Indonesian Journal of Computer Science
ISSN 2302-4364 (*print*) dan 2549-7286 (*online*)
Jln. Khatib Sulaiman Dalam, No. 1, Padang, Indonesia, Telp. (0751) 7056199, 7058325
Website: ijcs.stmikindonesia.ac.id | E-mail: ijcs@stmikindonesia.ac.id

Analisis Keamanan Website Menggunakan Open Web Application Security Web (OWASP)

I Wayan Sriyasa, Victor Ilyas Sugara
iws@unpak.ac.id, victor.ilyas@unpak.ac.id
Universitas Pakuan

Informasi Artikel

Diterima : 29 Jan 2024
Direview : 19 Mar 2024
Disetujui : 23 Apr 2024

Kata Kunci

Keamanan, OWASP, Resiko, Informasi, ZAP

Abstrak

Aplikasi web telah menjadi bagian integral dari kehidupan sehari-hari, memberikan layanan yang diperlukan untuk berkomunikasi, berbelanja, bertransaksi, dan berbagai aktivitas lainnya. Permasalahan yang muncul terkait keamanan web adalah ketidakmampuan sistem aplikasi untuk melindungi informasi sensitif dari ancaman siber. Top 10 OWASP adalah daftar yang diperbarui secara berkala yang memuat sepuluh kerentanan keamanan aplikasi web yang paling umum terjadi Level risiko yang bersifat medium memiliki confidence level yang sama, yakni 11.1%, dengan total level risiko Medium adalah 33.3%. Seluruh celah kemanan yang ditemukan hampir semuanya berkaitan dengan A05 Kesalahan Konfigurasi Keamanan & A06 Komponen yang Rentan dan Kedaluwarsa pada OWASP Top 10:2021.Rekomendasi perbaikan terhadap temuan sudah diberikan, dan diantaranya bersifat perbaikan didalam source code dan konfigurasi pada application server/web server yang diprioritaskan kepada temuan yang bersifat High, Medium dan Low.

.

A. Pendahuluan

Aplikasi web telah menjadi bagian integral dari kehidupan sehari-hari, memberikan layanan yang diperlukan untuk berkomunikasi, berbelanja, bertransaksi, dan berbagai aktivitas lainnya. Namun, dengan perkembangan teknologi, keamanan aplikasi web semakin menjadi perhatian utama. Ancaman seperti serangan peretasan, pencurian data, dan eksploitasi kerentanan telah meningkatkan risiko keamanan dalam lingkungan web.

Web atau *World Wide Web (WWW)* adalah sistem informasi yang terdiri dari berbagai dokumen dan sumber daya yang saling terhubung dan dapat diakses melalui internet. Ini merupakan salah satu layanan yang tersedia di dalam internet. Web memungkinkan pengguna untuk menavigasi antara berbagai halaman web yang berisi teks, gambar, audio, video, dan berbagai jenis konten lainnya[1].

Kata "web" sering kali merujuk pada:

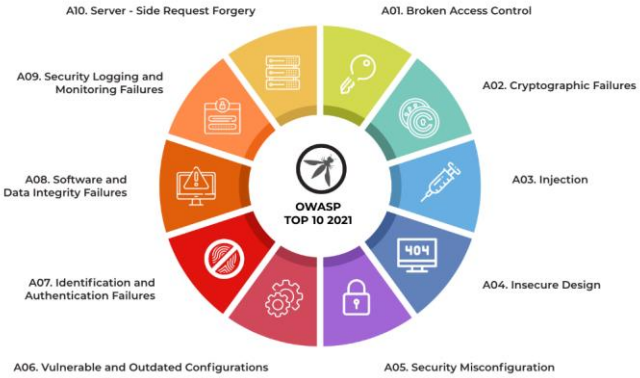
1. Halaman Web: Sebuah halaman atau dokumen yang diakses melalui internet. Halaman web ini sering kali terdiri dari teks, gambar, tautan *(link)*, dan elemen-elemen lain yang dapat diakses dengan menggunakan peramban web *(web browser)* [2]
2. *URL (Uniform Resource Locator)*: Cara untuk mengidentifikasi alamat spesifik suatu sumber daya di internet, seperti halaman web, gambar, atau dokumen lainnya. URL adalah alamat yang digunakan untuk mengakses dan menemukan informasi di web [3].
3. Server Web: Komputer atau sistem yang menyimpan dan menyediakan halaman web serta sumber daya lainnya kepada pengguna melalui internet. Server web menjalankan perangkat lunak tertentu (biasanya seperti Apache, Nginx, atau Microsoft IIS) untuk mengirimkan konten web kepada pengguna yang meminta informasi [4].
4. *Hyperlink* (Tautan): Sebuah elemen pada halaman web yang menghubungkan ke halaman web atau sumber daya lainnya [5] Dengan mengklik *hyperlink,* pengguna dapat berpindah dari satu halaman web ke halaman web lainnya.

Permasalahan yang muncul terkait keamanan web adalah ketidakmampuan sistem aplikasi untuk melindungi informasi sensitif dari ancaman siber. Kerentanan seperti *SQL injection*, *Cross-Site Scripting (XSS*), dan *Cross-Site Request Forgery (CSRF)* menjadi celah yang sering dimanfaatkan oleh peretas untuk meretas atau mencuri informasi dari aplikasi web [6]

OWASP adalah singkatan dari "*Open Web Application Security Project*". Ini merupakan sebuah komunitas global yang berfokus pada meningkatkan keamanan perangkat lunak. Tujuan utama dari OWASP adalah untuk memberikan sumber daya, alat, pedoman, dan standar terkait keamanan aplikasi web yang dapat digunakan oleh para profesional keamanan, pengembang perangkat lunak, dan organisasi untuk memahami, mengidentifikasi, dan mengatasi kerentanan keamanan pada aplikasi web [7]

Top 10 OWASP adalah daftar yang diperbarui secara berkala yang memuat sepuluh kerentanan keamanan aplikasi web yang paling umum terjadi, seperti *SQL Injection, Cross-Site Scripting (XSS),* dan lainnya[8]. Daftar ini memberikan

panduan tentang apa yang harus diperhatikan dan diatasi dalam keamanan aplikasi web, seperti pada gambar 1 berikut



Gambar 1. OWASP Top 10 2021

Berikut adalah daftar OWASP Top 10 2021:

1. A01:2021 – Kerusakan Akses Kontrol
   Akses Kontrol menetapkan sebuah peraturan yang dimana *user* tidak dapat melakukan sebuah aksi diluar *permission* yang diberikan. Kegagalan atas hal ini dapat mengakibatkan pengeluaran informasi yang tidak diizinkan, modifikasi, atau penghancuran dari semua data atau pemberlakuan sebuah fungsi bisnis di luar limit sebuah user [9]
2. A02:2021 – Kegagalan Kriptografi
   Menentukan kebutuhan perlindungan data dalam perjalanan (*Data in Transport*) dan pada saat disimpan *(Data at Rest).* Misalnya, kata sandi, nomor kartu kredit, catatan kesehatan, informasi pribadi, dan rahasia bisnis yang memerlukan pelindungan, terutama jika data tersebut termasuk dalam undang-undang privasi [10]
3. A03:2021 – Injeksi
   Beberapa injeksi yang biasa terjadi adalah SQL, NoSQL, perintah OS, pemetaan relasi objek *(Object Relation Mapping)*, LDAP, dan bahasa ekspresi *(Expression Language)*. *Source Code Review* adalah metode terbaik dalam mendeteksi apakah aplikasi tersebut berisiko untuk diinjeksi. Testing otomatis terhadap semua parameter-parameter, headers, URL, cookies, JSON, SOAP, and input data XML sangat disarankan [11]
4. A04:2021 – *Insecure Design*
   Desain yang tidak aman adalah sebuah representasi kategori yang luas dari banyak kelemahan yang berbeda, yang diekspresikan sebagai "desain kontrol yang tidak ada atau kurang efisien.". Ada perbedaan antara desain tidak aman dan implementasi tidak aman [12]
5. A05:2021 – Kesalahan Konfigurasi Keamanan
   Aplikasi akan menjadi rentan jika tidak memiliki pertahanan yang sesuai atau security hardening yang diperlukan diseluruh bagian dari stack aplikasi atau tidak benar dalam melakukan konfigurasi untuk *permission* pada *cloud services*. Fitur - fitur yang tidak digunakan masih di-*enable* atau

diinstall (contoh seperti *port, services, pages, accounts,* atau *privileges* yang tidak dipakai) [13]

6. A06:2021 – Komponen yang Rentan dan Kedaluwarsa
   Semua versi komponen sistem yang kadaluwarsa, termasuk OS, server web/aplikasi, sistem manajemen basis data (DBMS), aplikasi, API dan semua komponen, lingkungan *runtime*, dan Library, serta komponen yang tidak di *patch* [14]
7. A07:2021 – Kegagalan Identifikasi dan Otentikasi
   Terjadi kelemahan otentikasi dan *session management* sangat penting untuk melindungi dari serangan terkait otentikasi[15]
8. A08:2021 - Kegagalan Integritas Data dan Perangkat Lunak
   Kegagalan menjaga integritas data dan perangkat lunak disebabkan oleh kode dan infrastruktur yang tidak mencegah terjadinya pelanggaran integritas. Contohnya sebuah objek/data yang telah di enkoding/diserialisasi didalam struktur yang dapat dilihat dan dimodifikasi oleh penyerang rentan terhadap deserialisasi yang tidak aman [16]
9. A09:2021 – Kegagalan dalam Keamanan Logging dan Monitoring
   Tanpa mencatat dan memonitor, penjebolan tidak dapat dideteksi. Ketidakcukup melakukan log, deteksi, memonitor dan respon aktif terjadi setiap waktu [17]
10. A10:2021 – *Server-Side Request Forgery (SSRF)*
    Kecacatan SSRF muncul saat sebuah aplikasi web meminta *remote resource* tanpa melakukan validasi URL yang diberikan oleh pengguna. Ini memperbolehkan penyerang untuk memaksa aplikasi untuk mengirim *crafted request* ke destinasi yang tidak diharapkan, meskipun sudah dilindungi oleh firewall, VPN, atau tipe lain dari daftar aturan akses jaringan - *Access Control List (ACL)*[18]

Melalui upaya kolaboratifnya, OWASP bertujuan untuk meningkatkan pemahaman umum tentang keamanan aplikasi web, mempromosikan praktik terbaik dalam pengembangan perangkat lunak yang aman, dan menyediakan sumber daya yang dibutuhkan untuk membantu mencegah serangan serta melindungi aplikasi web dari kerentanan keamanan. Penelitian ini bertujuan untuk:

1. Menganalisis kerentanan keamanan yang umum terjadi pada aplikasi web.
2. Meninjau dan membandingkan teknik-teknik keamanan yang digunakan dalam melindungi aplikasi web dari serangan.
3. Mengusulkan strategi pencegahan dan pemulihan yang efektif untuk meningkatkan keamanan aplikasi web.

B. Metode Penelitian

Penelitian ini meliputi beberapa tahap yang bisa dilihat pada gambar 2

Gambar 2. Tahapan Penelitian

1. Penentuan Target *Host*

   Penentuan target *host* dalam konteks keamanan jaringan adalah proses memilih dan menetapkan perangkat atau sistem yang akan menjadi subjek dari pemindaian keamanan atau evaluasi keamanan lebih lanjut. Langkah ini penting dalam perencanaan dan pelaksanaan strategi keamanan, terutama saat menggunakan alat pemindaian kerentanan seperti Nessus atau alat keamanan jaringan lainnya.
2. Instalasi OWASP

   OWASP *(Open Web Application Security Project)* merupakan sebuah komunitas yang menyediakan berbagai alat, sumber daya, panduan, dan informasi terkait keamanan aplikasi web. OWASP memiliki beragam proyek open-source yang dapat digunakan untuk membantu meningkatkan keamanan aplikasi web. Namun, OWASP itu sendiri bukanlah sebuah perangkat lunak atau aplikasi yang diinstal seperti program pada umumnya. Sebaliknya, OWASP adalah sebuah komunitas yang menyediakan proyek-proyek terbuka dan sumber daya lainnya yang bisa diakses secara online. Salah satu proyek OWASP yang cukup terkenal adalah "OWASP ZAP" *(Zed Attack Proxy),* sebuah alat pengujian keamanan aplikasi web yang bersifat open-source dan sering digunakan untuk mengidentifikasi kerentanan keamanan pada aplikasi web [19]
3. Pemindaian Target

   Untuk pemindaian target menggunakan OWASP bisa memakai konfigurasi standard yang telah ditentukan.
4. Simpan Hasil

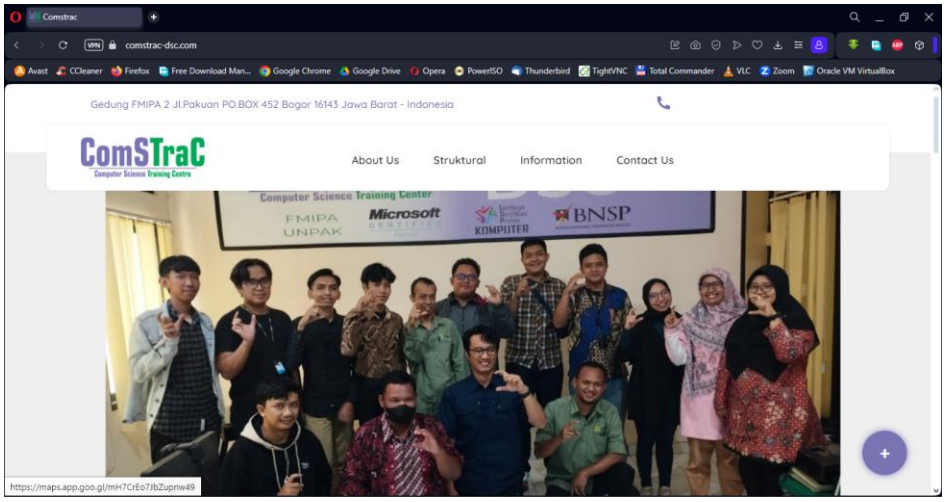
   Setelah pemindaian selesai, maka hasilnya bisa disimpan untuk dianalisa kerentanan-kerentanan yang terdapat pada host tersebut

untuk kemudian dilakukan perbaikan-perbaikan yang memenuhi standar keamanan

C. Hasil dan Pembahasan

1. Target Host

ComSTraC merupakan pusat pelatihan yang disediakan untuk sertifikasi nasional (bekerjasama dengan BNSP) dan internasional (bekerjasama dengan Microsoft). Pusat pelatihan ini diinisiasi untuk menjadi wadah peningkatan kualitas SDM, dan tidak hanya melayani sivitas akademika (mahasiswa, dosen dan karyawan) Universitas Pakuan tetapi juga terbuka untuk masyarakat luas yang berminat sertifikasi di bidang informatika. Sebagai media pertukaran informasi, ComSTraC membangun sebuah website yang diberi nama https://comstrac-dsc.com, yang bisa dilihat pada gambar 3 dan kemudian dijadikan sebagai target didalam penelitian ini.

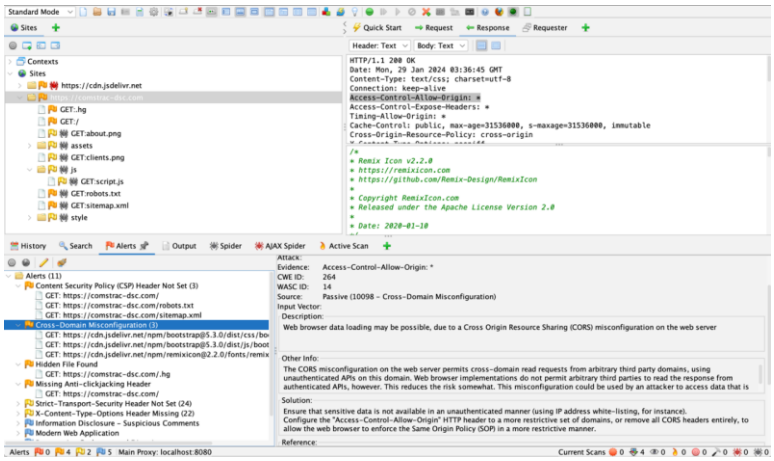


Gambar 3. Web Comstrac

2. Pemindaian Target

Profil Keamanan

Pemindaian target dilakukan seperti pada gambar 4, dengan memindai secara langsung url https://comstrac-dsc.com tanpa menggunakan *credential* karena saat ini wesite https://comstrac-dsc.com baru hanya menyediakan konten statis. Hasil dari pemindaian ini menemukan kerentanan sedang, dangkal hingga informatif [20].



Gambar 4. Pemindaian Target

Berikut adalah profil risiko yang diperoleh pada proses pemindaian target, dimana diketahui bahwa terdapat 9 celah keamanan dengan sebaran terlihat dalam Tabel 1.

Tabel 1. Profil Keamanan

	Risk			
	High	**Medium**	**Low**	**Informational**
https://comstrac-dsc.com	1	3	2	3

Level Risiko

Tabel risiko menunjukan peluang risiko yang terjadi untuk setiap celah keamanan yang ditemukan. Level risiko yang bersifat medium memiliki *confidence level* yang sama, yakni 11.1%, dengan total level risiko Medium adalah 33.3% yang dapat di lihat pada tabel 2.

Tabel 2. Level Risiko

		Confidence			
		High	**Medium**	**Low**	**Total**
Risk	**High**	0 (0.0%)	0 (0.0%)	1 (11.1%)	1 (11.1%)
	Medium	1 (11.1%)	1 (11.1%)	1 (11.1%)	3 (33.3%)
	Low	1 (11.1%)	1 (11.1%)	0 (0.0%)	2 (22.2%)
	Informational	0 (0.0%)	2 (22.2%)	1 (11.1%)	3 (33.3%)
	Total	2 (22.2%)	4 (44.4%)	3 (33.3%)	9 (100%)

Tabel 3 memperlihatkan jenis celah kemanan yang ditemukan berdasarkan framework OWASP Top 10:2021. Seluruh celah kemanan yang ditemukan hampir semuanya berkaitan dengan A05 Kesalahan Konfigurasi Keamanan & A06 Komponen yang Rentan dan Kedaluwarsa pada OWASP Top 10:2021.

Tabel 3. Jenis Celah Kemanan yang Ditemukan

Celah Keamanan	Risk	Count
Cloud Metadata Potentially Exposed	High	1 (11.1%)
Content Security Policy (CSP) Header Not Set	Medium	3 (33.3%)
Hidden File Found	Medium	4 (44.4%)
Missing Anti-clickjacking Header	Medium	1 (11.1%)
Strict-Transport-Security Header Not Set	Low	24 (266.7%)
X-Content-Type-Options Header Missing	Low	22 (244.4%)
Modern Web Application	Informational	1 (11.1%)
Re-examine Cache-control Directives	Informational	1 (11.1%)
User Agent Fuzzer	Informational	12 (133.3%)
Total		9

3. Analisis Kerentanan Target

3.1. *Cloud Metadata Potentially Exposed*

Serangan *Metadata Cloud* berupaya menyalahgunakan server NGINX yang tidak dikonfigurasi dengan benar untuk mengakses metadata instans yang dikelola oleh penyedia layanan cloud seperti AWS, GCP, dan Azure. Semua *provider* ini menyediakan metadata melalui alamat IP internal yang tidak dapat dirutekan '169.254.169.254' - ini dapat diekspos oleh server NGINX yang tidak dikonfigurasi dengan benar dan diakses dengan menggunakan alamat IP ini di bidang header Host.

OWASP Top 10:

A05 Kesalahan Konfigurasi Keamanan

A06 Komponen yang Rentan dan Kedaluwarsa

Bidang Serang: 169.254.169.254

Method: GET https://comstrac-dsc.com/latest/meta-data/

3.2. *Content Security Policy* (CSP) *Header Not Set*

Content Security Policy (CSP) adalah lapisan keamanan tambahan yang membantu mendeteksi dan memitigasi jenis serangan tertentu, *termasuk Cross Site Scripting* (XSS) dan serangan injeksi data. Serangan-serangan ini digunakan untuk segala hal mulai dari pencurian data hingga perusakan situs atau distribusi malware. CSP menyediakan serangkaian header HTTP standar yang memungkinkan pemilik situs web mendeklarasikan sumber konten yang disetujui dan boleh dimuat oleh browser di halaman tersebut — tipe yang tercakup adalah JavaScript, CSS, bingkai HTML, font, gambar, dan objek yang dapat disematkan seperti applet Java, ActiveX, file audio dan video.

OWASP Top 10:

A05 Kesalahan Konfigurasi Keamanan

A06 Komponen yang Rentan dan Kedaluwarsa

Bidang Serang: -

Method: GET https://comstrac-dsc.com/sitemap.xml

3.3.Hidden File Found

File sensitif diidentifikasi sebagai dapat diakses atau tersedia. Hal ini dapat membocorkan informasi administratif, konfigurasi, atau kredensial yang dapat dimanfaatkan oleh individu jahat untuk menyerang sistem lebih lanjut atau melakukan upaya Social Engineering.

OWASP Top 10:

A05 Kesalahan Konfigurasi Keamanan

A06 Komponen yang Rentan dan Kedaluwarsa

Bidang Serang: -

Method: GET https://comstrac-dsc.com/.hg

3.4.Missing Anti-clickjacking Header

Responsnya tidak mencakup Content-Security-Policy dengan arahan 'frame-ancestors' atau X-Frame-Options untuk melindungi dari serangan 'ClickJacking'.

OWASP Top 10:

A05 Kesalahan Konfigurasi Keamanan

A06 Komponen yang Rentan dan Kedaluwarsa

Bidang Serang: -

Method: GET https://comstrac-dsc.com/

Parameter : `x-frame-options`

3.5.Strict-Transport-Security Header Not Set

HTTP Strict Transport Security (HSTS) adalah mekanisme kebijakan keamanan web dimana Web Server menyatakan bahwa agen pengguna yang mematuhi (seperti browser web) harus berinteraksi dengannya hanya menggunakan koneksi HTTPS yang aman (yaitu HTTP berlapis TLS/SSL). HSTS adalah protokol jalur standar IETF dan ditentukan dalam RFC 6797.

OWASP Top 10:

A05 Kesalahan Konfigurasi Keamanan

A06 Komponen yang Rentan dan Kedaluwarsa

Bidang Serang: -

Method: GET https://comstrac-dsc.com/sitemap.xml

Parameter : `x-frame-options`

3.6.X-Content-Type-Options Header Missing

Header Anti-MIME-Sniffing X-Content-Type-Options tidak diseting ke 'nosniff'. Hal ini memungkinkan versi Internet Explorer dan Chrome yang lebih lama untuk melakukan MIME-sniffing pada isi respons, yang berpotensi menyebabkan isi respons ditafsirkan dan ditampilkan sebagai tipe konten selain tipe konten yang dinyatakan. Versi Firefox saat ini (awal 2014) dan lama akan menggunakan tipe konten yang dideklarasikan (jika ada yang disetel), daripada melakukan sniffing MIME.

Masalah ini masih berlaku untuk halaman tipe kesalahan (401, 403, 500, dan lain-lain) karena halaman tersebut sering kali masih terpengaruh oleh masalah injeksi, dalam hal ini masih ada kekhawatiran browser akan mengendus halaman dari tipe konten sebenarnya.

OWASP Top 10:

A05 Kesalahan Konfigurasi Keamanan

A06 Komponen yang Rentan dan Kedaluwarsa

Bidang Serang: -

Method: GET https://comstrac-dsc.com/style/floating-button.css

Parameter : `x-content-type-options`

3.7.Modern Web Application

Aplikasi tersebut tampaknya merupakan aplikasi web modern. Jika Anda perlu menjelajahinya secara otomatis maka *Ajax Spider* mungkin lebih efektif daripada yang standar. Ditemukan tautan yang tidak memiliki atribut href tradisional, yang merupakan indikasi bahwa ini adalah aplikasi web modern.

OWASP Top 10:

A05 Kesalahan Konfigurasi Keamanan

A06 Komponen yang Rentan dan Kedaluwarsa

Bidang Serang: -

Method: GET https://comstrac-dsc.com/

Parameter :

```
<a href=""><i class="ri-phone-fill"></i></a>
```

3.8.Re-examine Cache-control Directives

Header kontrol cache belum diseting dengan benar atau hilang, sehingga memungkinkan browser dan proksi menyimpan konten dalam *cache*. Untuk aset statis seperti css, js, atau file gambar, hal ini mungkin dimaksudkan, namun sumber daya harus ditinjau untuk memastikan bahwa tidak ada konten sensitif yang akan di-cache.

OWASP Top 10:

A05 Kesalahan Konfigurasi Keamanan

A06 Komponen yang Rentan dan Kedaluwarsa
Bidang Serang: -
Method: GET https://comstrac-dsc.com/
Parameter : `cache-control`

3.9. User Agent Fuzzer

Periksa perbedaan respons berdasarkan Agen Pengguna yang tidak jelas (misalnya situs seluler, akses sebagai Perayap Mesin Telusur). Membandingkan kode status respons dan kode hash isi respons dengan respons asli.
Bidang Serang: -
Method: GET https://comstrac-dsc.com/
Parameter : `Header User-Agent`

4. Rekomendasi Remediasi Kerentanan Target

4.1. Cloud Metadata Potentially Exposed

Jangan percaya data pengguna apa pun di konfigurasi NGINX. Dalam hal ini mungkin penggunaan variabel $host yang diatur dari header 'Host' dan dapat dikontrol oleh penyerang.

4.2. Content Security Policy (CSP) Header Not Set

Pastikan server web, server aplikasi, Load Balancer, dan lain telah dikonfigurasi untuk *setting* header CSP.

4.3. Hidden File Found

Pertimbangkan apakah komponen tersebut benar-benar diperlukan dalam produksi atau tidak, jika tidak maka nonaktifkan. Jika ya, pastikan akses ke sana memerlukan autentikasi dan otorisasi yang sesuai, atau batasi paparan ke sistem internal atau IP sumber tertentu.

4.4. Missing Anti-clickjacking Header

Browser Web modern mendukung *header HTTP Content-Security-Policy* dan *X-Frame-Options*. Pastikan salah satunya disetel di semua halaman web yang ditampilkan oleh situs/aplikasi. Jika halaman dibingkai hanya oleh halaman di server Anda (misalnya, itu bagian dari FRAMESET) maka sebaiknya menggunakan SAMEORIGIN, sebaliknya jika tidak pernah mengharapkan halaman dibingkai, harus menggunakan DENY. Alternatifnya, pertimbangkan untuk menerapkan arahan "frame-ancestor" CSP.

4.5. Strict-Transport-Security Header Not Set

Pastikan server web, server aplikasi, penyeimbang beban dan lain-lain. Anda dikonfigurasi untuk menerapkan CSP.

4.6. X-Content-Type-Options Header Missing

Pastikan aplikasi/server web menyetel header Tipe Konten dengan tepat, dan setting header X-Content-Type-Options ke 'nosniff' untuk semua halaman web. Jika memungkinkan, pastikan bahwa pengguna akhir menggunakan browser web yang sesuai standar dan modern yang tidak melakukan sniffing MIME sama sekali, atau yang dapat diarahkan oleh aplikasi web/server web untuk tidak melakukan sniffing MIME.

4.7. Modern Web Application

Tidak diperlukan rekomendasi.

4.8. Re-examine Cache-control Directives

Untuk konten yang aman, pastikan header HTTP kontrol cache disetel dengan "no-cache, no-store, must-revalidate". Jika suatu aset harus di-cache, pertimbangkan untuk menyetel arahan "public, max-age, immutable".

4.9. User Agent Fuzzer

Tidak diperlukan rekomendasi.

D. Kesimpulan

Berdasarkan pada hasil pemindaian dan analisis kerentanan terhadap celah keamanan pada aplikasi target https://comstrac-dsc.com, dapat disimpulkan aplikasi tersebut saat ini dibangun hanya dengan melibatkan konten statis, sehingga belum banyak variasi uji keamanan yang bisa diterapkan untuk versi aplikasi saat ini.

Profil kemanan dan risiko atas aplikasi tersebut berada pada level *low to moderate*, dimana komponen risiko Medium mendominasi dengan Confidence Level sebesar 33.3%, yang berarti kerentanan didominasi oleh kerentanan berkategori medium. Ditemukan sebanyak 9 (sembilan) temuan celah keamanan, dengan 1 (satu) temuan berkategori Risiko Tinggi (High), 3 (tiga) temuan berkategori Risiko Medium, 2 (dua) temuan berkategori Risiko Rendah (Low) dan 3 (tiga) temuan bersifat informasi (Informational).

Secara umum, seluruh celah keaman tersebut masuk kedalam kategori OWASP TOP 10:2021, antara lain A05 Kesalahan Konfigurasi Keamanan dan A06 Komponen yang Rentan dan Kedaluwarsa. Kedua jenis kategori tersebut berkaitan dengan konfigurasi sistem yang belum sempurna dan terdapat penggunaan komponen penyusun aplikasi yang sudah kadaluwarsa.

Rekomendasi perbaikan terhadap temuan sudah diberikan, dan diantaranya bersifat perbaikan didalam *source code* dan konfigurasi pada *application server/web server* yang digunakan pada aplikasi tersebut. Perbaikan diprioritaskan kepada temuan yang bersifat High, Medium dan Low, terhadap temuan celah keamanan yang bersifat Informational, tidak menjadi keharusan untuk dilakukan perbaikan.

Disarankan untuk dilakukan pengembangan lebih lanjut terhadap aplikasi target https://comstrac-dsc.com, sehingga aplikasi ini menjadi lebih interaktif dan informatif sehingga dapat tujuan utama pengembangan aplikasi dapat tercapai dengan tanpa mengabaikan unsur keamanan didalam pengembangannya.

E. Referensi

[1] Rahmat Hidayat, “Pengertian World Wide Web,” *Pemrograman web dengan html*, pp. 6–20, 2017.

[2] W. F. Kusuma, “Pengembangan Halaman Web Menggunakan XML dalam Perkembangan WEB 2.0,” *Jurnal Teknik Informatika*, vol. 6, no. 2, 2015.

[3] I. Indra, “Apa itu URL? Pengertian URL Beserta Fungsinya,” Niagahoster Blog.

[4] T. Suhesti, "Web Server dan Jenisnya," *Ilmuti.Org*, 2014.
[5] N. C. Agustin and I. Krismayani, "Kemampuan Literasi Digital Mahasiswa S1 Angkatan 2018 Fakultas Ilmu Budaya Universitas Diponegoro," *Jurnal Ilmu Perpustakaan*, vol. 8, no. 3, 2019.
[6] I. O. Riandhanu, "Analisis Metode Open Web Application Security Project (OWASP) Menggunakan Penetration Testing pada Keamanan Website Absensi," *Jurnal Informasi dan Teknologi*, 2022, doi: 10.37034/jidt.v4i3.236.
[7] OWASP, "OWASP Top 10:2021," 2021.
[8] Open Web Application Security Project (OWASP), "OWASP Top Ten Web Application Security 2021," Owasp.
[9] OWASP, "A01:2021 – Broken Access Control," https://owasp.org/Top10/A01_2021-Broken_Access_Control/.
[10] OWASP, "A02:2021 – Cryptographic Failures," https://owasp.org/Top10/A02_2021-Cryptographic_Failures/.
[11] OWASP, "A03:2021 – Injection," https://owasp.org/Top10/A03_2021-Injection/.
[12] OWASP, "A04:2021 – Insecure Design," *https://owasp.org/Top10/A04_2021-Insecure_Design/*, 2021.
[13] OWASP, "A05:2021 – Security Misconfiguration," https://owasp.org/Top10/A05_2021-Security_Misconfiguration/.
[14] OWASP, "A06:2021 – Vulnerable and Outdated Components," https://owasp.org/Top10/A06_2021-Vulnerable_and_Outdated_Components/.
[15] OWASP, "A07:2021 – Identification and Authentication Failures," https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures/.
[16] OWASP, "A08:2021 – Software and Data Integrity Failures," https://owasp.org/Top10/A08_2021-Software_and_Data_Integrity_Failures/.
[17] OWASP, "A09:2021 – Security Logging and Monitoring Failures," https://owasp.org/Top10/A09_2021-Security_Logging_and_Monitoring_Failures/.
[18] OWASP, "A10:2021 – Server-Side Request Forgery (SSRF)," https://owasp.org/Top10/A10_2021-Server-Side_Request_Forgery_%28SSRF%29/.
[19] G. Kusuma, "IMPLEMENTASI OWASP ZAP UNTUK PENGUJIAN KEAMANAN SISTEM INFORMASI AKADEMIK," *Jurnal Teknologi Informasi: Jurnal Keilmuan dan Aplikasi Bidang Teknik Informatika*, vol. 16, no. 2, 2022, doi: 10.47111/jti.v16i2.3995.
[20] T. Ariyadi, T. L. Widodo, N. Apriyanti, and F. S. Kirana, "Analisis Kerentanan Keamanan Sistem Informasi Akademik Universitas Bina Darma Menggunakan OWASP," *Techno.Com*, vol. 22, no. 2, 2023, doi: 10.33633/tc.v22i2.7562.