UJI KERENTANAN PADA SISTEM *PROCTORING* UJIANBERBASIS *LEARNING MANAGEMENT SYSTEM*

Rakha Rizqllah Pratama Saputra
Fakultas Teknik Elektro Telkom University
Bandung, Indonesia
khatama@student.telkomuniversity.ac.id

Dr. Yudha Purwanto , S.T, M.T
Fakultas Teknik Elektro Telkom University
Bandung, Indonesia
omyudha@student.telkomuniversity.ac.id

Muhammad Faris Ruriawan, S.T, M.T
Fakultas Teknik Elektro Telkom University
Bandung, Indonesia
muhammadfaris@student.telkomuniversity. ac.id

Pada Universitas X, sedang dikembangkan sistem pengawasan ujian yang biasa disebut proctoring. Proctoring adalah sebuah sistem pengawasan online yang dilakukan dengan cara merekam aktivitas yang dilakukan oleh peserta ujian, baik layar komputer yang digunakan maupun wajah peserta melalui webcam. Dalam pengembangan sistem proctoring dari LMS, dibutuhkan sebuah proses yang disebut VulnTest (Vulnerability Testing).

VulnTest (Vulnerability Testing) adalah proses untuk mengidentifikasi, mengevaluasi, dan mengklasifikasikan tingkat keparahan pada celah keamanan yang ada pada sebuah jaringan komputer, sistem, aplikasi, atau bagian lain yang ada di ekosistem IT berdasarkan risiko yang dapat ditimbulkan. Vulntest dibutuhkan untuk menguji sistem proctoring dengan mencari celah keamanan yang berpotensi sebagai kecurangan pada pelaksanaan ujian. Hasil yang didapatkan dari pengujian menunjukan celah keamanan dapat ditemukan dengan menggunakan OBS, Burpsuite, dan local storage browser menyebabkan peserta dapat mengelabui sistem pengawasan pada saat sebelum ujian dimulai dan saat ujian berlangsung.

Kata Kunci: Vulntest, LMS, Proctoring.

I. PENDAHULUAN

Learning management system (LMS) adalah aplikasi perangkat lunak yang dirancang untuk membuat, mendistribusikan, dan mengatur penyampaian materi pembelajaran dalam jaringan. [1]Penggunaan *Learning management system* (LMS) dalam melakukan proses pengajaran di universitas-universitas telah menyebar luas, tetapi penggunaan LMS secara rutin untuk menjadi sebuah budaya masih memiliki tantangannya sendiri. Pada universitas X penggunaan sudah secara rutin digunakan dan universitas x juga sedang mengembangkan sistem pengawasan ujian.

Sistem pengawasan ujian yang sedang dikembang harus melalui proses pengujian sebelum digunakan oleh mahasiswa.[2]Sistem informasi dan komunikasi memiliki beberapa kerentanan keamanan. Selain itu, perangkat lunak keamanan konvensional memerlukan upaya penyetelan dan mungkin tidak dapat mendeteksi banyak serangan web. Untuk alasan ini, keamanan menjadi tujuan global untuk banyak sistem teknologi termasuk *Learning Management Systems* (LMS) seperti MoodIe. *Vulnerability testing* (vulntest) dapat menjadi salah satu cara untuk menekan celah-celah keamanan yang dimiliki sebuah LMS.

Maka dari itu, demi terjaganya keamanan dan mencegah kecurangan yang terjadi sebuah *Learning Management Systems* (LMS) diperlukan *vulnerability testing* dalam pengembangan sebuah LMS. Contohnya jika sebuah LMS memiliki fitur pengawasan ujian atau yang biasa disebut *proctoring* seperti yang dimiliki oleh Universitas X. Vulntest harus dilakukan untuk mengurangi kemungkinan-kemungkinan kecurangan yang bisa dilakukan selama melakukan ujian. Dengan beberapa skenario-skenario pengujian berupa dengan mencari kelemahan dari *image detection* yang dimiliki oleh *proctoring* yang digunakan. Tidak hanya itu pengujian juga akan berfokus pada parameter-parameter yang nantinya akan menjadi celah keamanan.

II. KAJIAN TEORI

A. Website

Website atau situs merupakan kumpulan halaman-halaman yang dimiliki sebuah domain. [3]*Website* utamanya digunakan oleh organisasi untuk berbagi informasi. *Website* telah menjadi sebuah alat komunikasi utama yang sukses bergantung pada aksesibilitasnya, SEO (*Search Engine Optimization)* dan kegunaanya. Yang menjadi faktor penting kesuksesan dari 3 faktor tersebut adalah kegunaannya, yang berarti kesuksesan dan kegagalan dari *website* apapun tergantung pada kegunaannya. Oleh karena itu, kegunaan adalah pertimbangan sebagai dasar dan fitur penting untuk kesuksesan dari sebuah website.

B. Information Assurance (IA) Principle

Prinsip jaminan informasi atau [4]*information assurance* (IA) *Principle* bertindak sebagai pendukung sebuah aktivitas organisasi *security* untuk melindungi dan mempertahankan jaringan mereka dari serangan keamanan. IA memfasilitasi penanggulangan dan tindakan respons pada peringatan atau deteksi ancaman. Karena itu operator jaringan harus menggunakan prinsip IA untuk mengidentifikasi data yang sensitif, dan untuk menangkal kejadian yang memungkinkan implikasi keamanan pada jaringan. Prinsip IA membantu dalam mengidentifikasi celah keamanan pada jaringan, memantau jaringan untuk setiap percobaan penyusupan dan kegiatan yang mencurigakan, dan mempertahankan jaringan dengan melakukan mitigasi celah keamanan.

Aktivitas mempertahankan jaringan harus mengikuti prinsip IA untuk mencapai defense-in-depth keamanan jaringan :
1. *Confidentiality*: izin confidentiality (kerahasiaan) hanya untuk pengguna yang memiliki akses, menggunakan atau menyalin informasi. Autentikasi adalah hal yang krusial dalam kerahasiaan. Jika pengguna yang tidak memiliki terotorisasi mengakses informasi yang dilindungi, hal ini termasuk pembobolan kerahasiaan telah terjadi

2. *Integrity*: *Integrity* (keutuhan) melindungi data dan tidak terjadinya modifikasi, penghapusan atau korupsi data tanpa adanya otorisasi yang tepat. Prinsip penjaminan informasi juga bergantung pada fungsi yang layak pada otentikasi.
3. *Availability*: *Availability* (ketersediaan) adalah proses dalam melindungi sistem informasi atau jaringan yang menyimpan data sensitive, untuk membuatnya tersedia untuk *end user* kapanpun ada permintaan akses.

4. *Non-repudiation*: adalah sebuah layanan yang memvalidasi sebuah keutuhan transmisi dari *digital signature*, dimulai dari asal hingga kemana sama tujuannya. *Non-repudiation* menjamin akses yang melindungi informasi dengan memvalidasi *digital signature* dari pihak yg bersangkutan.

5. *Authentication*: adalah proses mengotorisasi pengguna dengan kredensial yang tersedia, dengan membandingkannya dengan yang ada di *database* dari pengguna yang terotorisasi dalam *authentication server*, untuk mendapatkan akses ke jaringan. Hal tersebut menjamin *file* dan data yang melintasi jaringan aman.

C. *Learning Management System* (LMS)

Berdasarkan penelitian dari Meyliana, Henry Antonius Eka Widjaja, dan kawan-kawan[1].LMS adalah sebuah kombinasi dari fasilitas pedagogis, interaksi manusia, konten pembelajaran dan dukungan evaluasi untuk meningkatkan aktivitas pengajaran dan pembelajaran di sekolah atau universitas. LMS harus mampu bertemu dengan kebutuhan pengguna, khususnya di pendistribusian konten pembelajaran.

D. *Vulnerability Assessment*

Menurut Halit Alptekin, Simge Demir, dan kawan-kawan .[5]*Vulnerability assessment* atau penilaian kerentanan adalah proses mengidentifikasi dan memprioritaskan kerentanan pada sebuah sistem. *Vulnerability scanners* dapat digunakan, sebagai contoh, mendeteksi celah keamanan untuk sebuah website dengan menjalankan *repository* dari deteksi uji keamanan, setiap pengujian didesain untuk mengeksekusi sebuah *vulnerability*.

E. NIST 800-115 *Penetration Testing*

Berdasarkan dokumen NIST 800-115. [6]P*enetration testing* adalah uji keamanan yang pada pengujiannya penguji meniru serangan aslinya untuk mengidentifikasi metode-metode untuk menghindari fitur-fitur keamanan pada sebuah aplikasi, sistem, atau jaringan. Hal ini sering melibatkan peluncuran serangan nyata pada sistem nyata dan data yang menggunakan alat dan Teknik yang biasa dilakukan oleh penyerang. Kebanyakan *Penetration test* melibatkan pada pencarian untuk kombinasi-kombinasi pada kerentanan pada satu atau lebih sistem yang dapat mendapatkan akses lebih dibandingkan yang seharusnya dapat ditemukan dari sebuah kerentanan. *Penetration testing* dapat juga berguna untuk menentukan:

1. Seberapa baik sistem mentoleransi pola serangan nyata .
2. Kemungkinan kecanggihan yang dibutuhkan seorang penyerang dalam mendapatkan sebuah sistem.
3. Tolak Ukur tambahan yang dapat memitigasi ancaman terhadap sistem
4. Kemampuan pertahanan untuk mendeteksi serangan dan merespon dengan tepat

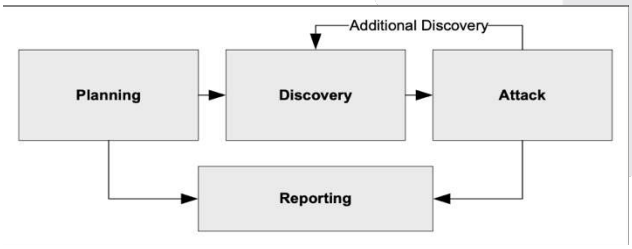
Penetration testing dapat menjadi tidak berharga, tetapi hal ini membutuhkan tenaga kerja dan membutuhkan para ahli untuk meminimalisir risiko untuk sistem yang menjadi target. Sistem memungkin terdampak atau sebaliknya tidak dapat dioperasikan selama *penetration testing*, meskipun perusahaan diuntungkan dengan mengetahui bagaimana

sistem dapat tidak dikendalikan oleh penyusup. Walaupun *penetration tester* berpengalaman dapat memitigasi risiko ini, hal ini tidak dapat dihilangkan sepenuhnya. *Penetration testing* harus dilakukan khusus setelah pertimbangan, pemberitahuan, dan perencanaan cermat.

Penetration testing sering memasukkan metode-metode non-teknikal pada serangan. Sebagai contoh, seorang *penetration tester* dapat membobol kendali keamanan fisikal dan prosedur untuk terhubung ke sebuah jaringan, mencuri perangkat, mendapatkan informasi sensitive (memungkinkan dengan memasang perangkat *keylogging*), atau mengganggu komunikasi. Kewaspadaan harus dilatih saat melakukan uji keamanan fisik—penjaga keamanan harus dibuat sadar tentang bagaimana untuk memverifikasi validitas dari kegiatan penguji, seperti melalui sebuah titik pertemuan atau dokumentasi. Cara non-teknis lainnya berarti serangan adalah dengan menggunakan *social engineering*, seperti berpura-pura menjadi pegawai *helpdesk* dan meminta untuk permohonan sebuah kata sandi pengguna, atau meminta pegawai *help desk* sebagai pengguna dan meminta untuk sbeuha kata sandi untuk diatur ulang. Informasi tambahan pada uji keamanan fisik, teknik *social engineering*, dan cara serangan non-teknis lainnya yang termasuk dalam *penetration testing* berada di luar cakupan publikasi ini[].

a. *Penetration testing phases.*

Pada NIST 800-115 memiliki *framework* yang dapat bisa digunakan oleh *pentester* untuk melakukan penyerangan ke sebuah sistem. *Framework* NIST 800- 115 memiliki empat fase penyerangan. fase pertama adalah *planning* atau perencanaan, dilanjutkan dengan fase *discovery* atau penemuan, lalu fase *attack* atau penyerangan, dan terakhir fase *reporting* atau pelaporan. Pada gambar 2.1 ditunjukan alur yang dimiliki oleh *framework* NIST 800-115.
[6]Fase *planning* menetapkan dasar untuk suksesnya sebuah *penetration test*. Pada fase tidak ada pengujian yang dilakukan. Fase *discovery* dari *penetration testing* memiliki 2 bagian. Bagian pertama dimulai dengan testing sesungguhnya,



dan mencakup pengumpulan informasi dan pendeteksian. Port jaringan dan identifikasi layanan dilakukan untuk mengidentifikasi target yang berpotensi.

Pada bagian kedua dari fase *discovery* adalah analisis kerentanan, yang melibatkan perbandingan layanan-layanan, aplikasi-aplikasi, dan sistem operasi dari *host* yang dideteksi terhadap kerentanan *database* dan pengetahuan tentang kerentanan dari penguji itu sendiri. Mengeksekusi sebuah serangan tepat pada jantung dari *penetration test* apapun. Proses dari verifikasi sebelumnya teridentifikasi kerentanan dengan melakukan eksploitasi. Jika sebuah serangan sukses dilakukan, kerentanan yang terverifikasi dan *safeguard* teridentifikasi untuk memitigasi paparan keamanan yang ada.

Fase *Reporting* dilakukan bersamaan dengan tiga fase penetrasi lainnya. Pada fase *planning* rencana penilaian atau ROE dikembangkan. Dalam fase *discovery* dan *attacking*, log tertulis biasanya disimpan dan laporan berkala dibuat untuk administrator dan/atau manajemen sistem. Di akhir pengujian, sebuah laporan biasanya dikembangkan untuk menggambarkan kerentanan yang teridentifikasi, menampilkan peringkat risiko, dan memberikan panduan tentang cara memitigasi kelemahan yang ditemukan.

b. Jenis *Penetration Testing* untuk web aplikasi

Dalam melakukan *penetration testing* para *pentester* dibagi menjadi beberapa jenis berdasarkan petunjuk yang didapatkan. jenis *penetration testing* ada 3 jenis ,yaitu *black box*, *grey box*, dan *whitebox*. Berikut penjelasan jenis *penetration testing* berdasarkan Narayanan Anantharaman dan Dr. Bharati Wukkadada.[7]

1. *Black Box*

Pada pengujian *Black box* penguji tidak memiliki petunjuk tentang lingkup aplikasi. hal tersebut adalah tanggung jawab dan kemampuan dari penguji untuk berpikir seperti seorang peretas dan mencoba mengeksploitasi aplikasi. Biasanya pengujian ini memakan waktu lebih lama dimana pengujian akan mencoba serangan dengan berbagai teknik yang akan melibatkan perangkat terotomatisasi seperti burp, metasploit, wireshark, etc.

2. *Grey Box*

Pada pengujian *grey box* penguji akan mendapatkan beberapa petunjuk tentang aplikasi yakni sebuah informasi minim yang tersedia. sebagai contoh pengujian akan mendapatkan akses ke satu modul untuk melakukan pengujian. Pengujian ini memungkinkan waktu yang lebih sedikit dibandingkan *black box* ,tetapi lebih banyak waktu dibandingkan *white box* sebagai penguji tetap harus menjelajah aplikasi dan mengerti alur dan fungsionalitas nya sendiri dan mengeksploitasinya. sebagai penguji harus memperhatikan beberapa pengetahuan dari aplikasi yang bersangkutan, cara terbaik untuk melakukan pengujian adalah dengan melakukan sebuah kombinasi dari kedua pengujian terotomatisasi dan manual. Sebagai contoh, menebak kata sandi dapat dilakukan secara manual atau dapat dilakukan dengan menggunakan pengujian terotomatisasi.

3.*WhiteBox*
Pada pengujian *white box* penguji akan diberikan akses penuh terhadap web aplikasi yang diuji. Sebelum pengujian dari aplikasi penguji akan diberikan panduan tentang aplikasi dari pemilik aplikasi. Pengujian berlangsung dengan waktu yang lebih sedikit dibandingkan pengujian *black* dan *grey box* dan

kualitas dari pengujian akan lebih mendalam karena penguji telah mengenal tentang aplikasinya.

F. OWASP Top 10 2021 (Open Web Application security Project Top 10)

OWASP adalah yayasan yang bergerak dibidang keamanan sebuah perangkat lunak. OWASP memiliki sebuah dokumen yang selalu diperbaharui setiap tahunnya, yaitu OWASP Top 10. [8]OWASP Top 10 adalah sebuah dokumen standar pengetahuan untuk para pengembang dan *web application security*. Pada gambar 2.2 ditunjukan perbedaan *security incident* antara 2017 dengan 2021.

OWASP Top 10 2021

No *Web application security risk*

1. A01 Broken Access Control
2. A02 Cryptographic Failures
3. A03 Injection
4. A04 Insecure Design
5. A05 Security Misconfiguration
6. A06 Vulnerable and Outdated Components
7. A07 Identification and Authentication Failures
8. A08 Software and Data Integrity Failures
9. A09 Security Logging and Monitoring Failures
10. A10 Server-side Request Forgery (SSRF)

2.7 Common Vulnerability Scoring System (CVSS)

pada umum nya setelah melakukan pengujian kerentanan seorang penguji akan menentukan kerentanan yang didapatkan dengan kalkulator CVSS. [9]*Common Vulnerability Scoring System* (CVSS) adalah metode yang digunakan menyediakan ukuran kualitatif dari tingkat keparahan. CVSS bukan ukuran dari risiko. CVSS terdiri dari tiga kelompok metrik: basis, temporal, dan lingkungan. Metrik basis menghasilkan skor mulai dari 0 hingga 10, yang kemudian dapat dimodifikasi dengan menskor metrik temporal dan lingkungan. Skor CVSS juga direpresentasikan sebagai string vektor, representasi tekstual terkompresi dari nilai yang digunakan untuk mendapatkan skor. dengan demikian, CVSS sangat cocok sebagai sistem pengukuran standar untuk industri, organisasi, dan pemerintah yang membutuhkan skor keparahan kerentanan yang akurat dan konsisten. Dua penggunaan umum CVSS adalah menghitung keparahan kerentanan yang ditemukan pada sistem seseorang dan sebagai faktor dalam memprioritaskan

aktivitas perbaikan kerentanan. *National Vulnerability Database* (NVD) menyediakan CVSS skor untuk hampir semua kerentanan yang diketahui.

2.7.1 Tingkat Keparahan Kerentanan NVD

Setelah melakukan perhitungan menggunakan kalkulator CVSS akan didapatkan beberapa angka yang dapat diartikan CVSS *Rating*.[9] NVD menyediakan peringkat keparahan kualitatif dari “low”, “medium”, dan “High” untuk CVSS v2.0 *Ratings* rentang skor dasar selain peringkat keparahan untuk CVSS v3.0 *Ratings* seperti yang ditentukan dalam spesifikasi CVSS v3.0. Pada gambar 2.3 adalah penilaian yang dimiliki oleh CVSS v2.0 dan CVSS 3.0.

2.8 Face Detection

Menurut Asep Hadian Sudrajat Ganidisastra dan Yoanes bandung.[10]Pada umumnya pengenalan wajah melakukan 3 langkah proses utama, yaitu: deteksi wajah, ekstraksi fitur, dan pengenal wajah. Banyak metode telah diusulkan di setiap langkah untuk meningkatkan akurasi. Pada deteksi wajah, beberapa metode telah diusulkan yang biasa digunakan adalah viola-jones/haar cascade method, *Local Binary Pattern* (LBP) *method*. *Multi-Task Cascaded* CNN (MTCNN), dan terakhir adalah YOLO-face.

III. Skenario pengujian

Skenario pengujian yang akan dilakukan pada sistem pengawasan ujian pada LMS Universitas X akan berdasarkan pada NIST 800-115. NIST 800-115 memiliki fase-fase yang dapat dijadikan panduan dalam melakukan *penetration testing*. Fase-fase tersebut adalah *Planning,Discovery,attacking,* dan *Reporting.* Berikut skenario pengujian yang dilakukan pada setiap fase:

1. *Planning* (Perencanaan) :Pada fase ini akan ditentukan target-target dari pengujian celah keamanan apa yang akan diuji nantinya

2. *Discovery* (Penemuan): Pada fase ini akan dilakukan pencarian informasi pada sistem pengawasan ujian yang nantinya dapat dieksploitasi pada fase serangan

3. *Attacking* (serangan): Dalam Serangan yang dilakukan akan memanfaatkan celah-celah keamanan yang ditemukan dari fase *discovery*. Serangan ini akan menggunakan software yang biasa digunakan saat pengetesan sebuah web, yaitu burpsuite dan untuk melakukan pengujian secara non-teknis menggunakan OBS (*Open Broadcaster Software*) yang dapat digunakan untuk orang awam untuk mengelabui sistem pengawasan saat melakukan ujian.

4. *Reporting*(pelaporan): setiap fase-fase yang akan dilakukan akan dijadikan laporan nantinya. Mulai dari perencanaan yang akan dilakukan selama pengujian, Penemuan celah-celah keamanan apa saja yang dapat dimanfaatkan oleh seorang penyerang, lalu serangan apa saja yang dapat dilakukan selama pengujian berlangsung.

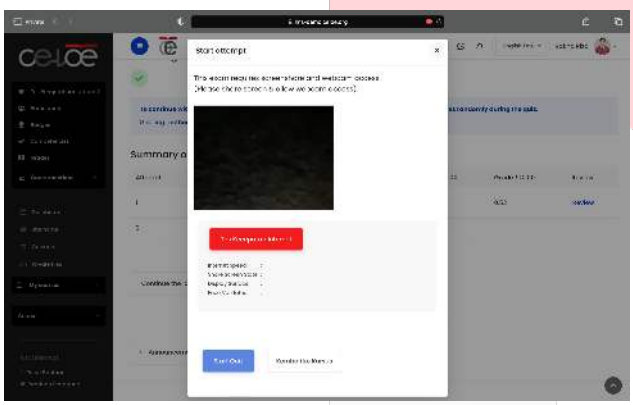
IV. HASIL DAN ANALISIS

untuk *vulnerability testing* atau uji kerentanan menggunakan *framework* NIST 800-115 dengan 4 fase yang dimiliki, yaitu *planning, discovery, attack,* dan *reporting*.

Tools yang akan digunakan pada pengujian ini adalah OBS untuk melakukan manipulasi pada input kamera yang digunakan, Burpsuite untuk melakukan perubahan pada *packet* yang akan dikirimkan ke sistem, dan *inspect element*

A. Planning

Pada fase *planning* atau perencanaan adalah penentuan target dan tujuan yang ingin dicapai. Target dalam pengujian ini adalah sistem pengawasan ujian berbasis *Learning Management System* pada universitas X dengan URL https://lms- demo.celoe.org/course/view.php?id=7. Tujuan dalam melakukan pengujian ini adalah mencari celah keamanan yang dapat dimanfaatkan untuk kecurangan dalam melakukan ujian.



B. Discovery

Pada fase *discovery* ini penyerang melakukan pengumpulan informasi atau *information gathering* sebelum melakukan penyerangan. Hasil *discovery* yang ditemukan pada fase ini adalah sebagai berikut:

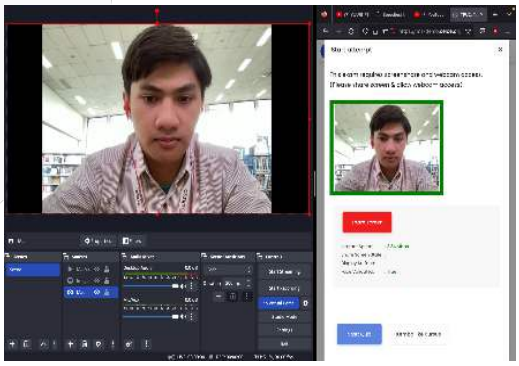
A. Selama ujian berlangsung memerlukan *webcam*
B. Saat *packet request* dirimkan terdapat beberapa yang ditemukan parameter seperti pada gambar 4.2:
   1. index
   2. methodname
   3. Args
   4. Courseid
   5. Cmid
   6. Profileimage
   7. Webcampicture yang menggunakan enkripsi base64
   8. Camera_name
   9. Camera_all
C. Mengharuskan share screen sebelum dan selama ujian .
D. Selama Ujian berlangsung terdapat *violation point* yang apabila melewati batas maka ujian akan dihentikan.

i. *4.1.3. Attack*

Fase *attack* atau penyerangan akan menggunakan hasil dari fase *discovery* .pada fase *attack* akan dilakukan pengujian menggunakan OBS (*Open Broadcaster Software*), Burpsuite, dan *inspect element* pada browser. Berikut hasil pengujian dengan memanfaatkan hasil *discovery* yang dilakukan sebelumnya:

a. Pengujian *Input* kamera dengan OBS (*Open Broadcaster Software*)

Pada fase *discovery* ditunjukan bahwa proses otentikasi akan menggunakan sebuah input sebuah *webcam*. Pengujian yang ini akan mengganti input yang seharusnya *webcam* menjadi *virtual cam* yang dimiliki oleh OBS. setelah digantinya pergantian input kamera proses otentikasi masih menunjukan *true* menandakan *virtual cam* dari OBS dapat digunakan pada proses otentikasi ini seperti yang ditunjukan pada gambar 4.5.



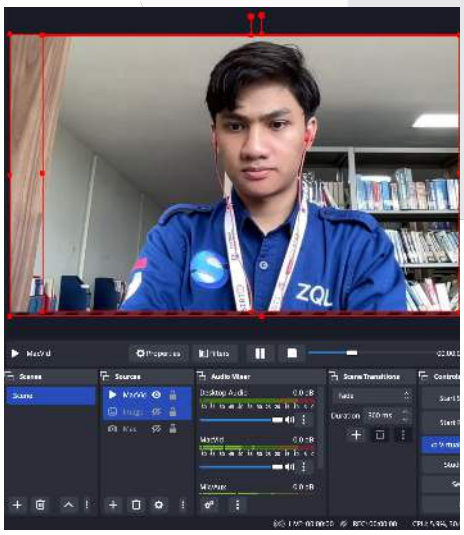
Gambar 4.5 OBS Mac WebCam

Tahap selanjutnya masih menggunakan OBS, namun mengganti *source virtual cam* yang sebelumnya mac *webcam* menjadi foto statis sebagai *source* untuk *virtual cam*. respons yang diberikan sistem seperti yang

ditunjukan pada gambar 4.6. sistem *proctoring* menunjukan hasil *true* pada foto yang digunakan di OBS *virtual cam* dan dapat melanjutkan pengerjaan quiz.

Gambar 4.6 OBS *Image*

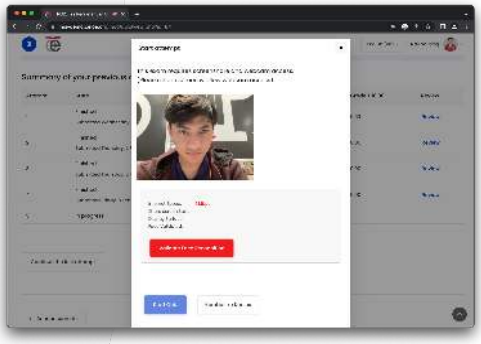
Pada pengujian selanjutnya menggunakan video sebagai *source* dari *virtual cam* pada OBS. Respons yang diberikan sistem seperti yang ditunjukan pada gambar 4.7. setelah mengganti *source* pada OBS sistem *proctoring* menunjukan nilai *true* dan dapat melanjutkan ke pengerjaan quiz.



Gambar 4.7 OBS Video

b. Perubahan kecepatan internet

Sebelum ujian akan dilakukan beberapa pengecekan terhadap peserta ujian maupun device yang dimiliki. Salah satu pengecekan yang dilakukan adalah pengecekan pada kecepatan internet. Celah keamanan yang ditemukan pada pengecekan kecepatan internet adalah parameter yang digunakan untuk melakukan pengecekan internet dapat diubah melalui *local storage* yang ada pada *browser*. Dengan diubahnya *value* dari parameter tersebut memungkinkan peserta ujian yang memiliki kecepatan *internet* kurang dari yang ditentukan dapat melakukan ujian. Seperti gambar 4.8 peserta ujian memiliki kecepatan *internet* kurang dari syarat yang ditentukan.



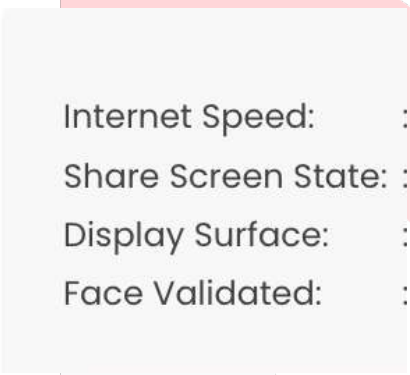
Gambar 4.8 Kecepatan internet red

Gambar 4.9 Parameter speed sebelum

Pada gambar 4.9 adalah parameter akan digunakan untuk mengubah kecepatan internet yang dimiliki dapat dilakukan dengan melakukan *inspect element*, lalu klik tab *application* dan pilih *local storage*. Pada parameter bernama *speed* dapat diubah *value*-nya menjadi speed yang menjadi syarat otentikasi, misal

20. Gambar 4.10 menunjukan *value* telah diubah dan pada gambar 4.11 menunjukan kecepatan internet yang dimiliki sudah sesuai ketentuan untuk melakukan ujian.

Gambar 4.10 Parameter speed setelah



Gambar 4.11 Kecepatan internet green

c. Perubahan parameter Cameraname dan camera_all

Perubahan pada parameter ini dilakukan jika penggunaan kamera yang gunakan dilarang saat ujian. Misal penggunaan OBS dilarang dalam melakukan ujian, namun peserta ujian dapat tetap menggunakannya tanpa terdeteksi menggunakan OBS. Celah keamanan yang digunakan bisa melalui perubahan *value* pada parameter Cameraname dan camera_all yang ditunjukan oleh gambar 4.12. Dua parameter ini memiliki fungsi untuk mendeteksi kamera yang digunakan pada saat *proctoring* dilakukan untuk parameter Cameraname dan pada parameter Camera_all berfungsi untuk mendeteksi semua kamera pada device peserta ujian.

Key	Value
cameraname	FaceTime HD Camera
camera_all	FaceTime HD Camera, OBS

Gambar 4.12 Parameter *Cam* Sebelum

Pada Gambar diatas menunjukan kamera yang digunakan pada perangkat peserta adalah FaceTime HD Camera dan semua kamera yang terdeteksi pada perangkat peserta adalah FaceTime HD Camera dan OBS Virtual Camera. Pada dua parameter cameraname dan camera_all dapat dilakukan perubahan pada *value* yang dimiliki pada setiap parameter sebelum melakukan *validate face recognition*. Setelah mengganti *value* maka isi dari parameter yang dikirim ke sistem sudah bukan parameter yang sebenarnya ada pada perangkat peserta ujian.

Key	Value
cameraname	TestingCAM
camera_all	TestinCAM

Gambar 4.13 Parameter *Cam* Sesudah

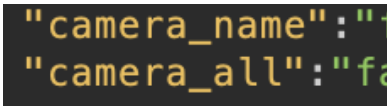
pada gambar 4.13 ditunjukan penyerang mengubah nama kamera yang digunakan menjadi TestinCAM. hasil dari perubahan *value* yang terekam pada sistem ditunjukan pada gambar 4.14. *value* pada parameter camera berhasil diubah dan tersimpan pada sistem.

camera_use	camera_all
TestingCAM	TestinCAM

Gambar 4.14 *Value* Parameter di Sistem

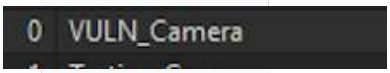
Selain melalui *inspect element* perubahan *value* ini dapat dilakukan dengan menggunakan burpsuite.

Perubahan ini dapat dilakukan saat proses otentikasi sebelum ujian dilakukan dan saat *proctoring* dilakukan ketika ujian berlangsung. Gambar 4.15 hasil dari *intercept request packet* menggunakan burpsuite .



Gambar 4.15 Parameter *Cam* Burpsuite

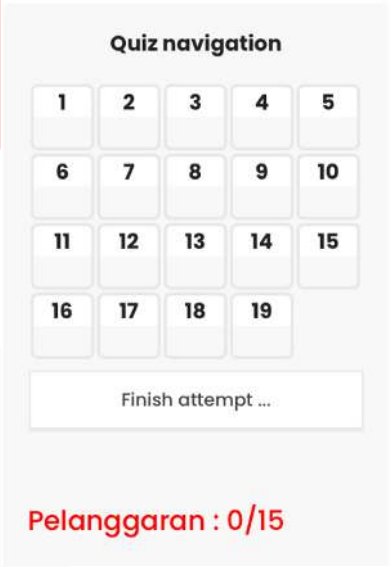
Pada parameter camera_name dan camera_all akan terisi fake_device_0 yang merupakan *default* dari *chromium* yang terhubung dengan burpsuite. Pada dua parameter tersebut dapat dilakukan perubahan *value* sebelum dikirim ke sistem. Sebagai contoh *value* tersebut diubah menjadi "VULN_Camera" pada parameter camera_name dan pada parameter camera_all diubah menjadi "VULN1, VULN2" seperti gambar 4.16.



Gambar 4.16 *Value Cam* dengan Burp di Sistem

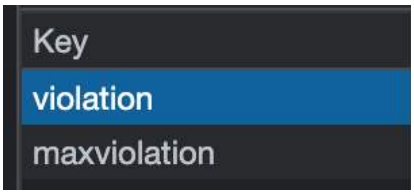
d. Perubahan poin pelanggaran yang dilakukan dan maksimal pelanggaran

Saat ujian berlangsung peserta ujian akan diawasi dengan proses *proctoring* yang dimiliki oleh sistem. Pelanggaran-pelanggaran yang terjadi akan mendapatkan *violation point* yang jika melebihi batas tertentu maka ujian yang sedang berlangsung akan selesai secara paksa. Pelanggaran-pelanggaran yang dapat menyebabkan bertambahnya *violation point* seperti berubah nya layar yang peserta ujian selain layar ujian, tidak terdeteksinya wajah peserta ujian selama ujian berlangsung, membuka *tab* lain pada *browser* peserta ujian. *Violation* maksimal yang dapat dilakukan yang diberikan pada peserta adalah 15. Namun, *violation* poin memiliki celah keamanan pada parameter yang digunakan dapat diubah *value*-nya dan peserta ujian dapat melakukan ujian tanpa harus berhenti secara paksa karena melebihi batas *violation*.



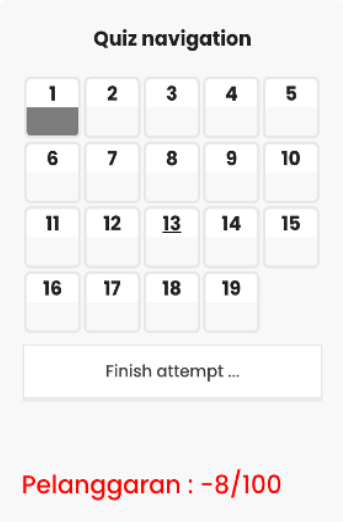
Gambar 4.17 *Violation Point*

Pada gambar 4.17 adalah kondisi *violation point* yang belum diubah. Dengan metode yang sama pada celah keamanan sebelumnya perubahan *value* akan dilakukan melalui *inspect element* pada saat sebelum ujian berlangsung. Parameter yang diubah *value* nya adalah parameter violation untuk mengubah *violation* yang telah dilakukan oleh peserta dan parameter maxviolation untuk mengubah batas maksimal *violation* yang dapat dilakukan oleh peserta ujian. Misal pada parameter violation diberi *value* -8 dan parameter *max violation value* nya menjadi 100 seperti gambar dibawah.



Gambar 4.19 Parameter Violation Sesudah

Pada gambar 4.20 ini menunjukan bahwa kedua parameter sudah berhasil diubah *value* nya. *Value* yang sudah diubah akan tersimpan pada saat ujian . setelah *value* diubah peserta ujian dapat melakukan pelanggaran sesuai dengan *violation* point yang telah mereka tentukan.



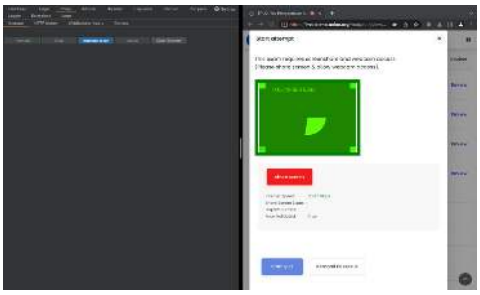
Gambar 4.20 Hasil perubahan parameter violation

e. *Broken authentication* menggunakan burpsuite

Proses otentikasi pada *proctoring* kuis dilakukan pada saat sebelum ujian dilakukan dan saat ujian berlangsung sistem akan mengambil gambar peserta yang sedang melakukan ujian berdasarkan waktu yang telah ditentukan di sistem. Terdapat celah keamanan pada sistem otentikasi yang dimiliki oleh sistem *proctoring* universitas X. celah keamanan yang pertama, yaitu dengan melakukan *intercept* pada *packet response* saat proses otentikasi sebelum ujian dimulai. Pertama, lakukan proses otentikasi seperti biasa tanpa harus menggunakan *input* kamera pada perangkat yang digunakan. Sebelum *packet request* yang akan dikirimkan pastikan sudah mengaktifkan *intercept* pada *packet response* lalu klik forward pada burpsuite. Setelah itu akan muncul respons dari sistem yang menunjukan bahwa proses otentikasi gagal atau *failed* seperti gambar 4.21.

Gambar 4.20 Respon Sistem

Lakukan perubahan *value* pada status dari “failed” menjadi “success” lalu klik forward. Berikut respons yang akan diberikan oleh sistem seperti pada gambar 4.22. respons sistem menunjukan bahwa otentikasi tersebut bernilai *true*.

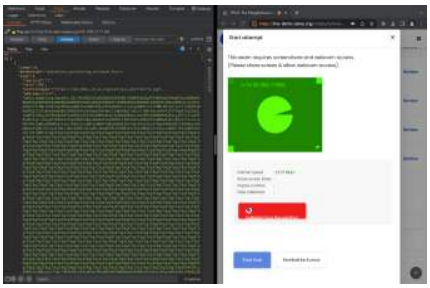


Gambar 4.21 Hasil perubahan parameter status

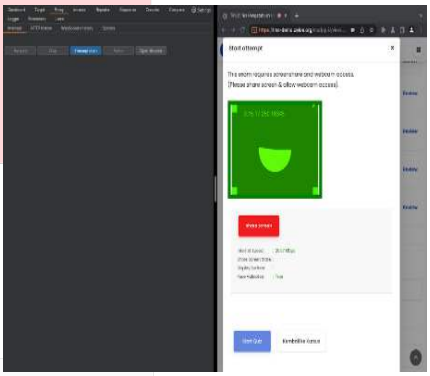
Celah keamanan kedua yang ditemukan pada proses otentikasi selanjutnya adalah melakukan modifikasi pada parameter webcampicture. Parameter webcampicture ini menggunakan *hashing* base64 untuk mengirimkan gambar hasil dari kamera yang digunakan. Karena pada burpsuite tidak dapat menambahkan input kamera selain fake_device_0 yang sudah menjadi *default* maka dapat lakukan perubahan juga pada parameter camera_name dan camera_all untuk mengelabui pengawas. Pada parameter webcampicture ganti *value* yang ada dengan foto dari pemilik akun yang sudah di *hash* dengan base64 seperti pada gambar 4.23.

Gambar 4.22 Hash base 64

Setelah melakukan *hashing* pada foto, salin semua hasil *hashing* tersebut lalu ubah isi dari parameter webcampicture dengan *hashing* yang didapatkan seperti pada gambar 4.24 lalu klik *forward*. Pada gambar 4.25 sistem *proctoring* menunjukan nilai *true* pada nilai *hashing* yang telah diubah. setelah mendapatkan nilai *true* peserta dapat melanjutkan ke proses otentikasi selanjutnya.



Gambar 4.23 Parameter setelah diubah



Gambar 4.24 Hasil hash true

b.

c. 4.2 Analisis Pengujian

Setelah melakukan pengujian kerentanan atau *vulnerability testing* pada tahap analisis akan ditentukan *scoring* pada celah keamanan yang ditemukan dengan menggunakan kalkulator CVSS (*Common Vulnerability Scoring System*). Hasil perhitungan yang didapatkan dengan menyesuaikan pada celah keamanan yang ditemukan. Pada *base score metrics* terdapat beberapa parameter seperti yang ditunjukan pada gambar 4.26 yang akan dikonversi menjadi *numerical value* lalu dilakukan perhitungan .

Gambar 4.25 Base score metrics

Pada parameter pertama *Attack Vector* (AV) diberi *value Network* (AV:N) karena selama pengujian berlangsung

penyerang dapat melakukan serangan secara jarak jauh tanpa harus terhubung dengan sebuah jaringan tertentu. Parameter kedua *Attack Complexity* (AC) diberi *value Low*(AC:L) karena penyerang tidak membutuhkan usaha dan pengetahuan yang cukup rumit dalam melakukan penyerangan. Pada parameter *Privileges Required* (PR) diberi *value Low* (PR:L) karena penyerang harus memiliki akses ke sebuah akun yang dapat melakukan ujian, namun tidak memerlukan hak akses setingkat administrator. Pada *User Interaction* (UI) diberi *value none*(UI:N) karena tidak membutuhkan interaksi dengan pengguna lain. Selanjutnya, parameter *Scope*(S) diberi *value Uchanged* (S:U) karena penyerangan yang dilakukan tidak memberikan dampak pada komponen-komponen lain. Pada parameter *Confidentiality Impact* (C) diberi *value* low (C:L) karena pada pengujian yang dilakukan menunjukan ada beberapa parameter yang tidak seharusnya ditunjukan, tetapi dapat terlihat pada saat *packet request* di-*intecept* contohnya. Pada parameter *Integrity Impact* (I) diberi *value High* (I:H) karena terdapat celah keamanan pada parameter yang akan dikirimkan ke sistem dapat diubah, sehingga menyebabkan perubahan respons yang seharusnya tidak dilakukan. Pada parameter *Availability Impact* (A) diberi *Value* None (A:N) karena pengujian yang dilakukan tidak menemukan gangguan pada layanan yang berlangsung. Setelah penentuan *value* pada setiap parameter setiap *value* memiliki *numerical value*-nya masing-masing seperti yang ada pada gambar di bawah ini.

Metric	Metric Value	Numerical Value
Attack Vector / Modified Attack Vector	Network	0.85
	Adjacent	0.62
	Local	0.55
	Physical	0.2
Attack Complexity / Modified Attack Complexity	Low	0.77
	High	0.44
Privileges Required / Modified Privileges Required	None	0.85
	Low	0.62 (or 0.68 if Scope / Modified Scope is Changed)
	High	0.27 (or 0.5 if Scope / Modified Scope is Changed)
User Interaction / Modified User Interaction	None	0.85
	Required	0.62
Confidentiality / Integrity / Availability / Modified Confidentiality / Modified Integrity / Modified Availability	High	0.56
	Low	0.22
	None	0

Gambar 4.26 Numerical value

Setelah *value* diubah menjadi *numerical value* angka yang didapatkan digunakan untuk melakukan perhitungan untuk mendapatkan CVSS *Score* yang sesuai. Pada parameter pertama *attack vector* memiliki *metric value network* dengan *numerical value* 0.85. Pada parameter kedua *attack complexity* memiliki *metric value low* dengan *numerical value* 0.77. Pada parameter ketiga *privileges required* memiliki *metric value network* dengan *numerical value* 0.62 karena pada parameter *scope* memiliki *metric value changed*. Pada parameter keempat *user interaction* memiliki *metric value none* dengan *numerical value* 0.85. Pada parameter kelima *confidentiality* memiliki *metric value low* dengan *numerical value* 0.22. Pada parameter keenam *integrity* memiliki *metric value high* dengan *numerical value* 0.56. Pada parameter ketujuh *availability* memiliki *metric value none* dengan *numerical value* 0.

ISS =	1 - [(1 - Confidentiality) × (1 - Integrity) × (1 - Availability)]
Impact =	
If Scope is Unchanged	6.42 × ISS
If Scope is Changed	$7.52 \times (ISS - 0.029) - 3.25 \times (ISS - 0.02)^{15}$
Exploitability =	8.22 × AttackVector × AttackComplexity ×
	PrivilegesRequired × UserInteraction
BaseScore =	
If Impact \<= 0	0, *else*
If Scope is Unchanged	Roundup (Minimum [(Impact + Exploitability), 10])
If Scope is Changed	Roundup (Minimum [1.08 × (Impact + Exploitability), 10])

Gambar 4.27 Rumus Base Score

Setelah perhitungan dilakukan akan didapatkan 3 hasil perhitungan, yaitu CVSS *base score*, *impact subscore*, *exploitability subscore*. Dari hasil perhitungan didapatkan CVSS *base score* dengan nilai 7,1, *impact subscore* dengan nilai 4,2, dan *exploitability subscore* dengan nilai 2.8. pada gambar 4.29 ditunjukan grafik hasil perhitungan.

Gambar 4.28 Base Scores

Dari hasil perhitungan yang dilakukan menunjukan *Base* CVSS *score* yang didapatkan adalah 7,1, *impact* 4,2, dan *exploitability* 2,8. Jika CVSS *Score* tersebut disesuaikan dengan tabel yang pada gambar 4.30 maka menunjukan peringkat *High* pada *Base* menunjukan kerentanan yang ada harus segera tangani oleh pihak pengembang. Pada *Impact* dengan *score* 4,2 menunjukan dampak yang diberikan pada celah keamanan berada pada nilai *medium* karena penyerang hanya dapat melakukan kecurangan, namun tidak sampai mengganggu layanan LMS atau mendapatkan hak akses yang tidak semestinya. Pada *Exploitability* mendapatkan *score* 2.8 menunjukan kompleksitas dari serangan yang dilakukan berada dinilai *low* karena penyerang dapat melakukan serangan atau kecurangan dengan pengetahuan yang minim.

Rating	CVSS Score
None	0.0
Low	0.1 - 3.9
Medium	4.0 - 6.9
High	7.0 - 8.9
Critical	9.0 - 10.0

Gambar 4.29 CVSS Rating

V. KESIMPULAN

Berdasarkan hasil dari pengujian dan analisis yang telah dilakukan pada Tugas Akhir ini, maka dapat disimpulkan sebagai berikut:

1. Setelah melakukan *vulnerability testing* pada *proctoring* LMS Universitas X menunjukan tingkat keamanan yang dimiliki masih sangat rendah. penyerang dapat dengan mudah melakukan kecurangan hanya dengan mengubah *value* di *local storage browser*. celah keamanan lainnya, untuk mengelabui *face detection* yang ada penyerang dapat menggunakan OBS dan Burpsuite.
2. *Dengan melakukan Vulnerability testing ini ditemukan celah keamanan pada proctoring yang dapat dimanfaatkan untuk melakukan kecurangan sebelum dan ketika ujian berlangsung. CVSS score yang didapatkan pada pengujian ini adalah 7.1. CVSS yang didapatkan termasuk dalam kategori HIGH menandakan celah keamanan yang ditemukan harus segera diperbaiki.*

REFERENSI

1. [1] Meyliana, “The Enhancement of Learning Managemens Systerm in Teaching Learning Process with the UTAUT2 and Trust Model,” *The Enhancement of Learning Managemens Systerm in Teaching Learning Process with the UTAUT2 and Trust Model,* 2019.
2. [2] F. El Hajj, “Multi-agent System Vulnerability detector for a secured E-learning Environment,” *Multi-agent System Vulnerability detector for a secured E-learning Environment,* 2016.
3. [3] S. S. Zarish, “Analyzing Usability of Educational Websites Using Automated Tools,” *Analyzing Usability of Educational Websites Using Automated Tools,* 2019.
4. [4] EC-Council, Certified Network Defender (CND) Version 2 w/ iLabs (Volumes 1 through 4) 2nd Edition, 2nd

penyunt., vol. 1, EC-Council Academia, 2020.
5. [5] H. Alptekin, "Towards Prioritizing Vulnerability Testing," *Towards Prioritizing Vulnerability Testing,* 2020.
6. [6] K. Scarfone, "Technical Guide to Information Security Testing and Assessment," *Technical Guide to Information Security Testing and Assessment,* 2008.
7. [7] N. Anantharaman, "Identifying the Usage of Known Vulnerabilities Components Based on OWASP A9," *Identifying the Usage of Known Vulnerabilities Components Based on OWASP A9,* 20.
8. [8] OWASP, "OWASP Top Ten," *OWASP Top Ten,* no. https://owasp.org/www-project-top-ten/, 2021.
9. [9] "NVD - Vulnerability Metrics," [Online]. Available: https://nvd.nist.gov/vuln-metrics/cvss. [Diakses 2 Februari 2023].
10. [10] A. H. S. Ganidisastra, "An Incremental Training on Deep Learning Face Recognition for M-Learning Online Exam Proctoring," *An Incremental Training on Deep Learning Face Recognition for M-Learning Online Exam Proctoring,* 2021.