerja. Tabel berikut menunjukkan 25 Besar CWE 2021.

Tabel 2.1 Daftar Kerentanan SANS/CWE Top 25

Rank	ID	Name
1.	*CWE 787*	*Out of bounds Write*
2.	*CWE 79*	*(Cross-Site-Scripting) Improper Neutralization of Input During Web Page Generation*
3.	*CWE 125*	*Out of bounds Read*
4.	*CWE 20*	*Improper Input Validation*
5.	*CWE 78*	*(OS Command Injection) Improper Neutralization of Special Elements used in an OS Command*
6.	*CWE 89*	*(SQL Injection) Improper Neutralization of Special Elements used in an SQL Command*

Rank	ID	Name
7.	*CWE 416*	*Use After Free*
8.	*CWE 22*	*Improper Limitation of a Pathname to a Restricted Directory (Path Traversal)*
9.	*CWE 352*	*Cross-Site Request Forgery (CSRF)*
10.	*CWE 434*	*Unrestricted Upload of File with Dangerous Type*
11.	*CWE 306*	*Missing Authentication for Critical Function*
12.	*CWE 190*	*Integer Overflow or Wraparound*
13.	*CWE 502*	*Deserialization of Untrusted Data*
14.	*CWE 287*	*Improper Authentication*
15.	*CWE 476*	*NULL Pointer Dereference*
16.	*CWE 798*	*Use of Hard coded Credentials*
17.	*CWE 119*	*Improper Restriction of Operations within the Bounds of a Memory Buffer*
18.	*CWE 862*	*Missing Authorization*
19.	*CWE 276*	*Incorrect Default Permissions*
20.	*CWE 200*	*Information Exposure*
21.	*CWE 522*	*Insufficiently Protected Credentials*
22.	*CWE 732*	*Incorrect Permission Assignment for Critical Resource*
23.	*CWE 611*	*Improper Restriction of XML External Entity Reference*
24.	*CWE 918*	*Server Side Request Forgery (SSRF)*
25.	*CWE 77*	*Improper Neutralization of Special Elements used in a Command (Command Injection)*

Berdasarkan tabel diatas dapat dilihat bawha *The 2021 Common Weakness Enumeration (CWE)* 25 atau *CWE Top 25* berisi daftar masalah paling umum dan sangat berdampak yang dialami selama dua tahun terakhir. Berdasarkan data kerentanan teratas yang dirilis oleh *OWASP* dan *SANS* memiliki kesamaan celah kerentanan sehingga dapat dilakukan pemetaan kesamaan kerentanan tersebut.

2.1.6 Pemetaan Kerentanan Berdasarkan OWASP-SANS/CWE

Pemetaan kerentanan dilakukan berdasarkan daftar sepuluh risiko keamanan *OWASP* pada tahun 2017 dan 25 risiko kerentanan *CWE/SANS* tahun 2019 seperti Tabel 2.2 berikut ini:

Tabel 2.2 Pemetaan berdasarkan OWASP-SANS/CWE (Li, 2020)

Peringkat OWASP	Kerentanan OWASP	ID SANS/CWE
1	*Injection*	*CWE 78: OS Command Injection (Improper Neutralization of Special Elements used in an OS Command)*
		CWE 89: SQL Injection
		CWE 94: Code Injection
		CWE 434: Unrestricted Upload of File with Dangerous Type
		CWE 494: Download of Code Without Integrity Check
		CWE 829: Inclusion of Functionality from Untrusted Control Sphere
2	*Broken Authentication*	*CWE 306: Missing Authentication for Critical Function*
		CWE 307: Improper Restriction of Excessive Authentication Attempts
		CWE 798: Use of Hard coded Credentials
		CWE 807: Reliance on Untrusted Inputs in a Security Decision
		CWE 862: Missing Authorization
		CWE 863: Incorrect Authorization
3	*Sensitive Data Exposure*	*CWE 311: Missing Encryption of Sensitive Data*
		CWE 319: Cleartext Transmission of Sensitive Information

Peringkat OWASP	Kerentanan OWASP	ID SANS/CWE
5	*Broken Access Control*	*CWE 73: External Control of File Name or Path*
		CWE 285: Improper Authorization
6	*Security Misconfiguration*	*CWE 250: Execution with Unnecessary Privileges*
		CWE 676: Use of Potentially Dangerous Function
		CWE 732: Incorrect Permission Assignment for Critical Resource
7	*Cross-Site Scripting (XSS)*	*CWE 79: Improper Neutralization of Input During Web Page Generation (Cross-Site Scripting)*
8	*Insecure Deserialization*	*CWE 134: Use of Externally Controlled Format String*
9	*Using Components with Known Vulnerabilities*	*CWE 190: Integer Overflow or Wraparound*
		CWE 327: Use of a Broken or Risky Cryptographic Algorithm
		CWE 759: Use of a One way Hash Without a Salt

Berdasarkan tabel diatas dapat diihat bahwa dari sepuluh daftar kerentanan yang dirilis oleh *OWASP* dan 25 daftar kerentanan *CWE/SANS* memiliki sembilan daftar kesamaan yang di petakan menurut kategori sepuluh kerentanan *OWASP* dan *ID-CWE*. Selain *OWASP Top* 10 komunitas *OWASP* juga merilis standar pengujian keamanan aplikasi berbasis *website* atau *WSTG (Web Security Testing Guide)* yang disebut sebagai *OWASP Testing Guide* dengan versi terbaru 4.2 yang dirilis pada tanggal 3 Desember 2020.

2.1.7 OWASP Testing Guide Version 4.2

OWASP Testing Guide Version 4.2 adalah panduan pengujian keamanan aplikasi berbasis web yang menjelaskan metodologi *penetration testing* terdiri dari 12 sub kategori pengujian, diantaranya: *Introduction and Objectives, Information Gathering, Configuration and*

Deployment Management Testing, Identity Management Testing, Authentication Testing, Authorization Testing, Session Management Testing, Input Validation Testing, Testing for Error Handling, Testing for weak Cryptography, Business Logic Testing, Client Side Testing.

2.1.8 Penetration Testing

Penetration Testing merupakan serangkaian serangan yang terkontrol digunakan untuk mengidentifikasi celah keamanan atau kerentanan suatu aplikasi berbasis web, jaringan komputer, dan sistem operasi (Utoro et al., 2020). Pada teknik pengujian penetrasi atau *penetration testing* terdapat tiga strategi langkah yang digunakan dalam pengujian *vulnerability assessment* berdasarkan ruang lingkup dan jenis, yaitu *Black Box Testing, White Box Testing,* dan *Grey Box Testing.* Pada pengujian *Black Box Testing* penguji memiliki tujuan untuk melakukan audit keamanan eksternal suatu target yang diuji keamananannya dengan memberikan simulasi berupa tindakan sesuai prosedur seperti yang dilakukan oleh *attacker* secara nyata yang mungkin menyerang tempat lain di luar batas target uji tanpa memiliki pengetahuan tentang target yang hendak diuji. Pengujian *Gray Box Testing* merupakan kombinasi dari dua pengujian, penguji kurang memiliki pengetahuan tentang arsitektur jaringan, namun mengetahui informasi dasar tentang pengujian konfigurasi sistem maupun jaringan (Yunus, 2019). Pada pengujian *White Box Testing* seorang penguji melakukan audit bagian internal dari sistem keamanan dengan mengetahui informasi menyeluruh mengenai konfigurasi jaringan dan sistem yang digunakan, kemudian penguji mensimulasikan tindakan (Guntoro et al., 2020).

2.1.9 Authentication Testing

Pengujian autentikasi atau *authentication testing* merupakan suatu tindakan yang dilakukan untuk mengkonfirmasi segala sesuatu yang mengatakan suatu hal itu adalah benar. Autentikasi objek dapat dilakukan dengan melakukan konfirmasi asal objek tersebut, sedangkan melakukan autentikasi terhadap seseorang dapat dilakukan dengan memverifikasikan identitasnya.

2.1.10 Authorization Testing

Authorization atau otorisasi merupakan suatu konsep yang digunakan untuk memberikan izin akses kepada seseorang yang diizinkan untuk mengakses sumber daya yang dituju.

Pengujian otorisasi berarti memahami proses otorisasi yang dilakukan setelah proses autentikasi berhasil dilakukan dengan menghindari mekanisme otorisasi menggunakan informasi tersebut. Penguji melakukan verifikasi setelah mendapatkan kredensial yang valid mengenai serangkaian peran dan hak istimewa yang terdefinisi secara baik.

2.1.11 Session Management Testing

Session management testing dapat diartikan sebagai sekumpulan kontrol dalam interaksi antara pengguna dengan *website*, mencangkup segala hal mulai dari bagaimana melakukan otentikasi pengguna hingga output yang dikeluarkan. Mekanisme yang digunakan untuk mengontrol dan mempertahankan status pengguna yang berinteraksi dengan aplikasi berbasis web merupakan salah satu komponen inti dalam aplikasi berbasis web. Terdapat beberapa cara aplikasi berbasis web dapat berinteraksi dengan penggunanya, tergantung pada sifat situs, keamanan, dan persyaratan ketersediaan aplikasi.

2.2 Penelitian Terkait

Penulis melakukan studi literatur terhadap 15 literatur, dari studi literatur dipilih tujuh literatur dari tahun 2017 sampai 2021 yang membahas penelitian serupa mengenai metode dalam melakukan pengujian keamanan *website*. Alasan pemilihan tersebut karena dari 15 literatur terdapat beberapa literatur yang memiliki kesamaan metode yang digunakan sehingga dipilih salah satu literatur yang lebih lengkap dalam menyampaikan tahapan atau *tools* yang digunakan pada setiap metode, kemudian dipetakan dalam sebuah tabel perbandingan berikut:

Tabel 2.3 Metode dan *Tools* dalam Analisis Celah Keamanan *Website*

Tahun	Metode	Penulis	*Tools*
2017	*DREAD (Damage Potential, Reproducibility, Exploitability, Affected User, Discoverability)*	Anggariyona Saputra; Nelmiawati; Maya Armys Roma Sitorus	-
2018	*Vulnerability Assesment*	Ari Marta Tania; Didik Setiyadi; Fata Nidaul Khasanah	*WebScarab, WPScan, Exploitwp_find_password,*

Tahun	Metode	Penulis	*Tools*
			Metasploit, THC-Hydra, Slowloris
2019	*OWASP Versi 4*	Moh Yunus	*Zenmap, Mozilla firefox, Google Chrome, Netsparker, HAVIJ 1.15*
2020	*Ethical Hacking (Footprinting dan Vulnerability Scanning)*	Erick Irawadi Alwi; Herdianti; Fitriyani Umar	*CMD (Command Prompt), Zenmap, Whois, Pentest-tools.com, Acunetix, OWASP ZAP*
	ISSAF dan OWASP	Guntoro; Loneli Costaner ; Musfawati	*OWASP ZAP, Mozilla firefox, Google Chrome, Brutus, WebScarab, Wfuzz, Dirb, OWASP CSRF Tester, Zenmap, Whois, Acunetix, SSL Scan, Low Orbit Ion Canon, SQLmap*
	PTES (Penetration Testing Execution Standards)	Setyo Utoro; Bayu Andi Nugroho; Meinawati; Septian Rheno Widianto	*Whois, Pentest-tools.com, OWASP ZAP, Google Chrome, SQLmap, TheHarvester, Nessus Vulnerability Scanner, Nmap, Wireshark*
2021	*OCTAVE (Operationally Critical Threat, Asset and Vulnerability Evaluation)*	Raden Ichsan Achmad Falach; Dr. ir. Lukman Abdurrahman, MIS; Iqbal Santoso, S.Si, MTI	-

Berdasarkan tabel di atas diperoleh kesimpulan pada setiap metode penelitian yang digunakan sebagai berikut. Menurut Saputra, dkk (2017) dalam pengujiaannya menggunakan metode *DREAD* dalam melakukan penilaian ancaman pada *website* dengan tahapan yang meliputi *Secure Transmission, Authentication, Session Management, Cryptography, Data Validation, Denial of Service, Specific Risk of Functionality, Error Handling* diperoleh kesimpulan bahwa metode DREAD dapat memberikan informasi nilai dan jenis kerentanan pada *website* target.

Menurut Tania, dkk (2018) dalam penelitiannya evaluasi keamanan *website* menggunakan metode *Vulnerability Assessment* terdapat dua pengujian yang berstatus tidak

aman, serta pengembangan penelitian selanjutnya dapat dilakukan berupa pengujian keamanan *website* menggunakan *Hacking Methodology*.

Selain itu, menurut Yunus (2019) dalam penelitiannya melakukan analisis celah keamanan aplikasi berbasis web berdasarkan framework *OWASP version* 4 dengan 5 kategori penelitian yaitu *Authentication_Testing*, *Authorization Testing*, *Session Management Testing*, *Input Validation Testing*, dan *Error Handling* dengan kombinasi *security tools project* diperoleh kesimpulan dari hasil pengujian yaitu metode *OWASP version* 4 dapat digunakan sebagai standar dalam melakukan penilaian analisis kerentanan dan keamanan suatu aplikasi berbasis web.

Adapun menurut Alwi, dkk (2020) dalam penelitian yang dilakukan mengenai analisis keamanan *website* menggunakan metode *Ethical hacking* menggunakan teknik *footprinting* dan *vulnerability scanning*, diperoleh kesimpulan dari hasil penelitian bahwa *website* target memiliki celah keamanan diantaranya *CORS (Cross-Origin Resource Sharing) origin validation failure* dengan tingkat risiko tinggi, *X-Frame-Options Header Not Set* dengan tingkat risiko sedang, *Directory listing is enabled* dengan tingkat risiko sedang, *HTML form without CSRF protection* dengan tingkat risiko sedang, *WordPress username enumeration* dengan tingkat risiko sedang, dan *Cookie No HttpOnly Flag* dengan tingkat risiko rendah.

Berdasarkan penelitian yang dilakukan Guntoro, dkk (2020) dalam melakukan analisis keamanan pada web server *Open Journal System* pada suatu universitas dengan metode pengujian yang digunakan yaitu *ISSAF* dan *OWASP*. Berdasarkan pengujian yang telah dilakukan menggunakan metode pengujian *ISSAF* terdapat kerentanan terhadap serangan *DoS* pada web server, selain itu hasil pengujian menggunakan metode *OWASP* menunjukkan bahwa web server tergolong aman, tetapi terdapat kelemahan ketika *user* melakukan kesalahan berulang pada saat *login* sistem tidak dapat memblokirnya.

Berdasarkan penelitian yang dilakukan Utoro, dkk (2020) dalam melakukan analisis keamanan *website e-learning* SMKN 1 Cibatu, berdasarkan pengujian yang dilakukan menggunakan metode *Penetration Testing Execution* diperoleh kesimpulan bahwa metode *PTES* dapat digunakan oleh aplikasi berbasis web sebagai standar penilaian keamanan, untuk pengembangan lebih lanjut dapat dilakukan penelitian menggunakan metode yang berbeda seperti *ISSAF (Information System Security Assessment Framework)* atau *OWASP (Open Web Application Security Project).*

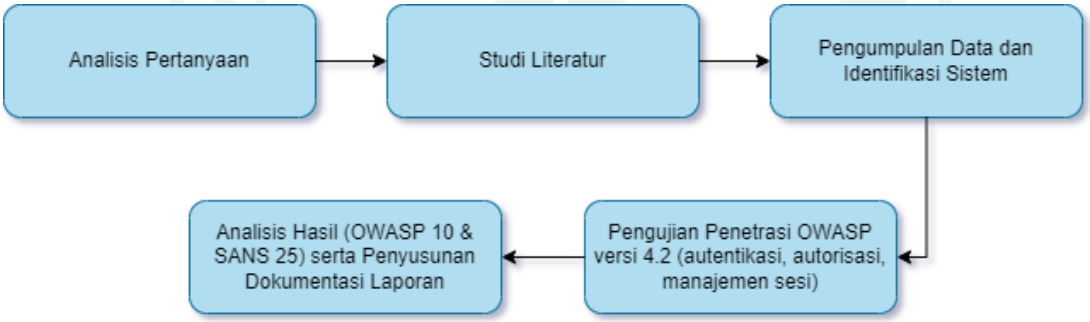
Berdasarkan penelitian yang dilakukan Ichsan, dkk (2021) menggunakan metode *OCTAVE Allegro* dalam menganalisis risiko dan merancang kontrol keamanan menggunakan

8 tahap pengujian, meliputi membangun kriteria pengukuran risiko, mengembangkan profil aset informasi, mengidentifikasi kontainer dari aset informasi, mengidentifikasi *area of concern*, mengidentifikasi skenario ancaman, mengidentifikasi risiko, menganalisa risiko, memilih pendekatan mitigasi, diperoleh kesimpulan bahwa dua jenis penanganan berupa *mitigate* dan *defer* dapat diterapkan pada *website* target.

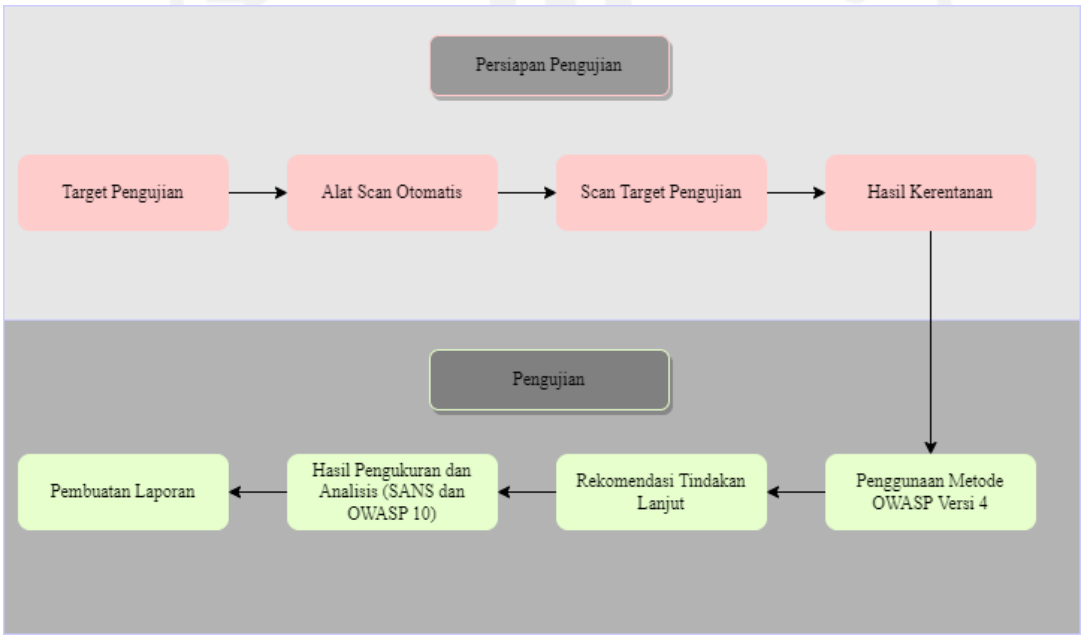
BAB III
METODOLOGI PENELITIAN

3.1 Metode Penelitian

Tahapan penelitian secara garis besar digambarkan pada Gambar 3.1. Penelitian ini mengadaptasi diagram proses pengujian yang dilakukan Yunus (2019) dengan perbedaan yang terletak pada versi pengujian terbaru Versi 4.2 yang dirilis oleh *OWASP* dengan fokus pengujian terletak pada pengujian autentikasi, pengujian otorisasi, dan pengujian manajemen sesi serta penambahan standar keamanan *SANS* sebagai perbandingan dan pelengkap daftar sepuluh kerentanan keamanan yang dirilis oleh *OWASP*.



Gambar 3.1 Tahapan Penelitian



Gambar 3.2 Proses Pengujian Penetrasi

Berdasarkan Gambar 3.1 dapat dilihat bahwa tahapan penelitian ini secara garis besar terdiri dari 5 tahap, sebagai berikut:

A. Analisis Pertanyaan

Pada tahapan analisis pertanyaan dilakukan pembuatan pertanyaan 5W + 1H dengan disertai jawaban yang berkaitan dengan penelitian yang akan dilaksanakan. Pertanyaan tersebut meliputi:

1. Apa yang diteliti dalam penelitian ini?
   Jawaban : Penelitian ini hendak meneliti standar keamanan suatu *website*.
2. Mengapa penelitian ini perlu dilakukan?
   Jawaban : Penelitian ini perlu dilakukan karena sebuah *website* mudah terkena risiko keamanan jika tidak memenuhi standar keamanan yang tersedia, untuk itu perlu dilakukan *filtering* input agar tidak mudah terkena SQL Injection, selain itu untuk menganalisis apakah *website* tersebut sudah dilengkapi dengan keamanan yang memenuhi standar yaitu dengan menggunakan acuan standar keamanan OWASP dan SANS.
3. Siapa yang dijadikan objek dalam penelitian?
   Jawaban : Yang hendak dijadikan objek penelitian adalah sebuah website "Cek-ejaan.com".
4. Dimana penelitian ini akan dilaksanakan?
   Jawaban : Penelitian akan dilaksanakan di lingkungan Universitas Islam Indonesia.
5. Kapan penelitian ini dilaksanakan?
   Jawaban : Penelitian ini telah dimulai pada bulan Februari 2021 atau awal semester enam dengan target selesai pada akhir semester tujuh.
6. Bagaimana alur dalam penelitian ini?
   Jawaban : alur penelitian telah dituangkan ke dalam *flowchart* tahapan penelitian seperti yang tercantum pada Gambar 3.1.

B. Studi Literatur

Pada tahapan studi literatur ini memiliki tujuan untuk menjelaskan kajian pustaka yang dapat digunakan untuk mendukung penelitian berdasarkan teori-teori penunjang yang diperoleh dari membaca jurnal, laporan penelitian terdahulu, buku, artikel maupun situs web di internet. Hasil dari dilakukannya studi literatur ini berupa sekumpulan referensi terkait berdasarkan rumusan masalah, dengan tujuan sebagai dasar teori dalam melakukan penelitian

serta memperkuat permasalahan (Dirgahayu et al., 2015). Pada tahap studi literatur ini dilakukan pengelompokkan terhadap 15 literatur terkait yang membahas mengenai metode pengujian celah keamanan *website* berdasarkan metode, tahapan pengujian dan *tools* yang digunakan.

C. Pengumpulan Data dan Identifikasi Sistem

Pada tahapan pengumpulan data dan identifikasi sistem ini dilakukan pengumpulan data target dan identifikasi *website* cek-ejaan.com dengan menggunakan *tools Nmap* yang terdapat pada *Kali Linux.* Adapun analisis alat dan kebutuhan sistem yang dibutuhkan dalam penelitian meliputi:

1. Perangkat Keras

Spesifikasi perangkat keras yang digunakan dalam pengumpulan data penelitian tercantum pada Tabel 3.1 berikut:

Tabel 3.1 Spesifikasi Perangkat Keras

Komponen	Spesifikasi
CPU	*Intel® Core™ i5 8250U Processor (6M Cache, up to 3.40 GHz)*
RAM	*8GB DDR4 2133MHz SDRAM*
Storage	*1TB SATA HDD 5400RPM*
Graphic	*Discrete graphics Nvidia GT 930MX 2GB / Nvidia GT 940MX 2GB*

Berdasarkan tabel di atas dapat dilihat spesifikasi perangkat keras yang akan digunakan dalam penelitian ini terdiri dari *CPU (Central Processor Unit)* yang memiliki spesifikasi *Intel® Core™ i5* dengan kapasitas memori internal pada *CPU* sebesar 6M dan *core speed up to 3.60 GHz,* selain itu komponen *RAM (Random Access Memory)* sebesar 8*GB* dan komponan penyimpanan *HDD (Hard Disk Drive)* sebesar 1*TB,* komponen *graphic* dengan spesifikasi *discrete Nvidia.*

2. Perangkat Lunak

Perangkat lunak yang digunakan dalam penelitian ini yaitu *Kali Linux*. *Kali Linux* merupakan distribusi *linux* berbasis *debian* yang bersifat *open source*, memiliki beberapa ratus *tools* yang digunakan untuk keamanan informasi, seperti pengujian penetrasi, forensik komputer, riset keamanan, rekayasa terbalik, dan audit keamanan tingkat lanjut.

D. Pengujian Penetrasi dengan metode *OWASP* versi 4.2 (*SANS* dan *OWASP* 10)

Pada tahap pengujian penetrasi ini terbagi menjadi dua tahap yaitu persiapan dan pengujian seperti yang tertera pada Gambar 3.2. Pada tahap persiapan dilakukan identifikasi celah keamanan *(vulnerability identification)* menggunakan alat scan otomatis *OWASP ZAP (Zed Attack Proxy).* Pada tahap pengujian dilakukan berdasarkan metode *OWASP* versi 4.2 dengan fokus pada tiga kategori pengujian yaitu autentikasi, otorisasi, dan manajemen sesi, kemudian dilakukan analisis hasil dan pengukuran berdasarkan standar keamanan *SANS* yang berisi 25 celah keamanan teratas *CWE/SANS* serta sepuluh kerentanan teratas oleh *OWASP Top* 10. Metode *OWASP* versi 4.2 sebagai panduan pengujian memiliki 11 kategori pengujian seperti yang tertera pada Tabel 3.2 berikut:

Tabel 3.2 Panduan Pengujian Keamanan Web OWASP Versi 4.2 (OWASP, 2020)

Kategori Pengujian	Kode Pengujian	Tahapan pengujian
Information Gathering	*WSTG-INFO-01*	*Conduct Search Engine Discovery and Reconnaissance for Information Leakage*
	WSTG-INFO-02	*Fingerprint Web Server*
	WSTG-INFO-03	*Review Webserver Metafiles for Information Leakage*
	WSTG-INFO-04	*Enumerate Applications on Webserver*
	WSTG -INFO-05	*Review Webpage Comments and Metadata for Information Leakage*
	WSTG-INFO-06	*Identify Application Entry Points*

Kategori Pengujian	Kode Pengujian	Tahapan pengujian
	WSTG-INFO-07	*Map Execution Paths through Application*
	WSTG-INFO-08	*Fingerprint Web Application Framework*
	WSTG-INFO-09	*Fingerprint Web Application*
	WSTG-INFO-10	*Map Application Architecture*
Configurationand Deploy Management Testing	*WSTG-CONF-01*	*Test Network Infrastructure Configuration*
	WSTG-CONF-02	*Test Application Platform Configuration*
	WSTG-CONF-03	*Test File Extensions Handling for Sensitive Information*
	WSTG-CONF-04	*Backup and Unreferenced Files for Sensitive Information*
	WSTG-CONF-05	*Enumerate Infrastructure and Application Admin Interfaces*
	WSTG-CONF-06	*Test HTTP Methods*
	WSTG-CONF-07	*Test HTTP Strict Transport Security*
	WSTG-CONF-08	*Test RIA Cross Domain Policy*
	WSTG-CONF-09	*Test File Permission*
	WSTG-CONF-10	*Test for Subdomain Takeover*
	WSTG-CONF-11	*Test Cloud Storage*

Kategori Pengujian	Kode Pengujian	Tahapan pengujian
Identity Management Testing	*WSTG-IDNT-01*	*Test Role Definitions*
	WSTG-IDNT-02	*Test User Registration Process*
	WSTG-IDNT-03	*Test Account Provisioning Process*
	WSTG-IDNT-04	*Testing for Account Enumeration and Guessable User Account*
	WSTG-IDNT-05	*Testing for Weak or Unenforced Username Policy*
Authentication Testing	*WSTG-ATHN-01*	*Testing for Credentials Transported over an Encrypted Channel*
	WSTG-ATHN-02	*Testing for Default Credentials*
	WSTG-ATHN-03	*Testing for Weak lock out mechanism*
	WSTG-ATHN-04	*Testing for Bypassing Authentication schema*
	WSTG-ATHN-05	*Testing for Vulnerable Password Functionality*
	WSTG-ATHN-06	*Testing for Browser Cache Weaknesses*
	WSTG-ATHN-07	*Testing for Weak Password Policy*
	WSTG-ATHN-08	*Testing for Weak Security Question Answer*
	WSTG-ATHN-09	*Testing for weak Password Change or Reset Functionalities*
	WSTG-ATHN-10	*Testing for Weaker Authentication in Alternative Channel*

Kategori Pengujian	Kode Pengujian	Tahapan pengujian
Authorization Testing	*WSTG-ATHZ-01*	*Testing Directory Traversal File Include*
	WSTG-ATHZ-02	*Testing for Bypassing Authorization Schema*
	WSTG-ATHZ-03	*Testing for Privilege Escalation*
	WSTG-ATHZ-04	*Testing for Insecure Direct Object References*
Session Management Testing	*WSTG-SESS-01*	*Testing for Bypassing Session Management Schema*
	WSTG-SESS-02	*Testing for Cookies Attributes*
	WSTG-SESS-03	*Testing for Session Fixation*
	WSTG-SESS-04	*Testing for Exposed Session Variables*
	WSTG-SESS-05	*Testing for Cross Site Request Forgery*
	WSTG-SESS-06	*Testing for Logout Functionality*
	WSTG-SESS-07	*Test Session Timeout*
	WSTG-SESS-08	*Testing for Session Puzzling*
	WSTG-SESS-09	*Testing for Session Hijacking*
Input Validation Testing	*WSTG-SESS-01*	*Testing for Reflected Cross Site Scripting*
	WSTG-SESS-02	*Testing for Stored Cross Site Scripting*

Kategori Pengujian	Kode Pengujian	Tahapan pengujian	
	WSTG-SESS-03	*Testing for HTTP Verb Tampering*	
	WSTG-SESS-04	*Testing for HTTP Parameter Pollution*	
	WSTG-SESS-05	*Testing for SQL Injection*	*Testing for Oracle*
			Testing for MySQL
			Testing for SQL Server
			Testing for PostgreSQL
			Testing for MS Access
			Testing for NoSQL injection
			Testing for ORM Injection
			Testing for Client-side
	WSTG-SESS-06	*Testing for LDAP Injection*	
	WSTG-SESS-07	*Testing for XML Injection*	
	WSTG-SESS-08	*Testing for SSI Injection*	
	WSTG-SESS-09	*Testing for XPath Injection*	
	WSTG-SESS-10	*Testing for IMAP/SMTP Injection*	
	WSTG-SESS-11	*Testing for Code Injection*	*Testing for Local File Inclusion*
			Testing for Remote File Inclusion
	WSTG-SESS-12	*Testing for Command Injection*	

Kategori Pengujian	Kode Pengujian	Tahapan pengujian
	WSTG-SESS-13	*Testing for Format String Injection*
	WSTG-SESS-14	*Testing for Incubated Vulnerabilities*
	WSTG-SESS-15	*Testing for HTTP Splitting Smuggling*
	WSTG-SESS-16	*Testing for HTTP Incoming Requests*
	WSTG-SESS-17	*Testing for Host Header Injection*
	WSTG-SESS-18	*Testing for Server-side Template Injection*
	WSTG-SESS-19	*Testing for Server-side Requests Forgery*
Testing for Error Handling	*WSTG-ERRH-01*	*Testing for Improper Error Handling*
	WSTG-ERRH-02	*Testing for Stack Traces*
Testing for Weak Cryptography	*WSTG-CRYP-01*	*Testing for Weak Transport Layer Security*
	WSTG-CRYP-02	*Testing for Padding Oracle*
	WSTG-CRYP-03	*Testing for Sensitive Information Sent via Unencrypted Channels*
	WSTG-CRYP-04	*Testing for Weak Encryption*
Business Logic Testing	*WSTG-BUSL-01*	*Test Business Logic Data Validation*
	WSTG-BUSL-02	*Test Ability to Forge Requests*

Kategori Pengujian	Kode Pengujian	Tahapan pengujian
	WSTG-BUSL-03	*Test Integrity Checks*
	WSTG-BUSL-04	*Test for Process Timing*
	WSTG-BUSL-05	*Test Number of Times a Function Can Be Used L mits*
	WSTG-BUSL-06	*Testing for the Circumvention of Work Flows*
	WSTG-BUSL-07	*Test Defenses Against Application Misuse*
	WSTG-BUSL-08	*Test Upload of Unexpected File Types*
	WSTG-BUSL-09	*Test Upload of Malicious Files*
Client Side Testing	*WSTG-CLNT-01*	*Testing for DOM-Based Cross Site Scripting*
	WSTG-CLNT-02	*Testing for JavaScript Execution*
	WSTG-CLNT-03	*Testing for HTML Injection*
	WSTG-CLNT-04	*Testing for Client-side URL Redirect*
	WSTG-CLNT-05	*Testing for CSS Injection*
	WSTG-CLNT-06	*Testing for Client-side Resource Manipulation*
	WSTG-CLNT-07	*Testing Cross Origin Resource Sharing*
	WSTG-CLNT-08	*Testing for Cross Site Flashing*

Kategori Pengujian	Kode Pengujian	Tahapan pengujian
	WSTG-CLNT-09	*Testing for Clickjacking*
	WSTG-CLNT-10	*Testing WebSockets*
	WSTG-CLNT-11	*Testing Web Messaging*
	WSTG-CLNT-12	*Testing Browser Storage*
	WSTG-CLNT-13	*Testing for Cross Site Script Inclusion*

Berdasarkan Tabel 3.2 dapat dilihat bahwa metode *OWASP* Versi 4.2 berisi 11 kategori yang digunakan dalam penugujian celah keamanan *website* atau sering disebut sebagai *WSTG (Web Security Testing Guide)*, masing-masing kategori memiliki beberapa jenis pengujian dengan total pengujian 104 pengujian yang dapat berubah dan berkembang seiring dengan perkembangan teknologi.

E. Analisis Hasil dan Pengumpulan Dokumentasi Laporan

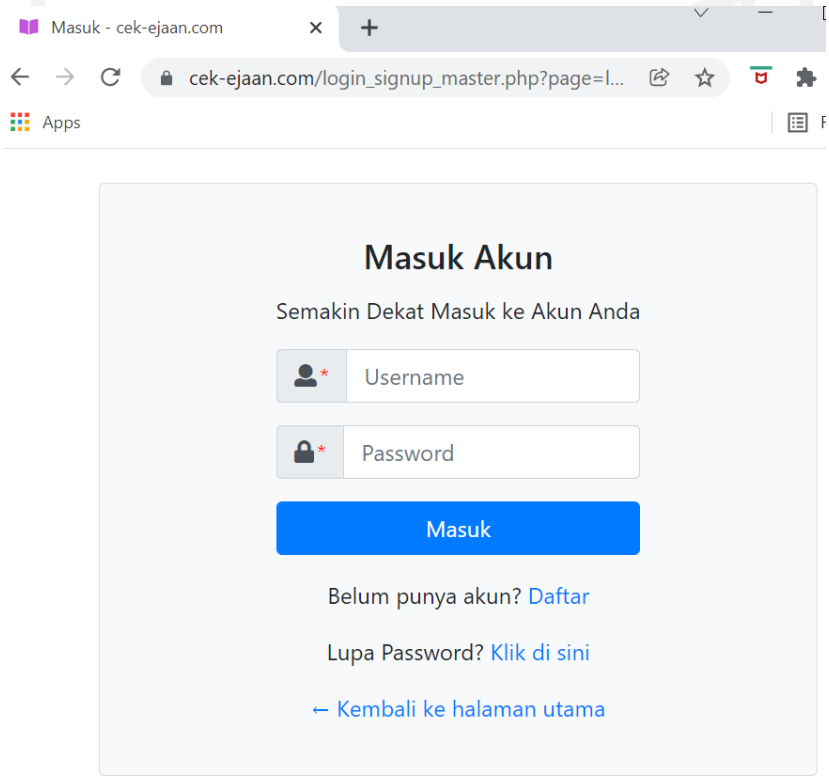
Pada tahapan analisis hasil dan pengumpulan dokumentasi laporan pengujian ini dilakukan pembuatan laporan dan dokumentasi pengujian menggunakan metode *OWASP* versi 4.2 dan berdasarkan daftar celah keamanan teratas *CWE/SANS Top* 25 dan *OWASP Top* 10.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Hasil

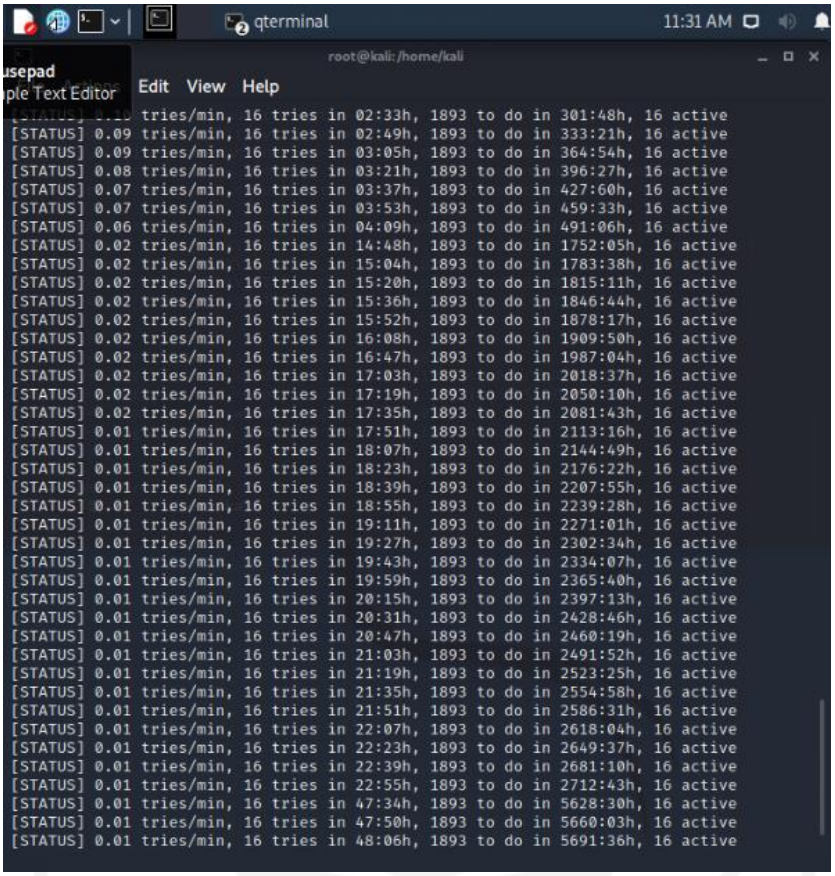
4.1.1 Authentication Testing

Hasil yang diperoleh dari proses *authentication testing* dengan kategori *pengujian* kredensial yang ditransfer melalui saluran terenkripsi *(OTG-AUTHN-001)* yang telah dilakukan pada *website* cek-ejaan.com yaitu pada halaman login *website* menerapkan *HTTPS* seperti pada Gambar 4.1.



Gambar 4.1 Halaman login *website*

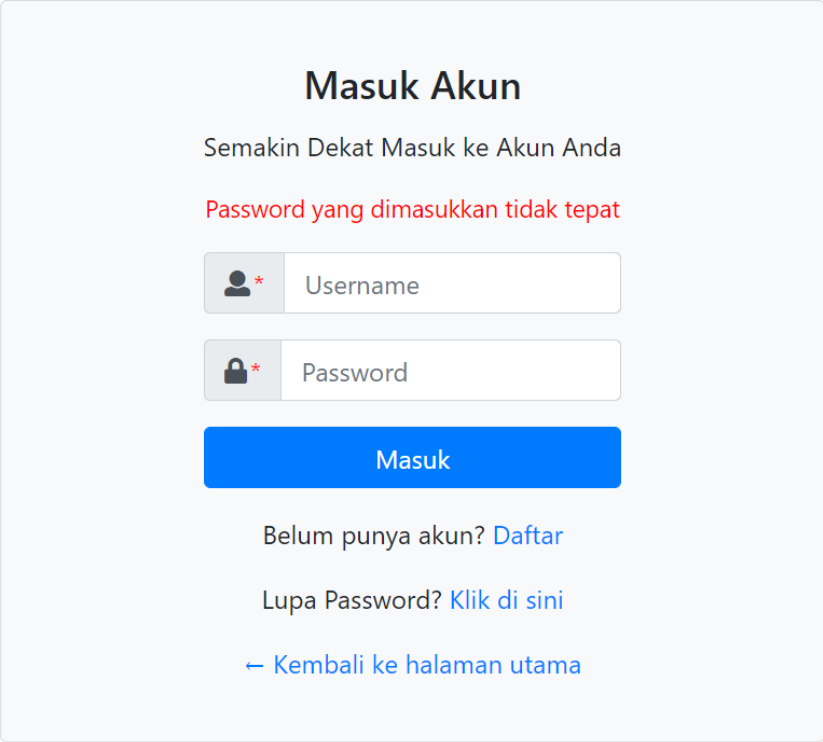
Gambar 4.1 menunjukkan halaman *login website* cek-ejaan.com yang diakses melalui saluran terenkripsi sehingga pada pengujian *WSTG-ATHN-01* dinyatakan tidak memiliki celah keamanan. Pengujian kategori kedua yaitu pengujian untuk kredensial *default (WSTG-ATHN-02)* dengan menggunakan *tools THC-Hydra* seperti Gambar 4.2.



Gambar 4.2 Hasil prediksi *username default* menggunakan *THC-Hydra*

Berdasarkan Gambar 4.2 dapat dilihat bahwa pada pengujian *WSTG-ATHN-02* dengan memprediksi kredensial *default* dan validasi dari halaman *login* menggunakan *tools THC-Hydra* diperoleh hasil tidak terdapat *username default* sehingga dalam kategori pengujian ini tidak memiliki celah keamanan.

Kategori pengujian ketiga dengan fokus pengujian *authentication* yaitu pengujian mekanisme penguncian yang lemah *(WSTG-ATHN-03)* dengan melakukan beberapa kali *login* menggunakan *password* yang salah untuk menguji kemampuan mekanisme penguncian akun seperti Gambar 4.3.



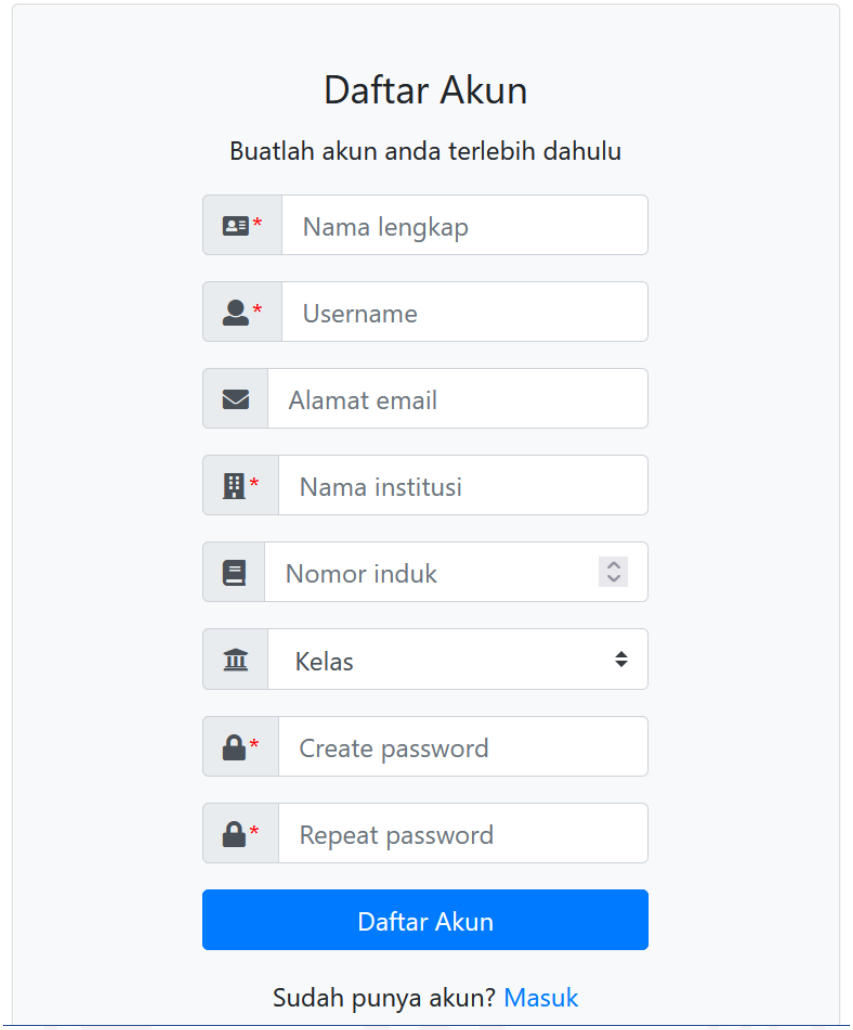
Gambar 4.3 Halaman login ketika *password* salah

Gambar 4.3 menunjukkan halaman *login website* cek-ejaan.com ketika dilakukan sepuluh kali percobaan *login* menggunakan *password* yang salah, kemudian berhasil login dengan *password* yang benar sehingga dari pengujian ini *website* dinyatakan terdapat celah keamanan karena tidak adanya mekanisme penguncian akun pada saat *user invalid login*.

Pengujian keempat yaitu pengujian untuk melewati skema autentikasi *(WSTG-ATHN-04)* menggunakan *tools SQLMap* seperti pada Gambar 4.4 dan *Mozilla firefox* seperti Gambar 4.5.

Gambar 4.4 Hasil modifikasi parameter *URL* menggunakan *SQLMap*

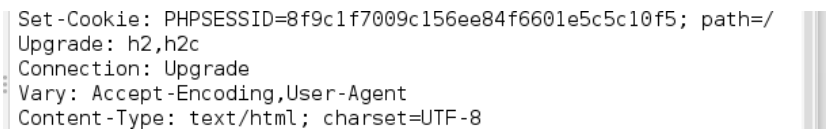
Berdasarkan Gambar 4.4 dapat dilihat bahwa pengujian dengan memodifikasi parameter *URL* menggunakan *tools SQLMap* gagal dilakukan karena terdeteksi oleh *WAF (Web Application Firewall)* yang terdapat pada *website* cek-ejaan.com.



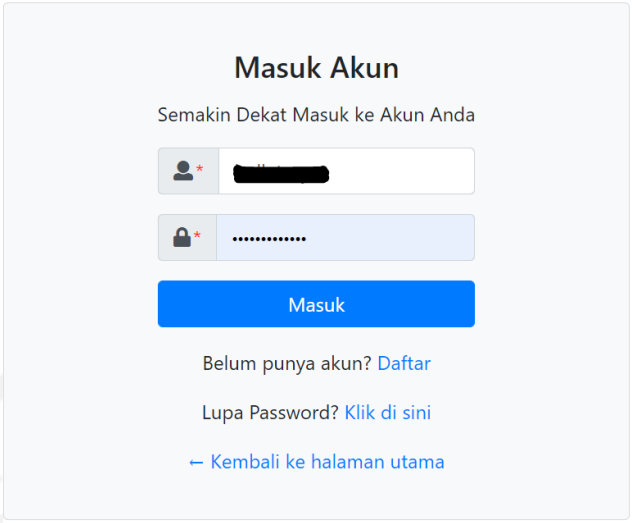
Gambar 4.5 Hasil akses halaman langsung menggunakan *Mozilla firefox*

Namun, berdasarkan Gambar 4.5 dapat dilihat pada saat penjelajahan paksa menggunakan browser Mozilla firefox dengan URL yang mengarah pada halaman *reset password* berhasil dilakukan tetapi mengarah pada halaman *create account* sehingga pada pengujian ini dinyatakan tidak lolos.

Kategori pengujian *authentication* kelima yaitu pengujian pada fungsi ingat *password (WSTG-ATHN-05)* menggunakan *tools OWASP ZAP* seperti yang tertuang pada Gambar 4.6 dan *Google Chrome* pada Gambar 4.7.



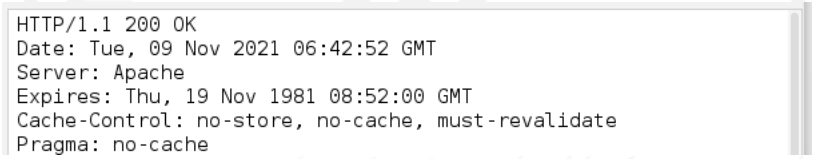
Gambar 4.6 Hasil *OWASP ZAP* Set-cookie *website*



Gambar 4.7 Hasil *autocomplete* aktif

Gambar 4.6 menunjukkan *set-cookie* yang tersimpan dalam *website* cek-ejaan.com berbentuk nilai *hash* sehingga *website* dikatakan cukup aman. Namun, berdasarkan Gambar 4.7 dapat dilihat bahwa status *autocomplete on* sehingga ditemukan celah keamanan pada pengujian ini.

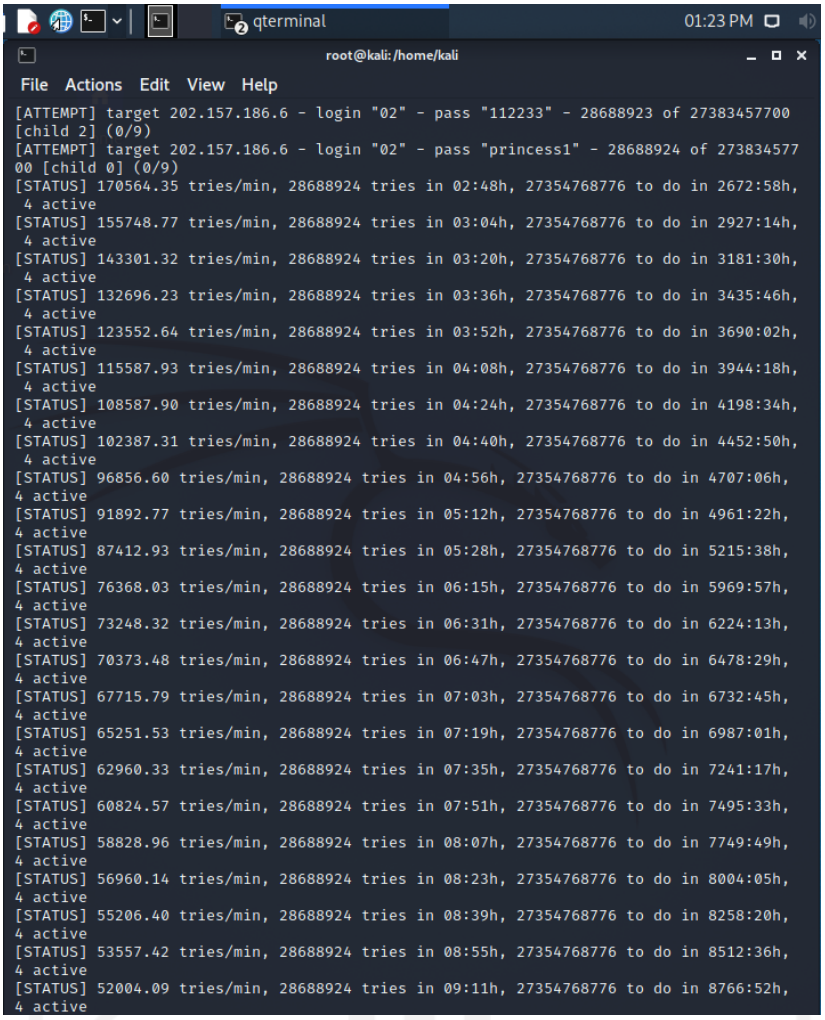
Kategori pengujian *authentication* keenam yaitu pengujian untuk *cache browser* yang lemah *(WSTG-ATHN-06)* menggunakan *tools Google Chrome* dan *OWASP ZAP* seperti pada Gambar 4.8.



Gambar 4.8 Hasil *OWASP ZAP no cache browser*

Gambar 4.8 menunjukkan hasil pengujian menggunakan *tools OWASP ZAP* dapat dilihat bahwa tidak ditemukan *cache browser*. Selain itu, pada saat memeriksa dari tombol *back* menggunakan *tools Google Chrome* sumber daya tidak dapat ditampilkan dan diharuskan melalui skema autentikasi atau *login* kembali sehingga pada pengujian ini tidak ditemukan celah keamanan.

Kategori pengujian *authentication* ketujuh yaitupengujian kebijakan kata sandi yang lemah *(WSTG-ATHN-07)* menggunakan *tools THC-Hydra* dengan melakukan *brute force* pada *website* cek-ejaan.com seperti pada gambar Gambar 4.9

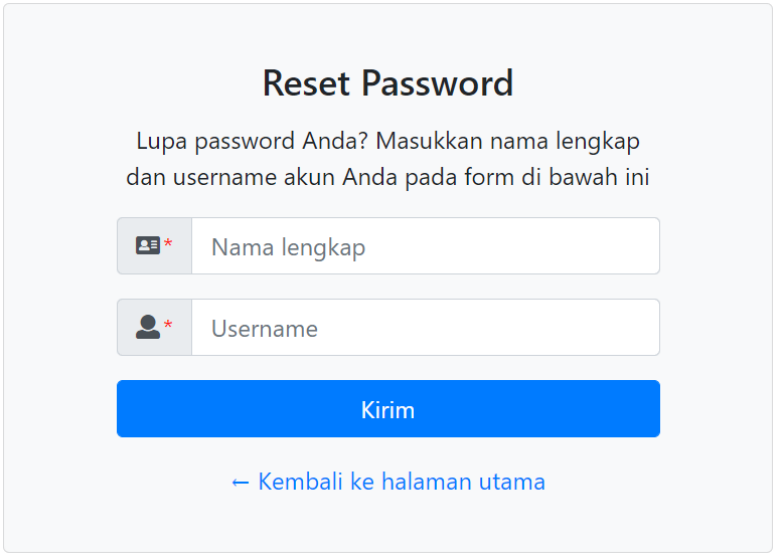


Gambar 4.9 Hasil *brute force* menggunakan *THC-Hydra*

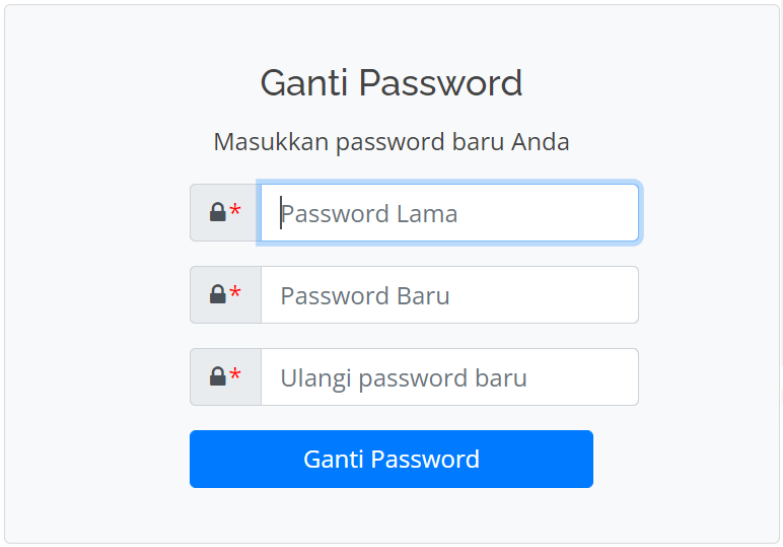
Berdasarkan Gambar 4.9 dapat dilihat bahwa proses *brute force* tidak ditemukan *password* yang sesuai, selain itu proses ini membutuhkan waktu yang lama. Hal ini dapat disebabkan oleh *firewall* yang digunakan dalam *website* yang dapat memblokir aktivitas mencurigakan, sehingga pada pengujian ini tidak dilanjutkan serta dapat disimpulkan bahwa tidak terdapat celah pada pengujian.

Kategori pengujian *authentication* berikutnya yaitu pengujian untuk jawaban pertanyaan keamanan yang lemah *(WSTG-ATHN-08)* dengan memeriksa skema pertanyaan keamanan pada *website* cek-ejaan.com diperoleh hasil bahwa *website* tidak menerapkan skema pertanyaan keamanan sehingga tidak terdapat kemungkinan celah pada kategori pengujian ini. Selanjutnya, kategori pengujian *authentication* sembilan yaitu pengujian untuk fungsi perubahan kata sandi yang lemah *(WSTG-ATHN-09)* dengan melakukan pemeriksaan

mekanisme perubahan atau pengaturan ulang kata sandi yang terdapat pada halaman lupa *password* seperti yang tertera pada Gambar 4.10 dan ganti *password* pada Gambar 4.11.



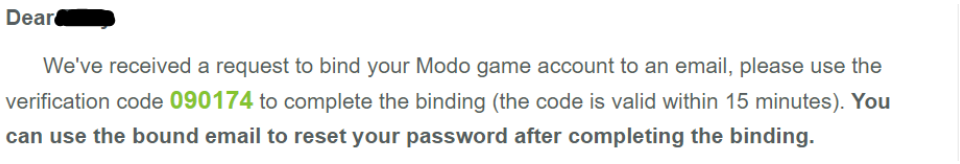
Gambar 4.10 Halaman lupa *password*



Gambar 4.11 Halaman ganti *password*

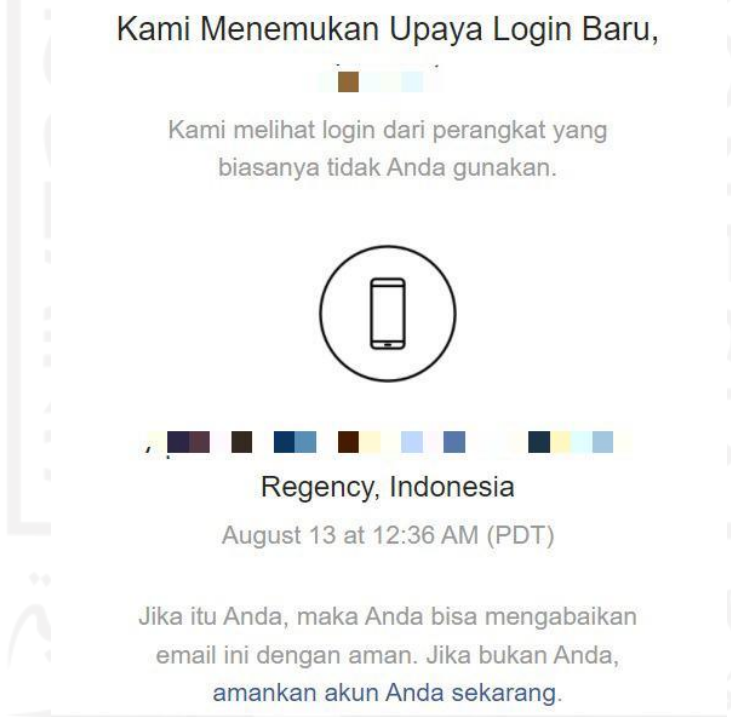
Berdasarkan Gambar 4.10 di atas dapat dilihat bahwa pada halaman lupa *password* hanya membutuhkan nama lengkap dan *username* untuk dapat merubah kata sandi. Gambar 4.11 menunjukkan bahwa fitur ganti *password* setelah *login* juga hanya membutuhkan *password* lama dan *password* baru tanpa melalui autentikasi dua faktor sehingga terdapat kemungkinan

celah keamanan karena mekansime perubahan kata sandi dianggap mudah atau lemah. Autentikasi dua faktor yang dimaksud dapat dilihat seperti Gambar 4.12.



Gambar 4.12 Contoh autentikasi dua faktor menggunakan kode

Berdasarkan Gambar 4.12 dapat dilihat bahwa fitur ganti *password* pada aplikasi x menggunkan autentikasi dua faktor yaitu dengan menggunakan kode verifikasi yang dikirimkan melalui email pengguna. Contoh autentikasi dua faktor yang lain seperti pada Gambar 4.13 di bawah ini.



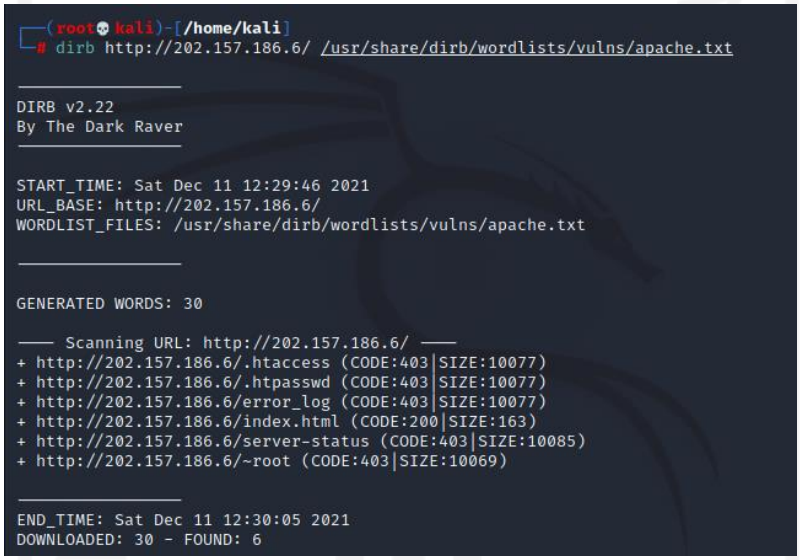
Gambar 4.13 Contoh autentikasi dua faktor deteksi upaya *login*

Berdasarkan Gambar 4.13 dapat dilihat bahwa pada aplikasi x berhasil mendeteksi upaya login baru pada perangkat yang tidak biasanya digunakan melalui media *email*. *Email* tersebut berisi *URL* yang mengarah pada upaya pengamanan akun apabila *user* tidak mengetahui atau tidak melakukan *login* menggunakan perangkat lain.

Kategori pengujian *authentication* terakhir yaitu pengujian autentikasi yang lemah pada saluran alternatif *(WSTG-ATHN-10).* Pada pengujian ini tidak ditemukan kemungkinan celah keamanan karena *website* cek-ejaan.com tidak memiliki saluran alternatif atau saluran autentikasi lain.

4.1.2 Authorization Testing

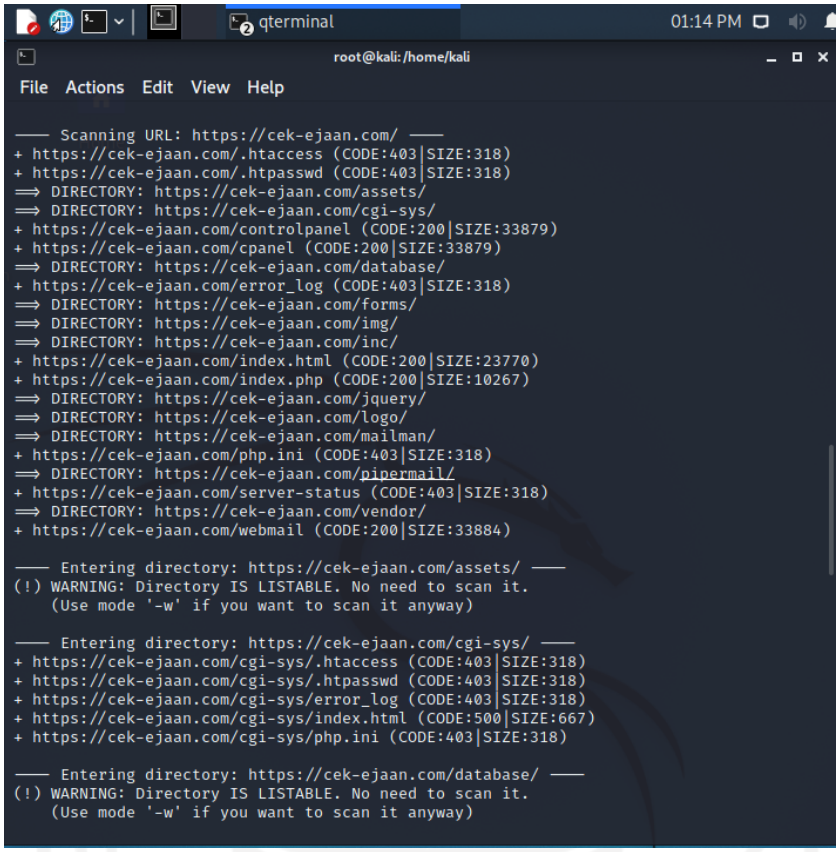
Hasil yang diperoleh dari proses *authorization testing* dengan kategori pengujian direktori traversal *(WSTG-ATHZ-01)* menggunakan *tools Dirb* dengan mengidentifikasi lokasi file *root directory* atau *root* dokumen web yang tertuang dalam Gambar 4.14.



Gambar 4.14 Hasil pencarian *root directory* menggunakan *tools Dirb*

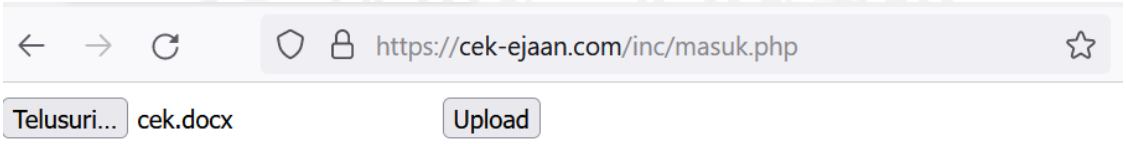
Berdasarkan Gambar 4.14 dapat dilihat bahwa terdapat *URL* yang mengarah pada *root directory* sehingga dalam pengujian ini ditemukan kemungkinan celah keamanan.

Selanjutnya, pengujian autorisasi dengan kategori pengujian untuk melewati skema otorisasi *(WSTG-ATHZ-02)* menggunakan *tools Dirb* dengan melakukan percobaan akses pada sumber daya khusus tanpa proses autentikasi seperti Gambar 4.15, kemudian menggunakan *tools Mozilla Firefox* untuk mencoba mengoperasikan fungsi pada sumber daya khusus seperti pada Gambar 4.16 dan Gambar 4.17.



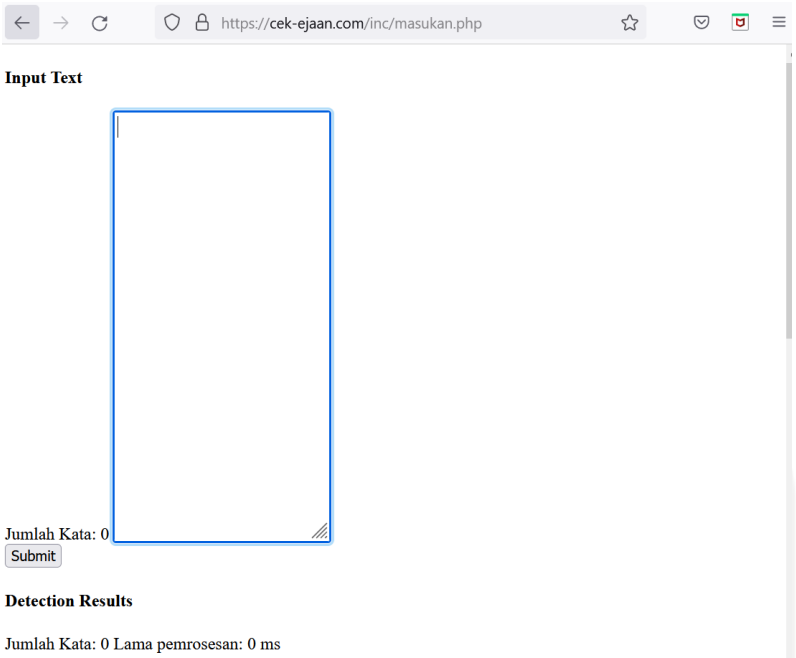
Gambar 4.15 Hasil akses sumber daya khusus menggunakan *tools Dirb*

Berdasarkan Gambar 4.15 dapat dilihat bahwa terdapat beberapa *URL* yang mengarah pada beberapa *directory website* cek-ejaan.com terutama pada direktori khusus seperti memasukan dokumen untuk cek ejaan yang hanya bisa diakses oleh pangguna *website*. Namun, dengan menggunakan *tools Dirb* beberapa direktori tersebut dapat diakses tanpa melalui skema autentikasi.



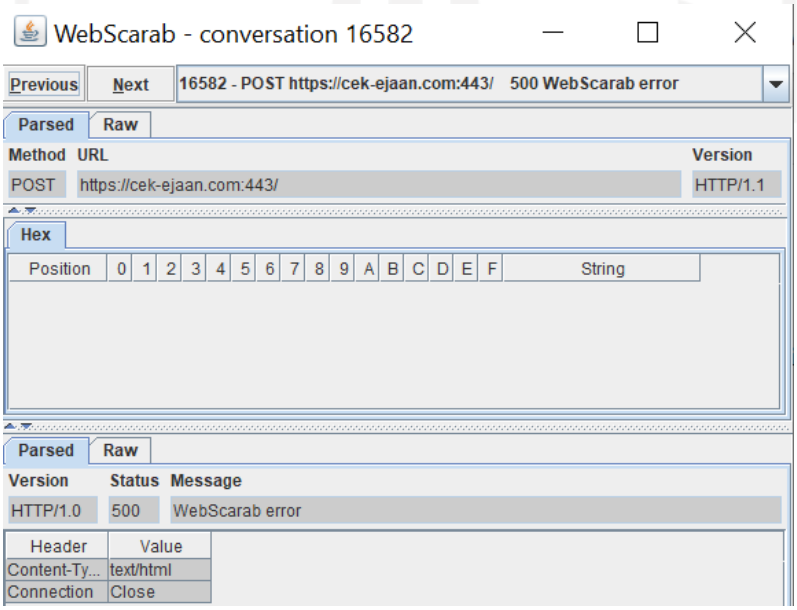
Gambar 4.16 Hasil akses sumber daya khusus menggunakan *tools Mozilla Firefox*

Gambar 4.16 menunjukkan hasil akses *URL* menggunakan *tools Mozilla Firefox* yang sebelumnya ditemukan pada *tools Dirb*. Namun, pada *website* cek-ejaan.com tidak ada fungsi *upload* dokumen.



Gambar 4.17 Hasi upload dokumen pada sumber daya khusus

Gambar 4.17 menunjukkan hasil dari *upload* dokumen yang mengarah pada halaman analisis teks pada *website* cek-ejaan.com yang hanya dapat diakses oleh pengguna sehingga berdasarkan hasil yang diperoleh, *website* memiliki kemungkinan celah keamanan pada kategori pengujian ini. Hasil pengujian *authorization* dengan kategori pengujian eksalasi hak istimewa *(WSTG-ATHZ-03)* menggunakan *tools WebScarab* seperti Gambar 4.18.



Gambar 4.18 Hasil manipulasi hak istimewa menggunakan *tools WebScarab*

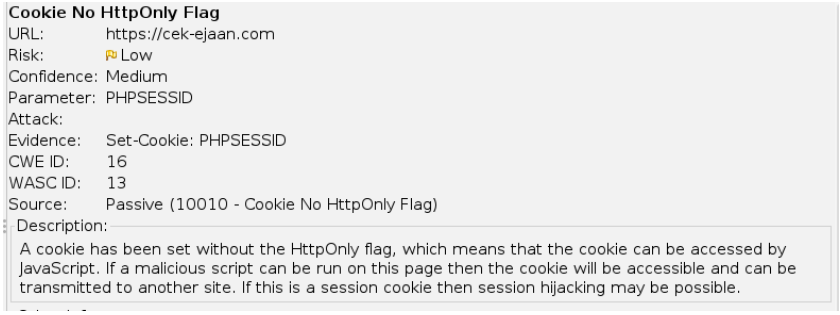
Berdasarkan Gambar 4.18 dapat dilihat bahwa *tools WebScarab* merespon *error* pada percobaan manipulasi hak isitimewa pengguna sehingga percobaan tidak dapat dilakukan dan dapat disimpulkan bahwa pada pengujian ini tidak terdapat kemungkinan celah keamanan. Hasil pengujian *authorization* kategori pengujian referensi objek langsung yang tidak aman *(WSTG-ATHZ-04)* dengan memodifikasi parameter *URL* yang terdapat pada halaman log aplikasi menggunakan *tools Mozilla Firefox* seperti Gambar 4.19.

Gambar 4.19 Hasil modofikasi parameter *URL* pada halaman log aplikasi

Berdasarkan Gambar 4.19 dapat dilihat bahwa hasil modifikasi parameter *URL* yang mengarah pada halaman *edit* atau *input* ejaan tidak dapat diakukan dan harus melalui mekanisme *login* sehingga pada pengujian ini tidak ditemukan celah keamanan.

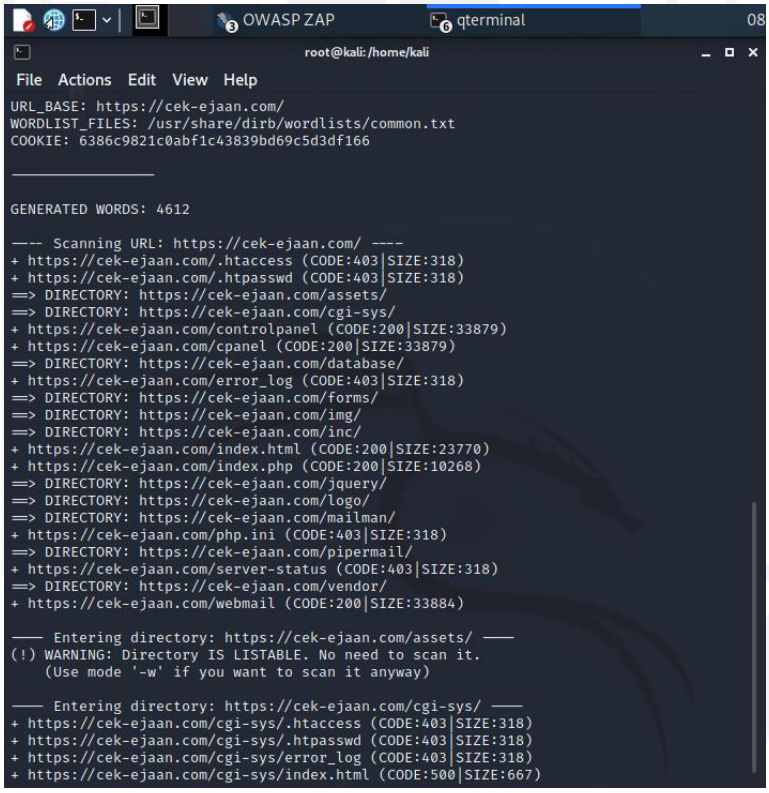
4.1.3 Session Management Testing

Hasil yang diperoleh pada pengujian *session management* kategori pengujian skema manajemen sesi *(WSTG-SESS-01)* dengan melihat status *cookie* untuk menanalisa kerentanan *website* terhadap serangan *hijacking* menggunakan tools *OWASP ZAP* tertuang pada Gambar 4.20.



Gambar 4.20 Hasil identifikasi status *cookie* menggunakan *tools OWASP ZAP*

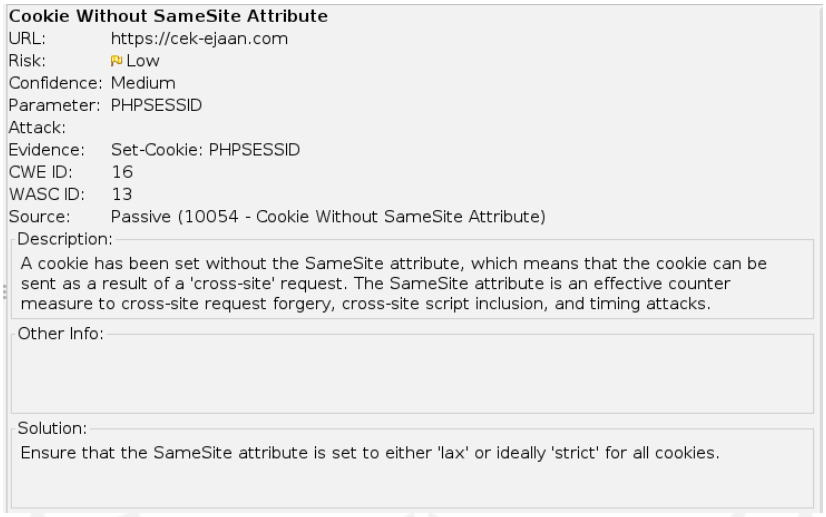
Berdasarkan Gambar 4.20 menunjukkan bahwa status *cookie no httponly flag* yang menyebabkan *cookie* dapat diakses melalui *java script.* Kemudian, hasil pengujian menggunakan *tools* dirb dengan mengakses *cookie* sesi seperti pada Gambar 4.21.



Gambar 4.21 Hasil akses halaman dengan *cookie* menggunakan *tools Dirb*

Gambar 4.21 menunjukkan hasil akeses halaman *website* menggunakan *cookie* dan ditemukan beberapa *directory* yang mengarah pada halaman *website* sehingga pada pengujian ini memiliki kemungkinan celah keamanan.

Kategori pengujian atribut *cookie* yang digunakan *(WSTG-SESS-02)* dengan memeriksa atribut *cookies* menggunakan *tools OWASP ZAP* diperoleh hasil seperti Gambar 4.22.

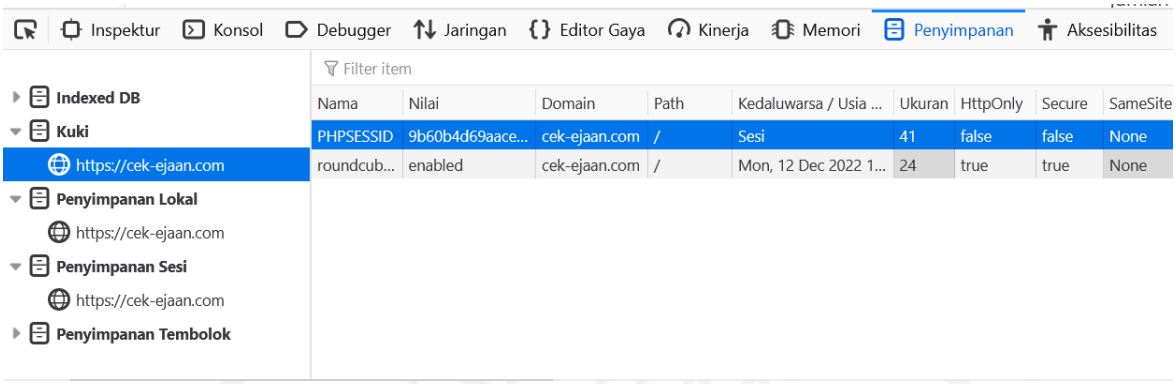


Gambar 4.22 Hasil identifikasi *samesite attribute cookie* menggunakan *tools OWASP ZAP*

Berdasarkan Gambar 4.22 dapat dilihat bahwa status *cookie without samesite attribute* dengan *level* risiko celah keamanan rendah yang berupa *cross-site request foregery*. Berikutnya hasil untuk *secure attribute cookie* menggunakan *tools OWASP ZAP* seperti pada Gambar 4.23.

Gambar 4.23 Hasil identifikasi *secure attribute cookie* menggunakan *tools OWASP ZAP*

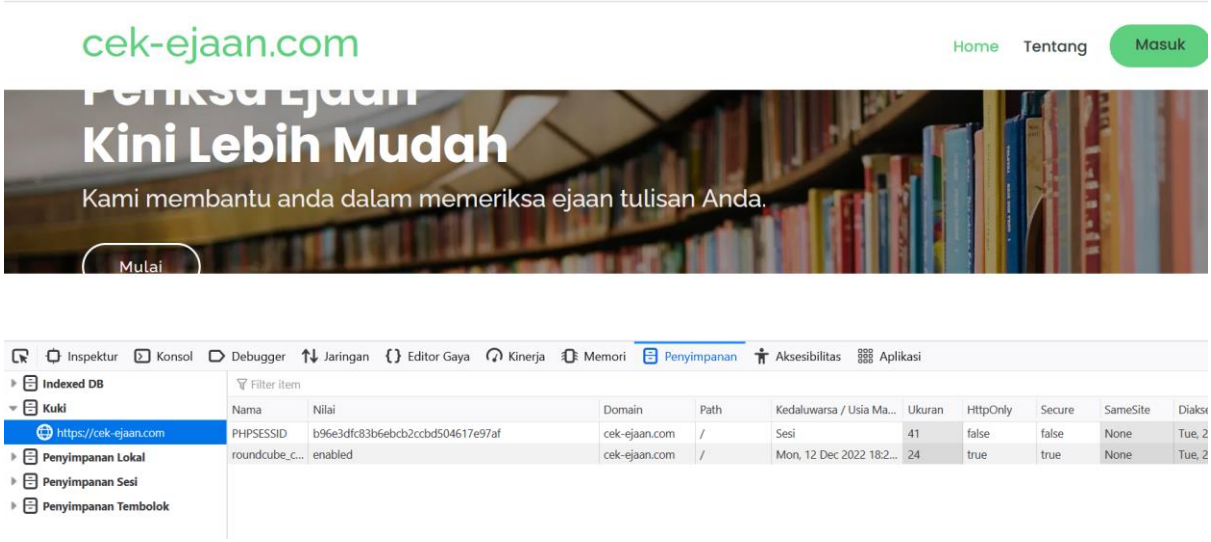
Gambar 4.23 menunjukkan hasil *cookie attribute secure* yang diperoleh berupa *without secure flag* dengan *level* risiko celah keamanan rendah yang berupa *cookie* dapat diakses menggunakan koneksi yang tidak terenkripsi atau melalui saluran yang tidak aman. Kemudian, hasil yang diperoleh dari pengujian menggunakan *tools Mozilla Firefox* seperti pada Gambar 4.24.



Gambar 4.24 Hasil identifikasi *attribute cookie* menggunakan *tools Mozilla Firefox*

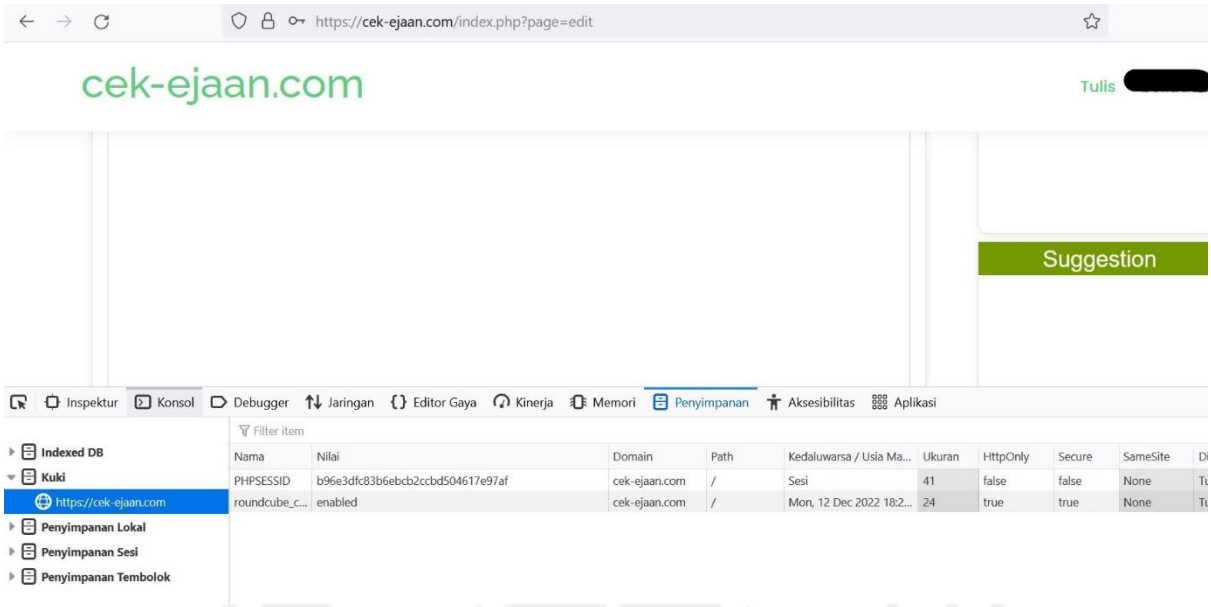
Berdasarkan Gambar 4.24 dapat dilihat bahwa atribut cookie yang digunakan berdomain cek-ejaan.com dengan *secure attribute "false"*, *httponly attribute "false"*, *path attribute "/"*, *expires attribute* "sesi", *samsite attribute "none"*, drai hasil yang diperoleh maka dapat disimpulkan bahwa *website* cek-ejaan.com memiliki risiko celah keamanan.

Hasil yang diperoleh pada kategori pengujian untuk fiksasi sesi yang digunakan *(WSTG-SESS-03)* dengan memeriksa *session ID* pada *cookie* sebelum melakukan autentikasi menggunakan *tools Mozilla Firefox* seperti yang tertuang pada Gambar 4.25.



Gambar 4.25 Hasil *cookie* sebelum login menggunakan *tools Mozilla Firefox*

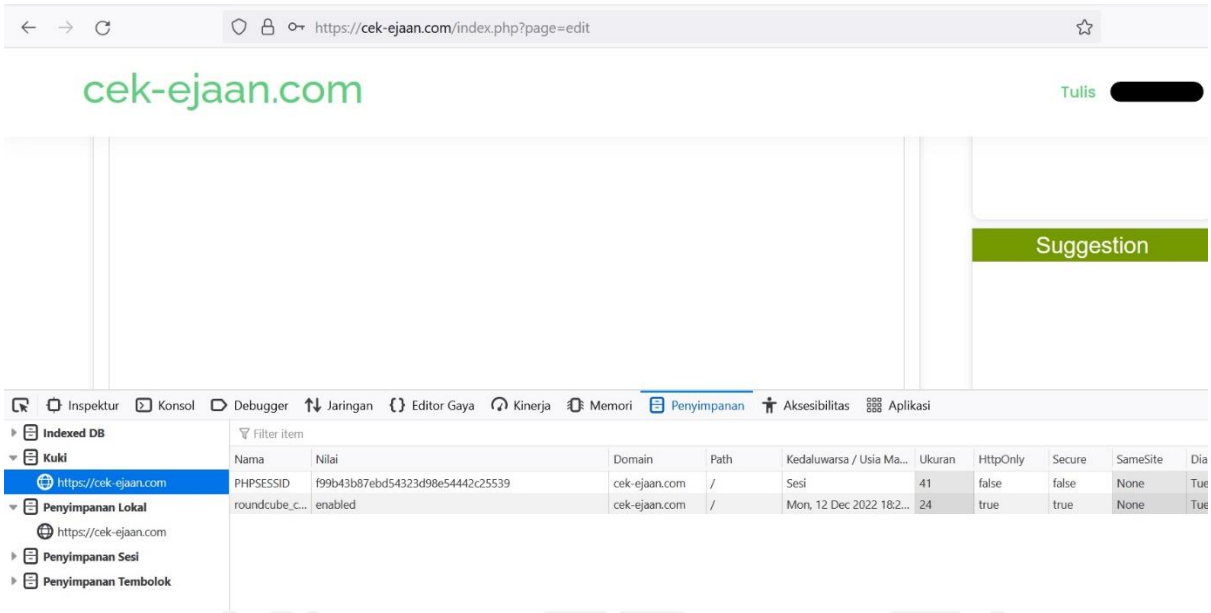
Berdasarkan Gambar 4.25 dapat dilihat nilai *hash* yang dihasilkan pada *cookie* setelah membuat permintaan *HTTP* sebelum melakukan proses autentikasi. Sedangkan setelah melakukan autentikasi tertera pada Gambar 4.26.



Gambar 4.26 Hasil *cookie* setelah *login* menggunakan *tools Mozilla Firefox*

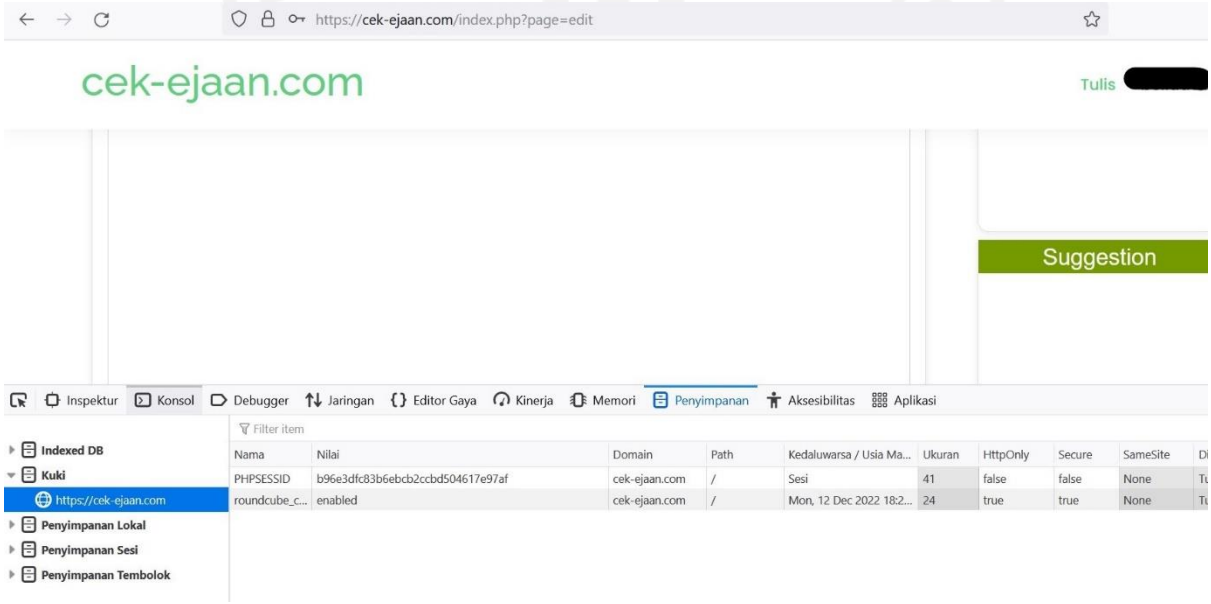
Gambar 4.26 menunjukkan bahwa tidak ada perubahan *Session ID* pada *cookie*, hal ini dapat dilihat dari nilai hash yang tidak mengalami perubahan baik sebelum maupun sesudah autentikasi sehingga pada pengujian ini terdapat risiko celah keamanan.

Kategori pengujian untuk variabel sesi *(WSTG-SESS-04)* dengan memeriksa token sesi *(Cookie, Session ID, Hidden Field)* yang dikirim melalui saluran terenkripsi apakah terdapat perbedaan setiap kali membuat permintaan *HTTP* menggunakan *tools Mozilla Firefox* diperoleh hasil yang tertuang pada Gambar 4.27 dan Gambar 4.28.



Gambar 4.27 Hasil *cookie* permintaan *HTTP user* x

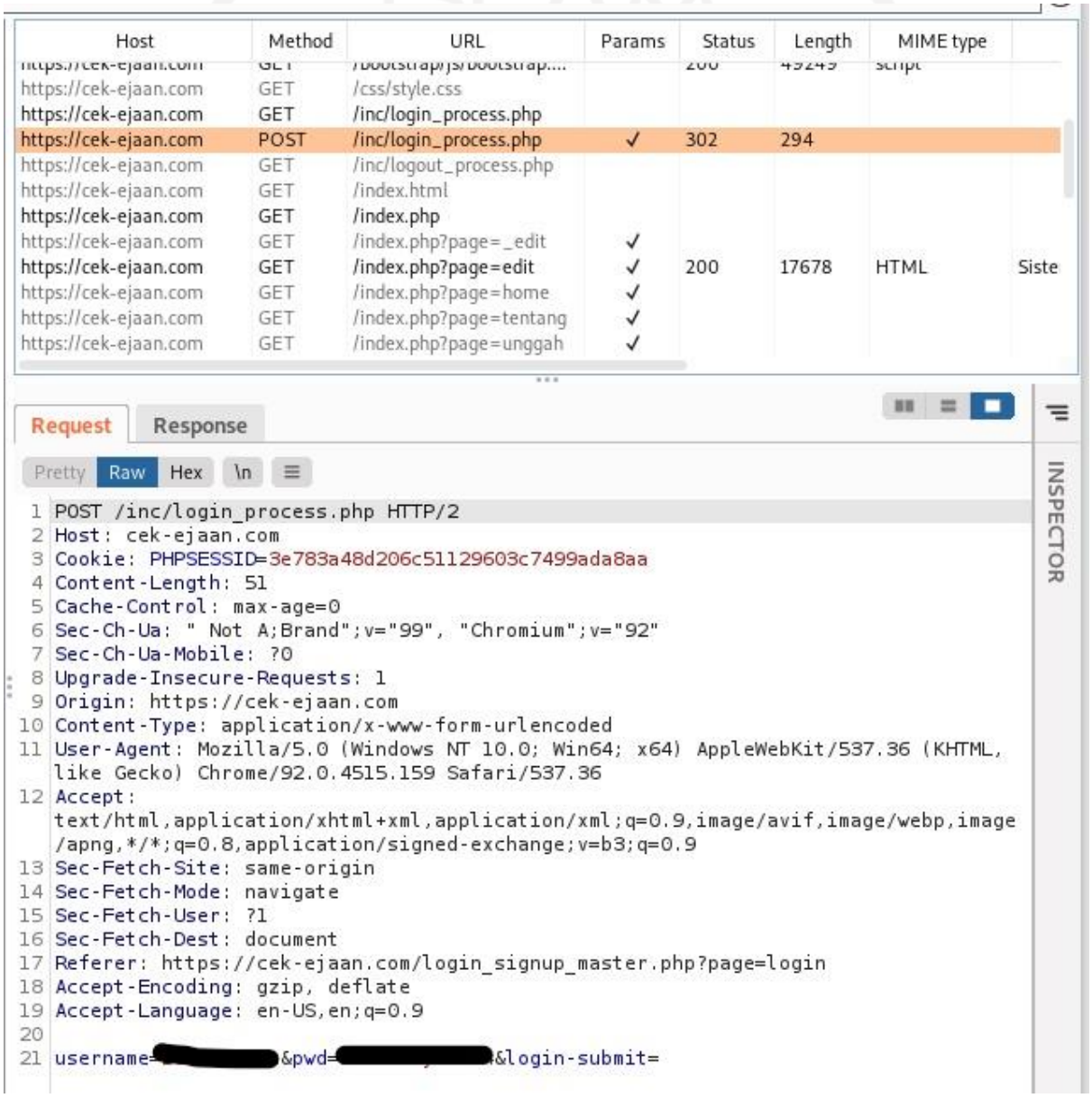
Berdasarkan Gambar 4.27 dapat dilihat bahwa dihasilkan *cookie* dengan nilai *hash* "f99b43b87ebd54323d98e54442c25539" pada permintaan *HTTP* pertama dengan pengguna x. Sedangkan pada Gambar 4.28 merupakan permintaan *HTTP* kedua dengan pengguna berbeda atau pengguna y.



Gambar 4.28 Hasil *cookie* permintaan *HTTP user* y

Berdasarkan Gambar 4.28 dapat dilihat bahwa *cookie* yang dihasilkan pada permintaan *HTTP* kedua memiliki perbedaan dengan *cookie* pada permintaan *HTTP* pertama yaitu dengan nilai *hash* "b96e3dfc83b6ebcb2ccbd504617e97af" sehingga pada pengujian ini tidak ditemukan risiko celah keamanan.

Pengujian kategori pemalsuan permintaan lintas situs *(WSTG-SESS-05)* dengan melakukan audit aplikasi untuk memastikan apakah terdapat kerentanan terhadap *Cross Site Request Forgery (CSRF)* menggunakan *tools Burpsuite* diperoleh hasil seperti Gambar 4.29.



Gambar 4.29 Hasil kerentanan *CSRF* menggunakan *tools Burpsuite*

Berdasarkan Gambar 4.29 dapat dilihat bahwa hasil menunjukkan URL aplikasi, parameter, dan nilai yang sah dari *website* cek-ejaan.com dapat diidentifikasi sehingga pada pengujian ini ditemukan celah keamanan berupa kerentanan terhadap serangan *CSRF*.

Pengujian kategori fungsionalitas *logout (WSTG-SESS-06)* dengan melakukan analisis terhadap batas waktu sesi apakah sesi dihentikan setelah *logout* dan apakah terdapat *locked invalid login* menggunakan *tools Mozilla Firefox* diperoleh hasil sesi dihentikan setelah *logout*. Namun, tidak ada mekanisme penguncian saat *invalid login* sehingga pada pengujian ini ditemukan kemungkinan celah keamanan.

Hasil yang diperoleh pada pengujian batas waktu *(WSTG-SESS-07)* dengan memeriksa batas waktu pada aplikasi web saat tidak digunakan pada durasi tertentu, serta memeriksa tombol back pada sesi yang sama menggunakan *tools Mozilla Firefox* yaitu tidak ada mekanisme logout otomatis pada saat *session time out* sehingga dalam pengujian dini ditemukan risiko celah keamanan pada *website* cek-ejaan.com.

Hasil yang diperoleh pada pengujian untuk sesi yang membingungkan *(WSTG-SESS-08)* dengan mengidentifikasi dan menghitung variabel valid setiap sesi yang digunakan menggunakan *tools Mozilla Firefox* yaitu tidak adanya perubahan pada ukuran atau jumlah variabel valid pada setiap sesi yang dihasilkan sehingga pada pengujian ini ditemukan risiko celah keamanan pada *website* cek-ejaan.com.

Hasil yang diperoleh pada pengujian pembajakan sesi *(WSTG-SESS-09)* dengan melihat status *cookie* untuk mengidentifikasi kerentanan *cookie* menggunakan *tools OWASP ZAP* yaitu status *cookie* tidak *secure* sehingga dalam pengujian ini ditemukan risiko celah keamanan.

4.2 Pembahasan

4.2.1 Metode OWASP Versi 4

Tahap ini dilakukan pemetaan hasil yang telah diperoleh dalam bentuk tabel seperti yang tertera pada Tabel 4.1, kemudian dilakukan pembahasan sesuai dengan standar keamanan OWASP dan SANS.

Tabel 4.1 Hasil Pengujian menggunakan metode *OWASP* versi 4.2

Kategori Pengujian	Tahapan Pengujian	Aktivitas	*Tools*	Status	Hasil
Authentication Testing	*Testing for Credentials*	Memastikan data kredensial	• *Google Chrome*	Tidak ditemukan	Menerapkan *HTTPS*

Kategori Pengujian	Tahapan Pengujian	Aktivitas	*Tools*	Status	Hasil
	Transported over an Encrypted Channel (WSTG-ATHN-01)	terenkripsi yang ditransfer dari klien ke server aplikasi web saat login dan menggunakan aplikasi diakses melalui HTTPS.			
	Testing for default credentials (WSTG-ATHN-02)	Memprediksi kredensial default dan validasi dari halaman login	• *THC-Hydra*	Tidak ditemukan	Tidak ditemukan username default
	Testing for Weak lock out mechanism (WSTG-ATHN-03)	Melakukan beberapa kali login dengan *password* yang salah untuk menguji kemampuan mekanisme penguncian akun.	• *Google Chrome*	Ditemukan	Tidak ada mekanisme penguncian pada *users invalid login*
	Testing for bypassing authenticationschema (WSTG-ATHN-04)	Melakukan pengujian skema otentikasi dengan metode permintaan halaman secara langsung atau penjelajahan paksa dan modifikasi parameter URL.	• *SQL Map* • *Mozilla Firefox*	Tidak ditemukan	*URL* tidak dapat dimodifikasi dan tidak dapat mengakses halaman lain secara langsung melalui penjelajahan paksa.
	Test remember passwo	Memastikan keamanan	• *Google Chrome*	Ditemukan	*Autocomplete* aktif

Kategori Pengujian	Tahapan Pengujian	Aktivitas	*Tools*	Status	Hasil
	rd functionality (WSTG-ATHN-05)	kredensial pengguna pada sesi yang dihasilkan dan dikelola dengan melihat *log password* yang disimpan serta pengaktifan *autocomplete*.	• *OWASP ZAP*		
	Testing for Browser cache weakness (WSTG-ATHN-06)	Memeriksa *cache browser* dari tombol "Kembali" untuk melihat sumber daya yang ditampilkan pada halaman sebelumnya apakah informasi sensitif dapat diakses saat tidak diautentikasi.	• *Google Chrome* • *OWASP ZAP*	Tidak ditemukan	*Cache browser* tidak ditemukan
	Testing for Weak password policy (WSTG-ATHN-07)	Memeriksa ketahanan aplikasi terhadap tebakan kata sandi dengan melakukan *brute force* menggunakan kamus kata sandi.	• *THC-Hydra*	Tidak ditemukan	Tidak ada kata sandi yang berhasil ditebak.
	Testing for Weak security question/ answer	Memeriksa skema pertanyaan keamanan dengan	-	Tidak ditemukan	Tidak terdapat skema pertanyaan keamanan

Kategori Pengujian	Tahapan Pengujian	Aktivitas	*Tools*	Status	Hasil
	(WSTG-ATHN-08)	mengumpulkan kemungkinan jawaban dari serangkaian pertanyaan keamanan.			
	Testing for weak password change or reset functionalities(WSTG-ATHN-09)	Memeriksa mekanisme perubahan atau pengaturan ulang kata sandi	• *Google Chrome*	Ditemukan	Fungsi perubahan kata sandi lemah
	Testing for Weaker authentication in alternativechannel (WSTG-ATHN-10)	Mengidentifikasi saluran autentikasi lain (alternatif)	-	Tidak ditemukan	Tidak terdapat saluran alternatif
Authorization Testing	*Testing directory traversal file include (WSTG-ATHZ-01)*	Mengidentifikasi lokasi file *root directory* atau *root dokumen web*.	• *Dirb* • *Mozilla Firefox*	Ditemukan	Terdapat *URL* yang mengarah pada *root directory* web atau *directory traversal*
	Testing for bypassing authorizationschema (WSTG-ATHZ-02)	Melakukan percobaan untuk mengakses dan mengoperasikan fungsi pada sumber daya khusus tanpa proses autentikasi.	• *Dirb* • *Mozilla Firefox*	Ditemukan	Berhasil melewati proses autentikasi dan mengakses sumber daya khusus

Kategori Pengujian	Tahapan Pengujian	Aktivitas	*Tools*	Status	Hasil
	Testing for Privilege Escalation (WSTG-ATHZ-03)	Mengidentifikasi *hidden field HTML* yang terkait dengan manipulasi hak istimewa	• *WebScarab*	Tidak ditemukan	Hak istimewa tidak dapat dimanipulasi
	Testing for Insecure Direct ObjecReferences (WSTG-ATHZ-04)	Memodifikasi parameter *URL* yang terdapat pada halaman log aplikasi untuk menguji skema otentikasi	• *Mozilla Firefox*	Tidak ditemukan	Parameter *URL* pada halaman log tidak dapat dimodifikasi
Session Management Testing	*Testing for Bypassing Session Management Schema (WSTG-SESS-01)*	Menganalisa kerentanan terhadap serangan hijacking dengan melihat status *cookie* dan mencoba mengakses halaman menggunakan *cookie*.	• *OWASP ZAP (Zed Attack Proxy)* • *Dirb*	Ditemukan	Status *cookie No HttpOnly Flag* sehingga dapat diakses melalui *JavaScript* dan rentan terhadap serangan *hijacking*
	Testing for Cookies attributes (WSTG-SESS-02)	Memeriksa atribut *cookies* yang digunakan, seperti *secure attribute, httponly attribute, domain attribute, path attribute, expires attribute*	• *OWASP ZAP (Zed Attack Proxy)*	Ditemukan	*No secure Attribute, No HttpOnly Attribute*
	Testing for Session	Memeriksa *session ID* atau	• *Mozilla Firefox*	Ditemukan	*Session ID* atau *cookie*

Kategori Pengujian	Tahapan Pengujian	Aktivitas	*Tools*	Status	Hasil
	Fixation (WSTG-SESS-03)	*cookie* setelah berhasil melakukan autentikasi apakah terjadi pembaharuan.			tidak berubah setelah *login*
	Testing for Exposed Session Variables (WSTG-SESS-04)	Memeriksa token sesi *(Cookie, Session ID, Hidden Field)* yang dikirim melalui saluran terenkripsi apakah terdapat perbedaan setiap kali membuat permintaan *HTTP*	• *Mozilla firefox*	Tidak ditemukan	Token sesi berubah setiap membuat permintaan *HTTP*
	Testing for Cross Site Request Forgery (WSTG-SESS-05)	Memastikan apakah terdapat kerentanan terhadap *Cross Site Request Forgery (CSRF)*	• *Burpsuite*	Ditemukan	Dapat mengidentifikasi *URL* aplikasi, parameter, dan nilai yang sah
	Testing for logout functionality (WSTG-SESS-06)	Melakukan analisa terhadap batas waktu sesi apakah sesi dihentikan setelah *logout* dan apakah terdapat *locked invalid login*	• *Mozilla Firefox*	Ditemukan	Sesi dihentikan setelah *logout*. Namun, tidak ada mekanisme penguncian saat *invalid login*
	Test Session Timeout	Memeriksa batas waktu	• *Mozilla Firefox*	Ditemukan	Tidak ada logout

Kategori Pengujian	Tahapan Pengujian	Aktivitas	*Tools*	Status	Hasil
	(WSTG-SESS-07)	pada aplikasi web saat tidak digunakan pada durasi tertentu, apakah akan *logout* secara otomatis, serta memeriksa tombol back pada sesi yang sama			otomatis atau *session time out* pada saat aplikasi tidak digunakan pada durasi tertentu
	Testing for Session puzzling (WSTG-SESS-08)	Mengidentifikasi dan menghitung variabel valid setiap sesi yang digunakan	• *Mozilla Firefox*	Ditemukan	Ukuran variabel valid sama pada setiap sesi
	Testing for Session Hijacking (WSTG-SESS-09)	Mengidentifikasi kerentanan *cookies session* dengan melihat status *cookies*	• *OWASP ZAP (Zed Attack Proxy)*	Ditemukan	Status cookie tidak *secure*

Berdasarkan Tabel 4.1 Hasil Pengujian menggunakan metode *OWASP* versi 4.2 dapat dilihat bahwa terdapat tiga kategori pengujian yang terbagi menjadi tiga kategori *authentication*, *authorization*, dan *session management* dengan total 23 sub kategori yang kemudian di uji untuk memperoleh hasil dan ada atau tidaknya risiko celah keamanan pada *website* cek-ejaan.com. Hasil dari pengujian yang dilakukan dianalisis dan dipetakan sesuai dengan daftar kerentanan yang di rilis *OWASP* dan *SANS* seperti yang tertera pada Tabel 4.2.

Tabel 4.2 Pemetaan Hasil Pengujian OWASP Versi 4 berdasarkan daftar OWASP-SANS

Peringkat OWASP	Kerentanan OWASP	ID CWE/SANS	Kode Pengujian OWASP Versi 4.2	Hasil OWASP Versi 4.2
A01:2021	*Broken Access Control*	*CWE-200: Exposure of Sensitive Information to an Unauthorized Actor* *CWE-201: Exposure of Sensitive Information Through Sent Data* *CWE-352: Cross-Site Request Forgery*	• *WSTG-ATHZ-01* • *WSTG-ATHZ-02* • *WSTG-SESS-01* • *WSTG-SESS-05*	Ditemukan
A02:2021	*Cryptographic Failures*	*CWE-259: Use of Hard coded Password* *CWE-327: Broken or Risky Crypto Algorithm* *CWE-331: Insufficient Entropy*	• *WSTG-SESS-02*	Ditemukan
A03:2021	*Injection*	*CWE-79: Cross-Site Scripting* *CWE-89: SQL Injection*	• *WSTG-ATHN-04*	Tidak ditemukan

Peringkat OWASP	Kerentanan OWASP	ID CWE/SANS	Kode Pengujian OWASP Versi 4.2	Hasil OWASP Versi 4.2
		CWE-73: External Control of File Name or Path		
A04:2021	*Insecure Design*	*CWE-209: Generation of Error Message Containing Sensitive Information*	• *WSTG-ATHN-03* • *WSTG-ATHN-05* • *WSTG-SESS-07*	Ditemukan
		CWE-256: Unprotected Storage of Credentials		
		CWE-501: Trust Boundary Violation		
		CWE-522: Insufficiently Protected Credentials		
A05:2021	*Security Misconfiguration*	*CWE-16: Configuration*	• *WSTG-ATHN-02* • *WSTG-ATHN-07*	Tidak ditemukan
		CWE-611: Improper Restriction of XML External Entity Reference.		

Peringkat OWASP	Kerentanan OWASP	ID CWE/SANS	Kode Pengujian OWASP Versi 4.2	Hasil OWASP Versi 4.2
A06:2021	*Vulnerable and Outdated Components*	*CWE-1104: Use of Unmaintained Third-Party Components* and the two CWEs from Top 10 2013 and 2017	-	-
A07:2021	*Identification and Authentication Failures*	*CWE-297: Improper Validation of Certificate with Host Mismatch* *CWE-287: Improper Authentication* *CWE-384: Session Fixation*	• *WSTG-ATHN-03* • *WSTG-ATHN-05* • *WSTG-ATHN-09* • *WSTG-SESS-03* • *WSTG-SESS-06* • *WSTG-SESS-07*	Ditemukan
A08:2021	*Software and Data Integrity Failures*	*CWE-829: Inclusion of Functionalityfrom Untrusted Control Sphere* *CWE-494: Download of Code Without Integrity Check* *CWE-502: Deserialization of Untrusted Data*	-	-

Peringkat OWASP	Kerentanan OWASP	ID CWE/SANS	Kode Pengujian OWASP Versi 4.2	Hasil OWASP Versi 4.2
A09:2021	*Security Logging and Monitoring Failures*	*CWE-778: Insufficient Logging* to include	-	-
		CWE-117: Improper Output Neutralization for Logs		
		CWE-223: Omission of Security-relevant Information		
		CWE-532: Insertion of Sensitive Information into Log File		
A10:2021	*Server-Side Request Forgery (SSRF)*	-	-	-

Berdasarkan Tabel 4.2 dapat dilihat hasil pemetaan dari pengujian penetrasi yang telah dilakukan dari sepuluh kerentanan yang dirilis *OWASP* dan *SANS* terdapat enam risiko celah keamanan yang termasuk dalam pengujian, empat risiko celah keamanan yang lainnya tidak dilakukan pengujian. Dari keenam risiko celah keamanan tersebut ditemukan empat kerentanan dalam *website* cek-ejaan.com, kerentanan tersebut meliputi: *broken access control, cryptographic failures, insecure design, identification and authentication failures.*

4.3 **Analisis**

Berdasarkan pengujian yang telah dilakukan terhadap *website* cek-ejaan.com terdapat beberapa kerentanan yang berbahaya sehingga perlu dilakukan perbaikan sebagai bentuk penanggulangan ancaman keamanan di masa mendatang. Menurut pemetaan yang telah dilakukan berdasarkan daftar celah keamanan yang dirilis oleh *OWASP* dan *SANS/CWE*

website cek-ejaan.com memiliki kerentanan *broken access control, cryptographic failures, insecure design, identification and authentication failures.*

Jenis kerentanan *broken access control* terdeteksi dari pengujian *WSTG-ATHZ-01, WSTG-ATHZ-02, WSTG-SESS-01,* dan *WSTG-SESS-05*. Dalam pengujian *WSTG-ATHZ-01* ditemukan *URL* yang mengarah pada direktori *root website* cek-ejaan.com. Sedangkan dalam pengujian *WSTG-ATHZ-02* berhasil melewati proses autentikasi dan mengakses sumber daya khusus yang hanya dapat diakses oleh *users*. Pengujian *WSTG-SESS-01* mendeteksi status atribut *cookies no HTTP only flag* sehingga memungkinkan untuk diakses melalui *java script* dan rentan terhadap *hijacking*. Pengujian *WSTG-SESS-05* dapat mengidentifikasi *URL* aplikasi, parameter, dan nilai yang sah.

Celah keamanan *Cryptographic Failures* terdeteksi pada *WSTG-SESS-02* ditemukan atribut *cookie* yang digunakan tidak aman, serta menggunakan pengujian dasar berupa protokol kriptografi yang digunakan lemah pada *cookie* seperti fungsi *hash* masih menggunakan *MD5*.

Celah keamanan *Injection* pada *scanning* otomatisasi menggunakan *tools OWASP ZAP* ditemukan temuan yang berupa ancaman dari serangan *CSS (Cross-Site Scripting)*. Namun pada saat dilakukan *pentest* berupa *SQL injection (WSTG-ATHN-04)* terdeteksi oleh *WAF (Web Application Firewall)* sehingga dinyatakan tidak lolos.

Jenis kerentanan *insecure design* terdeteksi pada pengujian *WSTG-ATHN-03, WSTG-ATHN-05, WSTG-SESS-07* yang berhubungan dengan konsumsi kontrol sumber daya *website*. Pada pengujian *WSTG-ATHN-03* ditemukan tidak adanya pembatasan pada upaya *login* yang gagal. Pengujian WSTG-ATHN-05 ditemukan status *autocomplete on.* Selain itu, pada pengujian *WSTG-SESS-07* tidak ada mekanisme *logout* otomatis atau *session time out* pada batas waktu tertentu saat aplikasi tidak digunakan.

Jenis kerentanan *security misconfiguration* terdeteksi saat scanning *port* menggunakan *tools Nmap* karena semua *port* pada *website* terbuka sehingga terdapat kemungkinan kerentanan. Namun, pada saat dilakukan pengujian pada setiap *port* gagal dan merespon dengan 404 *(Error not found),* selain itu *WSTG-ATHN-02* dan *WSTG-ATHN-07* terkait dengan penggunaan akun *default* tidak ditemukan sehingga dapat dikatakan bahwa pada pengujian yang dilakukan tidak memiliki kemungkinan celah *security misconfiguration*.

Celah keamanan *Identification and Authentication Failures* terdeteksi melalui skema pengujian autentikasi yang telah dilakukan yaitu terdapat beberapa celah yang ditemukan yaitu pada pengujian *WSTG-ATHN-03, WSTG-ATHN-05*, dan *WSTG-ATHN-09.* Selain itu, pada

pengujian *WSTG-SESS-03, WSTG-SESS-06,* dan *WSTG-SESS-07* terkait dengan *session ID* yang digunakan.

Beberapa rekomendasi penulis terhadap pengujian yang telah dilakukan sebelumnya antara lain:

a. Jenis kerentanan kontrol akses yang rusak dapat ditanggulangi dengan meningkatkan kontrol akses efektif pada sisi server yang dapat dipercaya atau meningkatkan batas API dan pengontrol akses untuk meminimalkan bahaya dari *tools* otomatis yang digunakan untuk menyerang. Selain itu, kerentanan ini dapat dicegah dengan cara memastikan file cadangan dan metadata tidak dimasukkan kedalam direktori *root website* dan menonaktifkan daftar direktori server web serta *directory browing* melalui *CPanel* atau dapat dilakukan pemblokiran dengan file *.htacess*.
b. Jenis kerentanan *cryptographic failures* dapat ditanggulangi dengan menghindari fungsi kriptografi dan skema *padding* yang tidak digunakan lagi, seperti *MD5* dan *SHA1*. Selain itu, jenis kerentanan ini dapat dicegah dengan menonaktifkan caching respon yang mengandung data sensitif.
c. Kerentanan *insecure design* dapat ditanggulangi dengan membatasi konsumsi sumber daya layanan maupun pengguna, seperti membatasi upaya *login* yang gagal. Selain itu, perlunya menerapkan autentikasi multi-faktor untuk mencegah pengisian kredensial otomatis.
d. Mencegah kegagalan dalam identifikasi dan otentikasi dapat dilakukan dengan memeriksa mekanisme pengubahan kata sandi yang lemah, perlunya dilakukan perubahan *username* dan kata sandi *default*, terapkan kompleksitas sandi sesuai dengan pedoman *NIST (National Institute of Standards and Technology)*, membatasi upaya *login* yang gagal, tidak mencantumkan *ID* sesi pada *URL* dan pastikan bahwa *ID* sesi tidak valid setelah *logout* atau pada periode waktu tertentu, jika memungkinkan gunakan autentikasi multi faktor.

BAB V
KESIMPULAN

4.1 Kesimpulan

Analisis kepatuhan keamanan *website* menggunakan standar *OWASP* dan *SANS* dengan melakukan uji penetration testing menggunakan *OWASP* Versi 4.2 arau *WSTG (Web Security Testing Guide)* versi 4.2 bertujuan untuk mengidentifikasi celah keamanan pada *website* cek-ejaan.com. Berdasarkan seluruh kegiatan penelitian yang telah dilakukan dapat diperoleh kesimpulan, sebagai berikut:

a. Menurut prinsip keamanan informasi *CIA (Confidentiality, Integrity, and Availability) website* cek-ejaan.com kurang menerapkannya. Hal ini terlihat dari ditemukannya beberapa celah keamanan pada pengujian penetrasi.
b. *Website* cek-ejaan.com tergolog cukup aman, karena informasi yang tersimpan dan jika dilihat dari activity yang dilakukan tidak mengandung infromasi sensitif.
c. *Website* cek-ejaan.com memiliki *firewall* yang cukup tangguh dalam memblokir serangan.
d. Metode *OWASP Top 10* cocok digunakan sebagai acuan dalam melakukan pengujian celah keamanan pada *website* cek-ejaan.com. Karena daftar celah keamanan tersebut terus diperbaharui sesuai dengan perubahan teknologi, selain itu jenis kerentanan yang terdapat pada *website* cek-ejaan.com sesuai dengan daftar celah keamanan yang ada pada *OWASP Top 10*.
e. Metode *SANS/CWE Top 25* digunakan sebagai pelengkap masih sangat baik digunakan karena jenis kerentanan yang disebutkan lebih spesifik dan terdapat hubungan dengan metode *OWASP Top 10*.
f. *Tools OWASP ZAP* sebagai alat otomatisasi sangat baik menempilkan secara rinci jenis kerentanan, kategori level, yang dilengkapi deskripsi dan solusi.

4.2 Saran

Berdasarkan penelitian yang telah terlaksana diperoleh beberapa saran yang dapat digunakan sebagai acuan untuk pengembang website cek-ejaan.com sebagai studi kasus dalam penelitian ini dan dapat diterapkan dalam pengembangan penelitian terkait, sebagai berikut:

a. Perlunya dilakukan pengukuran berdasarkan kategori level pada hasil pengujian yang telah ditemukan berdasarkan panduan *WSTG (Web Security Testing Guide)* versi 4.2.

b. Perlu dilakukan pengujian lebih lanjut terhadap daftar celah keamanan yang belum dilakukan.
c. Perlunya dilakukan penelitian lebih lanjut menggunakan metode yang berbeda.
d. Perlu dilakukan pengecekan terhadap *source code* aplikasi atau pengujian dari sisi web server.

DAFTAR PUSTAKA

Alwi, E. I., Herdianti, H., & Umar, F. (2020). Analisis Keamanan Website Menggunakan Teknik Footprinting dan Vulnerability Scanning. *INFORMAL: Informatics Journal*, *5*(2), 43. https://doi.org/10.19184/isj.v5i2.18941

Dewanto, A. P. (2018). *Penetration Testing padaDomain uii.ac.id Menggunakan OWASP 10*.

Dirgahayu, T., Prayudi, Y., & Fajaryanto, A. (2015). Penerapan Metode ISSAF dan OWASP versi 4 Untuk Uji Kerentanan Web Server. *Jurnal Ilmiah NERO*, *1*(3), 190–197. http://nero.trunojoyo.ac.id/index.php/nero/article/download/29/27

Elanda, A., & Tjahjadi, D. (2018). Analisis Manajemen Resiko Sistem Keamanan Ids (Intrusion Detection System) Dengan Framework Nist (National Institute of Standards and Technology) Sp 800-30 (Studi Kasus Disinfolahtaau Mabes Tni Au). *Infoman's*, *12*(1), 1–13. https://doi.org/10.33481/infomans.v12i1.45

Guntoro, G., Costaner, L., & Musfawati, M. (2020). Analisis Keamanan Web Server Open Journal System (Ojs) Menggunakan Metode Issaf Dan Owasp (Studi Kasus Ojs Universitas Lancang Kuning). *JIPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika)*, *5*(1), 45. https://doi.org/10.29100/jipi.v5i1.1565

Ichsan, R., Falach, A., Abdurrahman, L., Santoso, I., & Si, S. (2021). *Octave Allegro Risk Analysis and Information Security Control Design in Hospital Management Information System Billing Module Using Octave Allegro*. *8*(2), 2709–2722.

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

Nazwita, S. R. (2017). Analisis Sistem Keamanan Web Server dan Database Server Menggunakan Suricata. *Seminar Nasional Teknologi Informasi Komunikasi Dan Industri*, *0*(0), 2579–5406. http://ejournal.uin-suska.ac.id/index.php/SNTIKI/article/view/3368

OWASP. (2014). 4.0 Testing Guide. *OWASP Foundation*, *Cc*, 224. https://www.owasp.org/images/1/19/OTGv4.pdf

Putri, S. (2020). Pemanfaatan Internet untuk Meningkatkan Minat Baca Mahasiswa PLS IKIP Siliwangi. *Comm-Edu (Community Education Journal)*, *3*(2), 91. https://doi.org/10.22460/comm-edu.v3i2.3700

Saputra, A., Nelmiawati, N., & Sitorus, M. A. R. (2017). Penilaian Ancaman pada Website Transkrip Aktifitas Mahasiswa Politeknik Negeri Batam Menggunakan Metode DREAD. *Jurnal Integrasi*, *9*(1), 53. https://doi.org/10.30871/ji.v9i1.281

Tania, A. M., Setiyadi, D., & Khasanah, F. N. (2018). Keamanan Website Menggunakan Vulnerability Assessment. *Keamanan Website Menggunakan Vulnerability Assessment*,

2(2), 171–180.

Utoro, S., Nugroho, B. A., Meinawati, M., & Widianto, S. R. (2020). Analisis Keamanan Website E-Learning SMKN 1 Cibatu Menggunakan Metode Penetration Testing Execution Standard. *Multinetics*, *6*(2), 169–178. https://doi.org/10.32722/multinetics.v6i2.3432

Victor Tobing. (2020). Rekapitulasi Insiden Web Defacement. *Badan Siber Dan Sandi Negara*, *Juni 2020*(Maret), 1–27. https://bssn.go.id/rekap-serangan-siber-januari-april-2020

Yunus, M. (2019). Analisis Kerentanan Aplikasi Berbasis Web Menggunakan Kombinasi Security Tools Project Berdasarkan Framework Owasp Versi 4. *Jurnal Ilmiah Informatika Komputer*, *24*(1), 37–48. https://doi.org/10.35760/ik.2019.v24i1.1988

LAMPIRAN

A. *Report*

B. Hasil *scan* otomatis *OWASP ZAP*

REPORT PENGUJIAN CELAH KEAMANAN PADA WEBSITE CEK-EJAAN.COM

Disusun Oleh:

N a m a : Bella Tasya Kumala Dewi

NIM : 18523305

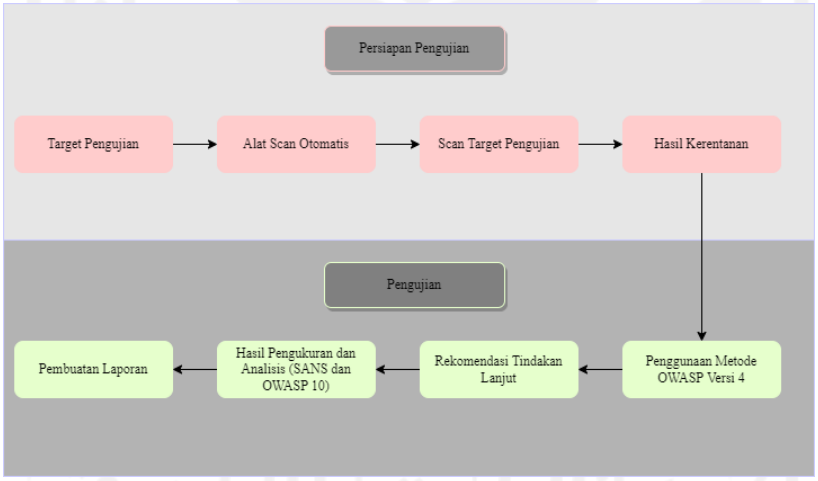
Yogyakarta, 5 Januari 2022

Target:

Website cek ejaan yang berdomain cek-ejaan.com dikelola oleh Jurusan Informatika pada Fakultas Teknologi Industri, Universitas Islam Indonesia. Universitas Islam Indonesia merupakan perguruan tinggi swasta yang terletak di Yogyakarta. Proses pengujian celah keamanan *website* cek-ejaan.com bertujuan untuk:

e. Menganalisis *website* cek-ejaan.com terkait penerapan aturan keamanan apakah sudah sesuai dengan standar yang ada yaitu menerapkan tiga aspek Confidentiality, Integrity, dan Availability.
f. Mengetahui risiko celah keamanan pada *website* cek-ejaan.com sehingga dapat segera dilakukan perbaikan untuk menghindari risiko yang timbul di masa mendatang.

Metode pengujian yang digunakan dalam *penetration testing* ini yaitu *WSTG (Web Security Testing Guide)* versi 4.2 yang dirilis oleh *OWASP* dengan fokus pengujian autentikasi, torisasi, dan manajemen sesi. Penguji mensimulasikan sebagai orang luar *(attacker)* yang tidak mengetahui informasi terkait *website* cek-ejaan.com pada proses penetration testing dengan alur yang digunakan seperti yang tertuang pada Gambar 1.



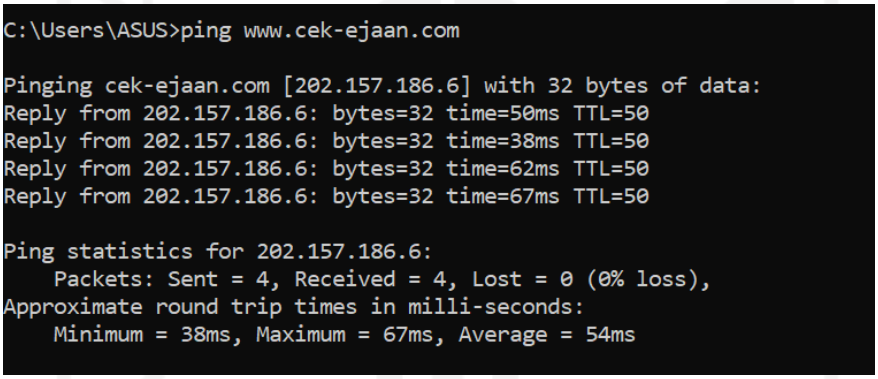
Gambar 1 Proses pengujian penetrasi

Berdasarkan Gambar 1 dapat dilihat bahwa terdapat dua tahap dalam pengujian penetrasi yaitu tahap persiapan dan tahap pengujian. Beberapa hal yang diperlukan kemudian dipersiapkan pada tahap pesiapan dalam empat langkah, diantaranya: 1) mencari informasi terkait target pengujian *(footprinting)*, 2) mempersiapkan alat scan otomatis untuk identifikasi celah, 3) proses scan target menggunakan alat scan otomatis, 4) analisis hasil kerentanan. Tahap pengujian terdiri dari empat langkah, diantaranya: 1) proses pengujian menggunakan

metode *OWASP* versi 4.2, 2) memberikan rekomendasi tindakan lanjut terkait ada atau tidaknya risiko celah pada setiap pengujian, 3) mednapatkan hasil pengukuran risiko (*low*, *medium*, atau *high*) berdasarkan analisis menggunakan daftar kerentanan yang dirilis oleh *OWASP* dan *SANS*, 4) pembuatan laporan hasil pengujian.

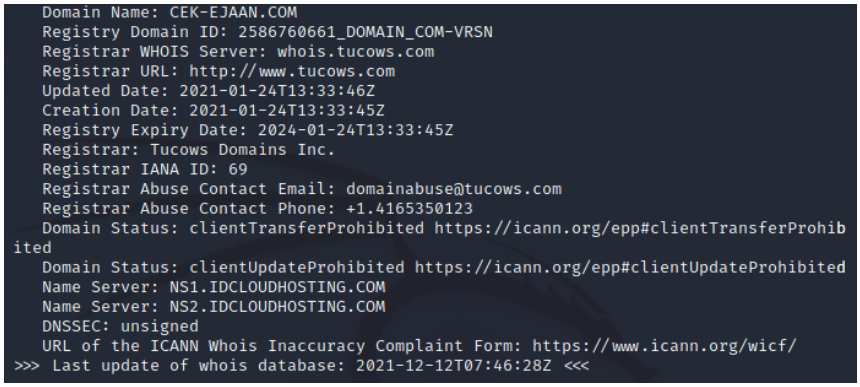
Pengumpulan Data dan Identifikasi Sistem (Target Pengujian)

Tahap ini dilakukan pengumpulan data menggunakan beberapa *tools* untuk mendapatkan informasi mengenai *website* cek-ejaan.com. Pada menggunakan *tools command prompt* didapatkan *IP adress website* cek-ejaan.com menggunakan perintah *ping*. *Command prompt* merupakan suatu aplikasi *CLI (Command Line Interpreter)* yang terdapat pada *OS Windows*.



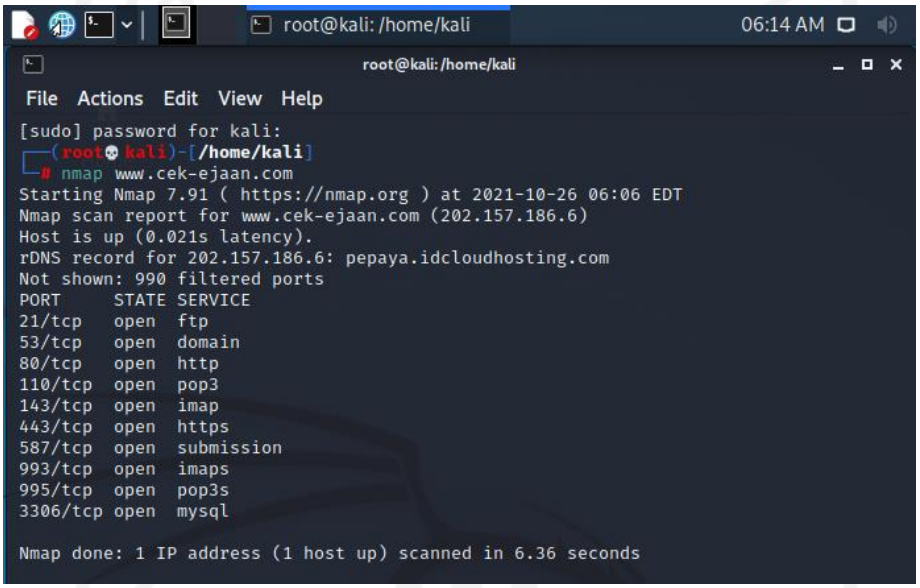
Gambar 2 Hasil *ping* target menggunakan *Command Prompt*

Berdasarkan Gambar 2 dapat dilihat bahwa dengan perintah *ping* dapat diketahui *IP address* dari suatu *website*, dalam penelitian ini menggunakan target *website* cek-ejaan.com dengan *IP address* 202.157.186.6. Selanjutnya dengan menggunakan *tools whois* yang merupakan salah satu *tools* yang terdapat dalam *OS Kali Linux*, dapat digunakan dalam mencari informasi terkait pemilik sebuah *domain* maupun alamat *IP* pada suatu *website* seperti pada Gambar 3.



Gambar 3 Hasil menggunakan *Whois*

Berdasarkan Gambar 3 didapatkan nama *domain*, *id domain* yang didaftarkan, dan lain sebagainya, dari data tersebut dapat dilihat bahwa informasi *contact* yang tertera berisi tentang penyedia *whois domain* sehingga dapat disimpulkan bahwa *website* cek-ejaan.com menjaga privasi atau menerapkan *whois privacy protection*. Pada tahap selanjutnya dilakukan *port scanning* untuk mengidentifikasi sistem menggunakan *tools Nmap* yang terdapat pada *Kali Linux* seperti yang tertuang pada Gambar 4.

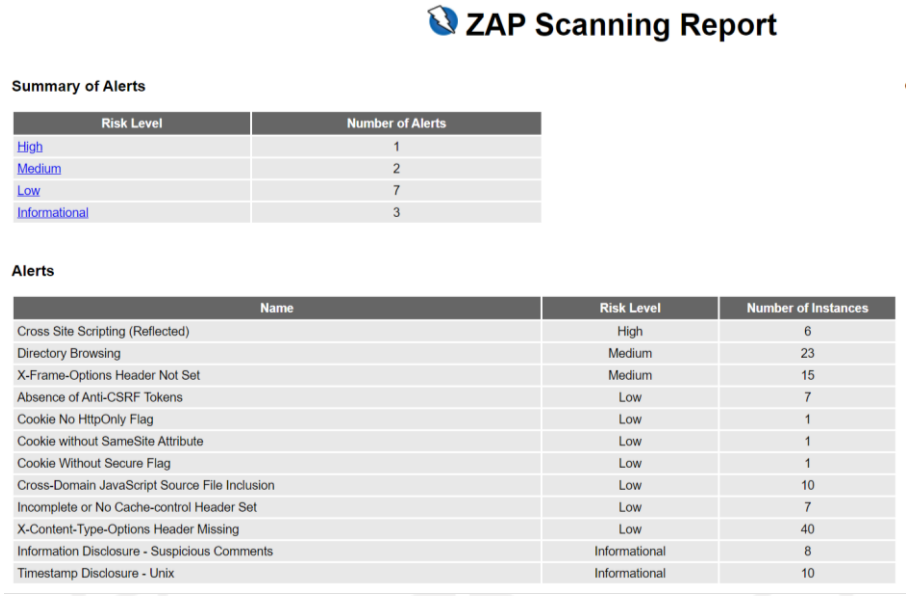


Gambar 4 Hasil port *scanning Nmap*

Nmap atau *network mapper* merupakan sebuah *tools security scanner* yang digunakan untuk *scanning* dan memeriksa *port* suatu *host* yang terbuka. Berdasarkan Gambar 4 dapat dilihat bahwa *port* pada *website* target terbuka semua sehingga dapat disimpulkan bahwa *website* target rentan untuk diexploitasi.

Scanning otomatis OWASP ZAP

Tahap ini dilakukan pengujian celah keamanan *(Vulnerability Identification)* pada *website* Cek-ejaan.com menggunakan alat scan otomatis *OWASP ZAP*. *OWASP Zed Attack Proxy (ZAP)* adalah pemindai aplikasi berbasis website yang bersifat *open source*. Dari proses *scanning* ditemukan *alerts* seperti pada Gambar 5.



ZAP Scanning Report

Summary of Alerts

Risk Level	Number of Alerts
High	1
Medium	2
Low	7
Informational	3

Alerts

Name	Risk Level	Number of Instances
Cross Site Scripting (Reflected)	High	6
Directory Browsing	Medium	23
X-Frame-Options Header Not Set	Medium	15
Absence of Anti-CSRF Tokens	Low	7
Cookie No HttpOnly Flag	Low	1
Cookie without SameSite Attribute	Low	1
Cookie Without Secure Flag	Low	1
Cross-Domain JavaScript Source File Inclusion	Low	10
Incomplete or No Cache-control Header Set	Low	7
X-Content-Type-Options Header Missing	Low	40
Information Disclosure - Suspicious Comments	Informational	8
Timestamp Disclosure - Unix	Informational	10

Gambar 5 Hasil *automated* scan *OWASP ZAP*

Berdasarkan Gambar 5 menunjukkan hasil *automated scan* menggunakan *OWASP Zed Attack Proxy (ZAP)* dengan celah keamanan yang terdeteksi pada *website* cek-ejaan.com yaitu sebanyak 12 *alerts* dengan jumlah kasus per *alerts* seperti yang tertera pada gambar, serta terbagi menjadi 3 kategori level risiko *high*, *medium*, *low*, dan *informational*, sehingga didapatkan kesimpulan bahwa website target tergolong cukup aman.

Metode OWASP Versi 4.2

Tahap ini dilakukan pengujian *website* target menggunakan *framework WSTG (Web Security Testing Guide) OWASP* Versi 4 dengan fokus pengujian *authentication, authorization,* dan *session management* menggunakan beberapa kombinasi tools. Hasil yang diperoleh dalam pengujian ini tertuang pada Tabel 1 dibawah ini.

Tabel 1 Hasil Pengujian menggunakan metode *OWASP* versi 4.2

Kategori Pengujian	Tahapan Pengujian	Aktivitas	*Tools*	Status	Hasil
Authentication Testing	*Testing for Credentials Transported over an Encrypted Channel*	Memastikan data kredensial terenkripsi yang ditransfer dari klien ke server aplikasi web saat login	• *Google Chrome*	Tidak ditemukan	Menerapkan *HTTPS*

Kategori Pengujian	Tahapan Pengujian	Aktivitas	*Tools*	Status	Hasil
	(WSTG-ATHN-01)	dan menggunakan aplikasi diakses melalui HTTPS.			
	Testing for default credentials (WSTG-ATHN-02)	Memprediksi kredensial default dan validasi dari halaman login	• *THC-Hydra*	Tidak ditemukan	Tidak ditemukan username default
	Testing for Weak lock out mechanism (WSTG-ATHN-03)	Melakukan beberapa kali login dengan *password* yang salah untuk menguji kemampuan mekanisme penguncian akun.	• *Google Chrome*	Ditemukan	Tidak ada mekanisme penguncian pada *users invalid login*
	Testing for bypassing authenticationschema (WSTG-ATHN-04)	Melakukan pengujian skema otentikasi dengan metode permintaan halaman secara langsung atau penjelajahan paksa dan modifikasi parameter URL.	• *SQL Map* • *Mozilla Firefox*	Tidak ditemukan	*URL* tidak dapat dimodifikasi dan tidak dapat mengakses halaman lain secara langsung melalui penjelajahan paksa.
	Test remember password functionality (WSTG-ATHN-05)	Memastikan keamanan kredensial pengguna pada sesi yang dihasilkan dan dikelola	• *Google Chrome* • *OWASP ZAP*	Ditemukan	*Autocomplete* aktif

Kategori Pengujian	Tahapan Pengujian	Aktivitas	*Tools*	Status	Hasil
		dengan melihat *log password* yang disimpan serta pengaktifan *autocomplete*.			
	Testing for Browser cache weakness (WSTG-ATHN-06)	Memeriksa *cache browser* dari tombol "Kembali" untuk melihat sumber daya yang ditampilkan pada halaman sebelumnya apakah informasi sensitif dapat diakses saat tidak diautentikasi.	• *Google Chrome* • *OWASP ZAP*	Tidak ditemukan	*Cache browser* tidak ditemukan
	Testing for Weak password policy (WSTG-ATHN-07)	Memeriksa ketahanan aplikasi terhadap tebakan kata sandi dengan melakukan *brute force* menggunakan kamus kata sandi.	• *THC-Hydra*	Tidak ditemukan	Tidak ada kata sandi yang berhasil ditebak.
	Testing for Weak security question/ answer (WSTG-ATHN-08)	Memeriksa skema pertanyaan keamanan dengan mengumpulkan kemungkinan jawaban dari serangkaian	-	Tidak ditemukan	Tidak terdapat skema pertanyaan keamanan

Kategori Pengujian	Tahapan Pengujian	Aktivitas	*Tools*	Status	Hasil
		pertanyaan keamanan.			
	Testing for weak password change or reset functionalities(WSTG-ATHN-09)	Memeriksa mekanisme perubahan atau pengaturan ulang kata sandi	• *Google Chrome*	Ditemukan	Fungsi perubahan kata sandi lemah
	Testing for Weaker authentication in alternativ echannel (WSTG-ATHN-10)	Mengidentifikasi saluran autentikasi lain (alternatif)	-	Tidak ditemukan	Tidak terdapat saluran alternatif
Authorization Testing	*Testing directory traversal file include (WSTG-ATHZ-01)*	Mengidentifikasi lokasi file *root directory* atau *root dokumen web*.	• *Dirb* • *Mozilla Firefox*	Ditemukan	Terdapat *URL* yang mengarah pada *root directory* web atau *directory traversal*
	Testing for bypassing authorizationschema (WSTG-ATHZ-02)	Melakukan percobaan untuk mengakses dan mengoperasikan fungsi pada sumber daya khusus tanpa proses autentikasi.	• *Dirb* • *Mozilla Firefox*	Ditemukan	Berhasil melewati proses autentikasi dan mengakses sumber daya khusus
	Testing for Privilege Escalation (WSTG-ATHZ-03)	Mengidentifikasi *hidden field HTML* yang terkait dengan	• *WebScarab*	Tidak ditemukan	Hak istimewa tidak dapat dimanipulasi

Kategori Pengujian	Tahapan Pengujian	Aktivitas	*Tools*	Status	Hasil
		manipulasi hak istimewa			
	Testing for Insecure Direct Obje cReferences (WSTG-ATHZ-04)	Memodifikasi parameter *URL* yang terdapat pada halaman log aplikasi untuk menguji skema otentikasi	• *Mozilla Firefox*	Tidak ditemukan	Parameter *URL* pada halaman log tidak dapat dimodifikasi
Session Management Testing	*Testing for Bypassing Session Managemen t Schema (WSTG-SESS-01)*	Menganalisa kerentanan terhadap serangan hijacking dengan melihat status *cookie* dan mencoba mengakses halaman menggunakan *cookie*.	• *OWASP ZAP (Zed Attack Proxy)* • *Dirb*	Ditemukan	Status *cookie No HttpOnly Flag* sehingga dapat diakses melalui *JavaScript* dan rentan terhadap serangan *hijacking*
	Testing for Cookies attributes (WSTG-SESS-02)	Memeriksa atribut *cookies* yang digunakan, seperti *secure attribute, httponly attribute, domain attribute, path attribute, expires attribute*	• *OWASP ZAP (Zed Attack Proxy)*	Ditemukan	*No secure Attribute, No HttpOnly Attribute*
	Testing for Session Fixation (WSTG-SESS-03)	Memeriksa *session ID* atau *cookie* setelah berhasil melakukan autentikasi	• *Mozilla Firefox*	Ditemukan	*Session ID* atau *cookie* tidak berubah setelah *login*

Kategori Pengujian	Tahapan Pengujian	Aktivitas	*Tools*	Status	Hasil
		apakah terjadi pembaharuan.			
	Testing for Exposed Session Variables (WSTG-SESS-04)	Memeriksa token sesi *(Cookie, Session ID, Hidden Field)* yang dikirim melalui saluran terenkripsi apakah terdapat perbedaan setiap kali membuat permintaan *HTTP*	• *Mozilla firefox*	Tidak ditemukan	Token sesi berubah setiap membuat permintaan *HTTP*
	Testing for Cross Site Request Forgery (WSTG-SESS-05)	Memastikan apakah terdapat kerentanan terhadap *Cross Site Request Forgery (CSRF)*	• *Burpsuite*	Ditemukan	Dapat mengidentifikasi *URL* aplikasi, parameter, dan nilai yang sah
	Testing for logout functionality (WSTG-SESS-06)	Melakukan analisa terhadap batas waktu sesi apakah sesi dihentikan setelah *logout* dan apakah terdapat *locked invalid login*	• *Mozilla Firefox*	Ditemukan	Sesi dihentikan setelah *logout*. Namun, tidak ada mekanisme penguncian saat *invalid login*
	Test Session Timeout (WSTG-SESS-07)	Memeriksa batas waktu pada aplikasi web saat tidak digunakan pada durasi	• *Mozilla Firefox*	Ditemukan	Tidak ada logout otomatis atau *session time out* pada saat

Kategori Pengujian	Tahapan Pengujian	Aktivitas	*Tools*	Status	Hasil
		tertentu, apakah akan *logout* secara otomatis, serta memeriksa tombol back pada sesi yang sama			aplikasi tidak digunakan pada durasi tertentu
	Testing for Session puzzling (WSTG-SESS-08)	Mengidentifikasi dan menghitung variabel valid setiap sesi yang digunakan	• *Mozilla Firefox*	Ditemukan	Ukuran variabel valid sama pada setiap sesi
	Testing for Session Hijacking (WSTG-SESS-09)	Mengidentifikasi kerentanan *cookies session* dengan melihat status *cookies*	• *OWASP ZAP (Zed Attack Proxy)*	Ditemukan	Status cookie tidak *secure*

Berdasarkan Tabel 1 dapat dilihat bahwa terdapat tiga kategori pengujian yang terbagi menjadi 23 sub kategori. Kategori pengujian *authentication* pada sub kategori pengujian *WSTG-ATHN-01* menggunakan *tools Google Chrome* diperoleh hasil tidak ditemukan celah keamanan karena *website* telah menerapkan *HTTPS.* Pengujian *WSTG-ATHN-02* menggunakan *tools THC-Hydra* tidak ditemukan celah keamanan karena tidak ditemukan *username* dan *password default.* Pengujian *WSTG-ATHN-03* menggunakan *tools Browser Google Chrome* ditemukan celah keamanan karena tidak adanya mekanisme penguncian akun pada saat *users invalid login*. Hasil pengujian *WSTG-ATHN-04* menggunakan *tools SQL Map* dan *Browser Mozilla Firefox* menunjukkan bahwa *URL* tidak dapat dimodifikasi serta akses halaman lain secara langsung melalui penjelajahan paksa tidak dapat dilakukan sehingga status website tidak memiliki risiko celah pada pengujian ini. Hasil pengujian *WSTG-ATHN-05* menggunakan *tools Google Chrome dan OWASP ZAP* menunjukkan bahwa status *autocomplete on* sehingga *website* memiliki risiko kerentanan pada pengujian ini. Hasil menunjukkan bahwa pada pengujian *WSTG-ATHN-06 website* target ditemukan celah

keamanan yang terletak pada cache browser. Pengujian autentikasi ke tujuh atau *WSTG-ATHN-07* dengan melakukan *brute force* menggunakan *tools THC Hydra* dinyatakan tidak memiliki celah keamanan karena tidak ditemukan *username* dan *password* yang sesuai pada *website* target. Hasil yang diperoleh pada pengujian *WSTG-ATHN-08* dengan memeriksa skema pertanyaan keamanan secara langsung yaitu tidak adanya skema pernyataan keamanan sehingga dalam hal ini *website* target memiliki risiko celah keamanan. Pengujian *WSTG-ATHN-09* dengan melakukan pemeriksaan *website* target terhadap mekanisme perubahan atau pengaturan ulang kata sandi dinyatakan memiliki celah keamanan karena hanya membutuhkan nama lengkap dan *username* ketika hendak merubah *password* pada *website* target sehingga mekanisme perubahan kata sandi dinyatakan lemah. Pengujian *WSTG-ATHN-10* yang dilakukan dengan mengidentifikasi saluran alternatif untuk autentikasi tidak ditemukan celah keamanan karena tidak adanya saluran autentikasi lain.

Ketegori pengujian *authorization* terbagi menjadi empat sub kategori, pada sub kategori pengujian pertama atau *WSTG-ATHZ-01* menggunakan *tools Dirb* dan *Mozilla Firefox* dengan melakukan identifikasi lokasi file *root directory* atau *root dokumen* dinyatakan memiliki celah keamanan karena ditemukan *URL* yang mengarah pada *root directory* web. Pengujian *WSTG-ATHZ-02* menggunakan *tools Dirb* dan *Mozilla Firefox* memiliki celah keamanan karena pada saat percobaan akses dan pengoperasian fungsi pada sumber daya khusus dapat dilakukan tanpa adanya proses autentikasi. Pengujian *WSTG-ATHZ-03* menggunakan *tools WebScarab* dinyatakan tidak lolos atau tidak terdapat celah karena hak istimewa tidak dapat dimanipulasi. Pengujian *WSTG-ATHZ-04* menggunakan *tools Mozilla Firefox* untuk menguji skema otentikasi tidak terdapat celah karena parameter *URL* yang terdapat pada halaman *log* aplikasi tidak dapat dimodifikasi.

Kategori pengujian ketiga yaitu *session management testing*. Pada kategori ini terbagi menjadi sembilan pengujian, pada pengujian pertama *(WSTG-SESS-01)* diperoleh hasil website tidak aman pada saat dilakukan pengecekan menggunakan *tools OWASP ZAP* dan *Dirb*, karena status cookie *No HttpOnly Flag* sehingga dapat diakses melalui *JavaScript* dan rentan terhadap serangan *hijacking.* Pengujian kedua *(WSTG-SESS-02)* dengan memeriksa atribut *cookies* yang digunakan menggunakan *tools OWASP ZAP* diperoleh hasil terdapat celah keamanan dikarenakan atribut cookies *no secure attribute dan no httponly attribute.* Pengujian ketiga *(WSTG-SESS-03)* diperoleh hasil *Session ID* atau *cookie* tidak berubah setelah *login* menggunakan *tools Mozilla Firefox* sehingga ditemukan celah keamanan dalam pengujian ini. Pengujian keempat *(WSTG-SESS-04)* dengan memeriksa token sesi yang dikirim melalui

saluran terenkripsi diperoleh hasil terdapat perbedaan setiap kali membuat permintaan *HTTP* sehingga pada pengujian ini tidak ditemukan risiko celah keamanan. Pengujian kelima *(WSTG-SESS-05)* diperoleh hasil berupa *URL* aplikasi, parameter, dan nilai yang sah dapat diidentifikasi sehingga dalam pengujian ini terdapat kerentanan pada *website* cek-ejaan.com Pengujian keenam *(WSTG-SESS-06)* diperoleh hasil berupa sesi dihentikan setelah *logout*. Namun, tidak ada mekanisme penguncian saat *invalid login* sehingga dalam pengujian ini ditemukan kemungkinan risiko celah keamanan. *Pengujian ketujuh (WSTG-SESS-07)* diperoleh hasil berupa tidak ada *logout* otomatis atau *session time out* pada saat aplikasi tidak digunakan pada durasi tertentu sehingga pada pengujian ini terdapat kemungkinan celah keamanan. *Pengujian kedelapan (WSTG-SESS-08)* diperoleh hasil berupa Ukuran variabel valid terdapat kesamaan pada setiap sesi yang dihasilkan sehingga pada pengujian ini terdapat risiko celah keamanan. *Pengujian terakhir (WSTG-SESS-09)* diperoleh hasil berupa status pada *cookie* tidak *secure* sehingga pada pengujian ini ditemukan risiko celah keamanan.

ZAP Scanning Report

Summary of Alerts

Risk Level	Number of Alerts
High	1
Medium	2
Low	7
Informational	3

Alerts

Name	Risk Level	Number of Instances
Cross Site Scripting (Reflected)	High	6
Directory Browsing	Medium	23
X-Frame-Options Header Not Set	Medium	15
Absence of Anti-CSRF Tokens	Low	7
Cookie No HttpOnly Flag	Low	1
Cookie without SameSite Attribute	Low	1
Cookie Without Secure Flag	Low	1
Cross-Domain JavaScript Source File Inclusion	Low	10
Incomplete or No Cache-control Header Set	Low	7
X-Content-Type-Options Header Missing	Low	40
Information Disclosure - Suspicious Comments	Informational	8
Timestamp Disclosure - Unix	Informational	10

Alerts Detail

High (Medium)	Cross Site Scripting (Reflected)
Description	Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied code into a user's browser instance. A browser instance can be a standard web browser client, or a browser object embedded in a software product such as the browser within WinAmp, an RSS reader, or an email client. The code itself is usually written in HTML/JavaScript, but may also extend to VBScript, ActiveX, Java, Flash, or any other browser-supported technology. When an attacker gets a user's browser to execute his/her code, the code will run within the security context (or zone) of the hosting web site. With this level of privilege, the code has the ability to read, modify and transmit any sensitive data accessible by the browser. A Cross-site Scripted user could have his/her account hijacked (cookie theft), their browser redirected to another location, or possibly shown fraudulent content delivered by the web site they are visiting. Cross-site Scripting attacks essentially compromise the trust relationship between a user and the web site. Applications utilizing browser object instances which load content from the file system may execute code under the local machine zone allowing for system compromise. There are three types of Cross-site Scripting attacks: non-persistent, persistent and DOM-based. Non-persistent attacks and DOM-based attacks require a user to either visit a specially crafted link laced with malicious code, or visit a malicious web page containing a web form, which when posted to the vulnerable site, will mount the attack. Using a malicious form will oftentimes take place when the vulnerable resource only accepts HTTP POST requests. In such a case, the form can be submitted automatically, without the victim's knowledge (e.g. by using JavaScript). Upon clicking on the malicious link or submitting the malicious form, the XSS payload will get echoed back and will get interpreted by the user's browser and execute. Another technique to send almost arbitrary requests (GET and POST) is by using an embedded client, such as Adobe Flash. Persistent attacks occur when the malicious code is submitted to a web site where it's stored for a period of time. Examples of an attacker's favorite targets often include message board posts, web mail messages, and web chat software. The unsuspecting user is not required to interact with any additional site/link (e.g. an attacker

	site or a malicious link sent via email), just simply view the web page containing the code.
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	POST
Parameter	username
Attack	">\x0000<script>alert(1);</script>
Evidence	">\x0000<script>alert(1);</script>
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	POST
Parameter	kelas
Attack	">\x0000<script>alert(1);</script>
Evidence	">\x0000<script>alert(1);</script>
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	POST
Parameter	namalengkap
Attack	">\x0000<script>alert(1);</script>
Evidence	">\x0000<script>alert(1);</script>
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	POST
Parameter	email
Attack	">\x0000<script>alert(1);</script>
Evidence	">\x0000<script>alert(1);</script>
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	POST

Parameter	nomor_induk
Attack	">\x0000<script>alert(1);</script>
Evidence	">\x0000<script>alert(1);</script>
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	POST
Parameter	institusi
Attack	">\x0000<script>alert(1);</script>
Evidence	">\x0000<script>alert(1);</script>
Instances	6
Solution	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid. Examples of libraries and frameworks that make it easier to generate properly encoded output include Microsoft's Anti-XSS library, the OWASP ESAPI Encoding module, and Apache Wicket. Phases: Implementation; Architecture and Design Understand the context in which your data will be used and the encoding that will be expected. This is especially important when transmitting data between different components, or when generating outputs that can contain multiple encodings at the same time, such as web pages or multi-part mail messages. Study all expected communication protocols and data representations to determine the required encoding strategies. For any data that will be output to another web page, especially any data that was received from external inputs, use the appropriate encoding on all non-alphanumeric characters. Consult the XSS Prevention Cheat Sheet for more details on the types of encoding and escaping that are needed. Phase: Architecture and Design

For any security checks that are performed on the client side, ensure that these checks are duplicated on the server side, in order to avoid CWE-602. Attackers can bypass the client-side checks by modifying values after the checks have been performed, or by changing the client to remove the client-side checks entirely. Then, these modified values would be submitted to the server.

If available, use structured mechanisms that automatically enforce the separation between data and code. These mechanisms may be able to provide the relevant quoting, encoding, and validation automatically, instead of relying on the developer to provide this capability at every point where output is generated.

Phase: Implementation

For every web page that is generated, use and specify a character encoding such as ISO-8859-1 or UTF-8. When an encoding is not specified, the web browser may choose a different encoding by guessing which encoding is actually being used by the web page. This can cause the web browser to treat certain sequences as special, opening up the client to subtle XSS attacks. See CWE-116 for more mitigations related to encoding/escaping.

To help mitigate XSS attacks against the user's session cookie, set the session cookie to be HttpOnly. In browsers that support the HttpOnly feature (such as more recent versions of Internet Explorer and Firefox), this attribute can prevent the user's session cookie from being accessible to malicious client-side scripts that use document.cookie. This is not a complete solution, since HttpOnly is not supported by all browsers. More importantly, XMLHTTPRequest and other powerful browser technologies provide read access to HTTP headers, including the Set-Cookie header in which the HttpOnly flag is set.

Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use an allow list of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. Do not rely exclusively on looking for malicious or malformed inputs (i.e., do not rely on a deny list). However, deny lists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright.

When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid

	because it only contains alphanumeric characters, but it is not valid if you are expecting colors such as "red" or "blue." Ensure that you perform input validation at well-defined interfaces within the application. This will help protect the application even if a component is reused or moved elsewhere.
Reference	http://projects.webappsec.org/Cross-Site-Scripting http://cwe.mitre.org/data/definitions/79.html
CWE Id	79
WASC Id	8
Source ID	1
Medium (Medium)	**Directory Browsing**
Description	It is possible to view the directory listing. Directory listing may reveal hidden scripts, include files, backup source files, etc. which can be accessed to read sensitive information.
URL	https://cek-ejaan.com/assets/vendor/icofont/
Method	GET
Attack	https://cek-ejaan.com/assets/vendor/icofont/
Evidence	Parent Directory
URL	https://cek-ejaan.com/assets/vendor/aos/
Method	GET
Attack	https://cek-ejaan.com/assets/vendor/aos/
Evidence	Parent Directory
URL	https://cek-ejaan.com/assets/js/
Method	GET
Attack	https://cek-ejaan.com/assets/js/
Evidence	Parent Directory

URL	https://cek-ejaan.com/assets/vendor/php-email-form/
Method	GET
Attack	https://cek-ejaan.com/assets/vendor/php-email-form/
Evidence	Parent Directory
URL	https://cek-ejaan.com/assets/
Method	GET
Attack	https://cek-ejaan.com/assets/
Evidence	Parent Directory
URL	https://cek-ejaan.com/assets/vendor/bootstrap/js/
Method	GET
Attack	https://cek-ejaan.com/assets/vendor/bootstrap/js/
Evidence	Parent Directory
URL	https://cek-ejaan.com/assets/vendor/jquery/
Method	GET
Attack	https://cek-ejaan.com/assets/vendor/jquery/
Evidence	Parent Directory
URL	https://cek-ejaan.com/assets/vendor/bootstrap/
Method	GET
Attack	https://cek-ejaan.com/assets/vendor/bootstrap/
Evidence	Parent Directory
URL	https://cek-ejaan.com/assets/vendor/jquery.easing/
Method	GET
Attack	https://cek-ejaan.com/assets/vendor/jquery.easing/
Evidence	Parent Directory

URL	https://cek-ejaan.com/img/
Method	GET
Attack	https://cek-ejaan.com/img/
Evidence	Parent Directory
URL	https://cek-ejaan.com/assets/img/
Method	GET
Attack	https://cek-ejaan.com/assets/img/
Evidence	Parent Directory
URL	https://cek-ejaan.com/assets/vendor/boxicons/
Method	GET
Attack	https://cek-ejaan.com/assets/vendor/boxicons/
Evidence	Parent Directory
URL	https://cek-ejaan.com/assets/vendor/counterup/
Method	GET
Attack	https://cek-ejaan.com/assets/vendor/counterup/
Evidence	Parent Directory
URL	https://cek-ejaan.com/assets/vendor/owl.carousel/assets/
Method	GET
Attack	https://cek-ejaan.com/assets/vendor/owl.carousel/assets/
Evidence	Parent Directory
URL	https://cek-ejaan.com/assets/css/
Method	GET
Attack	https://cek-ejaan.com/assets/css/
Evidence	Parent Directory

URL	https://cek-ejaan.com/assets/vendor/remixicon/
Method	GET
Attack	https://cek-ejaan.com/assets/vendor/remixicon/
Evidence	Parent Directory
URL	https://cek-ejaan.com/assets/vendor/animate.css/
Method	GET
Attack	https://cek-ejaan.com/assets/vendor/animate.css/
Evidence	Parent Directory
URL	https://cek-ejaan.com/assets/vendor/waypoints/
Method	GET
Attack	https://cek-ejaan.com/assets/vendor/waypoints/
Evidence	Parent Directory
URL	https://cek-ejaan.com/assets/vendor/
Method	GET
Attack	https://cek-ejaan.com/assets/vendor/
Evidence	Parent Directory
URL	https://cek-ejaan.com/inc/
Method	GET
Attack	https://cek-ejaan.com/inc/
Evidence	Parent Directory
Instances	23
Solution	Disable directory browsing. If this is required, make sure the listed files does not induce risks.
Reference	http://httpd.apache.org/docs/mod/core.html#options

	http://alamo.satlug.org/pipermail/satlug/2002-February/000053.html
CWE Id	548
WASC Id	48
Source ID	1
Medium (Medium)	**X-Frame-Options Header Not Set**
Description	X-Frame-Options header is not included in the HTTP response to protect against 'ClickJacking' attacks.
URL	https://cek-ejaan.com/login_signup_master.php?page=lostpass
Method	GET
Parameter	X-Frame-Options
URL	https://cek-ejaan.com/index.php?page=unggah
Method	GET
Parameter	X-Frame-Options
URL	https://cek-ejaan.com/
Method	GET
Parameter	X-Frame-Options
URL	https://cek-ejaan.com/index.php?page=_edit
Method	GET
Parameter	X-Frame-Options
URL	https://cek-ejaan.com/index.php?page=home
Method	GET
Parameter	X-Frame-Options
URL	https://cek-ejaan.com/index.php?page=tentang
Method	GET

Parameter	X-Frame-Options
URL	https://cek-ejaan.com/login_signup_master.php?page=lostpass
Method	POST
Parameter	X-Frame-Options
URL	https://cek-ejaan.com/index.php
Method	GET
Parameter	X-Frame-Options
URL	https://cek-ejaan.com
Method	GET
Parameter	X-Frame-Options
URL	https://cek-ejaan.com/index.php?page=edit
Method	GET
Parameter	X-Frame-Options
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	GET
Parameter	X-Frame-Options
URL	https://cek-ejaan.com/login_signup_master.php?page=login
Method	GET
Parameter	X-Frame-Options
URL	https://cek-ejaan.com/index.html
Method	GET
Parameter	X-Frame-Options
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	POST

Parameter	X-Frame-Options
URL	https://cek-ejaan.com/index.html
Method	POST
Parameter	X-Frame-Options
Instances	15
Solution	Most modern Web browsers support the X-Frame-Options HTTP header. Ensure it's set on all web pages returned by your site (if you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. Alternatively consider implementing Content Security Policy's "frame-ancestors" directive.
Reference	https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
CWE Id	1021
WASC Id	15
Source ID	3
Low (Medium)	**Absence of Anti-CSRF Tokens**
Description	No Anti-CSRF tokens were found in a HTML submission form. A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL/form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) exploits the trust that a user has for a web site. Like XSS, CSRF attacks are not necessarily cross-site, but they can be. Cross-site request forgery is also known as CSRF, XSRF, one-click attack, session riding, confused deputy, and sea surf. CSRF attacks are effective in a number of situations, including: * The victim has an active session on the target site. * The victim is authenticated via HTTP auth on the target site.

	* The victim is on the same local network as the target site. CSRF has primarily been used to perform an action against a target site using the victim's privileges, but recent techniques have been discovered to disclose information by gaining access to the response. The risk of information disclosure is dramatically increased when the target site is vulnerable to XSS, because XSS can be used as a platform for CSRF, allowing the attack to operate within the bounds of the same-origin policy.

URL	https://cek-ejaan.com/login_signup_master.php?page=login
Method	GET
Evidence	<form method="post" action="inc/login_process.php">
URL	https://cek-ejaan.com/login_signup_master.php?page=lostpass
Method	POST
Evidence	<form method="post" action="">
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	POST
Evidence	<form method="post" action="">
URL	https://cek-ejaan.com/login_signup_master.php?page=lostpass
Method	GET
Evidence	<form method="post" action="">
URL	https://cek-ejaan.com/index.html
Method	POST
Evidence	<form action="" method="post">
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	GET
Evidence	<form method="post" action="">

URL	https://cek-ejaan.com/index.html
Method	GET
Evidence	<form action="" method="post">
Instances	7
Solution	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid. For example, use anti-CSRF packages such as the OWASP CSRFGuard. Phase: Implementation Ensure that your application is free of cross-site scripting issues, because most CSRF defenses can be bypassed using attacker-controlled script. Phase: Architecture and Design Generate a unique nonce for each form, place the nonce into the form, and verify the nonce upon receipt of the form. Be sure that the nonce is not predictable (CWE-330). Note that this can be bypassed using XSS. Identify especially dangerous operations. When the user performs a dangerous operation, send a separate confirmation request to ensure that the user intended to perform that operation. Note that this can be bypassed using XSS. Use the ESAPI Session Management control. This control includes a component for CSRF. Do not use the GET method for any request that triggers a state change. Phase: Implementation Check the HTTP Referer header to see if the request originated from an expected page. This could break legitimate functionality,

	because users or proxies may have disabled sending the Referer for privacy reasons.
Other information	No known Anti-CSRF token [anticsrf, CSRFToken, __RequestVerificationToken, csrfmiddlewaretoken, authenticity_token, OWASP_CSRFTOKEN, anoncsrf, csrf_token, _csrf, _csrfSecret, __csrf_magic, CSRF] was found in the following HTML form: [Form 1: "pwd" "username"].

Reference	http://projects.webappsec.org/Cross-Site-Request-Forgery http://cwe.mitre.org/data/definitions/352.html
CWE Id	352
WASC Id	9
Source ID	3
Low (Medium)	**Cookie No HttpOnly Flag**
Description	A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site. If this is a session cookie then session hijacking may be possible.

URL	https://cek-ejaan.com
Method	GET
Parameter	PHPSESSID
Evidence	Set-Cookie: PHPSESSID
Instances	1
Solution	Ensure that the HttpOnly flag is set for all cookies.
Reference	https://owasp.org/www-community/HttpOnly
CWE Id	1004
WASC Id	13
Source ID	3

Low (Medium)	Cookie without SameSite Attribute
Description	A cookie has been set without the SameSite attribute, which means that the cookie can be sent as a result of a 'cross-site' request. The SameSite attribute is an effective counter measure to cross-site request forgery, cross-site script inclusion, and timing attacks.

URL	https://cek-ejaan.com
Method	GET
Parameter	PHPSESSID
Evidence	Set-Cookie: PHPSESSID
Instances	1
Solution	Ensure that the SameSite attribute is set to either 'lax' or ideally 'strict' for all cookies.
Reference	https://tools.ietf.org/html/draft-ietf-httpbis-cookie-same-site
CWE Id	1275
WASC Id	13
Source ID	3

Low (Medium)	Cookie Without Secure Flag
Description	A cookie has been set without the secure flag, which means that the cookie can be accessed via unencrypted connections.

URL	https://cek-ejaan.com
Method	GET
Parameter	PHPSESSID
Evidence	Set-Cookie: PHPSESSID
Instances	1
Solution	Whenever a cookie contains sensitive information or is a session token, then it should always be passed using an encrypted channel. Ensure that the secure flag is set for cookies containing such sensitive information.

Reference	https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/06-Session_Management_Testing/02-Testing_for_Cookies_Attributes.html
CWE Id	614
WASC Id	13
Source ID	3
Low (Medium)	**Cross-Domain JavaScript Source File Inclusion**
Description	The page includes one or more script files from a third-party domain.
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	POST
Parameter	//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js
Evidence	<script src="//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
URL	https://cek-ejaan.com/login_signup_master.php?page=login
Method	GET
Parameter	//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js
Evidence	<script src="//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
URL	https://cek-ejaan.com/login_signup_master.php?page=lostpass
Method	GET
Parameter	//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js

Evidence	<script src="//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
URL	https://cek-ejaan.com/login_signup_master.php?page=lostpass
Method	POST
Parameter	//cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js
Evidence	<script src="//cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
URL	https://cek-ejaan.com/login_signup_master.php?page=lostpass
Method	GET
Parameter	//cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js
Evidence	<script src="//cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
URL	https://cek-ejaan.com/login_signup_master.php?page=login
Method	GET
Parameter	//cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js
Evidence	<script src="//cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
URL	https://cek-ejaan.com/login_signup_master.php?page=lostpass
Method	POST

Parameter	//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js
Evidence	<script src="//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	GET
Parameter	//cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js
Evidence	<script src="//cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	GET
Parameter	//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js
Evidence	<script src="//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	POST
Parameter	//cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js
Evidence	<script src="//cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
Instances	10

Solution	Ensure JavaScript source files are loaded from only trusted sources, and the sources can't be controlled by end users of the application.
Reference	
CWE Id	829
WASC Id	15
Source ID	3
Low (Medium)	**Incomplete or No Cache-control Header Set**
Description	The cache-control header has not been set properly or is missing, allowing the browser and proxies to cache content.
URL	https://cek-ejaan.com/login_signup_master.php?page=lostpass
Method	GET
Parameter	Cache-Control
Evidence	max-age=172800
URL	https://cek-ejaan.com/index.html
Method	GET
Parameter	Cache-Control
Evidence	max-age=172800
URL	https://cek-ejaan.com/login_signup_master.php?page=lostpass
Method	POST
Parameter	Cache-Control
Evidence	max-age=172800
URL	https://cek-ejaan.com/login_signup_master.php?page=login

Method	GET
Parameter	Cache-Control
Evidence	max-age=172800
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	GET
Parameter	Cache-Control
Evidence	max-age=172800
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	POST
Parameter	Cache-Control
Evidence	max-age=172800
URL	https://cek-ejaan.com/index.html
Method	POST
Parameter	Cache-Control
Evidence	max-age=172800
Instances	7
Solution	Whenever possible ensure the cache-control HTTP header is set with no-cache, no-store, must-revalidate.
Reference	https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html#web-content-caching https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Cache-Control
CWE Id	525

WASC Id	13
Source ID	3
Low (Medium)	**X-Content-Type-Options Header Missing**
Description	The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.
URL	https://cek-ejaan.com/assets/vendor/icofont/icofont.min.css
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/assets/img/about.webp
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/assets/vendor/owl.carousel/owl.carousel.min.js
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/assets/img/apple-touch-icon.png
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/assets/vendor/bootstrap/css/bootstrap.min.css
Method	GET
Parameter	X-Content-Type-Options

URL	https://cek-ejaan.com/assets/img/course-3.jpg
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/assets/img/course-2.jpg
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/assets/vendor/jquery.easing/jquery.easing.min.js
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/index.html
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/assets/vendor/aos/aos.css
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/index.html
Method	POST
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	POST

Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/assets/img/favicon.png
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/img/logobar.png
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/assets/img/course-1.jpg
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/assets/vendor/aos/aos.js
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/assets/vendor/counterup/counterup.min.js
Method	GET
Parameter	X-Content-Type-Options
URL	https://cek-ejaan.com/assets/vendor/animate.css/animate.min.css
Method	GET
Parameter	X-Content-Type-Options
Instances	40

Solution	Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages. If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.
Other information	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.
Reference	http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx https://owasp.org/www-community/Security_Headers
CWE Id	693
WASC Id	15
Source ID	3
Informational (Medium)	**Information Disclosure - Suspicious Comments**
Description	The response appears to contain suspicious comments which may help an attacker. Note: Matches made within script blocks or files are against the entire content not only comments.
URL	https://cek-ejaan.com/login_signup_master.php?page=lostpass
Method	POST
Evidence	Username
URL	https://cek-ejaan.com/login_signup_master.php?page=login
Method	GET
Evidence	Username

URL	https://cek-ejaan.com/login_signup_master.php?page=lostpass
Method	GET
Evidence	Username
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	POST
Evidence	Username
URL	https://cek-ejaan.com/login_signup_master.php?page=signup
Method	GET
Evidence	Username
Instances	5
Solution	Remove all comments that return information that may help an attacker and fix any underlying problems they refer to.
Other information	The following pattern was used: \bUSERNAME\b and was detected in the element starting with: "<!-- Username -->", see evidence field for the suspicious comment/snippet.
Reference	
CWE Id	200
WASC Id	13
Source ID	3
Informational (Low)	**Information Disclosure - Suspicious Comments**
Description	The response appears to contain suspicious comments which may help an attacker. Note: Matches made within script blocks or files are against the entire content not only comments.
URL	https://cek-ejaan.com/assets/vendor/php-email-form/validate.js
Method	GET
Evidence	from

URL	https://cek-ejaan.com/assets/vendor/bootstrap/js/bootstrap.bundle.min.js
Method	GET
Evidence	from
URL	https://cek-ejaan.com/assets/vendor/jquery/jquery.min.js
Method	GET
Evidence	username
Instances	3
Solution	Remove all comments that return information that may help an attacker and fix any underlying problems they refer to.
Other information	The following pattern was used: \bFROM\b and was detected in the element starting with: " msg = 'Form submission failed and no error message returned from: ' + action + ' '; ", see evidence field for the suspicious comment/snippet.
Reference	
CWE Id	200
WASC Id	13
Source ID	3
Informational (Low)	**Timestamp Disclosure - Unix**
Description	A timestamp was disclosed by the application/web server - Unix
URL	https://cek-ejaan.com/index.php?page=_edit
Method	GET
Evidence	537445635
URL	https://cek-ejaan.com
Method	GET

Evidence	1040024401
URL	https://cek-ejaan.com/index.php?page=edit
Method	GET
Evidence	1895776819
URL	https://cek-ejaan.com/index.php
Method	GET
Evidence	887843818
URL	https://cek-ejaan.com
Method	GET
Evidence	890788654
URL	https://cek-ejaan.com/index.php?page=unggah
Method	GET
Evidence	346015480
URL	https://cek-ejaan.com/assets/img/course-3.jpg
Method	GET
Evidence	777777777
URL	https://cek-ejaan.com/index.php?page=tentang
Method	GET
Evidence	2026756173
URL	https://cek-ejaan.com/index.php?page=home
Method	GET
Evidence	1362101832
URL	https://cek-ejaan.com/
Method	GET

Evidence	1776515195
Instances	10
Solution	Manually confirm that the timestamp data is not sensitive, and that the data cannot be aggregated to disclose exploitable patterns.
Other information	537445635, which evaluates to: 1987-01-12 17:27:15

Reference	http://projects.webappsec.org/w/page/13246936/Information%20Leakage
CWE Id	200
WASC Id	13
Source ID	3