FORMAT: Jurnal Ilmiah Teknik Informatika
Akreditasi Sinta 5 RISTEKBRIN No. 164/E/KPT/2021; E-ISSN 2722-7162 | P-ISSN 2089-5615

Vol. 14, No. 1,
January 2025,
1-12. 2025

Penerapan Fault Injection based Security Testing untuk Menemukan Potensi Celah Keamanan Indor pada Platform XYZ

Faiz Hanafi [1] ; Andi Purnomo[2]

[1,2] *Fakultas Ilmu Komputer, Universitas Ary Ginanjar, Menara 165, Cilandak Timur, Pasar Minggu, Jakarta Selatan*

[1] f.hanafi@students.esqbs.ac.id, [2] andi.purnomo@uag.ac.id

Kata kunci:
Bug, Security Disclosure, Fault Injection, IDOR, Parameter

Abstract

Security Disclosure has become an important program in identifying and addressing vulnerabilities in software systems across various organizations and institutions by providing an opportunity for system testing through collaboration with external parties. This research aims to provide a deeper understanding of security disclosure through the Security Disclosure program on Platform XYZ, with a focus on Fault Injection techniques through parameter manipulation to discover IDOR (Insecure Direct Object Reference) vulnerabilities. The research begins by explaining the importance of the Security Disclosure program in the context of software security and system improvement. Furthermore, this study conducts testing on Fault Injection techniques through parameter manipulation, which is used to search for IDOR vulnerabilities on Platform XYZ. By manipulating input parameters, researchers can test the system by inducing unexpected behavior that can reveal sensitive information or grant unauthorized access to permissions. Case studies and real-world examples are used to demonstrate the effectiveness and impact of Fault Injection techniques in enhancing system security. The research findings have shown a high level of IDOR vulnerability with an impact on thousands of users. This research also highlights the importance of broader understanding of software security and security awareness for organizations and individual users. It is expected that this research will contribute to improving understanding of security disclosure and Fault Injection techniques through parameter manipulation, as well as promoting higher security awareness in protecting software systems from attacks and safeguarding

Pendahuluan

Platform XYZ merupakan sebuah platform penyedia layanan pembelajaran daring berbasis *Learning Management System* yang terkolaborasi dengan berbagai macam fitur tertentu, *Platform XYZ* menjadi pelopor penyedia layanan pembelajaran daring berbasis *Learning Management System* digital pertama di Indonesia yang telah menjadi Inovator di Program IBT 2017 yang dilaksanakan oleh Kemenristekdikti, *Platform* XYZ juga telah memberikan akses *Learning Management System* digital 30 Ribu siswa, guru, serta orang tua sebagai media pembelajaran daring yang dapat digunakan diseluruh Indonesia.

Sebagai *Platform* media pembelajaran yang digunakan oleh banyak user *Platform XYZ* memiliki sebuah *Responsible Security Disclosure Program* untuk memberikan jaminan mutu dan keamanan data pengguna yang ada di Indonesia, *Platform XYZ* pernah mengalami serangan yang hampir menyebabkan kerugian materil. Melalui *Responsible Security Disclosure Program* peneliti maupun penguji perangkat lunak dari pihak eksternal perusahaan dapat berkontribusi memberikan informasi terkait potensi celah keamanan (*Bug*) yang dapat diekploitasi.

Bug merupakan sebuah kesalahan sistem yang bersifat logikal maupun non logik, identifikasi sebuah *bug* perlu ditingkatkan dalam setiap development sebuah perangkat lunak untuk memberikan hasil yang maksimal serta menghindari adanya *behaviour* sistem yang tidak sesuai dengan semestinya, sebuah tahapan pengujian perangkat lunak yang diharapkan untuk ditemukannya sebuah kesalahan dapat dilakukan tahapan *Security Testing*, *Formal testing* dapat dilakukan dengan melakukan pengujian piranti lunak berdasarkan teori yang diterjemahkan dalam sebuah formula *pseudocode flow* program yang sedang diuji, metode ini memiliki batasan dalam penggunaanya yakni memakan waktu yang cukup lama jika dihadapkan dengan model yang memiliki data yang besar dan kompleksitas tinggi.

Terdapat metode lain yakni *Model Based Security Testing* melakukan pendekatan dengan melakukan pengujian dari *behaviour* piranti lunak pada tiap tahapan pengujian secara sistematis, yang dimulai dari pengujian masukan keluaran, proses diagram hingga melakukan pengujian lansung pada *test case* yang diharapkan. Metode *Fault injection-based security testing* berfokus pada poin interaksi pada *environment* aplikasi, termasuk *user-input*, *file system, Network Interface,* dan *Environment Variable.* Menurut Tian-yang dkk., (2010), Fault injection-based security testing dapat digunakan untuk melakukan pengujian pada scope

program yang mencakup pada environment variable untuk mendeteksi celah keamanan IDOR (Insecure Direct Object Reference) melalui parameter tampering. Celah keamanan *IDOR* menjadi salah satu celah keamanan yang berdampak *Risk* tinggi, pada Mei 2022 security researcher menemukan celah keamanan IDOR di Instagram yang berdampak pada attacker dapat melakukan pengubahan media reels pengguna user Instagram diseluruh dunia, bug ini dihargai pihak Facebook dengan nilai reward bounty USD $49500 (Sharma, 2022). Kasus serupa juga terjadi pada Agustus 2022, pada website Reddit, dimana attacker dapat mengubah dan menambahkan custom link social media user, temuan celah keamana ini dihargai dengan nilai reward bounty USD $5000 (Firat, 2022). Dalam pengukuran tingkat celah keamanan terdapat tingkatan yang digunakan untuk menentukan level yang biasa disebut dengan *severity*, secara umum severity ini dibagi menjadi empat tingkatan level yakni *Low, Medium, High, dan Critical.* Proses penentuan tingkatan level ini dapat disesuaikan dengan keadaan peneliti berbasis dengan pernyataan kondisi peneliti pada saat melakukan pengujian (Hackerone, 2022).

Metode lain yang biasa digunakan adalah *Fuzzy testing* yang merupakan sebuah metode yang paling tepat digunakan untuk menemukan sebuah *endpoint* maupun *direktory* dan *file* menggunakan *random string data* yang di-*inject* pada suatu variabel metode ini merupakan metode yang digunakan secara umum untuk melakukan *information gathering* pada target sebelum melakukan eksploitasi dengan mengumpulkan informasi berupa pranala, direktori maupun *hidden parameter* yang memungkinkan untuk diteliti lebih jauh, Proses pengujian melalui *fuzzy testing* ini biasanya juga digunakan dalam piranti lunak *Scanning Tools*. Dalam implementasinya Metode *Vulnerabilty Scanning* memberikan sebuah hasil yang kurang akurat dibandingkan dengan metode lainnya. Dalam pengujian *Vulnerabilty Scanning* melalui *scanning tools* masih memiliki potensi tidak ditemukannya celah keamanan yang berpotensi tinggi *(Critical)* seperti *IDOR*. Property Based Testing menurut Fink & Bishop (1997) adalah model piranti lunak yang dikembangkan oleh TASPEC language. Pada pengujian property based testing dilakukan pendekatan untuk melihat suatu requirement pada piranti lunak untuk menemukan celah keamanan sesuai dengan prioritas dan klasifikasi yang diharapkan. Sedangkan menurut MacIver dkk., (2019) property based testing lebih menitik beratkan untuk pengujian dengan menentukan properti yang berlaku untuk berbagai macam masukan, yang kemudian library akan mencoba menghasilkan kasus uji untuk membantahnya. Melalui *White Box Security Testing* dapat dilakukan pengujian piranti lunak yang baik, metode ini merupakan salah satu metode yang paling umum digunakan dalam pengujian *security testing* dengan membuat sebuah program untuk menyerang *code* yang ada demi menguji ketahanan piranti lunak dalam menerima serangan yang tersistemasi melalui aplikasi, Metode *Risk based Security Testing* merupakan metode yang menggabungkan sebuah pendekatan vektor pada proses tahapan *Software Development Lifecycle (SDLC),* untuk menemukan potensi (*Risk*) pada sebuah skenario serangan dan analisis piranti lunak yang ada.

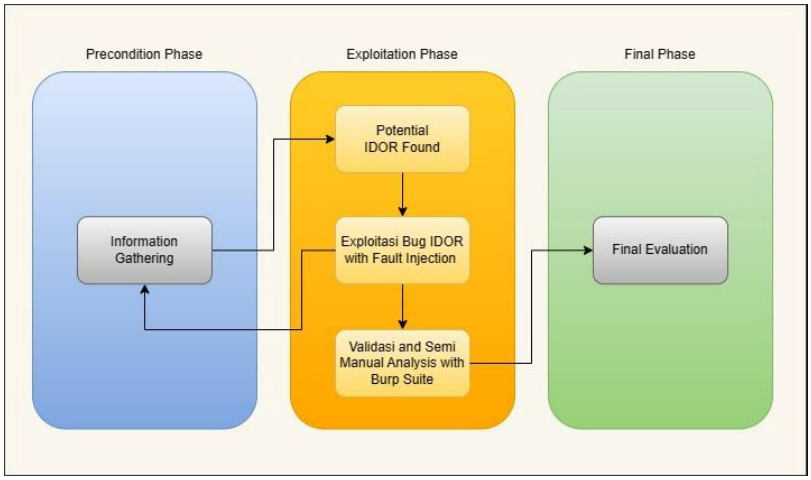
Metode *Fault injection-based security testing* menjadi pilihan bagi peneliti untuk melakukan pengujian piranti lunak yang sesuai dengan tipe celah keamanan *IDOR* melalui *parameter manipulation*, metode *Fault injection-based security testing* memiliki keunggulan dibandingkan dengan metode security testing lainnya, namun cara ini masih jarang digunakan dibandingkan dengan metode *White Box Testing* yang sudah sangat umum, pada penelitian ini *Fault Injection based security* testing akan diterapkan untuk pengujian dan penemuan celah keamanan *IDOR* secara manual, sehingga mendapatkan hasil yang mungkin terlewat dalam proses pengujian menggunakan metode lainnya.

Hasil dari penelitian ini diharapkan dapat menjadi sumber referensi ilmiah tentang ekploitasi celah keamanan IDOR pada *Platform* berbasis *web application* menggunakan teknik *fault injection* yang masih minim diterapkan dalam pengujian piranti lunak *Learning Management System* serta piranti lunak lainnya yang memiliki arsitektur sistem yang serupa untuk melakukan pendekatan pengujian yang maksimal sehingga dapat ditemukannya celah keamanan yang berpotensi merusak sebuah sistem piranti lunak.

Metode Penelitian

A. Tahapan Penelitian

Berikut adalah alur tahapan penelitian yang dilakukan.



Gambar 1. Alur Penelitian

Alur penelitian terbagi menjadi sebuah rangkaian skema *hacker* sebagai user penguji, pada satu bagian besar dengan ruang lingkup pengolahan informasi, dan *developer* sebagai *end user*. Berikut penjelasan tiap tahapan skema di atas.

- Tahap *Precondition*
  Skema *Precondition* ini berada diluar skema utama pada bidang besar yang mencakup tahapan lainnya, dalam situasi ini hacker bertindak sebagai peneliti melakukan pengujian dengan mencari objek yang membuka *security discolure program* untuk bersedia dilakukan pengujian sistem, dengan adanya hal ini maka hacker dapat menentukan untuk melakukan pengujian *IDOR* pada sistem LMS pada *Platform* XYZ.
- Melakukan *Gathering Information*
  Tahapan ini merupakan bagian dari tahapan utama yang tergabung dalam ruang lingkup pada satu skema besar, pada tahap ini *hacker* atau peneliti melakukan pengumpulan informasi terkait dengan program yang terkait, informasi ini meliputi hal-hal yang dapat digunakan untuk membuat model pendekatan pengujian *fault injection* untuk menemukan celah keamana *IDOR*, hacker akan mengumpulkan informasi melalui pembacaan *Souce Code* program LMS secara daring, crawling urls menggunakan python repo seperti *gau* (Leo, 2022), *waybackurls* (Tomnomnom, 2022), *subfinder* (ProjectDiscovery, 2022) juga dilakukan pengujian *intercept* parameter melalui *burp suite* untuk mengetahui *hidden parameter* saat *request* ke *server*. Dari proses ini hacker melakukan kode review, mengumpulkan *urls*, hingga *subdomain* sistem yang bisa ditemukan berdasarkan model celah keamanan *IDOR* untuk manipulasi paramater seperti "*urls?parameter*={1}" atau "urls/{1}" ditemukan. Proses ini merupakan salah satu proses paling menentukan untuk mendapatkan celah keamanan, jika dalam proses ini data dirasa telah cukup maka dilakukan proses alur selanjutnya yakni potential *IDOR*.
- Menemukan Potential *IDOR*
  Pada tahapan ini merupakan sebuah indikasi ketika parameter dapat ditemukan dan memiliki peluang untuk dilakukan manipulasi parameter melalui *fault injection parameter tampering*.
- Implementasi Exploitasi Bug *IDOR*
  Proses exploitasi dilakukan setelah menemukan potential *IDOR* parameter. Proses ini merupakan tahapan utama dari rangkaian tahapan sebelumnya, pada proses ini pula terjadi proses pengamatan behaviour pada sistem untuk dilakukan analisis lebih jauh dengan dilakukan pengujian *fault injection*. Setelah melakukan pengamatan terdapat dua kondisi dimana jika exploitasi tidak berhasil dilakukan maka proses akan diulang kembali melalui tahapn *information gathering*, cara ini dapat dilakukan berulang ulang sampai sebuah bug IDOR dapat dilakukan implementasi dan eksploitasi secara sempurna.
- Proses Validasi dan *Analysis Semi-Manual testing*
  Tahapan validasi ini memberikan sebuah penguatan statement tentang ditemukannya celah keamanan *IDOR*, dengan analisis ini dilakukan juga pengujian secara berulang dengan bantuan *burp suite* dan tools pendukung lainnya untuk memastikan celah yang ditemukan adalah celah yang valid dan dapat dideliver serta dilakukan *reporting* kepada tim *security developer Platform XYZ.*
- Proses Tahapan Evaluasi
  Proses evaluasi merupakan sebuah tahapan akhir dari tahapan yang tergabung dalam tahapan utama skema alur penelitan, pada tahap ini dilakukan evaluasi mendalam tentang penyebab sumber serta memberikan *statement preventif* untuk mencegah terjadinya ditemukannya celah yang sama pada tempat yang lain di sistem.
  Pada tahapan akhir ini juga dilakukan reporting untuk memberikan penjelasan serta tahapan dan evaluasi yang dilakukan *hacker* untuk dapat diterima oleh *developer* agar dapat diproses segera dan dilakukan *patch* perbaikan.

Jika sebuah kerentanan IDOR berhasil ditemukan dengan melakukan pengubahan parameter melalui *parameter manipulation* dilanjutkna proses pengujian dengan melakukan pengiriman *request* data dengan jumlah *intruder payload*. Sebelum melakukan proses pengiriman *request* ditentukan jumlah user yang akan diuji dengan teknik sampling sebagai berikut.
Berdasarkan data yang didapatkan dari pihak developer bahwa *user* aktif pengguna *LMS Platform XYZ* adalah 30 ribu *user*.

$$n = \frac{N}{1+(Ne^2)} \qquad \text{Rumus Sampling (0.1)}$$

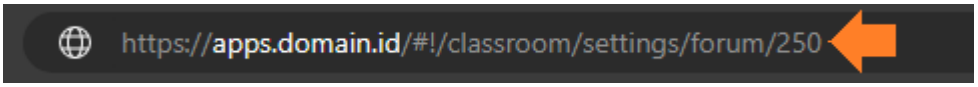
Maka dengan rumus sampling tersebut, jika N adalah jumlah pengguna dengan 30 ribu user, e adalah margin error dengan nilai 0.01. Berikut kalkulasi yang didapatkan.

$n = \frac{30000}{1+(30000\ x\ 0.01^2)}$; $n = \frac{30000}{1+(30000\ x\ 0.0001)}$; $n = \frac{30000}{1+(3)}$; $n = \frac{30000}{4}$
Dengan hasil n = 7500

Maka *user* aktif yang akan dilakukan pengujian dalam proses validasi melalui intruder adalah 7500 dari 30 ribu *user*.

Hasil dan Diskusi

Dalam skema yang akan dilakukan maka dilakukan pembuatan dua users dengan detail akun pertama yang bertindak sebagai korban untuk target pengujian *Account Victim* dengan Nomor induk userid *105639532102425562824*, akun ini digenerate dan didaftarkan menggunakan google account accountvictim@gmail.com, pada sisi lain telah tergenerate juga dengan *Account Attacker* yang sebelumya telah didaftarkan melalui akun gmail accountattacker@gmail.com, dengan Nomor induk user id *109289434223976181396*. Setiap user dapat ditambahkan juga nomor telepon pada setiap account. Setelah membuat kedua user maka dilakukan pengujian fitur yang ada pada *Platform* XYZ, Pada fitur kelas yang sesuai dengan skala *priority* yang dikategorikan oleh pihak pengembang laman akan dialihkan menuju endpoint *https://apps.domain.id/#!/classroom/my*. buat kelas dengan menggunakan fitur pada button kelas kemudian klik button pada *Buat Kelas*, setelah dibuat terdapat sebuah kelas dengan forum diskusi yang terletak pada sebuah laman forum *classroom endpoint url* https://apps.domain.id/#!/classroom/settings/forum/{id_forum} jika {id_forum} dilakukan parameter manipulation secara langsung pada url, maka tidak terjadi perubahan baik dalam data yang didapatkan maupun tampilan akan statis, tidak mengalihkan ke laman apapun.



Gambar 2. Domain

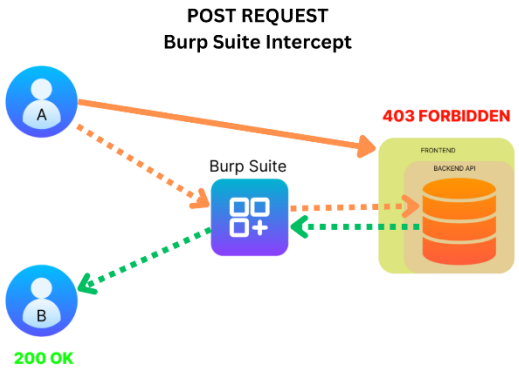
pada pengujian ini dilakukan juga pada fitur lain yang diukur dengan skala prioritas sesuai dengan kategori yang diberikan oleh pengembang.

Tabel 1. Parameter

URLs	Parameter	Parameter Custom	Result	Status	Priority
https://apps.domain.id/#!/classroom/settings/forum/250	250	251	Refresh	Non Valid	**Very High**
https://apps.domain.id/#!/dashboard	No Parameter	No Parameter	Refresh	Non Valid	**High**
https://apps.domain.id/#!/bank_materi/edit/19	19	20	Refresh	Non Valid	**Medium**
https://apps.domain.id/#!/questions_bank/question/60	60	61	Refresh	Non Valid	**Low**
https://apps.domain.id/#!/calendar_private/edit/39	39	40	Refresh	Non Valid	**Very Low**

Setelah melakukan pengujian pada fitur Kelas, Dashboard, Bahan Ajar, Bank Soal, dan Jadwal diketahui pada tabel bahwa laman tetap bersifat statis dengan melakukan muat ulang kembali pada laman sebelumnya setelah dilakukan manipulasi parameter dengan penambahan nilai 1 poin pada setiap parameter.
Dari hasil yang didapatkan maka pengujian lanjutan akan dilakukan dengan melakukan pengiriman *request* melalui *burp suite* dengan skema bahwa dapat diexploitasi melalui *backend api* menggunakan *vector* penyerangan yang lain.

Setelah mengetahui bahwa parameter manipulation tidak dimungkinkan dalam proses mengubah id secara langsung, maka dilakukan proses analisis intercept melalui aplikasi *burp suite*, proses dilakukan dengan menggunakan POST parameter method untuk mengirim *request* ke *server* backend-api dengan skema pada gambar berikut.



Gambar 3. Burp Suite Intercept

Berdasarkan skema pada gambar diatas dilakukan pengujian fitur yang ada pada *Platform XYZ* setelah dilakukan skenario *parameter manipulation* pada menggunakan *burp suite* maka dihasilkan sebuah *test plan* sebagai berikut.

Tabel 2. Hasil *Test Plan*

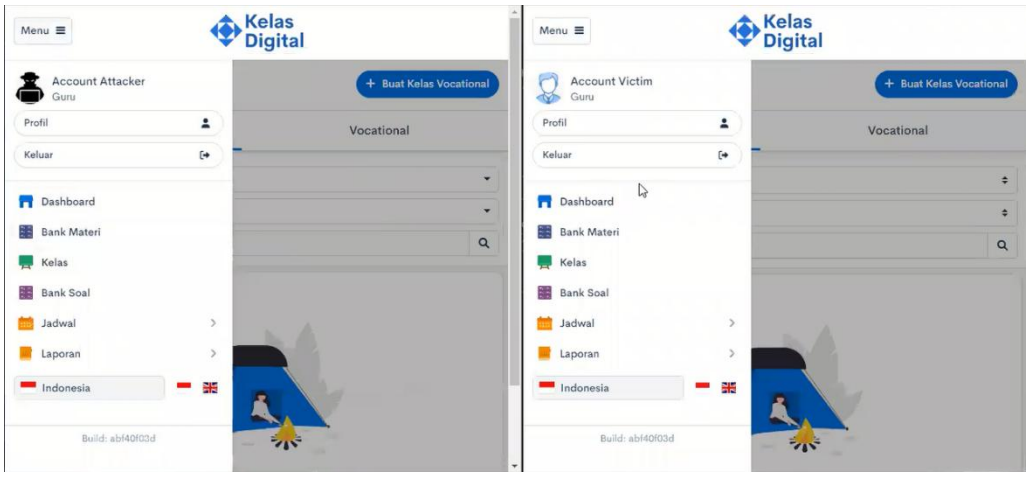
Scenario	URLs	Parameter	Fault Injection Parameter	Expected Result	Test Case ID	Priority
Melakukan pembuatan kelas dengan id_course dengan id_course user lain	https://apps.domain.id/#!/classroom/settings/forum/251	251	250	**403 Forbidden**	KLS-1	**Very High**
Melakukan pengubahan akun user lain	https://apps.domain.id/#!/dashboard	Tidak terdapat parameter pada url	Tidak Terdapat parameter pada url	**200 OK**	DSH-2	**High**
Melakukan pembuatan kelas dengan ide dengan id user lain	https://apps.domain.id/#!/bank_materi/edit/19	19	20	**403 Forbidden**	BAJ-3	**Medium**
Melakukan pembuatan soal dengan id_group dengan id_group user lain	https://apps.domain.id/#!/questions_bank/question/60	60	61	**403 Forbidden**	BAS-4	**Low**
Melakukan pembuatan calendar event dengan eventID dengan eventID user lain	https://apps.domain.id/#!/calendar_private/edit/39	39	40	**403 Forbidden**	CLD-5	**Very Low**

Pada tabel diatas merupakan test case yang terdapat pengujian yang berfokus pada lima fitur utama dengan *priority* dari *high* to *low*, test case memiliki kolom skenario sesuai dengan skenario pengujian pada fitur yang diuji, pada fitur -fitur terkait terdapat sebuah respond "*Status IDOR*" yang menunjukkan kondisi yang mendukung dan berpotensi besar dalam keberhasilan eksploitasi IDOR. Fitur yang menunjukkan hasil secara sempurna adalah pada fitur kelas forum, sedangkan pada fitur dashboard dengan tingkat priority dibawah fitur kelas tidak berpotensi IDOR dikarenakan tidak terdapatnya parameter untuk dilakukan manipulasi, pada tingkatan dibawah fitur *dashboard* yakni fitur bahan ajar memiliki hasil *responses* yang berstatus 200 OK, namun tidak terjadi perubahan pada user lain yang dituju sebagai korban, hal yang sama juga terjadi pada fitur *calendar* dengan response status adalah 200 OK dengan tanpa perubahan pada user taget dengan ID yang dituju.

Tabel 3. Status IDOR

Test Case ID	Result	Result JSON Response	Methode	Status IDOR	Priority
KLS-1	**200 OK**	**{"status":true, "message":"Data added successfully!"}**	**POST**	Success	**Very High**
DSH-2	**200 OK**	** <!-- Dashboard Content -->**	**GET**	Undefined	**High**
BAJ-3	**200 OK**	**{"status":true, "message":"Data added successfully!"}**	**POST**	Failed	**Medium**
BAS-4	**200 OK**	**{"status":"failed", "message":"Access is prohibited."}**	**POST**	Failed	**Low**
CLD-5	**200 OK**	**{"status":"1", "message":"Data updated successfully!"}**	**POST**	Failed	**Very Low**

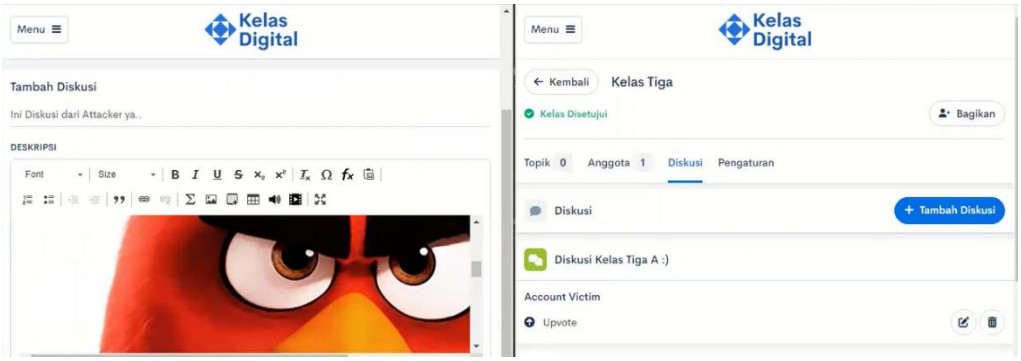
Berdasarkan berikut diketahui bahwa fitur bank soal menunjukkan hasil 200 OK namun pada json terdapat sebuah kondisi yang membatasi akses antar user sehingga walaupun berstatus 200 OK tidak ter-*generate* sebuah bank soal pada id target yang dituju. Dengan hasil yang didapatkan pada berikut adalah detail eksploitasi yang berhasil dilakukan pada fitur kelas.



Gambar 4. Request Melalui Burp Suite

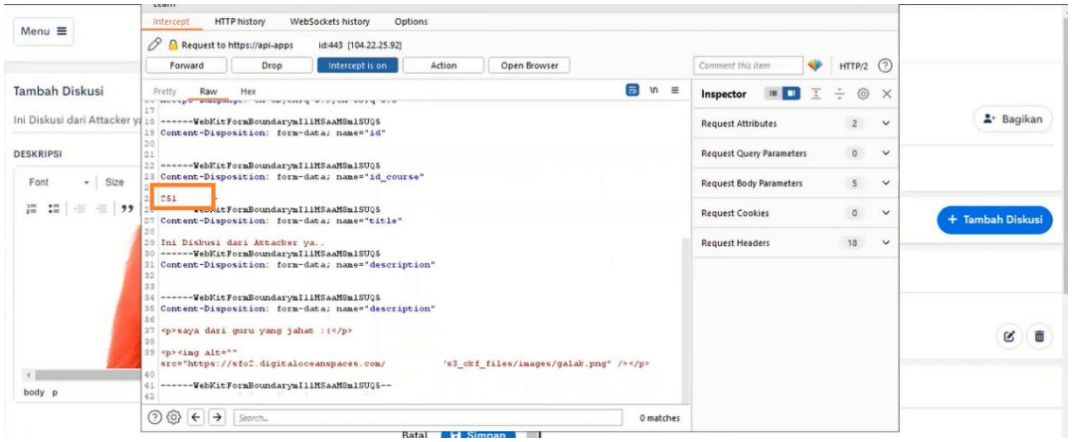
Pada gambar diatas dilakukan sebuah *request* melalui burp suite dengan mengirim paket *request* melalui parameter POST Method serta melakukan pengubahan parameter pada *id_course* forum. Proses dimulai dengan membuka dua akun pada waktu *realtime simultanuosly*.

Setelah membuat kedua kelas tergenerate id pada setiap kelas yakni pada account attacker memiliki kelas dengan *id_course* 251 dan sementara pada akun *victim* memiliki *id_course* 250, attacker mencoba untuk mengenerate sebuah diskusi forum dengan nama ini forum “Ini Diskusi dari Attacker ya” disertai dengan gambar pada forum tersebut.



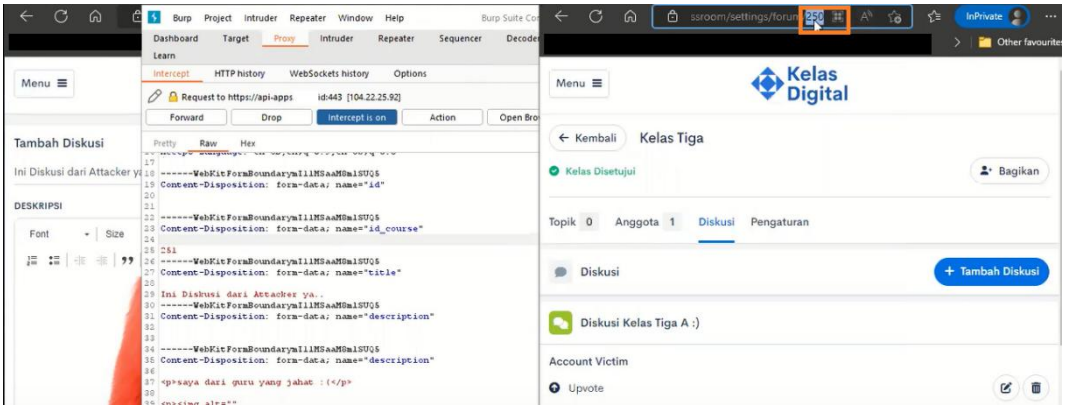
Gambar 5. Account Attacker

Pada saat pembuatan kelas tersebut telah disiapkan burp suite yang digunakan untuk melakukan pengujian dengan mengubah parameter *request id_course* dari 251 menjadi 250 dengan detail *request* sebagai berikut.



Gambar 6. Perubahan Parameter

Berikut adalah detail *request* pada aplikasi burp suite melalui parameter manipulation dengan pengujian fault injection dengan memasukkan parameter id yang salah dan tidak sesuai dengan program. Pada *request* ini terdapat parameter *id_course* 251 dengan endpoint POST */api/classroom_forum/save_forum_thread HTTP/2* diketahui bahwa proses ini juga melakukan pengiriman sebuah file berupa image bernama galak.png.
Pada kondisi ini burp suite akan menahan *request* untuk dapat diubah dan memanipulasi data *request* yang diinginkan sebelum dapat dikirimkan ke *server*. Maka dilakukan pengubahan parameter sebagai berikut.



Gambar 7. Perubahan Parameter

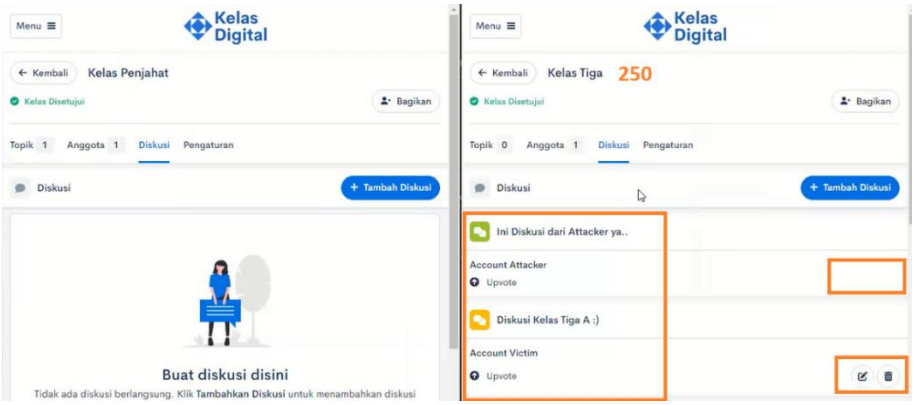
Berikut adalah pengubahan *request* pada burp suite dengan parameter semula 251 menjadi 250.

POST /api/classroom_forum/save_forum_thread HTTP/2
Host: api-apps.domain.id
Content-Disposition: form-data; name="id_course"
250 ///pengubahan pada paremeter ide_course

Dengan pengubahan ini maka *server* akan menerima bahwa attacker bertindak sebagai account victim yang melakukan pengiriman POST method menggunakan akun yang berbeda.
Ketika dilakukan *forwarding request*, maka yang terjadi adalah pada akun attacker tidak tergenerate sebuah forum, terdapat satu anomali yang memberikan sebuah pernyataan bahwa hasil *request* dapat dikirimkan dan diterima oleh *server* namun tidak pada attacker account, maka hasil dapat dilihat dari hasil respone berikut.

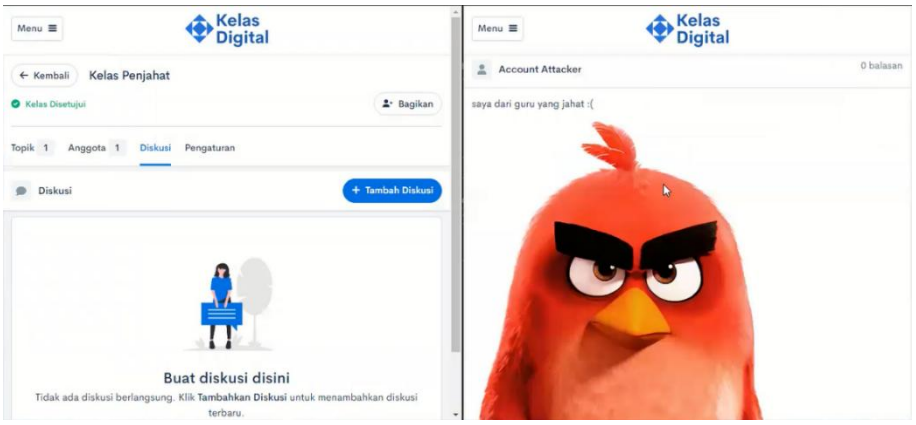
HTTP/2 200 OK
Host: api-apps.domain.id
{"status":true,"message":"Data added successfully!"}

Hasil yang didapatkan adalah *{"status":true,"message":"Data added successfully!"}*, setelah berhasil mengirim serangan, dilakukan pengecekan pada forum victim dengan memuat ulang laman id_forum 250.



Gambar 8. Sending Attack

Setelah dilakukan serangan, terlihat sebuah anomali yang membuat *user victim* tidak bisa melakukan edit, mengupdate maupun menghapus forum yang dibuat oleh penyerang, sebagai perbandingan pada forum yang digenerate oleh *user victim* terdapat *button edit* dan *delete*.
Jika user membuka forum yang dikirimkan oleh attacker maka muncul gambar yang dikirim oleh *attacker* sebagai bukti bahwa pengiriman *request* berhasil dan diterima secara valid oleh *server*.



Gambar 9. Pengujian Request

Berikut adalah hasil GET *Request* untuk melihat hasil respon yang didapatkan dari mengirim *request* GET melalui burp suite pada endpoint *GET /api/classroom_forum/250*.

Gambar 10. Request GET

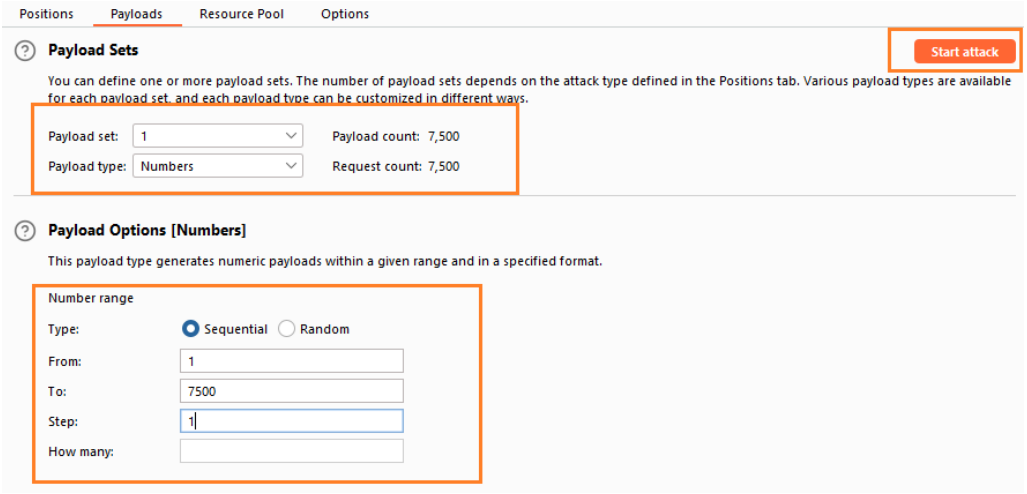
Terdapat forum yang dibuat oleh attacker dengan judul, nama file dan user sesuai dengan yang dikirimkan oleh attacker ke victim. Berikut adalah detail perintah *request* dan hasil dari respone pada endpoint *GET* /api/classroom_forum/250.


```
{
 "data": [
    "id_courses": "250",
    "title": "Ini Diskusi dari Attacker ya..",
    "nameUser": "Account Attacker ",
    "isEditable": "0"},  {
    "nameUser": "Account Victim",
    "isEditable": "1"  },  ],
 "status": "true"}
```


Proses selanjutnya dilakukan pengujian menggunakan intruder dari burp suite untuk melihat banyaknya sebuah *request* yang valid dan diterima oleh *server* menggunakan nilai variable *id_course*.

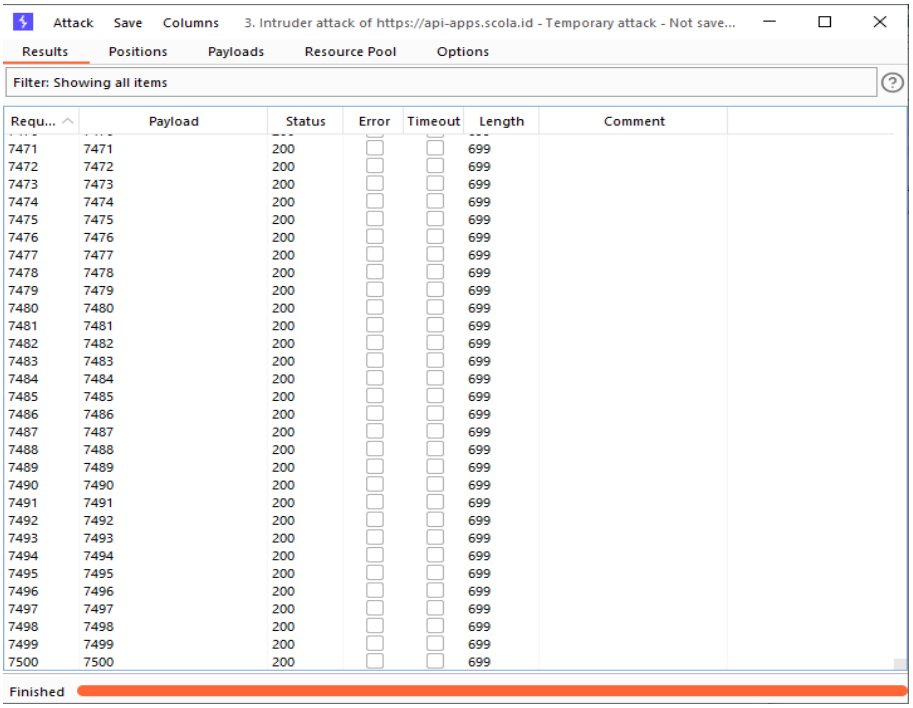
Gambar 11. Penggunaan Inturder

Sebuah parameter dikelilingi oleh tanda “$” untuk menunjukkan parameter mana saja yang akan digenerate untuk melakukan pengujian dengan jumlah *request* tertentu.



Gambar 12. Pemberian Parameter

Pada tab payload diisi sebuah nilai dengan payload tipe *Number* artinya bahwa payload yang akan digunakan dan dilakukan pengiriman *request* adalah berjenis integer angka, dengan jumlah range 1-7500, dengan jumlah *request count* 7500 lalu klik *start attack*. Proses akan berjalan dengan muncul popup sebagai berikut.



Gambar 13. Hasil *Payload*

Berdasarkan hasil exploitasi yang dilakukan dalam proses yang dilakukan telah diketahui bahwa exploitasi menggunakan metode fault injection berhasil untuk dilakukan pada sebuah parameter yang berada pada POST method Parameter, Setelah dilakukan pengujian bahwa LMS *Platform* XYZ pada domain.id tersebut memiliki kerentanan yang berpotensi tinggi hasil yang didapatkan pada pengujian dengan rincian sebagai berikut.

a) Pada *Endpoint classroom/settings/forum/{id_course}* dengan menggunakan endpoint tersebut IDOR dapat dilakukan melalui intercept burp suite dengan mengubah *id_course* parameter menjadi *id_course* pengguna lain melalui POST.
b) Hasil dari pengiriman *request* telah divalidasi dengan melakukan *request* pada url GET /api/classroom_forum/{id_course}.
c) Tingkat kerentanan pada celah keamanan ini berpotensi HIGH sesuai dengan skala prioritas fitur beserta pengguna yang terdampak dengan melihat potensi yang ditimbulkan pada 7500 pengguna.
d) Attacker dapat melakukan pengiriman forum pada setiap user yang diinginkan tanpa adanya batasan authorization.

Perlu dilakukan remediation dengan melakukan validasi pada setiap inputan *request* untuk mencegah celah keamanan IDOR

Kesimpulan

Implementasi dari fault injection dengan parameter manipulation untuk menemukan potensi celah keamanan IDOR telah berhasil dilakukan, menggunakan request POST pada backend API Platform XYZ hasil ini didapatkan setelah melakukan analisis bahwa pada frontend tidak bisa dilakukan secara langsung dengan mengubah parameter pada url yang ada. Pada pengujian yang dilakukan terdapat dampak kerentanan dengan Skala High dengan mempertimbangkan dampak yang bisa dilakukan hingga ribuan pengguna pada saat melakukan pengiriman request 7500 pengguna, serta priority dari fitur yang dikembangkan oleh developer dengan hasil responses request adalah 200 OK. Dampak yang bisa didapatkan pada kerentanan yang didapatkan adalah attacker bisa melakukan pengiriman forum pada pengguna lain, sehingga pengguna tidak dapat melakukan edit maupun delete forum, karena perbedaan role akses nama pengguna yang berhak. Pengguna yang bertindak sebagai role guru dapat kehilangan integritas, tidak terkontrolnya forum diskusi dikarenakan pengguna bukan pembuat forum, unauthorized access pada diskusi kelas, saling berbagi file berbahaya yang dikirimkan oleh Attacker.

Referensi

[1] Fink, G., & Bishop, M. (1997). Property-Based Testing; A New Approach to Testing for Assurance. In *ACM SIGSOFT Software Engineering Notes* (Vol. 22).

[2] Firat. (2022, December 23). *How these IDOR vulnerability earned 5000$ | Hackerone Reddit Bug Bounty*. https://infosecwriteups.com/how-these-idor-vulnerability-earned-5000-hackerone-reddit-bug-bounty-c685fcfbd8bc

[3] Hackerone. (2022). *Severity Hackerone*. https://docs.hackerone.com/hackers/severity.html#severity-caps

[4] Leo, C. (2022, July 24). *Fetch known URLs from AlienVault's Open Threat Exchange, the Wayback Machine, and Common Crawl.* Github Repository. https://github.com/lc/gau

[5] MacIver, D., Hatfield-Dodds, Z., & Contributors, M. (2019). Hypothesis: A new approach to property-based testing. *Journal of Open Source Software*, *4*(43), 1891. https://doi.org/10.21105/joss.01891

[6] ProjectDiscovery. (2022). *Subfinder is a subdomain discovery tool that discovers valid subdomains for websites. Designed as a passive framework to be useful for bug bounties and safe for penetration testing.* Github Repository. https://github.com/projectdiscovery/subfinder.git

[7] Sharma, N. (2022, June). *How I found a Critical Bug in Instagram and Got 49500$ Bounty From Facebook*. Medium.Com. https://infosecwriteups.com/how-i-found-a-critical-bug-in-instagram-and-got-49500-bounty-from-facebook-626ff2c6a853

[8] Tian-yang, G., Yin-sheng, S., & You-yuan, F. (2010). Research on Software Security Testing. *World Academy of Science, Engineering and Technology 70*.

[9] Tomnomnom. (2022). *Fetch all the URLs that the Wayback Machine knows about for a domain*. Github Repository. https://github.com/tomnomnom/waybackurls.git

Faiz Hanafi

M Universitas Ary Ginanjar
Fakultas Ilmu Komputer
Menara 165, Cilandak Timur, Pasar Minggu, Jakarta Selatan

Andi Purnomo

M Universitas Ary Ginanjar
Fakultas Ilmu Komputer
Menara 165, Cilandak Timur, Pasar Minggu, Jakarta Selatan