PENGUJIAN KEAMANAN PERANGKAT LUNAK PADA FAKTOR BROKEN AUTHENTICATION DAN SECURITY MISCONFIGURATION

Disusun Oleh:

N a m a : Heldi Saputera

NIM : 13523161

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2018

ii

HALAMAN PENGESAHAN DOSEN PEMBIMBING

PENGUJIAN KEAMANAN PERANGKAT LUNAK PADA FAKTOR BROKEN AUTHENTICATION DAN SECURITY MISCONFIGURATION

TUGAS AKHIR

Disusun Oleh:

N a m a : Heldi Saputera

NIM : 13523161

Yogyakarta, 4 Agustus 2018

Pembimbing,

(Hanson Prihantoro Putro S.T., M.T.)

iii

HALAMAN PENGESAHAN DOSEN PENGUJI

PENGUJIAN KEAMANAN PERANGKAT LUNAK PADA FAKTOR BROKEN AUTHENTICATION DAN SECURITY MISCONFIGURATION

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik Informatika
di Fakultas Teknologi Industri Universitas Islam Indonesia
Yogyakarta, 1 Nopember 2017

Tim Penguji

Hanson Prihantoro Putro S.T., M.T.

Anggota 1

Hendrik S.T., M.Eng.

Anggota 2

Mukhammad A Setiawan S.T., M.Sc., Ph.D.

Mengetahui,
Ketua Program Studi Teknik Informatika – Program Sarjana
Fakultas Teknologi Industri
Universitas Islam Indonesia

UNIVERSITAS ISLAM INDONESIA ★ YOGYAKARTA ★ FAKULTAS TEKNOLOGI INDUSTRI

(Raden Teduh Dirgahayu S.T., M.Sc., Ph.D.)

iv

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Heldi Saputera

NIM : 13523161

Tugas akhir dengan judul:

PENGUJIAN KEAMANAN PERANGKAT LUNAK PADA FAKTOR BROKEN AUTHENTICATION DAN SECURITY MISCONFIGURATION

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

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

Yogyakarta, 7 Agustus 2018

(Heldi Saputera)

HALAMAN PERSEMBAHAN

Dengan mengucap syukur *Alhamdulillah,* saya persembahkan karya ini untuk orang-orang yang saya cintai:

Kedua orang tua tercinta,
H. Harja Suwita dan Hj. Ratna Fermadi

Yang telah membimbing serta mengajarkan ilmu sejak kecil, selalu mendoakan, memberikan nasehat serta semangat. Semoga dapat membuat orang tua bangga dengan prestasi kecil ini.

Adik,
Fikri Daffa Raharja

Yang selama ini selalu menemani dalam merasakan manis dan pahitnya hidup. Semoga tujuan kita membahagiakan orang tua dapat terwujud. *Aamiin.*

HALAMAN MOTO

"*(Karena sesungguhnya sesudah kesulitan itu) atau kesukaran itu (ada kelapangan) yakni kemudahan*"

(QS. Al-Insyirah: 5)

KATA PENGANTAR

Assalamu'alaikum warahmatullahi wabarakatuh.

Alhamdulillahirobbil'alamin, puji syukur kita panjatkan kehadirat Allah SWT yang telah melimpahkan rahmat dan hidayah-Nya kepada kita semua. Sholawat beserta salam kita haturkan kepada nabi kita Nabi Muhammad SAW yang telah mengantarkan kita dari zaman jahiliyah menuju zaman penuh ilmu pengetahuan sehingga penulis dapat menyelesaikan laporan Tugas Akhir yang berjudul "Pengujian Alur Proses Bisnis Menggunakan State Transition Diagram".

Laporan tugas akhir ini disusun sebagai salah satu syarat untuk menyelesaikan pendidikan Strata 1 Jurusan Teknik Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia. Selama menjalani penulisan dan pengerjaan tugas akhir, penulis menyadari bahwa semua ini tidak terlepas dari berbagai pihak yang membantu dan mendukung penulis, oleh sebab itu penulis ingin menyampaikan rasa terimakasih sebesar-besarnya kepada:

1. Bapak Fathul Wahid, S.T., M.Sc., Ph.D., selaku Rektor Universitas Islam Indonesia.
2. Bapak Prof. Dr. Hari Purnomo, M.T., selaku Dekan Fakultas Teknologi Industri Universitas Islam Indonesia.
3. Dr. Raden Teduh Dirgahayu S.T., M.Sc. selaku Ketua Prodi Teknik Informatika Fakultas Teknologi Industri Universitas Islam Indonesia.
4. Bapak Hanson Prihantoro Putro, S.T., M.T., selaku pembimbing tugas akhir yang telah meluangkan waktu, membagikan ilmu, serta membimbing penulis dalam menyelesaikan tugas akhir.
5. Bapak dan Ibu dosen Jurusan Teknik Informatika yang telah memberikan ilmu yang bermanfaat kepada penulis.
6. Orang tua tercinta yang selalu penulis doakan yaitu bapak H. Harja Suwita dan Ibu Hj. Ratna Fermadi, serta adik Fikri Daffa Raharja atas dukungan materil maupun nonmateril kepada penulis.
7. Teman-teman "JJW", Jessica Noviana Raesita Putri, Alfandya, Anugrah Syauqi Yanuar, Aulia Ahmad Urfan Setya Putra, Bryan Yudho Haryono, Dodi Ahmad Shahrudin, Fikri Abdillah Fakhrudin, Gita Batari Hermayanthi, Haifa Azizah Utaryanto, Iga Umari Hanami, Muhammad Hafiz Siddiq, Nadya Aulia Oktavianti, Novendra Yoga Saputra, Ridho Akbar Dermawan yang selalu

viii

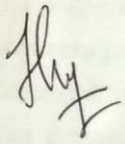
menemani, memberikan semangat serta menghiasi kehidupan perkuliahan penulis.

8. Teman-teman tim developer sistem KPTA FTI UII, Mas Yuridi, Aziz dan dan Luthfi yang selalu mendukung serta menemani penulis selama melakukan penelitian.
9. Teman-teman Teknik Informatika 13 "Eternity" yang selalu membantu, menemani serta membagikan ilmunya kepada penulis.
10. Teman-teman "Central Language Improvement" yang selalu memberikan semangat dan dukungan kepada penulis selama melakukan penelitian
11. Semua pihak yang telah membantu yang tidak dapat penulis tuliskan namanya satu persatu.

Penulis menyadari bahwa laporan tugas akhir ini jauh dari kata sempurna. Untuk itu penulis sangat menerima adanya kritik dan saran yang membangun untuk dapat dijadikan bahan evaluasi agar lebih baik kedepannya. Semoga laporan tugas akhir dapat memberikan manfaat bagi penulis serta orang lain. *Aamiin.*

Wassalamu'alaikum warahmatullahi wabarakatuh.

Yogyakarta, 4 Agustus 2018



(Heldi Saputera)

SARI

Pengujian perangkat lunak adalah proses mengeksekusi program atau aplikasi dengan maksud untuk menemukan *bug* dari suatu perangkat lunak yang dibuat. Pada penelitian ini, perangkat lunak akan diuji coba apakah sistem yang ada pada perangkat lunak berjalan dengan baik atau masih ditemukannya masalah agar nantinya perangkat lunak siap digunakan tanpa ada masalah. Penelitian ini mengambil studi kasus system KPTA FTI UII.

Secara umum pengujian keamanan perangkat lunak digunakan untuk mengecek keamanan sistem dan penanggulangan resiko yang akan terjadi sewaktu – waktu pada sistem. Pengujian ini mengikuti panduan OWASP *top* 10-2017. OWASP memiliki daftar 10 *web application security risks* pada tahun 2017. Dengan daftar tersebut, penguji akan mengkaji daftar apa saja yang cocok untuk mengecek kemanan pada sistem. Pada studi kasus Sistem KPTA FTI UII terdapat informasi dan status pengguna dalam melakukan tugas akhir. Data tersebut sangat penting dan harus dilindungi. Sebagai contoh profil *user* yang sudah terdaftar, informasi tersebut harus dilindungi seperti pemeriksaan keamanan jika ada aktivitas *login* dari perangkat lain dengan mengirimkan *email* verifikasi kemanan apakah yang melakukan *login* adalah pengguna asli atau orang lain.

Penelitian ini berhasil menemukan cara bagaimana melakukan pengujian pada faktor *broken authentication* dan *security misconfigurration*. Pengujian keamanan perangkat lunak pada faktor *broken authentication* dan *security misconfiguration* berhasil dilakukan dengan tepat sehingga sistem menjadi jauh lebih aman daripada sebelum dilakukan pengujian. Sebelum melakukan pengujian sistem KPTA FTI UII banyak ditemukan kesalahan pada sistem keamanan yang berakibat fatal jika celah tersebut diketahui oleh banyak orang. Dengan OWASP pengujian sistem KPTA FTI UII lebih spesifik dalam melakukan pengujian keamanan. Pada pengujian broken authentication ditemukan lima tahap pengujian yakni *insecure login login form, logout management, password attack, weak password* dan akses *view*. Pada pengujian *security misconfiguration* ditemukan dua tahap pengujian yakni *denial of service* dan pengecekan setiap *path* pada sistem.

Kata kunci: *black box testing, OWASP,* keamanan perangkat lunak, *security testing*

GLOSARIUM

Scenario test	Skenario yang dirancang untuk pengujian sistem
Re-engineering	Proses mempelajari ulang sistem secara keseluruhan
OWASP	Organisasi yang bergerak dalam keamanan perangkat lunak.
Broken authentication	Kejadian sebuah sistem mengizinkan prosedur yang tidak seharusnya dilakukan.
Security Misconfiguration	Terjadinya suatu kesalahan pada sistem dikarenakan adanya kode program yang salah.

DAFTAR ISI

HALAMAN JUDUL i
HALAMAN PENGESAHAN DOSEN PEMBIMBING ii
HALAMAN PENGESAHAN DOSEN PENGUJI iii
HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR iv
HALAMAN PERSEMBAHAN v
HALAMAN MOTO vi
KATA PENGANTAR vii
SARI ix
GLOSARIUM x
DAFTAR ISI xi
DAFTAR TABEL xiii
DAFTAR GAMBAR xiv
BAB I PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 2
1.4 Tujuan Penelitian 2
1.5 Manfaat Penelitian 3
1.6 Metodologi Penelitian 3
1.7 Sistematika Penelitian 3
BAB II STUDI LITERATUR 5
2.1 Sistem KPTA FTI UII 5
2.2 *Security Test* 6
2.3 OWASP 7
2.3.1 Broken Authentication 7
2.3.2 Security Misconfiguration 8
2.4 Studi Literatur 9
BAB III METODOLOGI 10
3.1 Proses *Re-engineering* 10
3.1.1 Use Case Diagram 10
3.1.2 Klasifikasi *Class* 12
3.2 Pembuatan *Scenario Test* 15

3.3 Pengujian Sistem 15
3.3.1 Pengujian Broken Authentication 17
3.3.2 Pengujian Security Misconfiguration 21
3.4 Perbaikan Sistem 22
BAB IV HASIL DAN PEMBAHASAN 23
4.1 Hasil Pengujian Broken Authentication Iterasi 1 23
4.2 Hasil Pengujian Security Misconfiguration Iterasi 1 31
4.3 Perbaikan Dari Iterasi 1 36
4.4 Hasil Pengujian Iterasi 2 38
BAB V KESIMPULAN 40
5.1 Kesimpulan 40
5.2 Saran 40
DAFTAR PUSTAKA 41
LAMPIRAN 42

DAFTAR TABEL

Tabel 3.1 Klasifikasi Sistem KPTA.....12
Tabel 3.2 Contoh *Scenario Test*.....15
Tabel 3.3 Tabel Pengujian Sistem 16
Tabel 4.1 Daftar Kemungkinan *Username* dan *Password*.....27
Tabel 4.2 Hasil Pengujian Akses *View* 29
Tabel 4.3 Hasil Pengujian Pengecekan *Path* 33
Tabel 4.4 Hasil Pengujian Iterasi 1 35
Tabel 4.5 Hasil Pengujian Iterasi 2 38

DAFTAR GAMBAR

Gambar 2.1 Halaman Awal Sistem KTPA FTI UII 5
Gambar 2.2 Macam-Macam Security Testing 6
Gambar 2.3 Top 10 The Most Critical Web Application Security Risks.....7
Gambar 3.1 Use Case Diagram Sistem KPTA FTI UII 11
Gambar 3.2 Proses Memasukkan Kode *Payload* Pada Form *Login* 18
Gambar 3.3 Membuat Daftar Kemungkinan *Username* Pada Aplikasi Burp Suite.....19
Gambar 3.4 Membuat Daftar Kemungkinan *Password* Pada Aplikasi Burp Suite 20
Gambar 4.1 Kode Page Source Pada Halaman *Login*.....23
Gambar 4.2 Halaman Awal Setelah *Login* Berhasil.....24
Gambar 4.3 Proses *Logout* Berhasil.....25
Gambar 4.4 Aplikasi Burp Suite Menampilkan *Username* dan *Password* sebelumnya.....26
Gambar 4.5 Membuat Daftar *Payload* untuk *username*.....26
Gambar 4.6 Membuat Daftar *Payload* untuk *password*.....27
Gambar 4.7 Hasil Penetrasi.....28
Gambar 4.8 Hasil Pengujian Akses *View*.....31
Gambar 4.9 Status Penyerangan DoS Terhadap Sistem 32
Gambar 4.10 Sistem Tidak Bisa Diakses.....33
Gambar 4.11 Hasil Pengujian Akses *Path* 35
Gambar 4.12 Perbaikan Kode Program Untuk Fungsi *Password* Berhasil.....37

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pengujian perangkat lunak adalah proses mengeksekusi program atau aplikasi dengan maksud untuk menemukan *bug* dari suatu perangkat lunak yang dibuat (Suhartono, 2016). Pada proses pengujian, perangkat lunak akan diuji coba apakah sistem yang ada pada perangkat lunak berjalan dengan baik atau masih ditemukannya masalah agar nantinya perangkat lunak siap digunakan tanpa ada masalah.

Pengujian perangkat lunak tidak selalu hanya menguji apakah perangkat lunak berjalan dengan baik atau belum, tetapi pengujian perangkat lunak juga bisa digunakan terhadap fungsionalitas, alur bisnis, *database*, dan lain – lain. Tujuan dari pengujian pada suatu perangkat lunak adalah menguji kualitas pada perangkat lunak agar diketahui apakah perangkat lunak sudah siap digunakan. Para *developer* kebanyakan melakukan pengujian hanya sebatas mengecek fungsionalitas dari sebuah perangkat lunak saja, tetapi keamanan dari sebuah perangkat lunak jarang diperhatikan.

Metode pengujian perangkat lunak yang sering dijumpai ada dua yaitu metode *white box* dan *black box*. Metode *white box* adalah metode untuk memeriksa apakah kode program bekerja seperti yang diharapkan (Williams, 2006) sedangkan metode *black box* adalah metode pengujian yang berfokus pada spesifikasi fungsional dari perangkat lunak, *tester* dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program (Mustaqbal, Firdaus, & Rahmadi, 2015). Tujuan dari pengujian pada suatu perangkat lunak adalah menguji kualitas pada perangkat lunak agar diketahui apakah perangkat lunak sudah siap digunakan. Penelitian ini berfokus pada keamanan perangkat lunak menggunakan panduan *The Open Web Application Security Project* (OWASP).

Secara umum pengujian keamanan perangkat lunak digunakan untuk mengecek keamanan sistem dan penanggulangan resiko yang akan terjadi sewaktu – waktu pada sistem. Pengujian ini mengikuti panduan OWASP *top* 10-2017. OWASP memiliki daftar 10 *web application security risks* pada tahun 2017, dengan daftar tersebut, penguji akan mengkaji daftar apa saja yang cocok untuk mengecek kemanan pada sistem. Pada studi kasus Sistem KPTA FTI UII terdapat informasi dan status pengguna dalam melakukan tugas akhir. Data tersebut sangat penting dan harus dilindungi. Sebagai contoh profil *user* yang sudah terdaftar, informasi tersebut harus dilindungi seperti pemeriksaan keamanan jika ada aktivitas *login*

dari perangkat lain dengan mengirimkan *email* verifikasi kemanan apakah yang melakukan *login* adalah pengguna asli atau orang lain.

Tidak hanya dalam melindungi informasi yang ada di dalam perangkat lunak, hal lain yang harus diuji adalah bagaimana sistem pada perangkat lunak melakukan manajemen insiden pada perangkat lunak jika ditemukannya kendala nanti seperti contoh pengguna tidak bisa melakukan proses *login* padahal *username* dan *password* yang dimasukkan sudah benar atau permasalahan *backup* data jika sewaktu-waktu data hilang.

Manfaat dilakukannya pengujian keamanan ini adalah untuk mengetahui sebaik apa sistem melindungi data pengguna, minimalisir kesalahan eksekusi dari sistem, serta mengurangi resiko *error* pada sistem. Dengan begitu, perangkat lunak bisa digunakan dengan aman. Oleh karena itu, dalam penelitian ini akan dilakukan pengujian keamanan perangkat lunak denga sn panduan OWASP pada sistem KPTA FTI UII. Penelitian ini dibuat untuk memastikan perangkat lunak ini aman digunakan.

1.2 Rumusan Masalah

Rumusan masalah pada penelitian ini adalah bagaimana melakukan pengujian keamanan perangkat lunak pada faktor broken authentication dan security misconfiguration ?

1.3 Batasan Masalah

Terdapat beberapa batasan masalah dalam penelitian ini yaitu sebagai berikut:

a. Penelitian ini dilakukan terhadap sistem KPTA FTI UII.
b. Pengujian yang dilakukan menggunakan OWASP sebagai metode penelitian pada perangkat lunak.
c. Metode OWASP yang digunakan hanya Broken Authentication dan Security Misconfiguration.
d. Teknik pengujian perangkat lunak yang diterapkan menggunakan teknik dasar.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah:

a. Untuk mengetahui sejauh mana tingkat kemanan pada sistem KPTA FTI UII apakah sudah aman atau belum.
b. Untuk mengetahui bagaimana cara menguji keamanan perangkat lunak pada sistem KPTA FTI UII menggunakan OWASP.

1.5 Manfaat Penelitian

Penelitian ini diharapkan memberi manfaat sebagai berikut:

a. Bagi peneliti

   Menambah ilmu dan pengalaman penulis dalam melakukan penelitian terhadap pengujian perangkat lunak.

b. Bagi pengelola sistem

   Bisa mengetahui seberapa baik keamanan sistem yang sudah diterapkan pada sistem KPTA FTI UII.

c. Bagi pengguna

   Dapat menggunakan perangkat lunak yang telah melalui proses pengujian sedemikian rupa sehingga mengurangi tingkat kesalahan dalam penggunaan.

1.6 Metodologi Penelitian

Metodologi yang dilakukan dalam penelitian ini adalah:

a. Studi Literatur

   Studi literatur dilakukan melalui jurnal, buku, internet dan sumber lain yang terkait dengan penelitian ini. Metode ini juga digunakan sebagai referensi landasan teori.

b. Melakukan proses *re-engineering*

   Proses *reengineering* ini dilakukan untuk mempelajari struktur aplikasi yang akan dijui nantinya kemudian menentukan metode OWASP apa yang akan dipakai.

c. Membuat *Scenario Test*

   Membuat skenario pengetesan yang akan dilakukan pada saat pengujian nanti, *scenario test* menjadi acuan untuk pengujian sistem nanti.

d. Pengujian Sistem

   Dari hasil membuat *scenario test* kemudian akan diujikan pada sistem. Dalam tahapan ini dapat diketahui apakah sistem sudah sesuai dengan panduan yang dibuat oleh OWASP.

e. Perbaikan Sistem

   Melakukan perbaikan sistem apabila terdapat kesalahan pada sistem.

1.7 Sistematika Penelitian

Sistematika yang digunakan adalah sebagai berikut:

BAB I PENDAHULUAN

Bab ini menjelaskan tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian dan sistematika penulisan.

BAB II LANDASAN TEORI

Memuat tentang dasar dan teori yang berkaitan dengan topik penelitian serta hal-hal yang nantinya akan digunakan dalam proses pengujian perangkat lunak.

BAB III METODOLOGI

Menjelaskan tentang langkah-langkah penyelesaian masalah yang ada pada penelitian. Seperti melakukan proses *re-engineering*, membuat *scenario test*, tahap pengujian, serta perbaikan sistem pada sistem KPTA.

BAB IV HASIL DAN PEMBAHASAN

Berisi tentang hasil serta pembahasan dari pengujian sistem KPTA. Disini akan dibahas hasil dari uji coba serta hasil dari perbaikan sistem.

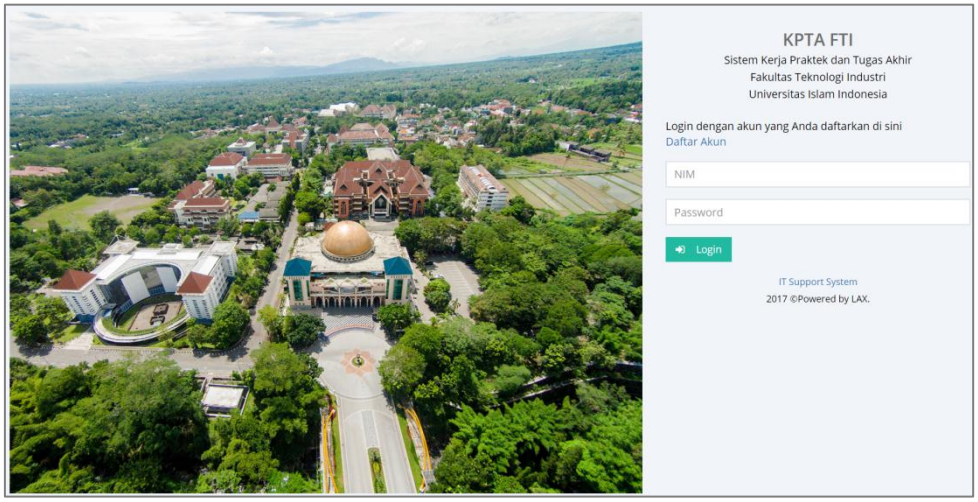
BAB V KESIMPULAN DAN SARAN

Bab ini menjelaskan tentang kesimpulan dan saran untuk perbaikan dan pengembangan sistem berikutnya.

BAB II
LANDASAN TEORI

2.1 Sistem KPTA FTI UII

Sistem KPTA FTI UII adalah sebuah sistem yang dibuat untuk menangani permasalahan tugas akhir. Sistem ini dibuat untuk mempermudah mahasiswa dalam mengatur proses tugas akhir seperti pendaftaran, pengajuan tugas akhir, pendaftaran *progress*, dan ujian pendadaran. Sistem ini telah aktif semenjak Juni 2017. Sistem KPTA FTI UII baru aktif dan digunakan oleh jurusan Teknik Informatika UII (Laksita, 2016). Gambar 2.1 menunjukkan halaman awal sistem KPTA FTI UII.



Gambar 2.1 Halaman Awal Sistem KTPA FTI UII

Gambar 2.1 adalah halaman *login* sistem KPTA FTI UII. Halaman ini digunakan untuk *login* oleh semua pengguna sistem yaitu mahasiswa, dosen, staf jurusan dan staf akademik. Saat ini efektifnya sistem ini hanya digunakan sampai tugas akhir terdaftar yaitu hanya dari daftar akun Sistem KPTA FTI UII. Setelah daftar akun, mahasiswa dapat langsung melakukan submit proposal tugas akhir yang sebelumnya sudah didiskusikan dengan calon dosen pembimbing. Setelah proposal terdaftar, nantinya hanya tinggal menunggu hasil persetujuan dari rapat dosen. Ketika disetujui, maka akan ditampilkan pada sistem. Jika tidak, maka mahasiswa harus melakukan submit ulang proposal baru. Selebihnya seperti progress dan ujian pendadaran belum bisa ditangani oleh sistem. Sistem KPTA FTI UII masih dalam

tahap pengembangan. Oleh karena itu, sistem ini dibutuhkan pengujian keamanan untuk mengetahui sebaik apa pengembang menerapkan keamanan pada sistem KPTA FTI UII.

2.2 *Security Test*

Security test adalah uji coba yang digunakan untuk mengetahui seberapa aman kah sebuah aplikasi web dalam menerapkan pengamanan sistem. Namun, *security testing* tidak menjamin bahwa aplikasi yang diuji coba aman tetapi hal ini penting dilakukan untuk melihat kemampuan sistem seberapa baik dalam hal perlindungan. Secara umum, *security test* terbagi dalam beberapa jenis sesuai dengan metodologi yang digunakan (Sofyan, 2014). Gambar 2.2 menampilkan macam-macam *security testing*.

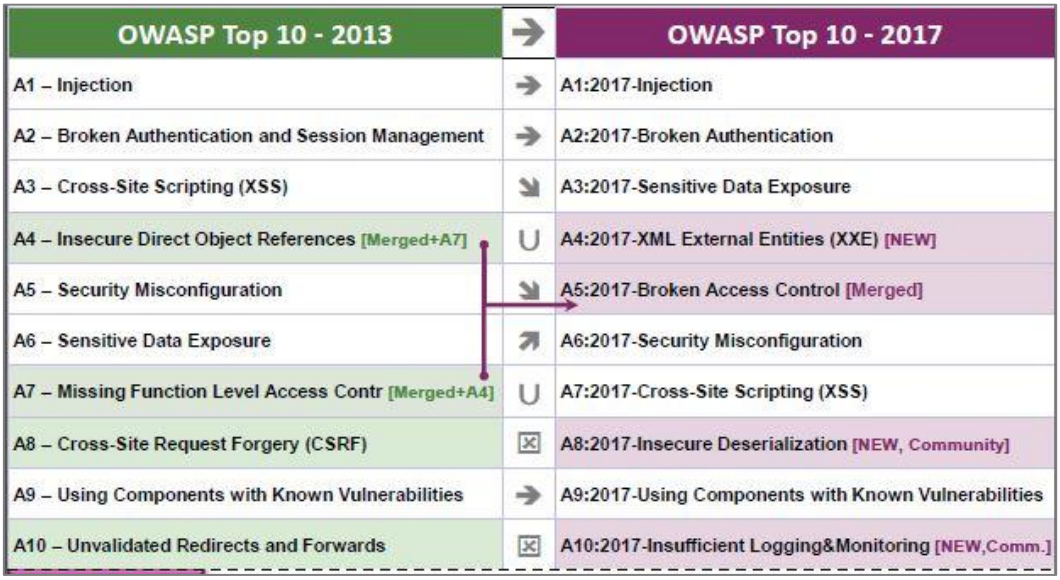
Gambar 2.2 Macam-Macam Security Testing

Sumber : (Sofyan, 2014)

Pada gambar 2.2, terdapat tujuh macam *security testing* sesuai dengan metodologi yang digunakan pada pengujian, contoh pada penelitian ini masuk dalam kategori *penetration testing* karena difokuskan untuk mensimulasi sebuah serangan terhadap aplikasi web dari serangan peretas. Dalam pengujiannya dibutuhkan metodologi tertentu untuk memeriksa sistem apakah bisa menangani serangan dan mengetahui potensi sebuah sistem untuk menanggulangi serangan dari luar. Pada pengujian ini digunakan OWASP sebagai acuan untuk melakukan pengujian keamanan pada sistem KPTA FTI UII.

2.3 OWASP

OWASP adalah sebuah organisasi yang bergerak dalam pengelolaan perangkat lunak khususnya dalam keamanan perangkat lunak. OWASP setiap tahunnya merilis *top* 10 *the most critical web application security risks.* Di dalamnya, OWASP membeberkan 10 permasalahan yang paling sering dijumpai dalam keamanan aplikasi web. Tidak hanya itu, OWASP juga menjelaskan mengapa hal itu bisa terjadi, resiko apa yang akan didapat, serta bagaimana cara mencegahnya. (OWASP, 2017). Gambar 2.3 menunjukkan daftar top 10 OWASP yang berisi daftar *top 10* OWASP pada tahun 2013 dan 2017.



Gambar 2.3 Top 10 The Most Critical Web Application Security Risks

Pada gambar 2.3 terdapat tabel OWASP *top 10* pada tahun 2013 dan 2017. Terdapat perbedaan seperti adanya daftar baru yang masuk pada tahun 2017 yaitu *XML External Entities* (XXE), kemudian adanya gabungan dua daftar yang ada pada 2013 menjadi *Broken Access control* pada tahun 2017.

2.3.1 Broken Authentication

Halaman *login* adalah bagian yang penting dalam sebuah aplikasi *web*. Karena pada halaman ini pengguna harus mengakses *username* dan *password* mereka untuk bisa mengakses sistem yang ada pada aplikasi *web*. Namun, halaman ini bisa menjadi ancaman jika pengamanannya kurang karena hal ini bisa menjadi pemicu bocornya informasi

pengguna contohnya. *Broken authentication* adalah suatu kejadian dimana sebuah sistem mengizinkan suatu prosedur yang tidak seharusnya dilakukan. Sebagai contoh pada saat pengguna melakukan akses *login* pada suatu aplikasi *web*. Pada saat pengguna berhasil *login*, informasi *username* dan *password* pengguna akan tersimpan pada *session id*, *session id* inilah yang bisa digunakan oleh peretas untuk mendapatkan informasi terkait *username* dan *password* pengguna aplikasi *web* untuk bisa masuk ke sistem dengan cara melakukan berbagai macam cara dengan tujuan untuk mengambil informasi pengguna dan lain sebagainya (OWASP, 2017). Pada pengujiannya, terdapat enam tahap pengujian yaitu *CAPTCHA bypassing, forgotten function, insecure login forms, logout management, password attacks, weak passwords* yang pada penelitian ini akan digunakan empat tahapan yaitu *insecure login forms, logout management, password attacks, weak passwords* ditambah satu tahap yakni mengkases *view* pada sistem tanpa melalui proses autentikasi

2.3.2 Security Misconfiguration

Security misconfiguration adalah terjadinya suatu kesalahan pada sistem aplikasi web dikarenakan adanya kode program yang salah ataupun sistem melakukan proses yang tidak perlu dilakukan. Hal ini membuat peretas bisa melakukan penetrasi kepada aplikasi web dengan cara menyisipkan virus, membuat url palsu untuk mencari *hidden* url, dan lain sebagainya. Tidak hanya itu, permasalahan ini juga kerap terjadi pada pengamanan yang kurang pada bagian bagian kecil pada sistem yang tidak terpikirkan sebelumnya bagi pengembang contohnya pada manajemen *session* (OWASP, 2017). Pada pengujiannya, terdapat 17 tahap yakni *arbitrary file access, cross-domain policy file, cross-origin resource sharing, cross ssite tracing, denial of service (large chucnk size), denial of service (slow http DoS), denial of service (SSL-exhaustion), denial of service (XML bomb), insecure ftp configuration, insecure SNMP configuration, insecure WebDAV configuration, local privilege escalation (sendpage), local privilege escalation (udev), man-in-the-middle attack* (HTTP), *man-in-the-middle attack* (SMTP), *old/backup & unreferenced files*, dan *robots file* yang pada penilitian ini hanya digunakan satu tahapan yaitu *denial of service (large chunck size)* ditambah tahapan untuk cek *path* sistem.

2.4 Studi Literatur

Security test pada aplikasi biasanya digunakan untuk melihat sejauh mana penerapan keamanan pada sistem. Dengan dilakukannya *security test,* hasil yang didapatkan bisa menjadi gambaran aman atau tidaknya sebuah aplikasi. Selama penulis melakukan studi literatur, ditemukan beberapa penelitian yang menggunakan OWASP sebagai acuannya.

Pada penelitian Analisis Celah Keamanan Manajemen Sesi terhadap Serangan *Session Hijacking* pada Web Aplikasi, penelitian ini menyertakan sumber dari OWASP dengan ditarik kesimpulan bahwa dalam melakukan uji coba keamanan pada suatu aplikasi, kita harus mengetahui secara spesifik pada bagian mana yang sering menjadi titik kelemahan pada suatu sistem, dengan itu pengujian jauh lebih efisien daripada harus menguji keseluruhan sistem pada aplikasi (Maulana, 2017).

OWASP juga diterapkan untuk menjadi pemetaan IoT attack surface areas. IoT masih banyak memiliki kendala pada hal keamanan seperti pada bagian autentikasi dan *update* sistem. Dalam hal ini, pengujian keamanannya membutuhkan metodologi yang tepat untuk memetakan permasalahannya, maka dari itu penelitian ini menggunakan OWASP sebagai metodologi untuk pengujiannya (Miessler, 2015).

Kemudian pada penelitian *Developing a Secure Web Application* yang dilakukan oleh Khairul Anwar Sedek juga menggunakan OWASP sebagai metodo*login*ya. Di dalam penelitiannya, peneliti menyebut bahwa para pengembang aplikasi web memerlukan sebuah pedoman untuk melakukan pengujian keamanan. Pedoman ini bisa digunakan untuk mempermudah para pengembang dalam mendapatkan standar capaian minimum dalam aplikasi web yang aman (Sedek, 2009).

Pada penelitian-penelitian sebelumnya, belum ada yang menjelaskan bagaimana pengujian keamanan dengan OWASP sebagai metode pengujian sistem informasi dan bagaimana cara melakukan pengujiannya. OWASP yang akan digunakan sebagai metode pengujian keamanan nantinya akan dijadikan suatu standar dalam melakukan pengujian keamanan khususnya pada faktor broken authentication dan security misconfiguration. Penelitian ini berfokus bagaimana cara melakukan pengujian keamanan pada sistem dan menjelaskan tahapan-tahapan yang dilakukan pada proses pengujiannya.

BAB III
METODOLOGI

3.1 Proses *Re-engineering*

Proses *re-engineering* adalah proses mempelajari sistem secara keseluruhan. Proses ini dilakukan karena sistem ini sudah dibuat dan masih dalam proses pengembangan lebih lanjut. Diperlukannya proses *re-engineering* ini adalah mempelajari struktur dari sistem KPTA FTI UII agar nanti pada saat melakukan proses pengujian menjadi lebih mudah karena sudah mempelajarinya terlebih dahulu.

Proses ini difokuskan untuk mengklasifikasi *class function* yang ada pada aplikasi KPTA, kemudian mengecek setiap fungsi pada setiap *class function* apakah pada pemrosesannya nanti membutuhkan autentikasi apa tidak. Proses *re-engineering* ini dilakukan pada sistem yang terakhir di-*update* pada tanggal 13 Maret 2018.

3.1.1 Use Case Diagram

Use case merupakan bagian dari UML (*Unified Modeling Language)* yang fungsinya untuk menggambarkan siapa saja yang terlibat untuk menggunakan sistem ini dan apa yang bisa pengguna lakukan. Secara umum, use case terdiri dari aktor, peran apa saja yang ada pada sistem, dan peran apa yang bisa dipakai oleh aktor yang terlibat dalam penggunaan sistem. Namun, pada penerapannya terkadang sebuah peran pada diagram ini bisa menjadi sebuah pemicu peran yang lain dikarenakan ada keterkaitan didalamnya.

Diagram ini menggambarkan empat aktor yang memiliki peran dalam mengakses aplikasi KPTA yakni mahasiswa, dosen, jurusan, dan staff akademik. Dengan use case diagram ini, penguji bisa mengetahui siapa saja yang berperan dalam mengakses sistem dan peran apa saja yang bisa dilakukan oleh setiap perannya. Penggunaan use case dalam proses - *re-engineering* ini adalah agar orang lain bisa memahami sekilas tentang apa saja yang bisa dilakukan, siapa yang terlibat, dan apa yang bisa dilakukan pada peran yang terlibat dalam penggunaan sistem ini. Gambar 3.1 menunjukkan use case diagram untuk sistem KPTA FTI UII.

Gambar 3.1 Use Case Diagram Sistem KPTA FTI UII

Sumber : (Kurniawan, 2018)

Gambar 3.1 menjelaskan use case diagram pada sistem KPTA FTI UII. Pada *use case ini*, digambarkan peran apa saja yang bisa dilakukan oleh tiap aktor. Use case diagram pada aktor mahasiswa memiliki beberapa fungsionalitas yang pada penerapannya di sistem KPTA nanti, aktor mahasiswa hanya bisa mengakses fungsi yang ada pada diagram.

Use case diagram pada aktor staf jurusan hanya memiliki empat fungsionalitas, namun perannya hampir mirip dengan aktor dosen yakni pada manajemen proposal. Fungsi manajemen proposal pada aktor staf jurusan dan dosen dapat melihat, update, menyetujui, atau menolak proposal yang diajukan oleh mahasiswa. Pada use case diatas tidak adanya aktor admin, namun aktor dosen dan staf jurusan pada use case diatas sekaligus menjadi admin karena fungsionalitasnya.

3.1.2 Klasifikasi *Class*

Sistem ini menggunakan *framework* Code Igniter (CI) dalam pembuatannya. *Framework* sendiri menggunakan konsep MVC (*Model, View, Controller*) dalam menerapkan lingkungan pengembangannya. Dengan MVC, pembuatan sistem menjadi lebih terstruktur dan lebih sederhana karena karena MVC memisahkan pembuatan logika pada sistem dengan pembuatan tampilan pada sistem. Pada proses ini, MVC pada sistem KPTA akan diklasifikasi menurut sesuai dengan hubungan antar *controller*, *view*, dan *modelnya.* Tabel 3.1 akan menampilkan hasil klasifikasi sistem KPTA.

Tabel 3.1 Klasifikasi Sistem KPTA

CONTROLLER	**METHOD**	***VIEW***	**MODEL**
Berita.php	Index(); Info(); Baru(); Baru_submit(); Kelola(); Hapus();	Panel; info; form; mnj;	M_user; M_berita;
Data.php	Index(); Ta(); Status(); Rekap_bimbingan(); Ta_diterima(); Lambatta(); Data_mahasiswa(); Data_alumni(); Bimbingan(); Bimbingan_ta_dosen();	Menu_ta; List; List_status; Chart_bimbingan_dosen; Chart_bimbingan_dari_bidmin; Chart_bimbingan_dari_staffjur; Filter_diterima; List_terlambat; Data_mahasiswa; Data_alumni; List_bimbingan; List_bimbingan2;	M_user; M_tugasakhir; M_kerja; M_konsestrasi; M_izin;
Dosen.php	Index(); Baru(); Baru_validate(); Baru_submit(); Kelola(); Ubah(); Ubah_info_validation(); Valid_telephone(); Ubah_submit();	List; Form_baru; Mnj; Mnj; Form_ubah;	M_user; M_dosen; M_izin; M_konsentrasi;
Jurusan.php	Index(); Kelola(); Ubah(); Ubah_submit(); Baru(); Baru_validate();	List; Form_ubah; Form_baru; Form_ubah_list;	M_user; M_jurusan; M_konsentrasi;

CONTROLLER	METHOD	*VIEW*	MODEL
	Baru_submit; Hapus_konsentrasi(); Delete();		
Karyawan.php	Index(); Kelola(); Ubah(); Ubah_validate(); Ubah_submit(); Baru(); Baru_validate(); Baru_submit();	List; Form_ubah; Form_baru;	M_user; M_karyawan;
Kerja.php	Mahasiswa(); Ta(); Seluruh_ta(); Daftarkp(); Daftarta(); Cek_judul_ascii(); Daftarta_tampil_rekomendasi(); Daftar_rekomendasi(); Baru_validate(); File_perjanjian_validate(); File_lampiran(); Daftar_ta_submit(); Revisi(); Revisi_submit(); Revisi_hapus(); Daftar(); Progress(); Sidang(); Revisi_akhir(); Status(); Input_nilai_akhir(); Input_nilai_akhir_submit(); Status_diterima_array(); Status_diterima(); Status_ditolak(); Ditolak_validate(); Status_ditolak_submit(); Tertolak(); Konsentrasi(); Info(); Info_submit(); Ubah(); Info_kerja_submit(); Detilstatus(); Lihat1(); Lihat(); Lihat_ditolak(); Bumbingan(); Kelola();	Blank; Menu_ta; Menu_seluruhnya; Form_daftar; Form_daftar_pilih_konsentrasi; Form_daftar; Form_revisi; List_utk_status; Upload;	M_kerja; M_status; M_dosen; M_user; M_mahasiswa; M_rekomendasi_konsentrasi; M_tugasakhir; M_kerja_status; M_kerja_user; M_ujian;

CONTROLLER	**METHOD**	***VIEW***	**MODEL**
Laporan.php	Kelola(); Baru(); Baru_submit(); Ubah(); Ubah_submit(); Hapus();	List; Form; Baru; Form_ubah; Kelola;	M_user; M_laporan;
Menu.php	Kelola(); Matrix(); Kelola_submit(); Ubah(); Ubah_submit(); Hapus();	List; Form; Kelola; Baru; Form_ubah;	M_menu; M_role; M_user;
User.php	*Login*(); Lagin_validate(); Sso_auth(); *Login*_submit(); Ubahpwd(); Ubahpwd_validate(); Ubahpwd_submit(); Info(); Ubah_info_validation(); Valid_telephone(); Kelola(); Ubah(); Ubah_validate(); Ubah_submit(); Ubah_karyawan(); Ubah_dosen(); Ubah_mahasiswa(); Monitoring_akun(); Daftar_users(); Pindah_akun(); Pindah_akun_auth(); Log_out();	Form_*login*; Form_pwd; Info_dsn; Mnj; Form_ubah; Menu_monitoring_akun; Daftar_user_*login*	M_user; M_menu M_mahasiswa; M_karyawan; M_dosen; M_izin; M_konsesntrasi;

Tabel 3.1 berisi *controller*, method, *view*, dan model yang digunakan pada sistem KPTA FTI UII. Klasifikasi ini dilakukan untuk mempermudah penguji untuk mempelajari sistem sehingga pada saat pengujian jika ditemukan *error*, penguji bisa mengetahui dimana letak kesalahan pada sistem lalu memperbaikinya. Pada kolom *controller*, terdapat delapan *controller* yang menangani sistem KPTA FTI UIII. Di dalam masing-masing *controller*, terdapat method yang menjalankan fungsi dari sistem. Method pada tiap dijabarkan pada kolom terpisah.

Pada kolom *view* berisi *view* yang digunakan pada *controller* terkait. Fungsi dari *view* adalah menampilkan halaman sistem yang nantinya akan diakses oleh pengguna. Pada kolom

model terdapat daftar database, fungsi dari model adalah media penyimpanan data pada sistem yang terkait oleh *controller*.

3.2 Pembuatan *Scenario Test*

Pada pembuatan *scenario test* ini dilakukan untuk membuat rancangan awal untuk tahapan pengujian sistem. Tabel *scenario test* ini berisi apa saja yang akan diuji pada sistem, kemudian tujuan pengujian, dan hasil pengujian. *Sceario test* digunakan untuk menjelaskan apa yang akan dilakukan pada pengujian, menjelaskan secara singkat tujuan dari pengujian, serta memudahkan penguji untuk memudahkan dalam melaporkan hasil dari pengujian. Tabel 3.2 menampilkan contoh tabel yang akan digunakan untuk *scenario test*.

Tabel 3.2 Contoh Tabel *Scenario Test*

ID	DESKRIPSI PENGUJIAN	TUJUAN	HASIL	STATUS

Tabel 3.2 merupakan contoh tabel *scenario test* yang akan menjadi alat untuk mencatat hasil keseluruhan dari pengujian. Pada kolom id diisi oleh id setiap pengujian, kode BR ditujukan untuk pengujian Broken Authentication, sedangkan kode SM ditujukan untuk pengujian *Security Misconfiguraton*. Deskripsi pengujian merupakan penjelasan singkat terkait pengujian yang dilakukan. Kolom tujuan berisi alasan singkat terkait dengan pengujian, kemudian status berisi hasil dari pengujian. Status dinyatakan aman ketika pengujian yang dilakukan tidak berhasil sedangkan status dinyatakan tidak aman ketika pengujian yang dilakukan berhasil.

3.3 Pengujian Sistem

Setelah dibuat *scenario test,* maka sistem akan dilakukan pengujian. Pada pengujian ini akan dilakukan dengan bantuan aplikasi dan juga akan diberikan gambaran bagaimana penerapannya pada saat dilakukan pengujian. Pada pengujian sistem ini akan dicatat hasil yang didapat. Jika pengujian yang dilakukan berhasil maka sistem dinyatakan memilki keamanan yang kurang dan jika pengujian gagal maka sistem dinyatakan aman. Pada

pengujian ini akan dilakukan lima tahapan penetrasi pada sistem yang terbagi menjadi lima pengujian untuk broken authentication dan dua pengujian untuk security misconfiguration.

Pada tabel 3.3 akan dijelaskan pengujian apa saja yang akan dilakukan, cara pengujian secara umum, tujuan pengujian, celah apa yang kemungkinan menjadi sasaran penetrasi, serta hasil dari pengujian. Fungsi dari pembuatan tabel ini adalah untuk menjelaskan gambaran umum terkait pengujian nantinya. Tabel 3.3 menampilkan gambaran umum hasil pengujian yang akan dilakukan

Tabel 3.3 Tabel Pengujian Sistem

ID	**DESKRIPSI PENGUJIAN**	**TUJUAN**	**CELAH**	**HASIL**	
				AMAN	**TIDAK AMAN**
BR-01	*Insecure login form* (pengecekan keamanan form login)	Mengecek keamanan form *login* dari boocornya informasi	*Page source* menampilkan data *username* dan *password* pengguna yang benar	Data *username* dan *password* pengguna tidak ditampilkan	Halaman *page source* menampilkan data *username* dan *password* pengguna
BR-02	*Logout management* (pengecekan proses *logout* pada sistem)	Mengecek keamanan proses *logout*	*Session* masih aktif	Sistem tidak bisa kemballi ke halaman sebelum proses *logout*	Sistem bisa kembali ke halaman sebelum *logout*
BR-03	*Password attacks (*melakukan pencarian *username* dan *password* pengguna*)*	Menyerang fungsi *password* untuk mencari data *username* dan *password* pengguna	*Password* yang sudah familiar digunakan	Kombinasi *username* dan *password* tidak ditemukan	Kombinasi *username* dan *password* ditemukan
BR-04	*Weak* password (pengecekan kombinasi *password)*	Mengecek kombinasi karakter pada *password*	Prosedur pemilihan kata pada *password*	Sistem tidak memproses password yang tidak sesuai dengan prosedur keamanan	Sistem memproses password yang tidak sesuai dengan prosedur keamanan

ID	DESKRIPSI PENGUJIAN	TUJUAN	CELAH	HASIL	
				AMAN	TIDAK AMAN
BR-05	Penngecekan *view* sistem tanpa melewati prosedur *login* ke sistem	Mengakses *view* pada sistem tanpa melalui prosedur *login*	Tidak adanya autentikasi pada sistem	Tidak ditemukan *view* yang bisa diakses tanpa prosedur *login*	ditemukan *view* yang bisa diakses tanpa prosedur *login*
SM-01	Denial of Service (melakukan prosedur DoS)	Mengetahui seberapa kuat sistem menghadapi serangan dari luar	*Port* atau *firewall* tidak berjalan dengan semestinya	Sistem berjalan normal ketika diakses	Sistem berjalan lebih lambat ketika diakses
SM-02	Pengecekan setiap *path* pada sistem yang memliki *role* berbeda	Mengecek kesesuaian akses *path* sistem	Tidak adanya autentikasi pada sistem	Tidak ditemukan *path* yang bisa diakses oleh beda *user*	ditemukan *path* yang bisa diakses oleh beda *user*

Tabel 3.3 merupakan gambaran umum pengujian yang akan dilakukan. Pada tabel tersebut terdapat hasil yang terbagi menjadi dua yaitu aman dan tidak aman. Pada kolom aman hasil yang ditujukan mengacu pada kolom tujuan sedangkan kolom tidak aman hasil yang ditujukan mengacu pada kolom celah.

3.3.1 Pengujian Broken Authentication

Pada pengujian broken authentication, hal yang dilakukan adalah melakukan penetrasi pada halaman *login* untuk mencoba masuk ke dalam sistem. Penetrasi ini melakukan bantuan beberapa aplikasi untuk membantu proses pengujian nantinya. Pengujian ini difokuskan pada celah keamanan *login* dan autentikasi pada sistem. Pengujian ini memiliki lima tahapan yang akan dijelaskan langkah – langkah pengujian pada tiap tahapan yaitu:

BR-01. Pengecekan form *login*

Pada pengujian ini, form *login* pada halaman sistem KPTA FTI UII akan dilakukan pengecekan apakah sistem berhasil menutupi informasi terkait *username* dan *password* pengguna atau tidak. Adapun langkah penelitiannya yaitu:

a. Akses halaman *login*

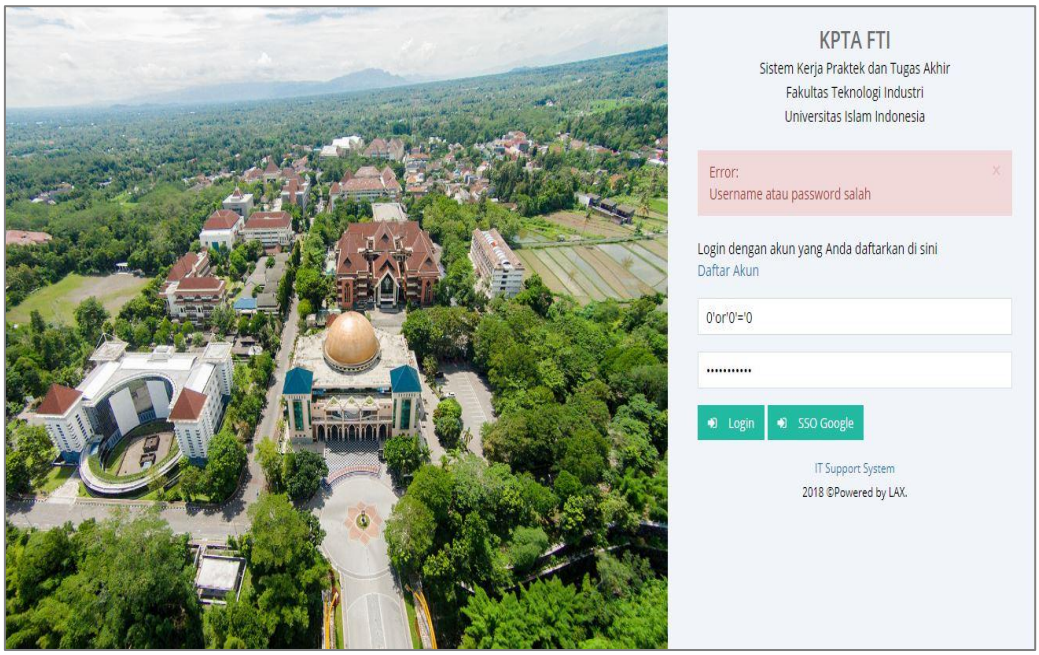
Tahapan ini sekaligus menjadi awal untuk mengakses sistem pertama kali.

b. Memasukkan kombinasi *username* dan *password* sembarang

Tujuan dari langkah ini adalah memastikan kombinasi yang dilakukan akan menghasilkan pesan *error* pada sistem.

c. Memasukkan kode *payload*

Kode *payload* disini digunakan untuk memancing sistem untuk memperlihatkan kombinasi *username* dan *password* yang digunakan untuk masuk ke dalam sistem. Kode *payload* yang digunakan adalah `0'or'0'='0`. Gambar 3.2 menunjukkan proses memasukkan kode *payload* pada sistem.



Gambar 3.2 Proses Memasukkan Kode *Payload* Pada Form *Login*

d. Akses *page source*

Setelah memasukkan kode *payload* pada sistem, langkah selanjutnya mengakses *page source* pada *browser* untuk cek apakah sistem memperlihatkan kombinasi *username* dan *password* yang benar.

BR-02. Pengecekan proses *logout*

Pengujian ini dilakukan untuk mengecek keamanan sistem setelah dilakukannya proses *logout*. Adapun langkah penelitiannya yaitu:

a. Melakukan *login*

Langkah ini sebagai proses agar bisa melakukan proses *logout* setelahnya.

b. Melakukan *logout*

Setelah melakukan *login*, langkah selanjutnya adalah melakukan proses *logout*.

c. Akses kembali ke halaman sebelumnya

Setelah proses *logout* sudah berhasil dilakukan,langkah selanjutnya adalah akses halaman sebelumnya.

BR-03 Melakukan penetrasi *password*

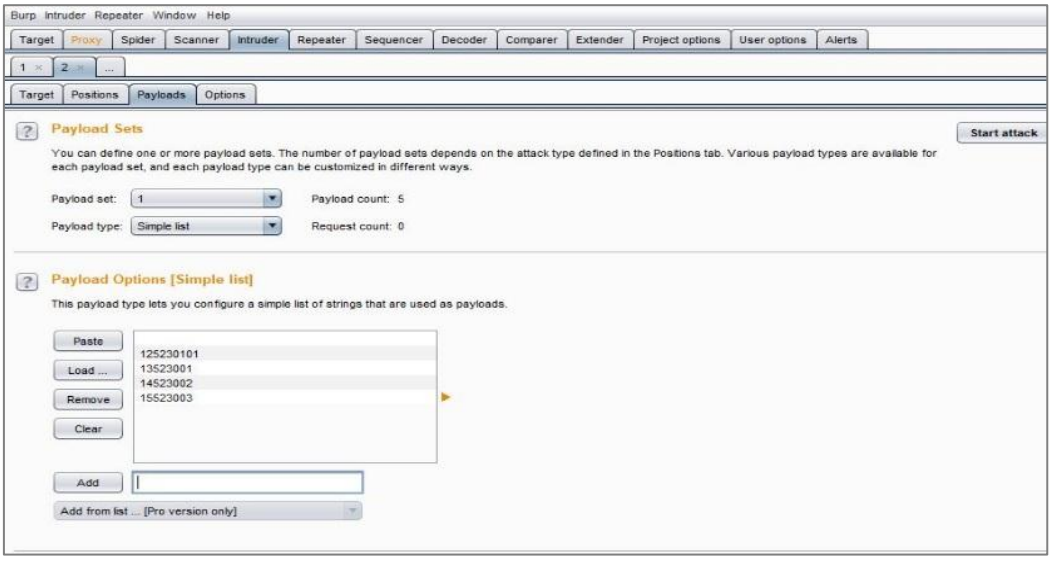
Pengujian ini dilakukan untuk mencari kemungkinan kombinasi *username* dan *password* yang benar untuk bisa masuk ke sistem. Adapun langkah penelitiannya yaitu:

a. Memasukkan *username* dan *password* secara acak

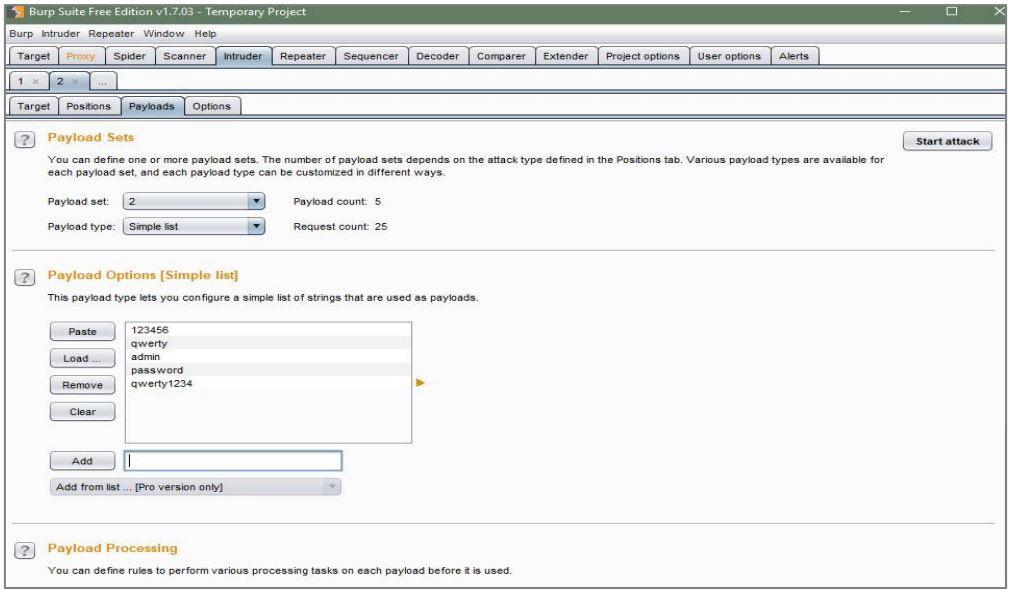
Langkah ini digunakan untuk menangkap status dan proses yang terjadi pada sistem ke aplikasi penetrasi *password*.

b. Membuat daftar kemungkinan *username* dan *password* yang benar

Setelah mendapatkan status dan proses yang terjadi sebelumnya ke aplikasi, langkah selanjutnya adalah membaut daftar kemungkinan *username* dan *password* yang benar untuk bisa masuk ke sistem. Aplikasi yang digunakan adalah Burp Suite. Gambar 3.3 menampilkan proses membuat daftar *username* dan gambar 3.4 menampilkan proses membuat daftar *password*.



Gambar 3.3 Membuat Daftar Kemungkinan *Username* Pada Aplikasi Burp Suite



Gambar 3.4 Membuat Daftar Kemungkinan *Password* Pada Aplikasi Burp Suite

c. Pencarian kombinasi *username* dan *password*

Setelah membuat daftar *username* dan *password* pada langkah sebelumnya, langkah selanjutnya adalah mulai melakukan penetrasi untuk mendapatkan kombinasi *username* dan *password* yang benar.

BR-04. Mengecek *password*

Pengujian ini dilakukan untuk mengetahui apakah sistem memiliki sebuah kondisi dalam menentukan pembuatan *password*. Adapun langkah penelitiannya yaitu hanya melakukan pendaftaran akun kemudian mencoba memuat *password* yang umum digunakan.

BR-05. Mengakses *view* sistem

Pengujian ini dilakukan untuk memastikan akses *view* pada sistem ada yang tidak memerlukan autentikasi terlebih dahulu. Adapun langkah penelitiannya yaitu :

a. Membuat daftar akses *view* pada setiap *controller*

Langkah ini untuk mempermudah langkah selanjutnya dalam melakukan pengujian.

b. Menjalankan *path*

Setelah membuat daftar pada langkah sebelumnya, yang akan dilakukan selanjutnya adalah menjalankan *path* satu persatu untuk mengecek apakah ada view yang bsia diakses tanpa autentikasi terlebih dahulu.

3.3.2 Pengujian Security Misconfiguration

Pengujian pada security misconfiguration dilakukan pada setiap fungsi *class* karena kesalahan ini bisa terjadi di setiap aplikasi. Hal yang dilakukan adalah dengan mengecek setiap *path* yang berjalan ketika pemroresan terjadi apakah ada konfigurasi yang bisa menjadi celah untuk dilakukannya pengujian, jika ada maka akan dilakukan penetrasi menggunakan aplikasi. Pengujian ini difokuskan pada celah keamanan *path* dan bisa bertambah pada saat pengujian jika ditemukan celah lainnya. Pengujian ini memiliki dua tahapan yang akan dijelaskan langkah – langkah pengujian pada tiap tahapan yaitu:

SM-01. Denial of Service

Pada pengujian ini, sistem dicoba untuk diberikan serangan dalam skala besar apakah sistem bisa menanganinya atau tidak. Adapun langkah penelitiannya yaitu:

a. Menyiapkan *virtual machine*

   Langkah ini adalah untuk membuat wadah untuk sistem operasi pendukung untuk melakukan pengujian DoS.

b. Memasang sistem operasi pendukung pengujian

   Memasang sistem operasi pendukung pengujian pada *virtual machine*.

c. Penyerangan ke sistem

   Setelah sistem operasi berhasil dipasang, langkah yang akan dilakukan adalah melakukan penyerangan DoS pada sistem.

SM-02. Pengecekan *path*

Pada pengujian ini, *path* akan dicek satu persatu pada akun pengguna lain untuk memastikan *path* yang diakses pada akun lain tidak bisa diakses. Adapun langkah penelitiannya yaitu:

a. Membuat daftar *path* yang akan diakses

   Pada langkah ini setiap *path* akan dibuat daftar sesuai dengan *role* penggunanya untuk memudahkan pengujian nantinya.

b. Melakukan pengecekan *path* keseluruhan

   Setelah dibuat daftar *path* yang akan diakses, langkah selanjutnya adalah mengecek tap *path* pada *role* yang berbeda untuk mengetahui apakah ada *path*yang bisa diakses pada beda *role*.

3.4 Perbaikan Sistem

Pada perbaikan sistem, sistem akan diperbaiki jika pada saat pengujian sistem, sistem berhasil diretas keamanannya, tahapan ini dilakukan dengan proses iterasi. Jika terdapat kesalahan, maka akan dilakukan perbaikan kemudian dilanjutkan ke iterasi selanjutnya untuk dilakukan pengujian kembali. Iterasi berakir jika pada pengujian tidak ditemukan kesalahan pada sistem. Tahapan ini juga sekaligus menjadi tahapan terakhir dalam uji coba.

BAB IV
HASIL DAN PEMBAHASAN

Pada pengujian keamanan Sistem KPTA FTI UII dilakukan sebanyak dua kali iterasi. Pada iterasi pertama dilakukan pengujian pertama, kemudian dilakukan perbaikan dan dilanjutkan pada iterasi selanjutnya untuk menguji kembali hasil perbaikan pada iterasi pertama. Pengujian dilakukan dengan OWASP khususnya pada metode broken authentication dan security misconfiguration. Kemudian membuat skenario tes serta tabel hasil pengujian. Fokus dari pengujian ini adalah keamanan pada authentikasi sistem dan akses *path* pada sistem.

4.1 Hasil Pengujian Broken Authentication Iterasi 1

BR-01. Pengecekan form *login*

Setelah kode *payload* diakses oleh sistem, muncul pesan yang menyatakan *username* atau *password* salah. Hal selanjutnya yang akan dilakukan adalah melihat hasil pada *page source* apakah sistem akan memperlihatkan kombinasi *username* dan *password* yang benar akibat dari penggunaan kode *payload* untuk akses *login* ke sistem. Berikut gambar 4.1 adalah hasil dari *page source* yang sudah diberikan kode *payload* sebelumnya.

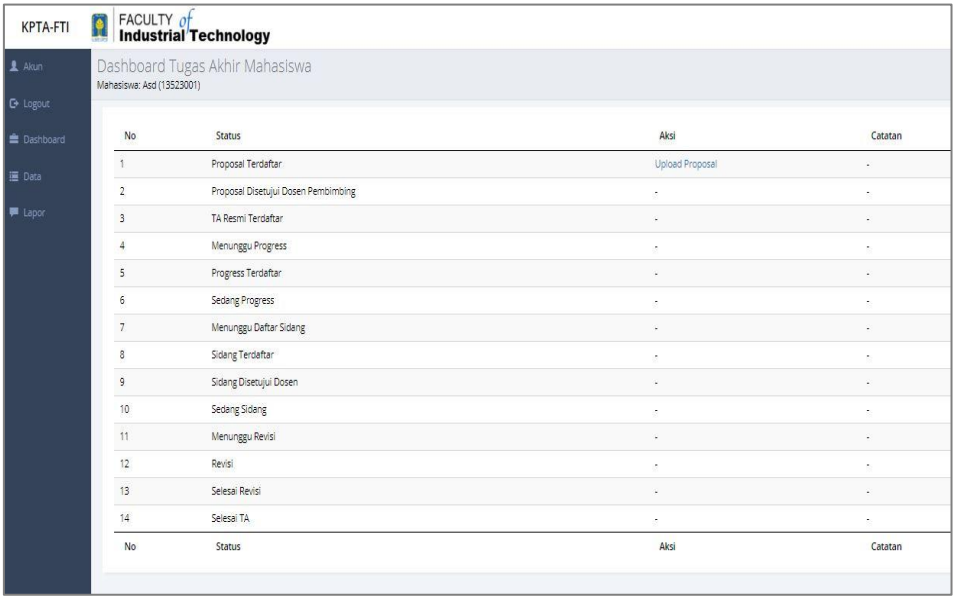
```
...
 <input       type=”text”       name=”user_nim”       class=”form_control”
placeholder=”NIM” required/>

<p class=”help=block”></p>
</div>
<div class=”form-froup”>
     <input  type=”password”  name=”user_password”  class=”form-control”
placeholder=”Password” required />
     <p class=”help-block”></p>
  </div>
...
```

Gambar 4.1 Kode Page Source Pada Halaman *Login*

Pada gambar 4.1 sistem berhasil untuk tidak menampilkan informasi *username* dan *password* yang valid ketika kode *payload* digunakan ketika melakukan proses *login*. Dengan begitu, form *login* pada sistem KPTA FTI UII aman dari bocornya informasi *username* dan *password* karena sistem berhasil menyembunyikan informasi terkait *username* dan *password* pengguna.

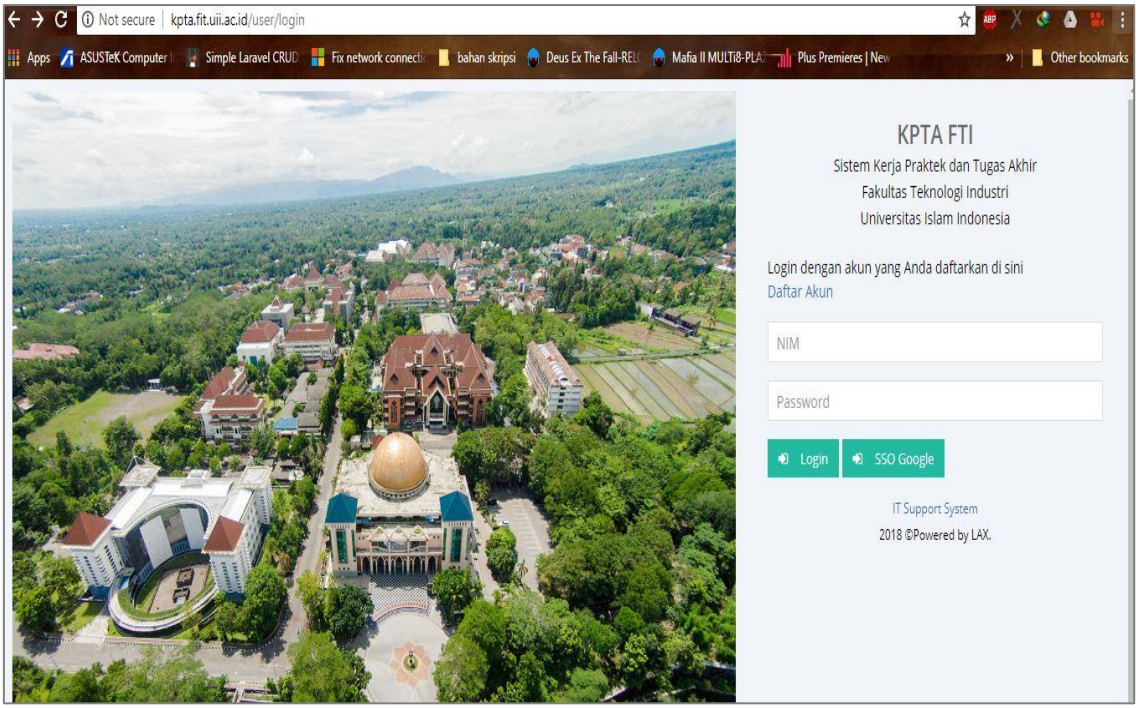
BR-02. Penegecekan Proses *Logout*

Pengujian ini mengharuskan penguji untuk akses masuk ke dalam sistem sebelum melakukan pengecekan. Setelah berhasil masuk, sistem mengarahkan pengguna ke halaman utama kemudian di sebelah kiri terdapat beberapa tombol untuk akses ke tiap fungsi pada sistem termasuk tombol *logout.* Gambar 4.2 menggambarkan halaman awal setelah pengguna berhasil masuk ke sistem.



Gambar 4.2 Halaman Awal Setelah *Login* Berhasil

Setelah melakukan akses *logout,* sistem otomatis mengarahkan kita ke halaman awal pada saat sistem pertama kali dijalankan. Kemudian, penguji mencoba mengakses kembali halaman sebelumnya apakah bisa kembali pada halaman sebelum proses *logout* dilakukan. Gambar 4.3 menunjukkan sistem berhasil melakukan proses *logout*.

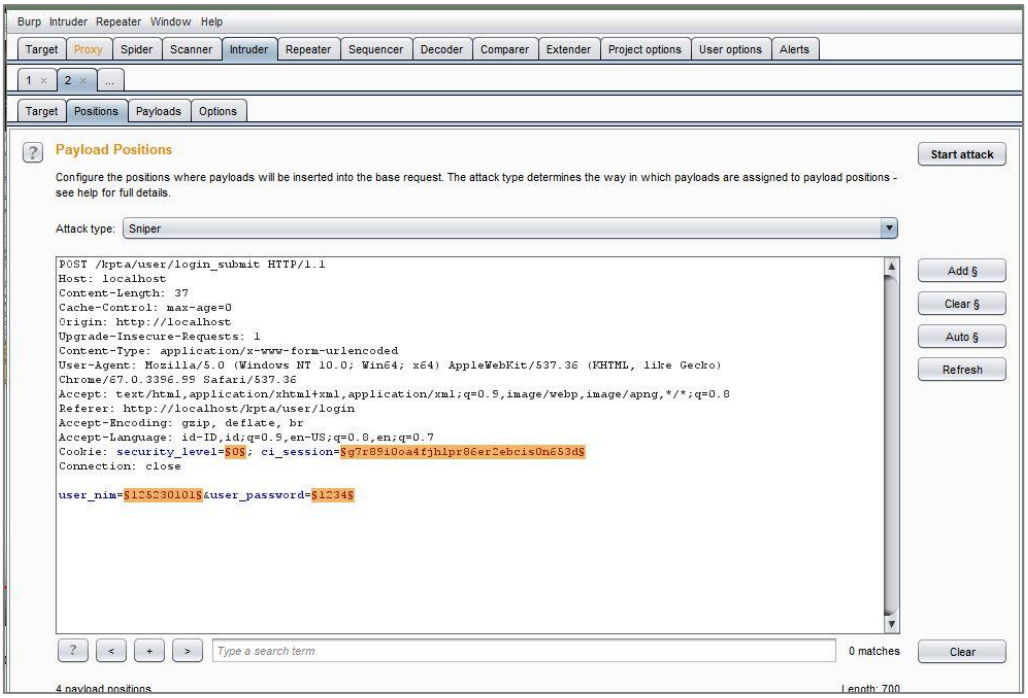


Gambar 4.3 Proses *Logout* Berhasil

Setelah dicoba, sistem tidak bisa mengakses halaman sebelumnya. Hal yang terjadi adalah sistem menampilkan halaman muka sistem jika ingin kembali ke halaman sebelumnya. Pada gambar 4.3 terdapat tanda panah kiri kanan yang aktif menandakan sistem diminta untuk kembali ke halaman sebelumnya, namun sistem tidak menampilkan halaman sebelumnya. Sistem berhasil mengamankan akses terlarang setelah proses *logout* dijalankan.

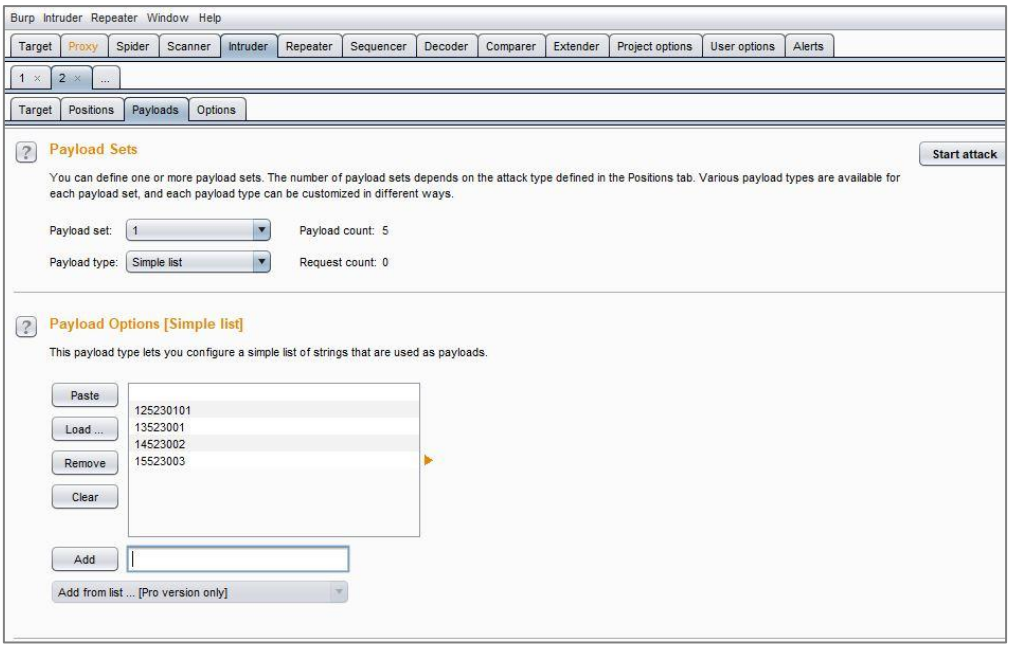
BR-03. Penetrasi *Password*

Pengujian ini menggunakan *tool* untuk mencari kombinasi *username* dan *password* yang benar untuk bisa masuk ke sistem. Setelah berhasil menangkap status dan proses *login* pada sistem, aplikasi akan menampilkan *username* dan *password* yang kita coba pakai pada sistem sebelumnya. Pada tahapan ini menandakan sistem bisa dilakukan penetrasi *password*. Gambar 4.4 menunjukkan proses mendapatkan kode *username* dan *password* yang sebelumnya dimasukkan.

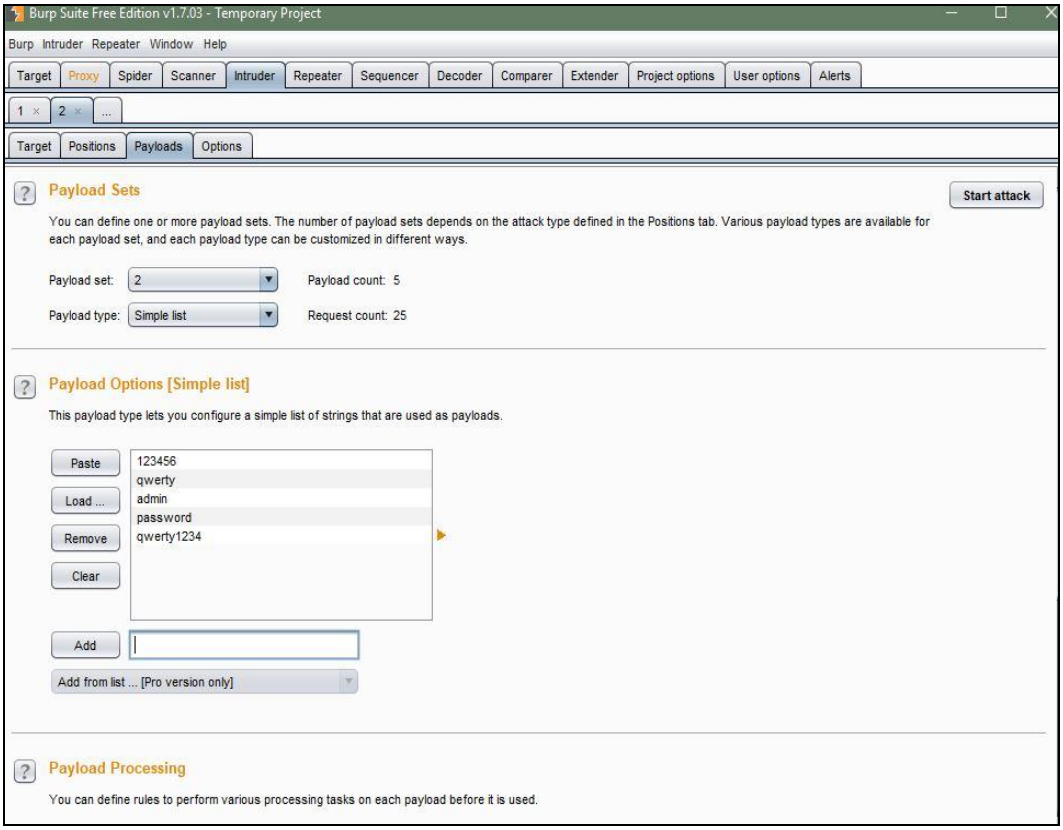


Gambar 4.4 Aplikasi Burp Suite Menampilkan *Username* dan *Password* sebelumnya

Pada gambar 4.5 dan gambar 4.6 Burp Suite siap melakukan penetrasi. Sebelum melakukan penetrasi, buat daftar *payload* untuk kemungkinan *username* dan *password* yang benar untuk masuk ke dalam sistem nantinya.



Gambar 4.5 Membuat Daftar *Payload* untuk *username*

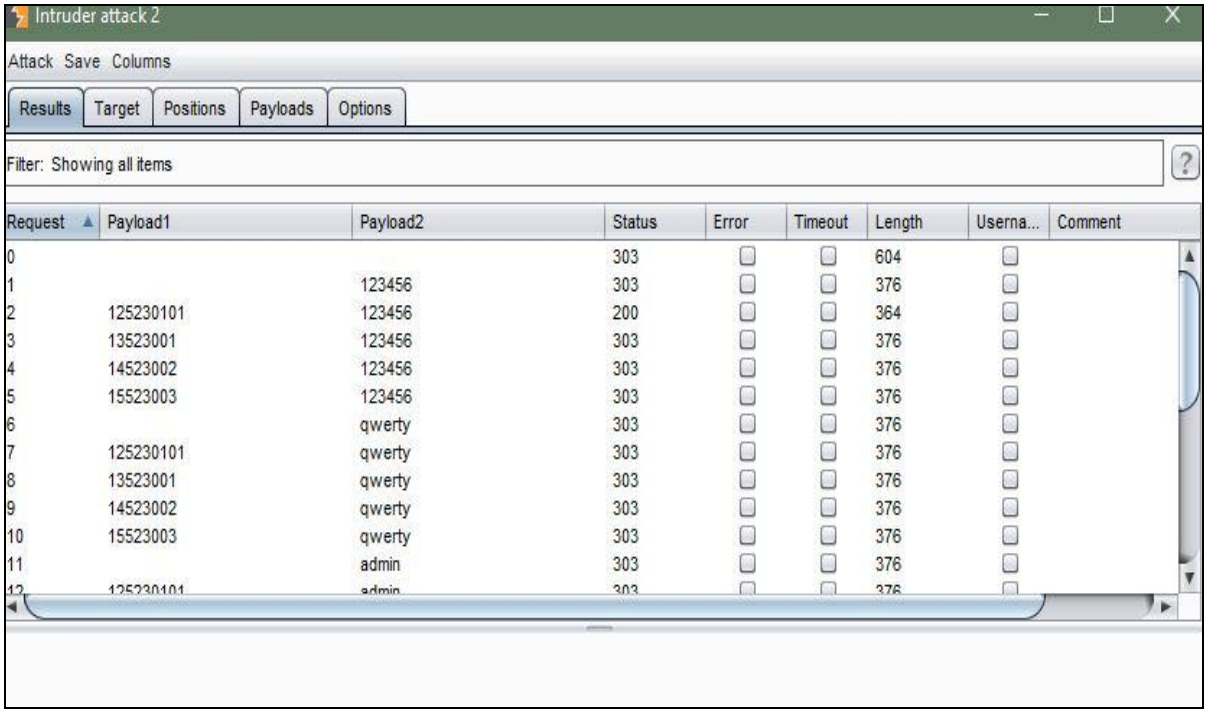


Gambar 4.6 Membuat Daftar *Payload* untuk *password*

Setelah daftar berhasil dibuat, penetrasi bisa dijalankan. Burp suite mencari kombinasi antar *payload* mana yang bisa digunakan untuk masuk ke dalam sistem. Setelah selesai melakukan kombinasi, berikut hasil pencarian Burp Suite. Pada tabel 4.1 akan menampilkan daftar kemungkinan *username* dan *password* yang bisa diakses dan gambar 4.7 menampilkan hasil penetrasi.

Tabel 4.1 Daftar Kemungkinan *Username* dan *Password*

USERNAME	*PASSWORD*
125230101	123456
13523001	Admin
14523002	Qwerty
11523123	Password



Gambar 4.7 Hasil Penetrasi Pencarian Password

Gambar 4.7 menunjukkan hasil dari penetrasi untuk mencari kombinasi *username* dan *password* untuk masuk ke dalam sistem. Kolom status menunjukkan bahwa kombinasi yang terjadi pada proses penetrasi benar atau tidak. Status 200 menunjukkan bahwa kombinasi tersebut benar dan ketika dilakukan proses login pada sistem penguji bisa masuk ke dalam sistem.

BR-04. Pengecekan *Password*

Pengujian ini berkaitan dengan pengujian sebelumnya. Pada pengujian sebelumnya diadapati bahwa *password* yang digunakan oleh pengguna sangat mudah untuk ditebak. Kemudian masalah selanjutnya terdapat pada pendaftaran akun. Pada bagian kata sandi, fungsi ini tidak memiliki aturan untuk menggunakan kombinasi huruf dan angka sehingga bagian ini tidak aman.

BR-05. Akses *View* Sistem

Pengujian ini mengecek *view* tiap *controller* pada sistem sehingga proses pengujiannya lebih lama karena harus mengakses satu per satu *path view* pada tiap *controller*-nya. Daftar

dan hasil dari pengujian akan ditampilkan dalam bentuk tabel. Akses *view* dilakukan tanpa melalui proses autentikasi terlebih dahulu. Tabel 4.2 menunjukkan hasil pengujian.

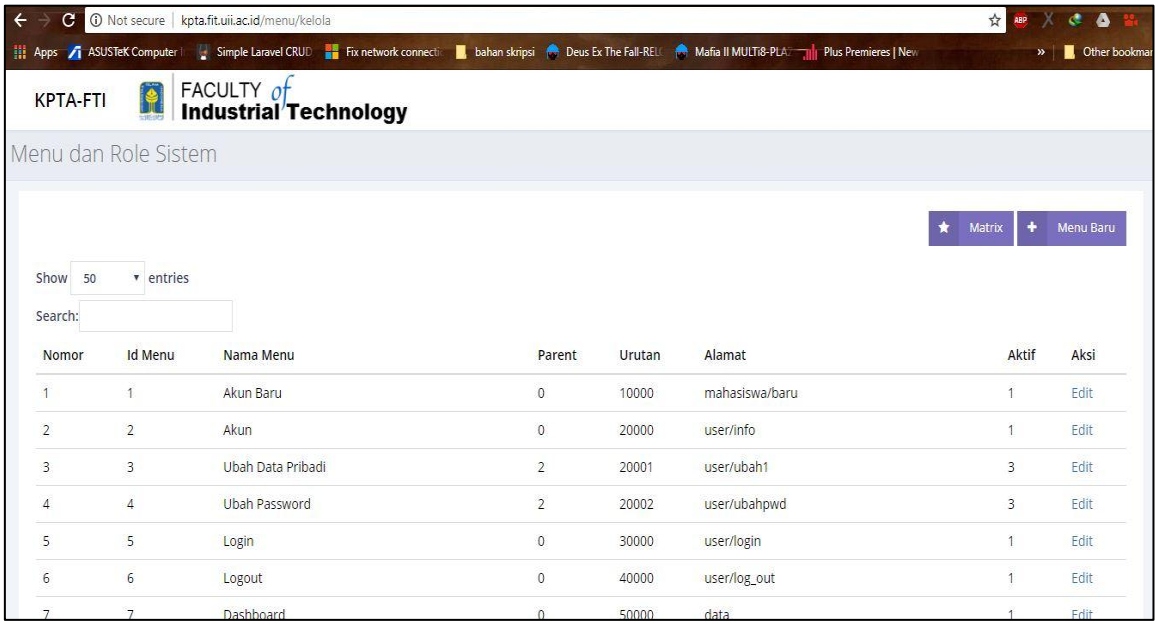
Tabel 4.2 Hasil Pengujian Akses *View*

CONTROLLER	PATH VIEW	HASIL
Berita.php	kpta/berita/index	Tidak Berhasil
	kpta/berita/info	Tidak Berhasil
	kpta/berita/baru	Tidak Berhasil
	kpta/berita/baru_submit	Tidak Berhasil
	kpta/berita/kelola	Tidak Berhasil
Data.php	kpta/data/index	Tidak Berhasil
	kpta/data/ta	Tidak Berhasil
	kpta/data/status	Tidak Berhasil
	kpta/data/rekap_bimbingan	Tidak Berhasil
	kpta/data/ta_diterima	Tidak Berhasil
	kpta/data/lambatta	Tidak Berhasil
	kpta/data/data_mahasiswa	Tidak Berhasil
	kpta/data/data_alumni	Tidak Berhasil
	kpta/data/bimbingan	Tidak Berhasil
	kpta/data/bimbingan_ta_dosen/…	Tidak Berhasil
Dosen.php	kpta/dosen/index	Tidak Berhasil
	kpta/dosen/baru	Berhasil
	kpta/dosen/kelola	Tidak Berhasil
	kpta/dosen/ubah/….	Tidak Berhasil
Karyawan.php	kpta/karyawan/index	Tidak Berhasil
	kpta/karyawan/kelola	Tidak Berhasil
	kpta/karyawan/ubah/…	Tidak Berhasil
	kpta/karyawan/baru	Tidak Berhasil
Kerja.php	kpta/kerja/mahasiswa	Tidak Berhasil
	kpta/kerja/ta	Tidak Berhasil
	kpta/kerja/seluruh_ta	Tidak Berhasil
	kpta/kerja/daftarkp	Tidak Berhasil
	kpta/kerja/daftarkp	Tidak Berhasil
	kpta/kerja/daftarta	Tidak Berhasil
	kpta/kerja/cek_judul_ascii	Tidak Berhasil

CONTROLLER	PATH VIEW	HASIL
	kpta/kerja/daftarta_tampil_rekomendasi	Tidak Berhasil
	kpta/kerja/revisi	Tidak Berhasil
	kpta/kerja/daftar/..	Tidak Berhasil
	kpta/kerja/status/..	Tidak Berhasil
	kpta/kerja/input_nilai_akhir/..	Tidak Berhasil
	kpta/kerja/status_ditolak/..	Tidak Berhasil
	kpta/kerja/konsentrasi	Tidak Berhasil
	kpta/kerja/tertolak	Tidak Berhasil
	kpta/kerja/lihat/..	Tidak Berhasil
	kpta/kerja/info/..	Tidak Berhasil
	kpta/kerja/ubah/..	Tidak Berhasil
	kpta/kerja/detilstatus/.	Tidak Berhasil
	kpta/kerja/lihat1/…	Tidak Berhasil
	kpta/kerja/lihat_ditolak/..	Tidak Berhasil
	kpta/kerja/bimbingan	Tidak Berhasil
	kpta/kerja/kelola	Tidak Berhasil
Laporan.php	kpta/laporan/kelola	Tidak Berhasil
	kpta/laporan/baru	Tidak Berhasil
	kpta/laporan/ubah/..	Tidak Berhasil
	kpta/laporan/hapus/..	Tidak Berhasil
Menu.php	kpta/menu/kelola	Berhasil
	kpta/menu/matrix	Berhasil
	kpta/menu/baru	Berhasil
	kpta/menu/ubah/..	Tidak Berhasil
	kpta/menu/hapus/..	Tidak Berhasil

CONTROLLER	PATH VIEW	HASIL
User.php	kpta/user/login	Tidak Berhasil
	kpta/user/ubahpwd	Tidak Berhasil
	kpta/user/info	Tidak Berhasil
	kpta/user/kelola	Tidak Berhasil
	kpta/user/ubah	Tidak Berhasil
	kpta/user/ubah_dosen	Tidak Berhasil
	kpta/user/ubah_mahasiswa	Tidak Berhasil
	kpta/user/monitoring_akun	Tidak Berhasil
	kpta/user/daftar_users	Tidak Berhasil
	kpta/user/pindah_akun_auth	Tidak Berhasil

Tabel 4.2 menunjukkan daftar *controller* , Setelah melakukan pengecekan, terdapat empat *view* yang bisa diakses tanpa memerlukan autentikasi terlebih dahulu. Sistem sepenuhnya tidak aman dalam pengujian akses *view*. Gambar 4.8 menunjukkan salah satu hasil *error* yang ditemukan ketika melakukan pengujian.



Gambar 4.8 Hasil Pengujian Akses *View*

4.2 Hasil Pengujian Security Misconfiguration Iterasi 1

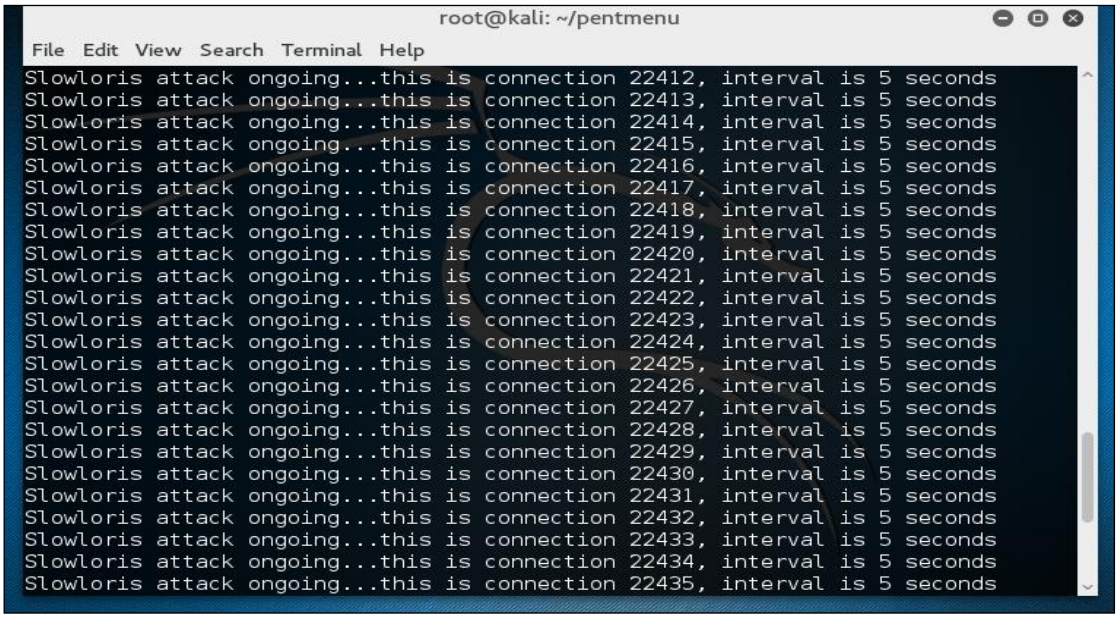
SM-01. *Denial of Service*

Pengujian dilakukan dengan melakukan serangan secara besar terhadap sistem dengan menggunakan sistem operasi yang terpasang pada *virtual machine*. Sistem operasi pendukung

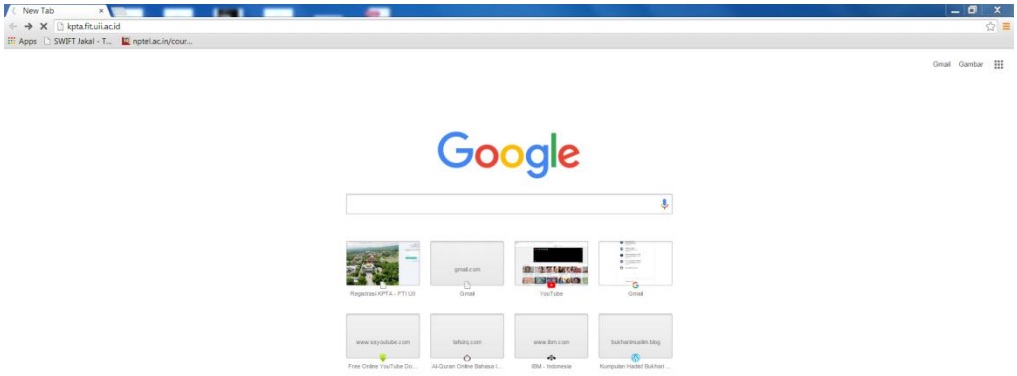
pengujian tahap ini adalah Kali Linux karena sistem operasi tersebut sudah menyediakan aplikasi untuk melakukan pengujian ini.

Dalam melakukan penetrasi, *virtual machine* disesuaikan IP addres-nya dengan sistem operasi yang dipakai sekarang, kemudian pada sistem operasi Kali Linux dijalankan aplikasi untuk DoS yang sudah disediakan oleh sistem operasi tersebut. Pengujian ini mneggunakan besar paket yaitu sebesar 1000 *bytes*.

Pada saat penetrasi, sistem tidak bisa diakses dikarenakan sistem diserang secara besar besaran sampai status DoS berhenti. Gambar 4.9 menunjukkan proses DoS sedang dilakukan dan gambar 4.10 sistem tidak bisa diakses ketika dilakukan pengujian DoS.



Gambar 4.9 Status Penyerangan DoS Terhadap Sistem



Gambar 4.10 Sistem Tidak Bisa Diakses

Sistem sama sekali tidak merespon permintaan penguji untuk mengakses sistem KTPA, pada gambar 4.9 sistem hanya memuat halaman sistem namun tidak bisa menampilkannya dikarenakan pekerjaan sistem dibuat menjadi berat akibat serangan DoS. Dengan kata lain, sistem tidak berhasil menangani serangan DoS.

SM-02. Pengecekan *Path*

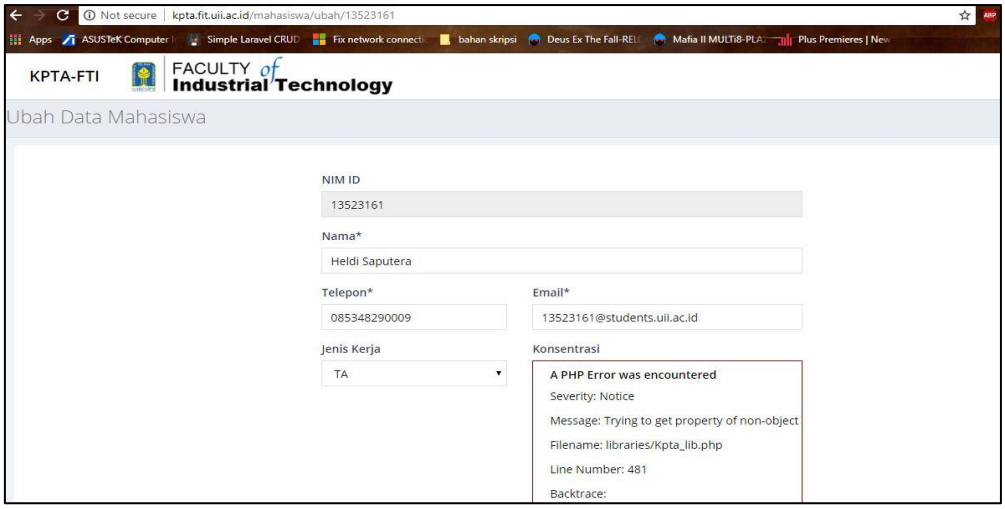
Pengecekan *path* dilakukan dengan masuk sebagai *role* berbeda, kemudian langkah selanjutnya menjalankan pertukaran *path* yang hanya bisa diakses oleh *role* masing-masing ke *role* berbeda. Hasil pengujian serta *path* akan ditampilkan pada tabel. Tabel 4.3 menampilkan hasil pengujian pengecekan *path*.

Tabel 4.3 Hasil Pengujian Pengecekan *Path*

ROLE	PATH	STATUS
Dosen	kpta/kerja/ta	Tidak Berhasil
	kpta/user/info	Tidak Berhasil
	kpta/dosen	Tidak Berhasil
	kpta/data/lambatta	Tidak Berhasil
	kpta/data/bimbingan	Tidak Berhasil
	kpta/data/status	Tidak Berhasil

	kpta/data/rekap_bimbingan	Tidak Berhasil
	kpta/user/monitoring_akun	Tidak Berhasil
	kpta/kerja/kelola	Tidak Berhasil
	kpta/kerja/ubah	Tidak Berhasil
	kpta/user/kelola	Tidak Berhasil
	kpta/user/kelola	Tidak Berhasil
	kpta/mahasiswa/kelola	Berhasil
	kpta/mahasiswa/ubah	Berhasil
	kpta/dosen/kelola	Tidak Berhasil
	kpta/dosen/ubah	Tidak Berhasil
	kpta/karyawan/kelola	Tidak Berhasil
	kpta/karyawan/ubah	Tidak Berhasil
	kpta/menu/kelola	Berhasil
	kpta/laporan/kelola	Tidak Berhasil
	kpta/laporan/ubah	Berhasil
Mahasiswa	kpta/kerja/mahasiswa	Tidak Berhasil
	kpta/user/info	Tidak Berhasil
	kpta/data/ta	Tidak Berhasil
	kpta/mahasiswa	Tidak Berhasil
	kpta/dosenkpta/laporan/baru	Tidak Berhasil

Pada tabel diatas menunjukkan hasil dari pengecekan terdapat empat *path* dosen yang bisa diakses oleh *role* mahasiswa. Sedangkan pada *role* siswa tidak ditemukan kesalahan pada sistem. Ini mendandakan masih ada *path* yang terlupakan untuk di autentikasi terlebih dahulu. Gambar 4.11 menampilkan salah satu hasil pengujian pada akses *path*.



Gambar 4.11 Hasil Pengujian Akses *Path*

Setelah melakukan pengujian, hasil pengujian akan dibuat dalam sebuah tabel. Tabel 4.4 menunjukkan hasil pengujian pada iterasi 1.

Tabel 4.4 Hasil Pengujian Iterasi 1

ID	DESKRIPSI PENGUJIAN	HASIL	STATUS
BR-01	Melakukan pencarian *username* dan *password*	Data *username* dan *password* pengguna tidak ditampilkan	Aman
BR-02	Melakukan penetrasi *password*	Sistem bisa kembali ke halaman sebelum *logout*	Aman
BR-03	Cek *password*	Kombinasi *username* dan *password* ditemukan	Tidak aman
BR-04	Mengakses *view* sistem	Sistem memproses password yang tidak sesuai dengan prosedur keamanan	Tidak aman

ID	DESKRIPSI PENGUJIAN	HASIL	STATUS
BR-05	Penyerenangan dengan Denial of Service	ditemukan *view* yang bisa diakses tanpa prosedur *login*	Tidak aman

SM-01	Mengakses setiap *path* pada sistem	Sistem berjalan lebih lambat ketika diakses	Tidak aman
SM-02	Melakukan penetrasi *password*	ditemukan *path* yang bisa diakses oleh beda *user*	Tidak aman

Kesimpulan yang bisa diambil setelah iterasi 1 yaitu sebagian besar pengujian tidak aman. Pada saat pengujian sistem memiliki celah yang kurang aman pada bagian autentikasi dan fungsi *password*. Tidak amannya fungsi *password* pada sistem KPTA dikarenakan masih kurangnya dalam pengecekan kombinasi *password* dan belum ditambahnya sebuah kondisi minimal mengandung angka dan huruf.

Beberapa akses *view* bisa diakses tanpa autentikasi dikarenakan pada bagian kode program ada bagian yang terlupakan untuk dimasukkan sehingga sistem menganggap kondisi tersebut bisa diakses dengan atau tanpa autentikasi terlebih dahulu.

Sistem tidak bisa menangani serangan DoS karena serangan ini hampir terjadi di setiap sistem pada umumnya. Serangan DoS pasti selalu berhasil jika keadaan akses pada sistem tidak dilindungi oleh *firewall.*

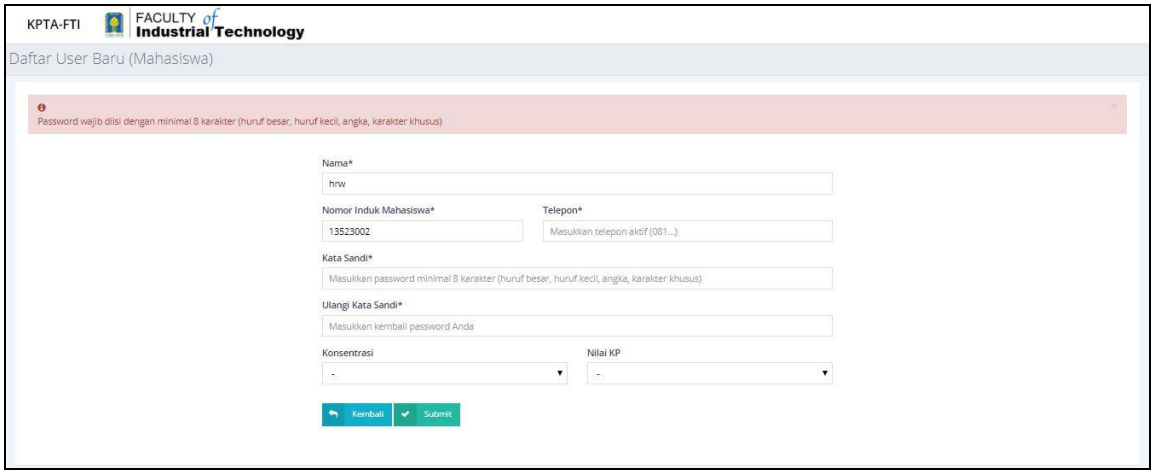
Beberapa akses *path* di *role* yang berbeda bisa diakses dikarenakan adanya bagian pada kode program yang lupa dimasukkan oleh pengembang sistem akibat kelalaiannya sehingga *path* yang harusnya tidak bisa diakses oleh *user* lain menjadi bisa diakses.

4.3 Perbaikan Dari Iterasi 1

a. Perbaikan fungsi *password*

Perbaikan yang dilakukan untuk fungsi *password* yaitu menambahkan kondisi pada kode program bahwa dalam pembuatan *password* harus memakai kombinasi huruf dan angka sehingga kemungkinan akun diretas menjadi lebih kecil. Perbaikan ini adalah hasil dari pengujian penetrasi *password* dan cek *password*. Secara umum, dalam hal aturan dalam penggunaan *password* yang baik tidak ada. Tiap pengembang bebas menentukan aturan dalam pembuatan *password*. Dari sisi pengguna sendiri harus mengikuti aturan pengembang sistem dalam pembuatannya (de Carnavalet & Mannan, 2014). Dengan perubahan ini *password* pengguna menjadi lebih sulit untuk didapatkan oleh peretas. Perbaikan kode program dilakukan pada fungsi validasi *password* dan *password_check.* Setelah kode program ditambahkan kondisi, sistem menolak *password* yang tidak lolos

dalam tahap validasi, ini dikarenakan *password* yang dibuat tidak memenuhi kondisi pada saat divalidasi oleh sistem. Gambar 4.12 menujukkan hasil perbaikan pada sistem.



Gambar 4.122 Perbaikan Kode Program Untuk Fungsi *Password* Berhasil

Pada gambar 4.12 dilakukan pengecekan dengan membuat akun baru terlebih dahulu. Pada saat memasukkan *password* yang hanya mengandung angka yaitu 12345678 setelah disubmit muncul pesan bahwa *password* diisi dengan minimal 8 karakter (huruf besar, huruf kecil, kaakter khusus). Hal ini perlu dilakukan agar mengurangi pengguna dalam memilih kode *password* yang sering digunakan seperti tanggal lahir, tempat lahir, nomor telepon, dan sebagainya.

b. Perbaikan *view*

Perbaikan *view* diperlukan untuk memperbaiki program yang seharusnya tidak bisa diakses langsung tanpa proses autentikasi. Untuk mengetahui letak kesalahannya, hal yang dilakukan pertama adalah mengecek kode program yang ditemukan kesalahan pada akses. Dalam hal ini kode program yang bermasalah ada pada *controller* dosen dan menu. Setelah mengecek kode program, ditemukan ada bagian kode yang terlupakan oleh pengembang sistem sehingga timbulnya *error*. Oleh karena itu, perbaikan yang dilakukan adalah menambahkan kode program yang kurang. Perbaikan ini dengan menambahkan kode program yaitu kondisi autentikasi. Fungsi dari kode program tersebut adalah membuat method tersebut hanya bisa diakses harus dalam keadaan autentikasi. Perbaikan ini juga meliputi tiga *path* yang memiliki permasalah yang sama.

c. Pengamanan dari DoS

Pengamanan dari DoS ini merupakan pencegahan yang bersifat sementara karena DoS sendiri adalah tipe serangan yang bisa terjadi di banyak sistem bukan pada sistem tertentu. Oleh karena itu pengamanan dari serangan DoS ini perlu diterapkan jika sewaktu – waktu ada penyerangan secara tiba – tiba. Untung tindakan pencegahannya yaitu menyalakan *firewall* sehingga serangan dari DoS tidak langsung menyerang ke sistem.

d. Perbaikan *path*

Pada perbaikan *path* sendiri perlu untuk dilakukan karena setiap *role* meiliki akses tertentu pada sistem, jika pada bagian itu tidak aman. Pengguna lain bisa mengakses secara bebas dan itu berbahaya jika *path* tersebut merupakan *path* yang penting. Perbaikan yangn dilakukan sama dengan perbaikan yang ada pada pengecekan *view* yaitu meberikan kode program yang mengharuskan autentikasi sesuai dengan role-nya terlebih dahulu untuk mengakses *path* terkait.

4.4 Hasil Pengujian Iterasi 2

4.4.1 Tabel Pengujian

Tabel 4.5 merupakan hasil pengujian pada iterasi kedua. Tabel yang digunakan sama dengan tabel 4.4 namun ada perubahan pada hasil perbaikan di iterasi 1.

Tabel 4.5 Hasil Pengujian Iterasi 2

ID	DESKRIPSI PENGUJIAN	HASIL	STATUS
BR-01	Melakukan pencarian *username* dan *password*	Data *username* dan *password* pengguna tidak ditampilkan	Aman
BR-02	Melakukan penetrasi *password*	Sistem bisa kembali ke halaman sebelum *logout*	Aman
ID	**DESKRIPSI PENGUJIAN**	**HASIL**	**STATUS**
BR-03	Cek *password*	Kombinasi *username* dan *password* tidak ditemukan	Aman

BR-04	Mengakses *view* system	Sistem tidak memproses password yang tidak sesuai dengan prosedur keamanan	Aman
BR-05	Penyerenangan dengan Denial of Service	Tidak ditemukan *view* yang bisa diakses tanpa prosedur *login*	Aman
SM-01	Mengakses setiap *path* pada system	Sistem berjalan normal ketika diakses	Aman
SM-02	Melakukan penetrasi *password*	ditemukan *path* yang bisa diakses oleh beda *user*	Tidak aman

Kesimpulan yang dapat diambil dari pengujian iterasi 2 seperti pada tabel 4.4 adalah:

a. Semua pengujian berstatus aman

   Setelah melakukan perbaikan dari hasil pengujian pada iterasi pertama, iterasi kedua tidak ditemukan *error* pada saat pengujian ulang. Pada pengujian iterasi kedua semua pengujian sudah dinyatakan aman tidak ditemukan kesalahan dalam penggunaannya sehingga sistem menjadi lebih aman.

b. Proses autentikasi sistem sesuai dengan rancangan

   Pengujian iterasi kedua ini proses autentikasi pada sistem sudah diperbaiki dan dijalankan sesuai dengan rancangan awal, sehingga *error* yang ditemukan pada saat mengakses *view* tertentu atau *path* tertentu sudah diminimalisir sehingga tidak sembarang pengguna bisa mengaksesnya.

BAB V
KESIMPULAN

5.1 Kesimpulan

Berdasarkan hasil penelitian yang dilakukan, kesimpulan yang dapat diambil yaitu,

a. Penelitian ini berhasil menemukan cara bagaimana melakukan pengujian pada faktor broken authentication dan security misconfiguration. Pada pengujian broken authentication ditemukan lima tahap pengujian yakni *insecure login login form, logout management, password attack, weak password* dan akses *view*. Pada pengujian *security misconfiguration* ditemukan dua tahap pengujian yakni *denial of service* dan pengecekan setiap *path* pada sistem.
b. Pengujian keamanan perangkat lunak pada faktor *broken authentication* dan *security misconfiguration* berhasil dilakukan dengan tepat sehingga sistem menjadi jauh lebih aman daripada sebelum dilakukan pengujian. Sebelum melakukan pengujian sistem KPTA FTI UII banyak ditemukan kesalahan pada sistem keamanan yang berakibat fatal jika celah tersebut diketahui oleh banyak orang. Dengan OWASP pengujian sistem KPTA FTI UII lebih spesifik dalam melakukan pengujian keamanan.

5.2 Saran

Saran dan perbaikan untuk sistem KPTA FTI UII agar menjadi lebih baik lagi adalah,

a. Dalam melakukan pengembangan bisa lebih teiliti lagi agar kesalahan pada sistem bisa diminimalisir.
b. Setiap ada perbaikan pada sistem sertakan dokumentasi perbaikan agar memudahkan pengembang yang lain bisa memahami sistem yang sudah diperbaiki dan lebih mudah melakukan pengembangan selanjutnya.
c. Diharapkan metode yang tidak digunakan dalam penelitian ini bisa dilakukan pada penelitian lainnya.

DAFTAR PUSTAKA

de Carnavalet, X. d., & Mannan, M. (2014). *From Very Weak to Very Strong: Analyzing Password-Strength Meters*.

Kurniawan, L. A. (2018). *PENGUJIAN ALUR PROSES BISNIS MENGGUNAKAN STATE TRANSITION DIAGRAM.* Yogyakarta: UII.

Laksita, R. M. P. (2016). Pengembangan SIMSON sebagai Sistem Informasi Manajemen KP yang Terintegrasi pada Semua Jurusan di FTI UII. Yogyakarta: Universitas Islam Indonesia.

Maulana, M. A. (2017). Analisis Celah Keamanan Manajemen Sesi terhadap Serangan Session Hijacking pada Web Aplikasi.

Miessler, D. (2015). Securing the Internet of Things: Mapping Attack Surface Areas Using the OWASP IoT Top 10. *RSA Conference.*

Mustaqbal, M. S., Firdaus, R. F., & Rahmadi, H. (2015). PENGUJIAN APLIKASI MENGGUNAKAN BLACK BOX TESTING BOUNDARY VALUE ANALYSIS. *Jurnal Ilmiah Teknologi Informasi Terapan*, 33.

OWASP. (2017). Retrieved from OWASP Top 10-2017: https://www.owasp.org/index.php/Top_10-2017_Top_10

Sedek, K. A. (2009). Developing a Secure Web Application Using OWASP Guidelines.

Sofyan, D. (2014). *Security Testing Pada Aplikasi yang Telah Dibangun*.

Suhartono, J. (2016, Desember 16). Software Testing.

Tamilbotnet. (2018, April 9). Retrieved from https://www.youtube.com/watch?v=pxvnU9G2nwM

Tank, K. (2015, October 9). Retrieved from https://www.youtube.com/watch?v=g-HeOCMZDxE&t=502s

Williams, L. (2006). White-Box Testing. 1.

LAMPIRAN