**BAB II**

**DESKRIPSI TEORITIK DAN KERANGKA BERPIKIR**

1. **Deskripsi Teoritik**

Deskripsi teoritik merupakan sebuah definisi atau teori-teori yang berhubungan dengan topik atau judul yang digunakan dalam pembahasan penelitian, termasuk pemodelan pengembangan sistem yang digunakan.

**A.1. Pengertian Penerapan**

Penerapan adalah suatu perbuatan mempraktekkan suatu teori, metode, dan hal lain untuk mencapai tujuan tertentu dan untuk suatu kepentingan yang diinginkan oleh suatu kelompok atau golongan yang telah terencana dan tersusun sebelumnya.

Menurut (Henra Saputra Tanjung, 2019), penerapan adalah perbuatan menerapkan.

Menurut (Andrilla, 2014), penerapan merupakan sebuah tindakan yang dilakukan baik secara individu maupun kelompok dengan maksud untuk mencapai tujuan yang telah dirumuskan.

Menurut (Ekonomi et al., 2016), penerapan diartikan sebagai suatu proses, cara, perbuatan menerapkan atau mempraktikkan.

Menurut (Sri Endang Rahayu, 2016), penerapan (implementasi) merupakan aktivitas untuk menjalankan suatu program yang terencana berdasarkan norma tertentu untuk mencapai tujuan kegiatan.

Menurut ( Kamus Besar Bahasa Indonesia (KBBI) Online), penerapan adalah proses, cara, perbuatan menerapkan.

Berdasarkan pendapat para ahli di atas dapat disimpukan bahwa penerapan adalah cara yang dilakukan dalam kegiatan agar dapat mencapai tujuan yang diinginkan.

**A.2. Pengertian *Website***

*Website* adalah kumpulan halaman dalam suatu *domain* yang memuat tentang berbagai informasi agar dapat dibaca dan dilihat oleh pengguna internet melalui sebuah mesin pencari. Informasi yang dapat dimuat dalam sebuah *website* umumnya berisi mengenai konten gambar, ilustrasi, video, dan teks untuk berbagai macam kepentingan.

Biasanya untuk tampilan awal sebuah *website* dapat diakses melalui halaman utama (*homepage*) menggunakan *browser* dengan menuliskan *URL* yang tepat. Didalam sebuah *homepage*, juga memuat beberapa halaman *web* turunan yang saling terhubung satu dengan yang lain.

Menurut (Nofyat, 2018), Halaman *website* biasanya berupa dokumen yang ditulis dalam format *Hyper Text Markup Language* (HTML), yang bisa diakses melalui HTTP, HTTPS adalah suatu protokol yang menyampaikan berbagai informasi dari *server website* untuk ditampilkan kepada para *user* atau pemakai melalui *web browser*.

Menurut (Imam Rosyadi, 2018), *Website* merupakan lokasi yang akan digunakan untuk mengumpulkan file-file halaman *web*. File-file dokumen *web* tersebut terdiri dari gambar, script *CSS*, *audio* dan sebagainya. Dengan banyaknya file-file tersebut, maka terbentuk suatu *website.*

Menurut (Zulfikar Yusya Mubarak, 2017), Website merupakan kumpulan dari halaman-halaman yang berhubungan dengan file-file lain yang saling terkait.

Menurut (Buhori Muslim, 2016), Website adalah sebuah tempat yang memungkinkan seseorang menyatakan dirinya, hobinya, pengetahuannya, produk yang dijualnya dan apapun juga yang dapat di akomodasikan oleh teks, tulisan, gambar, video, animasi dan file multimedia lainnya.

Menurut (Irfan Agus, 2017), Pengertian website adalah sering juga disebut Web, dapat diartikan suatu kumpulan-kumpulan halaman yang menampilkan berbagai macam informasi teks, data, gambar diam ataupun bergerak, data animasi, suara, video maupun gabungan dari semuanya, baik itu yang bersifat statis maupun yang dinamis, yang dimana membentuk satu rangkaian bangunan yang saling berkaitan dimana masing-masing dihubungkan dengan jaringan halaman atau hyperlink.

Berdasarkan pendapat para ahli diatas dapat disimpulkan bahwa *website* adalah sekumpulan halaman yang memuat gambar, teks dan suara untuk menampilkan kumpulan informasi.

**A.3. PengertianOWASP**

*Open Web Application Security Project* (OWASP) adalah sebuah organisasi internasional yang bersifat non-profit, didirikan oleh OWASP *foundation* pada 21 April 2004 di Amerika Serikat. Pada OWASP merupakan peningkatan keamanan perangkat lunak didalam mengembangkan, memperoleh, mengoperasikan dan memelihara aplikasi terpercaya untuk menjamin dapat keamanan yang telah di buat atau pun yang dikembangkan. OWASP sendiri memiliki tujuan dalam keamanan untuk mengamankan *software*, sehingga orang-orang dan organisasi dapat membuat keputusan terhadap resiko yang telah ditemukan dalam keamanan yang benar.

OWASP melakukan penelitian dan mensosialisasikan hasilnya untuk meningkatkan kesadaran akan keamanan aplikasi. OWASP memiliki beberapa project diantaranya OWASP *Web Application Penetration Testing* , *WebGoat*, *Webscarab, Risk Rating* dan OWASP *top10*.

Menurut (Rochman et al., 2021), OWASP (*Open Web Application Security Project*) yang dikeluarkan oleh owasp.org sebuah organisasi non-profit yang berdedikasi pada keamanan aplikasi berbasis web.

Menurut (Agung Tri Laksono & Joko Dwi Santoso, 2021), OWASP (*Open Web Application Security Project*) Ini adalah komunitas terbuka di seluruh dunia yang memungkinkan organisasi untuk meningkatkan keamanan aplikasi yang mereka percayai.

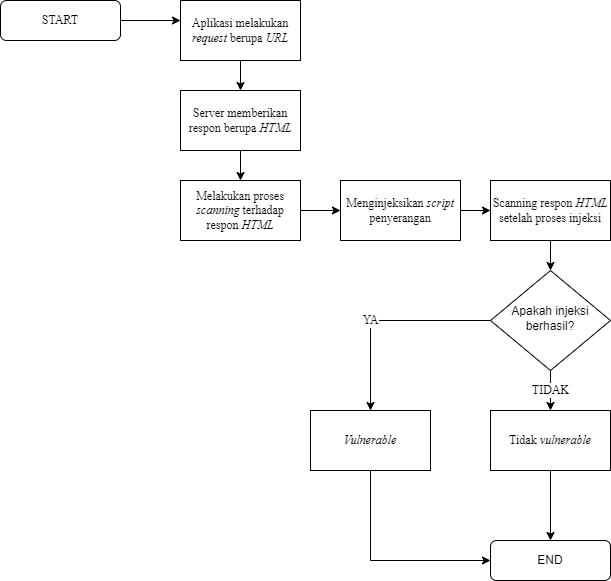
Menurut (Gabriel Rizkiawan Santoso et al., 2015), OWASP (*Open Web Application Security Project*) adalah suatu organisasi nirlaba yang memiliki nilai yaitu meningkatkan kemanan dari suatu *software.*

Menurut (Eka Pratama & Wiradarma, 2019), OWASP menyediakan banyak alat, panduan, dan metodologi pengujian untuk keamanan siber di bawah lisensi sumber terbuka, khususnya OWASP *Testing Guide* (OTG).

Menurut (Wibowo & Sulaksono, 2021), OWASP (*Open Web Application Security Project*) bertujuan untuk memberi tahu pengembang, perancang, arsitek, dan pemilik bisnis tentang risiko yang terkait dengan kerentanan keamanan yang paling berkembang di aplikasi *web*.

Berdasarkan pendapat para ahli di atas dapat disimpukan bahwa OWASP (*Open Web Application Security Project*) adalah sebuah organisasi yang menangani tentang tingkat keamanan sebuah aplikasi. OWASP juga bertujuan untuk menjaga keamanan aplikasi yang dimiliki oleh para pemilik aplikasi tersebut.

**A.3.1. Metodologi OWASP *Web Application Penetration Testing***

Pada OWASP (*Open Web Application Security Project*) versi 4 terdapat 11 sub kategori untuk menguji keamanan dari sebuah *website* yang disebut dengan *web application penetration testing methodology*. Secara garis besar metode yang digunakan OWASP adalah injeksi dengan menggunakan *request* dan *response method* yaitu memanfaatkan *HTTP* *Verb* (*POST, GET, PUT, PATCH, and DELETE*) untuk kemudian dilihat apakah terdapat kerentanan yang dapat mengakibatkan dampak terhadap aplikasi . Berikut adalah *Flow Chart penetration testing* dengan OWASP :

Gambar 2.1 Metodologi *Web Application Penetration* OWASP (Sumber: https://www.owasp.org)

* + 1. Aplikasi melakukan *request* berupa *URL* ke *server*.
    2. *Server* memberikan respon berupa *HTML*.
    3. Aplikasi melakukan proses *scan* terhadap respon

*HTML* dan menginjeksikan *script* injeksi.

* + 1. *Server* memberikan respon berupa *HTML*.
    2. Aplikasi melakukan proses *scan* terhadap respon

*HTML* untuk memeriksa hasil proses injeksi.

* + 1. Aplikasi memberikan laporan hasil proses *scan*.

Berikut adalah *Testing Guide* yang ada pada *OWASP* :

* + 1. *Introduction and Objectives*

Membuat tabel *checklist* mengenai *testing* yang akan dilakukan dan mencari informasi mengenai *fingerprint* aplikasi seperti bahasa pemrograman yang digunakan aplikasi, dimana aplikasi di simpan , *database* apa yang digunakan, serta *metafiles* yang berisi informasi- informasi mengenai aplikasi .

* + - * *Testing Checklist*
      * *Information Gathering*
      * *Conduct Search Engine Discovery and Reconnaissance for Information Leakage (OTG-INFO-001)*
      * *Fingerprint Web Server (OTG-INFO-002)*
      * *Review Webserver Metafiles for Information Leakage (OTG-INFO-003)*
      * *Enumerate Applications on Webserver (OTG- INFO-004)*
      * *Review Webpage Comments and Metadata for Information Leakage (OTG-INFO-005)*
      * *Identify application entry points (OTG-INFO- 006)*
      * *Map execution paths through application (OTG-INFO-007)*
      * *Fingerprint* *Web* *Application* *Framework (OTG-INFO-008)*
      * *Fingerprint Web Application (OTG-INFO- 009)*
      * *Map Application Architecture (OTG-INFO- 010)*
    1. *Configuration and Deployment Management Testing*

Melakukan pengujian pada konfigurasi jaringan, pengujian penanganan ekstensi file, mencari *backup* file yang biasanya berisi informasi *sensitive*, pengujian *http methods*.

* + - * *Test Network/Infrastructure Configuration (OTG-CONFIG-001)*
      * *Test Application Platform Configuration (OTG-CONFIG-002)*
      * *Test File Extensions Handling for Sensitive Information (OTG-CONFIG-003)*
      * *Review Old, Backup and Unreferenced Files for Sensitive Information (OTG-CONFIG-004)*
      * *Enumerate Infrastructure and Application Admin Interfaces (OTG-CONFIG-005)*
      * *Test HTTP Methods (OTG-CONFIG-006)*
      * *Test HTTP Strict Transport Security (OTG- CONFIG-007)*
      * *Test RIA cross domain policy (OTG-CONFIG- 008)*
    1. *Identity Management Testing*

Melakukan pengujian terhadap manajemen akun yang dimiliki oleh aplikasi, seperti *role* dan akses aplikasi, proses registrasi *user* baru , dan *provisioning* akun.

* + - * *Test Role Definitions (OTG-IDENT-001)*
      * *Test User Registration Process (OTG-IDENT- 002)*
      * *Test Account Provisioning Process (OTG- IDENT-003)*
      * *Testing for Account Enumeration and Guessable User Account (OTG-IDENT-004)*
      * *Testing for Weak or unenforced username policy (OTG-IDENT-005)*
    1. *Authentication Testing*

Pengujian terhadap autentikasi aplikasi dengan mencari celah untuk bisa masuk sebagai *user* yang memiliki hak akses seperti mencari celah mekanisme pada *directory traversal*, *backdoor* yang sebelumnya dibuat, fitur lupa password / *reset* password.

* + - * *Testing for Credentials Transported over an Encrypted Channel (OTG-AUTHN-001)*
      * *Testing for default credentials (OTG-AUTHN- 002)*
      * *Testing for Weak lock out mechanism (OTG- AUTHN-003)*
      * *Testing for bypassing authentication schema (OTG-AUTHN-004)*
      * *Test remember password functionality (OTG- AUTHN-005)*
      * *Testing for Browser cache weakness (OTG- AUTHN-006)*
      * *Testing for Weak password policy (OTG- AUTHN-007)*
      * *Testing for Weak security question/answer (OTG-AUTHN-008)*
      * *Testing for weak password change or reset functionalities (OTG-AUTHN-009)*
      * *Testing for Weaker authentication in alternative channel (OTG-AUTHN-010)*
    1. *Authorization Testing*

Pengujian terhadap otorisasi aplikasi dengan tidak menggunakan jalan masuk yang seharusnya dan tidak melakukan *login* untuk masuk sebagai pemilik akses, dengan memanfaatkan *get* dan *post method http*.

* + - * *Testing Directory traversal/file include (OTG- AUTHZ-001)*
      * *Testing for bypassing authorization schema (OTG-AUTHZ-002)*
      * *Testing for Privilege Escalation (OTG- AUTHZ-003)*
      * *Testing for Insecure Direct Object References (OTG-AUTHZ-004)*
    1. *Session Management Testing*

Pengujian terhadap *Session* yang ditinggalkan oleh aplikasi yaitu *cookies* yang bisa dimanfaatkan untuk bisa masuk sebagai pemilik akses tanpa harus *login*.

* + - * *Testing for Bypassing Session Management Schema (OTG-SESS-001)*
      * *Testing for Cookies attributes (OTG-SESS- 002)*
      * *Testing for Session Fixation (OTG-SESS-003)*
      * *Testing for Exposed Session Variables (OTG- SESS-004)*
      * *Testing for Cross Site Request Forgery (CSRF) (OTG-SESS-005)*
      * *Testing for logout functionality (OTG-SESS- 006)*
      * *Test Session Timeout (OTG-SESS-007)*
      * *Testing for Session puzzling (OTG-SESS-008)*
    1. *Input Validation Testing*

Pengujian terhadap validasi terhadap *script* yang dapat di eksekusi sehingga menyebabkan risiko yang berbahaya terhadap aplikasi seperti *SQL Injection*, *XML Injection*, *CSS*, dan lain-lain.

* + - * *Testing for Reflected Cross Site Scripting (OTG-INPVAL-001)*
      * *Testing for Stored Cross Site Scripting (OTG- INPVAL-002)*
      * *Testing for HTTP Verb Tampering (OTG- INPVAL-003)*
      * *Testing for HTTP Parameter pollution (OTG- INPVAL-004)*
      * *Testing for SQL Injection (OTG-INPVAL-005)*
      * *Testing for LDAP Injection (OTG-INPVAL- 006)*
      * *Testing for ORM Injection (OTG-INPVAL- 007)*
      * *Testing for XML Injection (OTG-INPVAL-008)*
      * *Testing for SSI Injection (OTG-INPVAL-009)*
      * *Testing for XPath Injection (OTG-INPVAL- 010)*
      * *IMAP/SMTP Injection (OTG-INPVAL-011*
      * *Testing for Code Injection (OTG-INPVAL- 012)*
      * *Testing for Command Injection (OTG- INPVAL-013)*
      * *Testing for Buffer overflow (OTG-INPVAL- 014)*
      * *Testing for incubated vulnerabilities (OTG- INPVAL-015)*
      * *Testing for HTTP Splitting/Smuggling (OTG- INPVAL-016)*
    1. *Testing for Error Handling*

Pengujian terhadap penanganan eror yang terjadi , dari penanganan tersebut biasanya akan muncul beberapa informasi yang sifatnya *credential* seperti *database*, *bugs*, atau komponen-komponen yang terkait dengan aplikasi.

* + - * *Analysis of Error Codes (OTG-ERR-001)*
      * *Analysis of Stack Traces (OTG-ERR-002)*
    1. *Testing for Weak Cryptography*

Pengujian terhadap *kriptografi* yang dimiliki oleh aplikasi dalam melakukan *enkripsi* informasi-informasi yang sifatnya adalah sensitif.

* + - * *Testing for Weak SSL/TLS Ciphers, Insufficient Transport Layer Protection (OTG-CRYPST-001)*
      * *Testing for Padding Oracle (OTG-CRYPST- 002)*
      * *Testing for Sensitive information sent via unencrypted channels (OTG-CRYPST-003)*
    1. *Business Logic Testing*

Pengujian terhadap celah yang ada pada proses bisnis aplikasi , setiap aplikasi memiliki detail logika yang berbeda-beda dalam menjalankan proses bisnis.

* + - * *Test Business Logic Data Validation (OTG- BUSLOGIC-001)*
      * *Test Ability* *to* *Forge* *Requests* *(OTG- BUSLOGIC-002)*
      * *Test Integrity Checks (OTG-BUSLOGIC-003)*
      * *Test for Process Timing (OTG-BUSLOGIC- 004)*
      * *Test Number of Times a Function Can be Used Limits (OTG-BUSLOGIC-005)*
      * *Testing for the Circumvention of Work Flows (OTG-BUSLOGIC-006)*
      * *Test Defenses Against Application Mis-use (OTG-BUSLOGIC-007)*
      * *Test Upload of Unexpected File Types (OTG- BUSLOGIC-008)*
      * *Test* *Upload* *of* *Malicious Files* *(OTG- BUSLOGIC-009)*
    1. *Client Side Testing*

Pengujian terhadap celah yang dapat muncul pada sisi *client*, dengan memanfaatkan *script* yang dapat dijalankan dapat membahayakan *client* lain.

* + - * *Testing for DOM based Cross Site Scripting (OTG-CLIENT-001)*
      * *Testing for JavaScript Execution (OTG- CLIENT-002)*
      * *Testing for HTML Injection (OTG-CLIENT- 003)*
      * *Testing for Client Side URL Redirect (OTG- CLIENT-004)*
      * *Testing for CSS Injection (OTG-CLIENT-005)*
      * *Testing for Client Side Resource Manipulation (OTG-CLIENT-006)*
      * *Test Cross Origin Resource Sharing (OTG- CLIENT-007)*
      * *Testing for Cross Site Flashing (OTG- CLIENT-008)*
      * *Testing for Clickjacking (OTG-CLIENT-009)*
      * *Testing WebSockets (OTG-CLIENT-010)*
      * *Test Web Messaging (OTG-CLIENT-011)*
      * *Test Local Storage (OTG-CLIENT-012)*

**A.3.2. Metodologi OWASP *Risk Rating***

OWASP *Risk Rating Methodology* adalah metodologi OWASP yang digunakan sebagai proses penilaian risiko (*risk assessment*) dengan input hasil pengujian yang dilakukan dan hasil akhir evaluasi . Akan ada 5 tahap dalam melakukan penilaian risiko yaitu :

* Fase 1 - Mengidentifikasi risiko,
* Fase 2 - Estimasi tingkat kemungkinan risiko terjadi (*Likelihood*)
* Fase 3 - Estimasi tingkat pengaruh terhadap proses bisnis (*Business Impact*)
* Fase 4 - Menentukan nilai risiko (*Severity*)
* Fase 5 - Menentukan prioritas perbaikan dari risiko

Fase 1 - Mengidentifikasi risiko

Langkah pertama yaitu mengidentifikasi risiko. Dalam mengidentifikasi risiko, perlu adanya informasi terkait jenis risiko apa saja yang mungkin terjadi, bentuk dan proses penyerangan risiko yang dapat di lakukan.

Fase 2 - Estimasi tingkat kemungkinan risiko terjadi (*Likelihood*)

Langkah kedua adalah menentukan faktor *likelihood*. Secara sederhana perhitungan *likelihood* dapat dilakukan dengan langsung membagi risiko ke dalam beberapa kategori yakni *high*, *medium*, *low*. Ada beberapa faktor yang dapat berpengaruh dalam penentuan *likelihood*, yang pertama adalah *threat agent*.:

* + - * *Skill Level*

Bagaimana *technical skill* yang dimiliki oleh *threat agents* ?

**Tabel 2.1 *Skill Level Risk Rating***

|  |  |
| --- | --- |
| ***Security*** ***penetration skills (9)*** | Menggunakan tools yang di buat sendiri dengan menyesuaikan infrastruktur aplikasi dan off-script ( diluar scenario penyerangan pada umumnya ) |
| ***network*** ***and programming skills (6)*** | Menggunakan tools yang di buat sendiri dengan menyesuaikan infrastruktur aplikasi |
| ***advanced*** ***computer user (5)*** | Menggunakan lebih dari 1 tools dengan setting parameter |
| ***some technical skills (3)*** | Menggunakan 1 tools dengan setting parameter |
| ***no technical skills (1)*** | Menggunakan 1 tools atau sama sekali tidak menggunakan |

(Sumber: https://www.owasp.org)

* + - * *Motive*

Bagaimana motivasi *threat agents* untuk menemukan dan membobol celah keamanan ?

**Tabel 2.2 *Motive Risk Rating***

|  |  |
| --- | --- |
| ***Low or no reward (1)*** | Tidak terdapat keuntungan atau keuntungan kecil |
| ***Possible reward (4)*** | Mungkin bisa menjadi keuntungan |
| ***High reward (9)*** | Keuntungan besar |

(Sumber: https://www.owasp.org)

* + - * *Opportunity*

Bagaimana kebutuhan dan peluang yang dibutuhkan *threat agents* untuk menemukan dan membobol celah keamanan ?

**Tabel 2.3 *Opportunity Risk Rating***

|  |  |
| --- | --- |
| ***Full access or expensive resources required (0)*** | Membutuhkan akses penuh atau membutuhkan sumber daya yang mahal |
| ***Special access or resources required (4)*** | Membutuhkan akses special atau membutuhkan sumber daya |
| ***Some access or resources required (7)*** | Membutuhkan beberapa akses atau membutuhkan sumber daya |
| ***No access or resources required (9)*** | Tidak memerlukan akses atau membutuhkan sumber daya |

(Sumber: https://www.owasp.org)

* + - * *Size*

Seberapa besar kelompok *user* yang termasuk dalam *threat agents* ?

**Tabel 2.4 *Size Risk Rating***

|  |  |
| --- | --- |
| ***Developers (2)*** | Kategori developer aplikasi |
| ***System*** ***administrators (2)*** | Kategori sistem administrator |
| ***Intranet users (4)*** | Kategori pengguna internet |
| ***Partners (5)*** | Kategori partner |
| ***Authenticated users (6)*** | Kategori user yang memilki akses |
| ***Anonymous internet users(9)*** | Kategori user di internet |

(Sumber: https://www.owasp.org)

Faktor berikutnya adalah *vulnerability factors*, dimana faktor ini dipakai untuk mengestimasi kemungkinan *vulnerability* ditemukan dan dipergunakan. *Vulnerability factors* juga dibagi ke dalam beberapa kriteria yakni sebagai berikut:

* + - * ***Ease of Discover***

Seberapa mudah kelompok *threat agents* ini dalam menemukan celah keamanan?

**Tabel 2.5 *Ease of Discover Risk Rating***

|  |  |
| --- | --- |
| ***Practically impossible (1)*** | Bisa di deteksi dengan manual test (tools tidak tersedia) dan cenderung off script |
| ***Difficult (3)*** | Bisa di deteksi dengan menggunakan lebih dari 1 tools |
| ***Easy (7)*** | Terdapat tools yang khusus untuk mendeteksi celah keamanan |
| ***Automated tools available (9)*** | Terdapat automated tools yang dapat mendeteksi celah keamanan |

(Sumber: https://www.owasp.org)

* + - * ***Ease of Exploit***

Seberapa mudah kelompok *threat agents* ini untuk membobol celah keamanan ?

**Tabel 2.6 *Ease of Exploit Risk Rating***

|  |  |
| --- | --- |
| ***Theoretical (1)*** | Bisa di bobol dengan manual test (tools tidak tersedia) dan cenderung off script |
| ***Difficult (3)*** | Bisa di bobol dengan menggunakan lebih dari 1 tools |
| ***Easy (5)*** | Terdapat tools yang khusus untuk membobol celah keamanan |
| ***Automated tools available (9)*** | Terdapat automated tools yang dapat membobol celah keamanan |

(Sumber: https://www.owasp.org)

* + - * ***Awareness***

Seberapa diketahuinya celah keamanan oleh kelompok *threat agents* ini ?

**Tabel 2.7 *Awareness Risk Rating***

|  |  |
| --- | --- |
| ***Unknown (1)*** | Tidak terdapat dalam daftar CWE (Common Weakness Enumerity) |
| ***Hidden (4)*** | Tidak terdapat informasi mengenai cara penyerangan tetapi terdapat mitigasi pada CWE (Common Weakness Enumerity) |
| ***Obvious (6)*** | Terdapat informasi mengenai cara penyerangan yang tidak detail dan mitigasi pada CWE (Common Weakness Enumerity) |
| ***Public knowledge (9)*** | Terdapat informasi mengenai cara penyerangan detail dan mitigasi pada CWE (Common Weakness Enumerity) |

(Sumber: https://www.owasp.org)

* + - * ***Intrusion Detection***

Bagaimana deteksi dari pembobolan sistem ?

**Tabel 2.8 *Intrusion Detection Risk Rating***

|  |  |
| --- | --- |
| ***Active*** ***detection in application (1)*** | Terdeteksi sebagai penyerangan pada aplikasi |
| ***Logged and reviewed (3)*** | Tercatat pada log dan terdapat review |
| ***Logged without review (8)*** | Tercatat pada log namun  tidak terdapat review |
| ***Not logged (9)*** | Tidak tercatat pada log |

(Sumber: https://www.owasp.org)

Fase 3 - Estimasi tingkat pengaruh terhadap proses bisnis (*Business Impact*)

Langkah berikutnya adalah menghitung *impact* dari risiko yang ditemukan. Ada 2 jenis faktor dari *impact* yaitu *technical* dan *business impact* factor. Berikut adalah beberapa faktor dalam *technical impact* faktor:

* + - ***Loss of Confidentiality***

Seberapa besar data bisa yang di ungkapkan dan seberapa sensitif ?

**Tabel 2.9 *Loss of Confidentiality Risk Rating***

|  |  |
| --- | --- |
| ***Minimal non- sensitive data disclosed* (2)** | Sedikit ( <2 ) dari data *informational* ( berisi data informasi yang tidak *confidential* ) terekspos |
| ***Minimal critical data disclosed* (6)** | Sedikit ( <2 ) dari data yang *critical* ( berisi data *confidential* ) terekspos |
| ***Extensive non- sensitive data disclosed* (6)** | Banyak ( >2 ) data *informational* ( berisi data informasi yang tidak *confidential* ) terekspos |
| ***Extensive critical data disclosed* (7)** | Banyak ( >2 ) data yang *critical* ( berisi data *confidential* ) terekspos |
| ***All data disclosed* (9)** | Seluruh data terekspos ( akses penuh *CPanel administration* ) |

(Sumber: https://www.owasp.org)

* + - ***Loss of Integrity***

Seberapa besar data yang bisa rusak dan seberapa besar tingkat keparahan nya ?

**Tabel 2.10 *Loss of Integrity Risk Rating***

|  |  |
| --- | --- |
| ***Minimal slightly corrupt data* (1)** | Sedikit ( <2 ) dari data *corrupt* dengan tingkat ringan |
| ***Minimal seriously corrupt data* (3)** | Sedikit ( <2 ) dari data *corrupt* dengan tingkat berat |
| ***Extensive slightly corrupt data* (5)** | Banyak ( >2 ) dari data *corrupt* dengan tingkat ringan |
| ***Extensive seriously corrupt data* (7)** | Banyak ( >2 ) dari data *corrupt* dengan tingkat berat |
| ***All data totally corrupt* (9)** | Seluruh data *corrupt* |

(Sumber: https://www.owasp.org)

* + - ***Loss of Avalaibility***

Seberapa banyak layanan yang bisa hilang dan seberapa vital?

**Tabel 2.11 *Loss of Availability Risk Rating***

|  |  |
| --- | --- |
| ***Minimal secondary services interrupted* (1)** | Sedikit (<1) layanan sekunder terganggu |
| ***Minimal******primary services interrupted* (5)** | Sedikit (<1) layanan utama terganggu |
| ***Extensive******secondary services interrupted* (5)** | Banyak (>2) layanan sekunder terganggu |
| ***Extensive******primary services interrupted* (7)** | Banyak (>2) layanan utama terganggu |
| ***All services completely lost* (9)** | Seluruh layanan hilang |

(Sumber: https://www.owasp.org)

* + - ***Loss of Accountability***

Apakah tindakan yang dilakukan threat agent dapat di lacak ?

**Tabel 2.12 *Loss of Accountability Risk Rating***

|  |  |
| --- | --- |
| ***Fully******traceable* (1)** | Bisa di lacak |
| ***Possibly traceable* (7)** | Memungkinkan untuk bisa di lacak |
| ***Completely anonymous* (9)** | Anonim ( tidak bisa di lacak ) |

(Sumber: https://www.owasp.org)

Berikut adalah beberapa faktor dalam *business impact* faktor :

* + - ***Financial Damage***

Seberapa besar kerugian finansial yang dihasilkan dari pembobolan ?

**Tabel 2.13 *Financial Damage Risk Rating***

|  |  |
| --- | --- |
| ***Less than the cost to fix the vulnerability* (1)** | Memiliki kerugian kurang dari biaya *fixing* celah keamanan |
| ***Minor on effect annual profit* (3)** | Memiliki kerugian kecil pada *annual profit* |
| ***Significant effect on annual* *profit* (7)** | Memiliki kerugian besar pada *annual profit* |
| ***Bankruptcy* (9)** | Kerugian yang menyebabkan kebangkrutan |

(Sumber: https://www.owasp.org)

* + - ***Reputation Damage***

Apakah pembobolan dapat menghasilkan hilangnya reputasi yang membahayakan bisnis ?

**Tabel 2.14 *Reputation Damage Risk Rating***

|  |  |
| --- | --- |
| ***Minimal damage* (1)** | Kerusakan kecil |
| ***Loss of major accounts* (4)** | Kehilangan akun utama |
| ***Loss of goodwill* (5)** | Kerusakan pada reputasi kinerja |
| ***Brand damage* (9)** | Kerusakan pada nama baik |

(Sumber: <https://www.owasp.org>)

* + - ***Non-Compliance***

Bagaimana pembobolan yang dilakukan terhadap jenis pelanggaran ?

**Tabel 2.15 *Non-Compliance Risk Rating***

|  |  |
| --- | --- |
| ***Minor violation* (2)** | Pelanggaran kecil |
| ***Clear violation* (5)** | Pelanggaran besar dengan bukti yang jelas dan terdapat peraturan mengenai itu |
| ***High profile violation* (7)** | Pelanggaran besar dengan bukti yang tidak jelas dan terdapat peraturan mengenai itu |

(Sumber: <https://www.owasp.org>)

* + - ***Privacy Violation***

Seberapa besar informasi personal yang dapat diungkapkan?

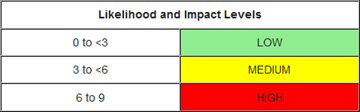
**Tabel 2.16 *Privacy Violation Risk Rating***

|  |  |
| --- | --- |
| ***One individual* (3)** | 1 orang |
| ***Hundreds******of******people* (5)** | 100 - 999 orang |
| ***Thousands of people* (7)** | 1000 - 1000000 orang |
| ***Millions of people* (9)** | 1000000 orang lebih |

(Sumber: <https://www.owasp.org>)

Fase 4 - Menentukan nilai risiko (*Severity*)

Tahap berikutnya adalah menentukan *severity* dari setiap risiko yang ditemukan dengan cara mencari rata- rata dari faktor setiap risiko. Setelah itu ditentukan levelnya melalui *likelihood and impact levels*. Setiap risiko mempunyai bobot *likelihood* dan *impact* yang berbeda, mulai dari *low,* lalu *medium*, dan yang paling tinggi adalah *high.* Gambar menunjukkan *likelihood and impact level*.



**Gambar 2.2 *Likelihood dan Impact Levels*** (Sumber: <https://www.owasp.org>)

Berikut adalah contoh dari pengisian *threat agents* dan *vulnerability factors* yang dapat mennentukan nilai *Likelihood*



**Gambar 2.3 *Threat Agents dan Vulnerability Factors*** (Sumber: <https://www.owasp.org>)

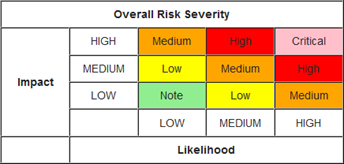
Berikut adalah contoh dari pengisian impact yang menentukan nilai masing-masing dari *Impact Levels*



**Gambar 2.4 *Technical dan Business Impact*** (Sumber: <https://www.owasp.org>)

Fase 5 - Menentukan prioritas perbaikan dari risiko

Nilai *Severity* yang terbesar akan menjadi prioritas dalam perbaikan (*patching*) , sehingga kerugian terhadap *technical* dan *financial* bisa diatasi. Untuk menentukan bagaimana nilai *severity*-nya adalah menggunakan tabel dibawah :



**Gambar 2.5 *Overall Risk Severity*** (Sumber: <https://www.owasp.org>)

**A.4. Pengertian *Kali Linux***

*Kali Linux* merupakan reinkarnasi dari *BackTrack*, sebuah *distro Linux* yang dibuat khusus untuk mengunakan keperluan dalam *penetration* dan *testing* pada sebuah sistem keamanan komputer. *Kali Linux* sudah dilengkapi dengan berbagai *tools* *Linux* yang dapat digunakan untuk melakukan *penetration testing* untuk keamanan aplikasi *web*. *Kali Linux* merupakan pengujian keamanan yang tidak perlu melakukan *meng-install* atau membuat kode *program*/*script* baru. *Kali Linux* bisa digunakan *hacker* yang dapat memahami setiap aksi yang dilakukannya dan mampu membuat *tool/script-nya* sendiri dalam *penetrasi testing* ([www.kali.org](http://www.kali.org)).

Menurut (Fikriyadi Fikriyadi et al., 2020), *Kali Linux* dirancang khusus untuk pengujian keamanan jaringan, dilengkapi dengan aplikasi pendukung yang digunakan dalam kegiatan *hacking* dan juga digunakan sebagai pengujian keamanan jaringan.

Menurut (Reza Rizky Adha et al., 2021), *Kali linux* adalah salah satu Distribusi *Linux* turunan dari *Debian linux* yang dikembangkan dengan tujuan sebagai sistem operasi yang dapat digunakan untuk *testing* keamanan suatu jaringan.

Menurut (Petar Cisar & Robert Pinter, 2019), *Kali linux* adalah distribusi *Linux* berbasis *Debian* yang berfokus pada *penetration testing* lanjutan dan *ethical hacking*. Ini berisi beberapa ratus alat yang ditujukan untuk berbagai tugas keamanan informasi, seperti pengujian penetrasi, pemeriksaan keamanan, komputer forensik dan rekayasa balik.

Menurut (Arjunsinh Parmar & Kunal M. Pattani, 2017), *Kali Linux* adalah distribusi *Linux* yang diturunkan dari *Debian* yang dirancang untuk forensik digital dan *penetration testing*.

Menurut (Muhammad Idham Rusdi & Dianradika Prasti, 2019), *Kali Linux* merupakan pembangunan kembali *BackTrack Linux* secara sempurna, mengikuti sepenuhnya kepada standar pengembangan *Debian*.

Berdasarkan pendapat para ahli di atas dapat disimpukan bahwa *Kali Linux* merupakan sitem operasi dari turunan *Debian*. *Kali linux* dirancang untuk pengujian keamanan dan *penetration testing* pada sebuah aplikasi.

**A.5. Pengertian *Penetration Testing* (Uji Penetrasi)**

*Penetration Testing* merupakan sebuah aktivitas pengujian keamanan dari sebuah sistem. Dari hasil pengujian tersebut, didapatkan sejumlah celah keamanan pada sistem yang kemudian menjadi bahan rekomendasi kepada organisasi yang memiliki sistem tersebut untuk dibenahi.

Menurut (Sahren Sahren, 2019), *Penetration Testing* merupakan metode evaluasi keamanan sistem komputer atau jaringan dengan mensimulasikan serangan dari sumber yang berbahaya dan merupakan bagian dari security audit.

Menurut (Yum Thurfah Afifa Rosaliah, 2021), *Penetration Testing* adalah salah satu cara untuk mensimulasikan metode yang mungkin akan digunakan oleh penyerang untuk menghindari atau menerobos mekanisme keamanan dan mendapatkan akses secara ilegal ke dalam suatu sistem.

Menurut (Devi Christiani Angir, 2015), *Penetration Testing* adalah metode untuk mengevaluasi keamanan sistem komputer atau jaringan dengan mensimulasikan serangan dari sumber yang berbahaya.

Menurut (Samsumar & Gunawan, 2017), *Penetration Testing* adalah sebuah metode untuk melakukan evaluasi terhadap keamanan dari sebuah sistem dan jaringan komputer. Evaluasi dilakukan dengan cara melakukan sebuah simulasi serangan (*attack*).

Menurut (Yunanri Yunanri, 2016), *Penetration testing* adalah metode evaluasi keamanan pada sistem komputer atau jaringan dengan mengidentifikasikan kelemahan, *vulnerabilities* dan *the absence of patches.*

Berdasarkan pendapat dari para ahli diatas, dapat disimpulkan bahwa *Penetration Testing* merupakan simulasi berupa penyerangan terhadap sistem layaknya dilakukan oleh seorang *hacker* untuk memastikan adanya celah keamanan tersebut..

**A.6. Pengertian *Black Box Testing***

*Black Box Testing* merupakan teknik yang membutuhkan keahlian dari seorang penguji untuk melakukan penyerangan terhadap sistem. Pada skenario pengujian ini, penguji akan berperan sebagai seorang *hacker* yang melakukan penyerangan dari luar, maupun sebagai seorang *hacker* yang telah berhasil memanfaatkan jaringan internal organisasi. Dalam pelaksanaannya, penguji tidak diberikan informasi apapun mengenai sistem yang akan diuji, baik informasi mengenai arsitektur jaringan maupun konfigurasi sistem. Proses ini dapat dilakukan dari luar maupun dari dalam wilayah dimana sistem tersebut berada.

Menurut (M. Sidi Mustaqbal et al., 2015), *Black Box Testing* berfokus pada spesifikasi fungsional dari perangkat lunak. *Tester* dapat mendefinisikan kumpulan kondisi *input* dan melakukan pengetesan pada spesifikasi fungsional program.

Menurut (Sholeh et al., 2021), *Black Box Testing* adalah metode pengujian yang berfokus pada persyaratan fungsional perangkat lunak*.*

Menurut (Taufik Hidayat & Mahmudin Muttaqin, 2018), *Black Box Testing* merupakan pengujian yang berfokus pada spesifikasi fungsional dari perangkat lunak, *tester* dapat mendefinisikan kumpulan kondisi *input* dan melakukan pengetesan pada spesifikasi fungsional program.

Menurut (Umi Salamah & Fata Nidaul Khasanah, 2017), *Black Box Testing* merupakan salah satu jenis metode pengujian yang memperlakukan perangkat lunak yang tidak diketahui kinerja internalnya. Sehingga para *tester* memandang perangkat lunak seperti layaknya sebuah “kotak hitam” yang tidak penting dilihat isinya, tapi cukup dikenali proses testing dibagian luar.

Menurut (Setiyani, 2019), *Black Box Testing* bertujuan untuk menemukan fungsi yang tidak benar , kesalahan antarmuka, kesalahanpada struktur data , kesalahan perfomansi , kesalahan inisialisasi dan terminasi.

Berdasarkan pendapat dari para ahli diatas, dapat disimpulkan bahwa *Black Box Testing* merupakan pengujian yang dilakukan tanpa informasi yang lengkap dan mencoba mencari informasi secara lengkap pada sistem yang ada didalamnya.

**A.7. Pengertian *Vulnerability***

*Vulnerability* adalah suatu titik lemah atau kelemahan dalam sebuah prosedur keamanan kontrol administratif, kontrol internet, dan lain-lain sebagai yang dapat dieksploitasi melalui suatu trik untuk memperoleh akses yang tidak sah terhadap informasi atau untuk mengganggu proses secara kritis. Dalam sebuah sistem, vulnerability dapat dikatakan sebagai sebuah kelemahan yang dapat dimanfaatkan oleh pihak tidak bertanggung jawab untuk melanggar kebijakan keamanan sistem. Setiap *website* berpotensi memiliki *vulnerability*, maka dari itu tidak ada jaminan bahwa suatu *website* benar-benar aman dari segala celah.

Penguji (*tester*) harus melakukan pengujian (*penetration testing*) terlebih dahulu untuk menemukan *vulnerability* atau kelemahan dalam suatu *website*. Ada 2 metode untuk melakukan pengujian, yakni *white box testing* dan *black box testing*. *White box testing* adalah sebuah metode pengujian dimana penguji mengetahui semua informasi yang diperlukan dari sebuah *website* untuk melakukan pengujian. Sedangkan *black box testing* adalah sebuah metode pengujian dimana penguji tidak mengetahui sama sekali tentang informasi dan infrastruktur dari sebuah target. Dengan demikian pada *black box testing*, penguji harus menggali informasi yang dibutuhkan dari awal kemudian melakukan analisis serta menentukan jenis serangan yang akan digunakan dalam proses pengujian.

Menurut (Didi Juardi, 2017), *Vulnerability* adalah suatu poin kelemahan dimana suatu sistem rentan terhadap serangan. Sebuah ancaman (*threats*) adalah suatu hal yang berbahaya bagi keberlangsungan sistem.

Menurut (Devi Christiani Angir, 2015), *Vulnerability* adalah suatu kelemahan yang mengancam nilai *integrity, confidentiality,* dan *availability* dari suatu aset.

Menurut (Muhammad Athallariq Rabbani et al., 2020), *Vulnerability* merupakan kelemahan dalam suatu sistem yang memungkinkan penyerang untuk menjalankan sistem, mengakses data yang tidak sah, dan dapat melakukan serangan.

Menurut (Dias Utomo et al., 2017), *Vulnerability* adalah segala sesuatu yang berjalan pada komputer yang dapat secara langsung atau tidak langsung memicu kebocoran kerahasiaan, integritas, ketersediaan informasi atau layanan di manapun pada jaringan.

Menurut (Agung Tri Laksono & Joko Dwi Santoso, 2021), *Vulnerability* adalah sebuah titik kelemahan yang dapat dieksploitasi melalui akses tidak sah ke informasi dalam prosedur keamanan, administratif, kontrol, internet, dan sebagainya.

Berdasarkan pendapat dari para ahli diatas, dapat disimpulkan bahwa *Vulnerability* merupakan celah atau kerentanan yang memungkinkan dapat dimanfaatkan oleh *hacker*.

**A.8. Pengertian *Hacker***

*Hacker* adalah orang yang masuk ke sistem komputer orang lain tanpa izin sang pemilik.Tujuannya adalah untuk mengakses data tertentu atau melakukan berbagai tindakan ilegal lainnya.

Menurut (Antoni, 2018), *Hacker* sendiri adalah orang/pelaku yang gemar ngoprek komputer, memiliki keahlian membuat dan membaca program tertentu serta terobsesi mengamati keamanannya.

Menurut (Sadino & Liviana Kartika Dewi, 2016), *Hacker* adalah orang yang senang memprogram dan percaya bahwa informasi adalah sesuatu hal yang sangat berharga dan ada yang bersifat dapat dipublikasikan dan rahasia.

Menurut (Faysal Banua Suwiknyo et al., 2021), *Hacker* adalah orang yang menggunakan keterampilan yang dimiliki untuk melakukan berbagai aksi kejahatan siber, mulai merusak sistem, data pribadi, hingga mengekspos data yang diperoleh ke publik.

Menurut (Yunanri Yunanri, 2016), *Hacker* merupakan seseorang yang memiliki kemampuan dalam pemrograman serta jaringan komputer.

Menurut (Yanuardi Longgo & Junus J. Beliu, 2017), *Hacker* adalah orang yang memiliki keinginan yang kuat untuk mengetahui atau mempelajari suatu sistem komputer secara detail dan bagaimana cara meningkatkan kapabilitasnya.

Berdasarkan pendapat dari para ahli diatas, dapat disimpulkan bahwa *Hacker* adalah orang yang coba menguji sistem keamanan komputer tanpa izin dan bertujuan untuk mengakses data, merusak sistem, mengekspos data yang diperoleh ke publik.

**A.9. Pengertian UML**

UML (*Unified Modelling Language*) adalah suatu metode dalam pemodelan secara visual yang digunakan sebagai sarana perancangan sistem berorientasi objek. Awal mulanya, UML diciptakan oleh *Object Management Group* dengan versi awal 1.0 pada bulan Januari 1997.

UML juga dapat didefinisikan sebagai suatu bahasa standar visualisasi, perancangan, dan pendokumentasian sistem, atau dikenal juga sebagai bahasa standar penulisan *blueprint* sebuah *software.*

Menurut (Ahmat Josi, 2017), UML (*Unified Modelling Language*) adalah standart bahasa untuk mendefinisikan dari *requirement*, membuat analisa & desain dan menggambarkan arsitektur dalam pemrograman yang berorientasi pada objek.

Menurut (M. Ramaddan Julianti et al., 2019), UML (*Unified Modelling Language*) merupakan teknik pengembangan sistem yang menggunakan bahasa grafis sebagai alat untuk pendokumentasian dan melakukan spesifikasi pada sistem.

Menurut (Embun Fajar Wati & Arvin Anggoro Kusumo, 2016), UML (*Unified Modelling Language*) adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma ‘berorientasi objek”.

Menurut (Muhamad Tabrani. & nsan Rezqy Aghniya, 2019), UML adalah sebuah teknik pengembangan sistem yang menggunakan bahasa grafis sebagai alat untuk pendokumentasian dan melakukan spesifikasi pada sistem.

Menurut (Rachmat Agusli et al., 2017), UML (*Unified Modelling Language*) merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung.

Berdasarkan pendapat dari para ahli diatas, dapat disimpulkan bahwa UML (*Unified Modeling Language*) dapat diartikan sebagai bahasa visual untuk menggambarkan definisi-definisi tentang *requirement*, membuat analisis dan desain serta menggambar arsitektur dalam pemrograman berorientasikan objek dengan menggunakan teks-teks pendukung.

Berikut Jenis-Jenis Diagram UML

1. *Use Case Diagram*

*Use Case Diagram* adalah satu jenis dari *diagram* UML (*Unified Modelling Language*) yang menggambarkan hubungan interaksi antara sistem dan aktor. *Use Case* dapat mendeskripsikan tipe interaksi antara si pengguna sistem dengan sistemnya. *Use Case* merupakan sesuatu yang mudah dipelajari. Langkah awal untuk melakukan pemodelan perlu adanya suatu *diagram* yang mampu menjabarkan aksi aktor dengan aksi dalam sistem itu sendiri, seperti yang terdapat pada *Use Case*.

1. *Aktor* merupakan orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
2. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

Berikut adalah simbol-simbol yang ada pada *diagram use case* :

**Tabel 2.17** *Use Case Diagram*

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Gambar** | **Nama** | **Keterangan** |
| 1 |  | Actor | Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan use case. |
| 2 |  | Dependency | Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (independent) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (independent). |
| 3 |  | Generalization | Hubungan dimana objek anak (descendent) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (ancestor). |
| 4 |  | Include | Menspesifikasikan bahwa use case sumber secara eksplisit. |
| 5 |  | Extend | Menspesifikasikan bahwa use case target memperluas perilaku dari use case sumber pada suatu titik yang diberikan. |
| 6 |  | Association | Apa yang menghubungkan antara objek satu dengan objek lainnya. |
| 7 |  | Sistem | Menspesifikasikan paket yang menampilkan sistem secara terbatas. |
| 8 |  | Use Case | Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu *actor* |
| 9 |  | Collaboration | Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi). |
| 10 |  | Note | Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi |

1. *Activity* *Diagram*

*Activity* *Diagram* atau dalam bahasa Indonesia berarti *diagram* aktivitas, merupakan sebuah *diagram* yang dapat memodelkan berbagai proses yang tejadi pada sistem. Seperti layaknya runtutan proses berjalannya suatu sistem dan digambarkan secara vertikal. *Activity* *Diagram* adalah salah satu contoh *diagram* dari UML dalam pengembangan dari *Use Case*.

Berikut adalah simbol-simbol yang ada pada diagram aktivitas :

**Tabel 2.18** *Activity Diagram*

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Gambar** | **Nama** | **Keterangan** |
| 1 | No Gambar Nama Keterangan  1  Actor Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan use case.  2  Dependency Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (independent) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (independent).  3  Generalization Hubungan dimana objek anak (descendent) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (ancestor).  4  Include Menspesifikasikan bahwa use case sumber secara eksplisit.  5  Extend Menspesifikasikan bahwa use case target memperluas perilaku dari use case sumber pada suatu titik yang diberikan. | *Activity* | Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain. |
| 2 |  | *Action* | *State* dari sistem yang mencerminkan eksekusi dari suatu aksi. |
| 3 |  | *Initial Node* | Bagaimana objek dibentuk atau diawali. |
| 4 |  | *Activity Final Node* | Bagaimana objek dibentuk dan diakhiri. |
| 5 |  | *Decision* | Digunakan untuk menggambarkan suatu keputusan/tindakan yang harus diambil pada kondisi tertentu. |
| 6 |  | *Line Connector* | Digunakan untuk menghubungkan satu simbol dengan simbol lainnya. |

1. *Sequence* *Diagram*

*Sequence* *Diagram* merupakan *diagram* yang menjelaskan interaksi objek berdasarkan urutan waktu. *Sequence* dapat menggambarkan urutan atau tahapan yang harus dilakukan untuk dapat menghasilkan sesuatu, seperti yang tertera pada *Use Case diagram*.

Berikut adalah simbol-simbol yang ada pada *diagram* *sequence* :

**Tabel 2.19** *Sequence Diagram*

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Gambar** | **Nama** | **Keterangan** |
| 1 | Description: https://1.bp.blogspot.com/-Rf0Vx9qxKrE/WH9l4Y3kteI/AAAAAAAABlU/TK86L_dPGusW3_4IRR30hF33zfoB11VNQCLcB/s1600/Pengertian%2Bdan%2BSimbol%2BSequence%2BDiagram%252C%2BBoxes%252C%2BMessage%2Bdan%2BLooping%2BLogic.jpg | *Boundary Class* | Menangani komunikasi antara lingkungan sistem dan ke dalam sistem. Mereka dapat menjamin *interface* ke pengguna atau sistem lain |
| 2 | Description: Description: Description: Simbol Control Class dari Sequence Diagram | *Control Class* | Memodelkan urutan kelakukan (*behavior*) khusus untuk satu atau lebih *use case*. Pada awal phasa *Elaboration*,sebuah *control class* ditambahkan untuk setiap pasangan *actor* atau *use case*. Control class bertanggung jawab untuk aliran kejadian-kejadian dalam *use case*. |
| 3 | Description: Description: Description: Simbol Entity Class dari Sequence Diagram | *Entity Class* | Biasanya merupakan ***class*** yang dibutuhkan *system* untuk menyelesaikan beberapa kewajiban. ***Entity class***biasanya ditemukan dalam phasa eloborasi. |
| 4 | Description: Description: Description: https://3.bp.blogspot.com/-LXyyH1G5Vzo/WH9mv4oTnFI/AAAAAAAABlo/LN_Io3nMrzg7h_M7pH9sK3wH_sJlMJcNgCLcB/s1600/Pengertian%2Bdan%2BSimbol%2BSequence%2BDiagram%252C%2BBoxes%252C%2BMessage%2Bdan%2BLooping%2BLogic.jpgDescription: Description: Description: https://2.bp.blogspot.com/-nKoGCp3TQHc/WH9mkdHdvcI/AAAAAAAABlk/gBKFrNu5UaUr-biAVEm27V6yQgiEI69VwCLcB/s1600/Pengertian%2Bdan%2BSimbol%2BSequence%2BDiagram%252C%2BBoxes%252C%2BMessage%2Bdan%2BLooping%2BLogic.jpg | *Message* dengan garis berpanah menunjukkan arah *message* | *Message* digunakan untuk mengambarkan hubungan antara objek yang ada dalam sistem. |

1. *Class* *Diagram*

*Class* *Diagram* atau *diagram* kelas merupakan suatu *diagram* yang digunakan untuk menampilkan kelas-kelas berupa pake-paket untuk memenuhi salah satu kebutuhan paket yang akan digunakan nantinya.

Namun, pada Class *diagram* desain modelnya dibagi menjadi 2 bagian. Class *diagram* yang pertama merupakan penjabaran dari *domain* model yang merupakan abstraksi dari basis data. *Class Diagram* yang kedua merupakan bagian dari modul program MVC pattern (*Model View Controller*), dimana terdapat *class boundary* sebagai *class* *interface*, *class control* sebagai tempat ditemukannya algoritma, dan *class entity* sebagai tabel dalam basis data dan *query* program.

1. Atribut merupakan *variable-variabel* yang dimiliki oleh suatu kelas.
2. Operasi atau metode adalah fungsi yang dimiliki oleh suatu kelas.

Berikut adalah simbol-simbol yang ada pada diagram kelas :

**Tabel 2.20** *Class Diagram*

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Gambar** | **Nama** | **Keterangan** |
| 1 | Description: Description: Description: Nama Kelas di Class Diagram | Kelas pada struktur *system* | Simbol ini adalah simbol untuk sebuah kelas pada struktur sistem. Penulisan di sana tidak diperbolehkan menggunakan spasi. Simbol ini memiliki 3 susunan, yaitu kotak pertama adalah nama kelas, kedua atribut dan terakhir operasi. |
| 2 | Description: Description: Description: Interface dalam class diagram | Antarmuka/*interface* | Lingkaran ini adalah simbol untuk *interface* atau dalam bahasa indonesianya antar muka. Konsep yang digunakan pun sama dengan pemrogramman berorientasi object (OOP). |
| 3 |  | *Association/*asosiasi | Simbol ini sering disebut dengan simbol *Association* atau dalam bahasa indonesia yaitu asosiasi. Garis ini adalah garis yang digunakan untuk menghubungkan atau merelasikan kelas satu dengan kelas yang lainnya dengan makna umum. |
| 4 |  | *indirected association*/ asosiasi berarah | Nama dari simbol ini adalah *indirected* *association* atau dalam bahasa indonesia adalah asosiasi berarah. Simbol ini merupakan simbol relasi antar kelas seperti yang di atas, namun yang membedakan pada relasi ini adalah cara penggunaannya. Simbol ini digunakan jika kelas yang satu digunakan oleh kelas yang lainnya. |

1. **Penelitian Sebelumnya**

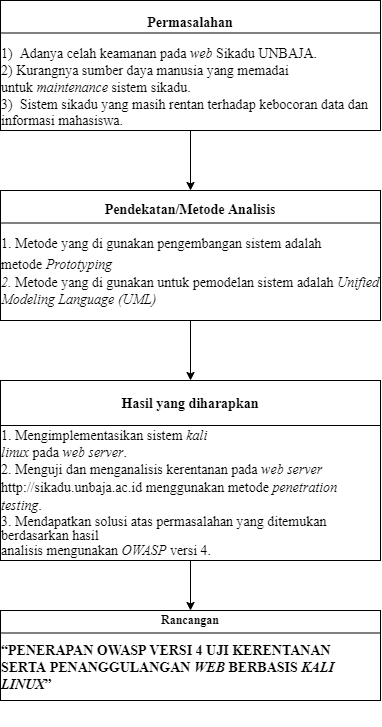
Penelitian-penelitian sebelumnya yang menjadi bahan rujukan dan referensi antara lain:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Judul Penelitian** | **Penulis, Tahun Terbit, Judul Jurnal** | **Permasalahan Penelitian** | **Metode Penelitian** | **Hasil Penelitian** |
| 1 | **Analisis Keamanan *Website* *Open Journal System* Menggunakan Metode *Vulnerability Assessment*** | (Riadi et al., 2020)  , Jurnal Teknologi Informasi dan Ilmu Komputer | Permasalahan yang dihadapi pada sistem OJS meliputi *network, port discover,* proses audit *exploit* sistem OJS. | *vulnerability assessment* | Hasil pengujian yang dilakukan menunjukkan bahwa pada OJS versi 2.4.7 memiliki banyak celah atau kerentanan tidak di rekomendasi untuk di gunakan, gunakanlah versi terbaru yang dikeluarkan oleh pihak OJS *Public knowledge project* (PKP). |
| 2 | **Analisis Perbandingan**  ***Penetration Testing Tool***  **Untuk Aplikasi**  ***Web*** | (Vito Tarigan et al., 2017), Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer | Aplikasi *web*  Yang dirancang dengan tidak baik, bahkan tidak dapat mengakses halaman posting –autentikasi. | *w3af, wapiti, dan Arachni.* | Hasil analisis yang didapat dengan menggunakan *tool w3af*, *wapiti*, dan  *Arachni* setelah melakukan 5 kali percobaan memberikan hasil tool  *Arachni* yang mendapatkan kerentanan pada identifikasi tersebut lebih banyak dari *tool w3af* dan *wapiti*. |
| 3 | **Uji Keamanan *Website* Terhadap Serangan *Path Traversal* Pada *Website* Pendataan Warga** | (Neng Ita Sopia Fazriani et al., 2019), Jurnal Riset dan Inovasi Pendidikan | Keamanan *website* dengan serangan *path traversal* yang ada pada *website* pendataan warga. | OWASP ZAP | Hasil implementasi serangan *Path Traversal* pada Aplikasi Pendataan Warga, yaitu ketika akses ke direktori tidak ditutup akan menjadi celah untuk melakukan serangan *Path Traversal* sehingga *attacker* bisa mengakses atau mengambil file penting dari direktori pada aplikasi tersebut. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 4 | **ANALISIS KEAMANAN *WEBSITE* DENGAN *INFORMATION SYSTEM SECURITY ASSESSMENT FRAMEWORK* (ISSAF) DAN *OPEN WEB APPLICATION SECURITY PROJECT* (OWASP)DI RUMAH SAKIT XYZ** | (Rochman et al., 2021), Jurnal Indonesia Sosial Teknologi | Kelemahan *webserver* Sistem Informasi HRD. | *INFORMATION SYSTEM SECURITY ASSESSMENT FRAMEWORK* (ISSAF) DAN *OPEN WEB APPLICATION SECURITY PROJECT* (OWASP) | Hasil pencarian celah keamanan (*vulnerability testing*) dan pengujian celah keamanan (*penetration testing*) ditemukan beberapa kelemahan yang terdapat pada *website* target. Kelemahan tersebut dapat *diexploitasi* hingga database target dapat diakses oleh pihak yang tidak berwenang atau tidak memiliki hak akses. |
| 5 | **Pengujian Celah Keamanan *Website***  **Menggunakan Teknik**  ***Penetration Testing***  **dan Metode OWASP TOP 10 pada**  **Website**  **SIM xxx** | (Yum Thurfah Afifa Rosaliah, 2021), Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA) | Kebocoran data suatu *website*  yang berisikan informasi para  *user* | OWASP TOP 10 | Hasil yang diperoleh, terdapat banyak  *username* dan *password* yang ditemukan dari hasil percobaan *bruteforce* menggunakan *hydra* dengan bantuan beberapa informasi *post* *http* dari hasil *scanning burproxy* yang dilakukan. |

Berdasarkan penelitian sebelumnya terdapat persamaan dan perbedaan dalam penelitian kali ini. Adapun persamaan dan perbedaan nya adalah :

1. **Persamaan dengan penilitian terdahulu**
2. Penelitian ini sama-sama membahas tentang menguji kerentanan *web server*.
3. Penelitian ini sama-sama membahas tentang pengamanan data informasi.
4. Penelitian ini sama-sama membahas tentang uji *penetration*.
5. **Perbedaan dengan penelitian terdahulu**
6. Perbedaan jenis metode yang digunakan untuk melakukan pengujian *penetration*.
7. Perbedaan *tools* yang dipakai untuk melakukan uji *penetration*.
8. Perbedaan versi aplikasi yang digunakan untuk uji *penetration*.
9. **Kerangka Berpikir**

Sebuah kerangka berpikir merupakan susunan dari pokok permasalahan hingga menghasilan suatu solusi. Kerangka berpikir penelitian ini dibuat dalam bentuk skema seperti gambar berikut. Berdasarkan hasil pengamatan yang dilakukan dan kajian teori yang telah disusun, maka selanjutnya dapatlah dibangun kerangka berpikir dapat dilihat pada gambar berikut:

**Gambar 2.6 Kerangka Berpikir**

**DAFTAR PUSTAKA**

Agung Tri Laksono, & Joko Dwi Santoso. (2021). Analysis of Website Security of SMKN 1 Pangandaran Against SQL Injection Attack Using OWASP Method. *The IJICS (International Journal Of Informatics and Computer Science)*, *5*, 209–216.

Ahmat Josi. (2017). PENERAPAN METODE PROTOTIPING DALAM PEMBANGUNAN WEBSITE DESA (STUDI KASUS DESA SUGIHAN KECAMATAN RAMBANG). *Jurnal Teknik Informatika*, *9*, 50–57.

Andrilla, R. (2014). *PENERAPAN STAKEHOLDER RELATIONSHIP MANAGEMENT PLUS (SRM+) DALAM PENGELOLAAN COMMUNITY DEVELOPMENT DI AREA OPERASIONAL TOTAL E&P INDONESIE*. *2*(3), 333–346.

Antoni, A. (2018). KEJAHATAN DUNIA MAYA (CYBER CRIME) DALAM SIMAK ONLINE. *Nurani: Jurnal Kajian Syari’ah Dan Masyarakat*, *17*(2), 261–274. <https://doi.org/10.19109/nurani.v17i2.1192>

Arjunsinh Parmar, & Kunal M. Pattani. (2017). Sniffing GSM Traffic Using RTL-SDR And Kali Linux OS . *International Research Journal of Engineering and Technology (IRJET)*, *4*(1).

*Arti kata terap2 - Kamus Besar Bahasa Indonesia (KBBI) Online*. (n.d.). Retrieved May 26, 2022, from <https://kbbi.web.id/terap-2>

Devi Christiani Angir, A. N. J. A. (2015). Vulnerability Mapping pada Jaringan Komputer di Universitas X. *JURNAL INFRA*, *3*, 1–7.

Dias Utomo, Muchammad Sholeh, & Arry Avorizano. (2017). Membangun Sistem Mobile Monitoring Keamanan Web Aplikasi Menggunakan Suricata dan Bot Telegram Channel. *Journal Uhamka.Ac.Id*, *2*, 81–87.

Didi Juardi. (2017). Kajian Vulnerability Keamanan Jaringan Internet Menggunakan Nessus. *SYNTAX Jurnal Informatika*, *6*, 11–19.

Eka Pratama, I. P. A., & Wiradarma, A. A. B. A. (2019). Open Source Intelligence Testing Using the OWASP Version 4 Framework at the Information Gathering Stage (Case Study: X Company). *International Journal of Computer Network and Information Security*, *11*(7), 8–12. <https://doi.org/10.5815/ijcnis.2019.07.02>

Ekonomi, J., Islam, B., Pratiwi, A., Ekonomi, F., & Samarinda, I. (2016). *Al-Tijary Pengaruh Kualitas Penerapan Good Corporate Governance (GCG) terhadap Kinerja Keuangan pada Bank Umum Syariah di Indonesia (Periode 2010-2015)*. *2*(1), 55–76. <https://doi.org/10.21093>

Embun Fajar Wati, & Arvin Anggoro Kusumo. (2016). Penerapan Metode Unified Modeling Language (UML) Berbasis Desktop Pada Sistem Pengolahan Kas Kecil Studi Kasus Pada PT Indo Mada Yasa Tangerang. *Jurnal Informatika*, *5*, 24–36.

Faysal Banua Suwiknyo, Tonny Rompi, & Harly Stanly Muaja. (2021). TINDAK KEJAHATAN SIBER DI SEKTOR JASA KEUANGAN DAN PERBANKAN. *Lex Privatum*, *9*, 183–192.

Fikriyadi Fikriyadi, Ritzkal Ritzkal, & Bayu Adhi Prakosa. (2020). Security Analysis of Wireless Local Area Network (WLAN) Network with the Penetration Testing Method. *Jurnal Mantik*, *4*, 1658–1662.

Henra Saputra Tanjung. (2019). PENERAPAN MODEL REALISTIC MATHEMATIC EDUCATION (RME) UNTUK MENINGKATKAN PEMAHAMAN KONSEP DAN HASIL BELAJAR SISWA KELAS XI SMAN 3 DARUL MAKMUR KABUPATEN NAGAN RAYA. *JURNAL ILMIAH PENDIDIKAN MATEMATIKA*, *6*, 101–112.

M. Ramaddan Julianti, Muhammad Iqbal Dzulhaq, & Ahmad Subroto. (2019). Sistem Informasi Pendataan Alat Tulis Kantor Berbasis Web pada PT Astari Niagara Internasional. *JURNAL SISFOTEK GLOBAL*, *9*, 92–97.

M. Sidi Mustaqbal, Roeri Fajri Firdaus, & Hendra Rahmadi. (2015). PENGUJIAN APLIKASI MENGGUNAKAN BLACK BOX TESTING BOUNDARY VALUE ANALYSIS (STUDI KASUS : APLIKASI PREDIKSI KELULUSAN SMNPTN). *Jurnal Ilmiah Teknologi Infomasi Terapan*, *3*, 31–36.

Muhamad Tabrani., & nsan Rezqy Aghniya. (2019). IMPLEMENTASI METODE WATERFALL PADA PROGRAM SIMPAN PINJAM KOPERASI SUBUR JAYA MANDIRI SUBANG. *Jurnal Interkom: Jurnal Publikasi Ilmiah Bidang Teknologi Informasi Dan Komunikasi*, *14*, 41–50.

Muhammad Athallariq Rabbani, Avon Budiyono, & Adityas Widjajarto. (2020). IMPLEMENTASI DAN ANALISIS SECURITY AUDITING MENGGUNAKAN OPEN SOURCE SOFTWARE DENGAN FRAMEWORK MITRE ATT&CK. *E-Proceeding of Engineering*, *7*, 7080–7087.

Muhammad Idham Rusdi, & Dianradika Prasti. (2019). Penetration Testing Pada Jaringan Wifi Menggunakan Kali Linux. *PROSIDING SEMANTIK*, *2*, 260–269.

Neng Ita Sopia Fazriani, Banta Cut, & Sanusi. (2019). Uji Keamanan Website Terhadap Serangan Path Traversal Pada Website Pendataan Warga. *Jurnal Riset Dan Inovasi Pendidikan*, *1*, 15–20.

Petar Cisar, & Robert Pinter. (2019). Some ethical hacking possibilities in Kali Linux environment. *Journal of Applied Technical and Educational Sciences (JATES)*, *9*, 129–149.

Rachmat Agusli, Sutarman, & Suhendri. (2017). Sistem Pakar Identifikasi Tipe Kepribadian Karyawan Menggunakan Metode Certainty Factor. *JURNAL SISFOTEK GLOBAL*, *7*, 21–27.

Reza Rizky Adha, Mochammad Fahru Rizal, & Setia Juli Irzal Ismail. (2021). MEMBANGUN SISTEM KEAMANAN JARINGAN BERBASIS FIREWALL DAN IDS MENGGUNAKAN TOOLS OPNSENSE. *EProceedings of Applied Science*, *7*, 2846–2856.

Riadi, I., Yudhana, A., & W, Y. (2020). Analisis Keamanan Website Open Journal System Menggunakan Metode Vulnerability Assessment. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *7*(4), 853–860. <https://doi.org/10.25126/jtiik.2020701928>

Rochman, A., Rohian Salam, R., & Agus Maulana, S. (2021). Analisis Keamanan Website dengan Information System Security Assessment Framework (Issaf) dan Open Web Application Security Project (Owasp) di Rumah Sakit Xyz. *Jurnal Indonesia Sosial Teknologi*, *2*(4), 506–519. <https://doi.org/10.36418/jist.v2i4.124>

Sadino, & Liviana Kartika Dewi. (2016). INTERNET CRIME DALAM PERDAGANGAN ELEKTRONIK. *Jurnal Magister Ilmu Hukum*, *1*, 9–17.

Sahren Sahren, R. A. D. M. A. (2019). Penetration Testing Untuk Deteksi Vulnerability Sistem Informasi Kampus. *Prosiding Senaris*, *1*, 994–1001.

Samsumar, L. D., & Gunawan, K. (2017). ANALISIS DAN EVALUASI TINGKAT KEAMANAN JARINGAN KOMPUTER NIRKABEL (WIRELESS LAN); STUDI KASUS DI KAMPUS STMIK MATARAM. *Jurnal Ilmiah Teknologi Infomasi Terapan*, *4*(1), 73–82. <https://doi.org/10.33197/jitter.vol4.iss1.2017.152>

Setiyani, L. (2019). PENGUJIAN SISTEM INFORMASI INVENTORY PADA PERUSAHAAN DISTRIBUTOR FARMASI MENGGUNAKAN METODE BLACK BOX TESTING. *Techno Xplore : Jurnal Ilmu Komputer Dan Teknologi Informasi*, *4*(1), 1–9. <https://doi.org/10.36805/technoxplore.v4i1.539>

Sholeh, M., Gisfas, I., Cahiman, & Fauzi, M. A. (2021). Black Box Testing on ukmbantul.com Page with Boundary Value Analysis and Equivalence Partitioning Methods. *Journal of Physics: Conference Series*, *1823*(1), 012029. <https://doi.org/10.1088/1742-6596/1823/1/012029>

Sri Endang Rahayu, H. F. (2016). PENERAPAN MODEL PEMBELAJARAN CONTEXTUAL TEACHING AND LEARNING (CTL) UNTUK MENINGKATKAN PEMAHAMAN MATERI PASAR VALUTA ASING PADA MATA KULIAH EKONOMI INTERNASIONAL 2 (STUDI MAHASISWA SEMESTER 5 JURUSAN EKONOMI PEMBANGUNAN FAKULTAS EKONOMI UMSU). *JURNAL ILMIAH MANAJEMEN DAN BISNIS*, *17*, 94–106.

Taufik Hidayat, & Mahmudin Muttaqin. (2018). Pengujian Sistem Informasi Pendaftaran dan Pembayaran Wisuda Online menggunakan Black Box Testing dengan Metode Equivalence Partitioning dan Boundary Value Analysis. *Jutis (Jurnal Teknik Informatika)*, *6*.

Umi Salamah, & Fata Nidaul Khasanah. (2017). Pengujian Sistem Informasi Penjualan Undangan Pernikahan Online Berbasis Web Menggunakan Black Box Testing. *Journal of Information Management*, *2*, 35–44.

Vito Tarigan, B., Kusyanti, A., & Yahya, W. (2017). *Analisis Perbandingan Penetration Testing Tool Untuk Aplikasi Web* (Vol. 1, Issue 3). <http://j-ptiik.ub.ac.id>

Wibowo, R. M., & Sulaksono, A. (2021). Web Vulnerability Through Cross Site Scripting (XSS) Detection with OWASP Security Shepherd. *Indonesian Journal of Information Systems*, 149–159. <https://doi.org/10.24002/ijis.v3i2.4192>

Yanuardi Longgo, & Junus J. Beliu. (2017). ANCAMAN DISINTEGRASI BANGSA MELALUI PEMANFAATAN MEDIA SOSIAL. *Jurnal Transformasi Sosial*, 39–52.

Yum Thurfah Afifa Rosaliah, J. J. B. H. (2021). Pengujian Celah Keamanan Website Menggunakan Teknik Penetration Testing dan Metode OWASP TOP 10 pada Website SIM. *Senamika*, *2*, 752–761.

Yunanri Yunanri, I. R. A. Y. (2016). Analisis Keamanan Webserver Menggunakan Metode Penetrasi Testing ( PENTEST ). *Annual Research Seminar (ARS)*, *2*, 300–304.