# **BAB II LANDASAN TEORI**

## Sistem Keamanan

Selalu ada aspek negatif dari sebuah pemanfaatan teknologi. Teknologi informasi tidak lepas dari masalah ini. Ada banyak manfaat dari teknologi informasi. Sayangnya salah satu aspek negatifnya adalah masalah keamanan (*security*).

Menurut Kizza (2005:49) :

*“Security is a continuous process of protecting an object from attack.”* (“Keamanan adalah proses berkelanjutan untuk melindungi suatu objek dari serangan.”)

Kemudian, menurut Salomon (2006:308) :

*“Computer security is a vast area that affects the performance of businesses, the quality of services provided by governments, and the daily lives of many on Earth (and perhaps elsewhere in the universe).”*

(“Keamanan komputer adalah area luas yang memengaruhi kinerja bisnis, kualitas layanan yang disediakan oleh pemerintah, dan kehidupan sehari-hari banyak orang di Bumi.”)

Keamanan terus menjadi ancaman yang lebih kompleks dan lebih besar. Keamanan semakin kompleks karena semakin banyak lubang keamanan yang ditemukan dan karena sistem operasi menjadi lebih rumit.

Menurut Rahardjo (2017:13) keamanan elemen tidak hanya pada jaringan saja, melainkan memiliki beberapa elemen yang lain, yang diantaranya komputer (*host*, server, *workstation*), jaringan (beserta perangkatnya), dan aplikasi (*software*, *database*).

### **Jenis Keamanan**

#### *Computer Security*

Menurut Rahardjo (2017), C*omputer security* adalah keamanan *computer*, yang terdiri dari server, *workstation*, *notebook*, dan sejenisnya. Seringkali disebut juga dengan *host security*. Permasalahan yang muncul pada keamanan komputer terkait dengan sistem operasi yang digunakan, *patch* yang terpasang, konfigurasi yang digunakan, serta keberadaan aplikasi yang ada. Seringkali sistem operasi yang digunakan sudah kadaluwarsa dan sudah ditemukan beberapa kerentanan (*vulnerability*) pada sistem operasinya yang memungkinkan terciptanya celah keamanan.

#### *Network Security*

Menurut Rahardjo (2017) saat ini sebagian besar sistem terhubung dengan jaringan (apapun jenis jaringannya). Ada beberapa masalah terkait dengan keamanan jaringan, seperti misalnya penyadapan data, modifikasi data, dan juga serangan *Denial of Service* terhadap jaringan dengan membanjiri jaringan dengan paket yang sangat banyak (sangat besar) yang menyebabkan jaringan menjadi *down*.

#### *Application Security*

Menurut Rahardjo (2017) boleh jadi komputer (server) yang digunakan sudah aman dan jaringan yang digunakan sudah aman, tetapi aplikasi (*software*) yang digunakan memiliki kerentanan sehingga data dapat diambil (atau diubah) oleh orang yang tidak berhak. Contoh yang sering terjadi saat ini adalah *SQL injection*. Terkait dengan *application* *security* adalah keamanan sistem *database*. Ternyata penanganan masalah keamanan *database* mirip dengan penanganan keamanan jaringan (bukan aplikasi).

### **Ancaman Keamanan**

Ancaman keamanan menurut pengertian dari Ariyus (2008:8) yang dapat terjadi terhadap informasi adalah :

#### *Interruption* (ancaman terhadap *availability* informasi, data yang ada dalam sistem komputer dirusak atau dihapus sehingga jika data atau informasi tersebut dibutuhkan maka pemiliknya akan mengalami kesulitan untuk mengaksesnya, bahkan mungkin informasi itu hilang.),

#### *Interception* (ancaman terhadap kerahasiaan informasi disadap sehingga orang yang tidak berhak dapat mengakses komputer di mana informasi tersebut disimpan.),

#### *Modification* (ancaman terhadap integritas. Orang yang tidak berhak berhasil menyadap lalu-lintas informasi yang sedang dikirim dan kemudian mengubahnya sesuai keinginan orang tersebut.),

#### *Fabrication* (ancaman terhadap integritas. Orang yang tidak berhak berhasil meniru atau memalsukan informasi sehingga orang yang menerima informasi tersebut menyangka bahwa informasi tersebut berasal dari orang yang dikehendaki oleh si penerima informasi.).

### **Aspek-Aspek Keamanan**

Sedangkan menurut pengertian dari Ariyus (2008:9) terdapat beberapa aspek keamanan yang diperlukan agar dapat terhindar dari tindakan kejahatan, diantaranya adalah :

#### Aspek *authentication* (penerima informasi dapat memastikan keaslian pesan, bahwa pesan itu datang dari orang yang dimintai informasi),

#### Aspek *integrity* (keaslian pesan yang dikirim melalui jaringan dan dapat dipastikan bahwa informasi yang dikirim tidak dimodifikasi oleh orang yang tidak berhak),

#### Aspek *nonrepudiation* (hal yang berhubungan dengan si pengirim, yang mana pengirim tidak dapat mengelak bahwa dialah yang mengirim informasi tersebut),

#### Aspek *authority* (informasi yang berada pada sistem jaringan tidak dapat dimodifikasi oleh pihak yang tidak berhak untuk mengaksesnya),

#### Aspek *confidentiality* (usaha untuk menjaga informasi dari orang yang tidak berhak mengakses),

#### Aspek *privacy* (lebih ke arah data-data yang bersifat pribadi),

#### Aspek *availability* (aspek availabilitas berhubungan dengan ketersediaan informasi ketika dibutuhkan),

#### Aspek akses *control* (aspek ini berhubungan dengan cara pengaturan akses ke informasi, biasanya yang mana berhubungan dengan masalah otentikasi dan privasi).

### **Permasalahan Mendasar Keamanan Internet**

Menurut Indrajit (2014:45) fakta dan statistik terjadinya sejumlah kecenderungan yang meningkat di dalam dunia maya, seperti : jumlah pengguna dan pelanggan yang semakin bertambah, nilai transaksi perdagangan yang meningkat nilainya, frekuensi transaksi yang meningkat tajam, tumbuh beranekaragamnya komunitas baru, dan lain sebagainya memanfaatkan internet.

Karena semakin banyak orang yang memanfaatkan internet, maka *value* dari dunia maya ini semakin meningkat. Akibatnya, semakin banyak pihak yang merasa berkepentingan dengan keberadaan internet, dari mereka yang ingin memanfaatkan berbagai peluang yang ada, hingga para kriminal yang ingin memperoleh keuntungan melalui perbuatan-perbuatan yang tidak baik.

## Kriptografi

Arti kata kriptografi berasal dari bahasa Yunani, yaitu kripto dan graphia. Arti dari kata *Kripto* adalah *secret* (rahasia) dan arti dari kata *graphia* adalah *writing* (tulisan).

### **Definisi Kriptografi**

Pengertian kriptografi menurut Ariyus (2008:13) :

“Ilmu dan seni untuk menjaga keamanan pesan ketika pesan dikirim dari suatu tempat ke tempat lain.”

Pengertian kriptografi menurut Salomon (2006:263) :

*“Cryptography is the art and science of making data impossible to read.”* “(Kriptografi adalah seni dan ilmu pengetahuan untuk membuat data tidak mungkin untuk dibaca.)”

Tugas dari berbagai metode enkripsi adalah mulai dengan data yang sederhana dan mudah dibaca (*plaintext*) dan mengacaknya, sehingga menjadi *ciphertext* yang tidak dapat dibaca. Setiap metode enkripsi juga harus menentukan bagaimana *ciphertext* dapat didekripsi kembali ke dalam *plaintext* asalnya.

Algoritma kriptografi terdiri dari tiga fungsi dasar, diantaranya adalah enkripsi, dekripsi, dan kunci.

### **Komponen Kriptografi**

Komponen kriptografi menurut Ariyus (2008:10) diantaranya adalah sebagai berikut:

#### Enkripsi (cara pengamanan data yang dikirimkan sehingga terjaga kerahasiaannya)

#### Dekripsi (kebalikan dari enkripsi)

#### Kunci (kunci yang dipakai untuk melakukan enkripsi dan dekripsinya, terdiri dari 2, yaitu kunci rahasia dan kunci umum)

#### *Ciphertext* (suatu pesan yang telah melalui proses enkripsi)

#### *Plaintext* (teks-asli atau teks-biasa merupakan pesan yang ditulis atau diketik yang memiliki makna)

#### Pesan (berupa data atau informasi yang dikirim)

#### *Cryptanalysis* (analisis kode atau suatu ilmu untuk mendapatkan teks-asli tanpa harus mengetahui kunci yang sah secara wajar)

### **Macam – Macam Kriptografi**

Macam-macam Algoritma Kriptografi menurut Ariyus (2008:44) dapat dibagi menjadi tiga bagian berdasarkan kunci yang dipakainya, yaitu:

#### Algoritma Simetris

Algoritma simetris atau sering disebut algoritma kriptografi konvensional adalah algoritma yang menggunakan kunci yang sama untuk proses enkripsi dan proses deskripsi. Algoritma kriptografi simetris dibagi menjadi dua kategori yaitu algoritma aliran (*Stream Ciphers*) dan algoritma blok (*Block Ciphers*). Dimana pada algoritma aliran, proses penyandiannya akan beriorientasi pada satu *bit/byte* data. Sedangkan pada algoritma blok, proses penyandiannya berorientasi pada sekumpulan *bit/byte* data (per blok). Adapun contoh algoritma kunci simetris adalah *DES (Data Encryption Standard), Blowfish, Twofish, MARS, IDEA, 3DES (DES* diaplikasikan 3 kali), *AES (Advanced Encryption Standard)* yang bernama *asli Rijndael*.

#### Algoritma Asimetris

Kriptografi asimetris adalah algoritma yang menggunakan kunci yang berbeda untuk proses enkripsi dan deskripsi. Dimana kunci enkripsi dapat disebarkan kepada umum dan dinamakan sebagai kunci publik *(public key),* sedangkan kunci deskripsi disimpan untuk digunakan sendiri dan dinamakan sebagai kunci pribadi *(private key).* Oleh karena itu, kriptografi ini dikenal pula dengan nama kriptografi kunci publik *(public key cryptography).* Adapun contoh algoritma yang menggunakan kunci asimetris adalah *RSA (Riverst Shamir Adleman)* dan *ECC (Elliptic Curve Cryptography).* Adapun pada kriptografi asimetris, dimana setiap pelaku sistem informasi akan memiliki sepasang kunci, yaitu kunci publik dan kunci pribadi, dimana kunci publik di distribusikan kepada umum, sedangkan kunci pribadi disimpan untuk diri sendiri. Artinya bila A ingin mengirimkan pesan kepada B, A dapat menyandikan pesannya dengan menggunakan kunci publik B, dan bila B ingin membaca surat tersebut, ia perlu mendeskripsikan surat itu dengan kunci privatnya. Dengan demikian kedua belah pihak dapat menjamin asal surat serta keaslian surat tersebut.

#### Hash Function.

*Hash function* adalah suatu fungsi yang berguna untuk mengkompresi/memperkecil sebuah *string* yang panjang menjadi sebuah *string* yang lebih pendek. Dalam dunia kriptografi, *hash function* bukan merupakan suatu barang yang baru. Merupakan salah satu cabang dalam kriptografi, *hash function* memiliki daya tarik tersendiri dikarenakan cukup banyak aplikasi yang menggunakan *hash function* dalam penerapannya. *Hash function* digunakan sebagai otentikasi, integritas dan *digital signature*, salah satu aplikasinya yaitu penggunaan *password* dalam aplikasi digital atau internet.

## Konsep Dasar Web

Pengertian Web menurut Sebesta (2014:7):

“adalah sebuah koleksi perangkat lunak dan protokol yang sudah dipasang pada sebagian komputer pada Internet.”

Sebagian komputer menjalankan web server, yang mana menyediakan dokumen, tapi kebanyakan menjalankan *web clients*, atau *browsers*, yang meminta dokumen dari server dan menampilkannya kepadapengguna.

Web terdiri dari beberapa komponen yang antara lain:

### **Komponen Web**

#### *HTML (Hypertext Markup Language)*

Menurut Wempen (2011) :

*“*Hypertext Markup Language (*HTML*) adalah bahasa markup yang mendasari dunia jaringan luas yang mana adalah utas bersama yang menyatukan hampir setiap situs web, dari situs perusahaan berskala besar seperti *Microsoft* hingga proyek kelas satu halaman di lokal sekolah dasar*.”*

### ***Golden Rules of Hypertext***

Menurut Shneiderman & Plaisant (2005:74) pembuatan desain terhadap *user interface* harus berdasarkan 8 prinsip aturan emas, yang diantara lainnya adalah :

#### *Strive for consistency*

Konsistensi yang dimaksud adalah konsistensi terhadap menggunakan ikon yang sudah dikenal, warna, hierarki menu, ajukan bertindak dan *user flow* untuk situasi serupa dengan konsisten. Tujuannya adalah membantu pengguna menjadi terbiasa dengan lanskap produk digital sehingga mereka dapat mencapai tujuan mereka dengan lebih mudah.

#### *Shortcuts*

Dengan tersedianya layanan *shortcuts*, maka akan memudahkan pengguna dalam mengoperasikannya. Selain dalam mengoperasikannya, mereka juga dapat menavigasinya dengan lebih cepat.

#### *Offer informative feedback*

Pengguna harus tahu setiap saat di mana mereka berada dan apa yang terjadi. Setiap tindakan harus ada *feedback* yang sesuai dan dapat dibaca pengguna dalam waktu yang wajar.

#### *Design dialogue to yield closure*

Dialog harus dimunculkan ketika tindakan yang mereka lakukan telah selesai, agar pesan yang disampaikan jelas dan tidak membuat pengguna menjadi bingung.

#### *Prevent errors*

Sebuah pesan yang baik menunjukkan kepada pengguna tentang apa yang terjadi, dan mengapa hal itu terjadi. Karena hal tersebut telah ditampilkan, maka pengguna mengerti bagaimana cara mencegah dan menyelesaikan masalah yang terjadi.

#### *Permit easy reversal of actions*

Fitur ini mengurangi kecemasan, karena pengguna tahu bahwa kesalahan dapat diurungkan, dengan demikian dapat mendorong eksplorasi lebih untuk pilihan yang tidak dikenal.

#### *Support internal locus of control*

Operator yang berpengalaman sangat menginginkan pengertian bahwa mereka bertanggung jawab atas antarmuka dan bahwa antarmuka merespons tindakan mereka. Tindakan antarmuka yang mengejutkan, urutan entri data yang membosankan, ketidakmampuan untuk mendapatkan atau kesulitan dalam memperoleh informasi yang diperlukan, dan ketidakmampuan untuk menghasilkan tindakan yang diinginkan semua membangun kecemasan dan ketidakpuasan.

#### *Reduce short-term memory load*

Manusia hanya mampu mempertahankan sekitar lima item dalam ingatan jangka pendek pada satu waktu. Oleh karena itu, antarmuka harus sesederhana mungkin dengan hierarki informasi yang tepat

## Algoritma *BASE64*

Pengertian Algoritma *BASE64* menurut Knudsen (1998:9):

“*BASE64* adalah sistem untuk mewakili *array* *byte* sebagai karakter *ASCII*.”

Menurut Knudsen, *BASE64* ini berguna, misalnya, ketika anda ingin mengirim data *byte* mentah melalui media, seperti *email*, yang mungkin tidak mendukung apa pun kecuali *ASCII* 7-*bit*.

### **Transformasi *BASE64***

Transformasi kriptografi *BASE64* menurut Sumartono, Siahaan, & Arpan (2016:64) banyak digunakan di dunia internet sebagai media data format untuk mengirimkan data, penggunaan tersebut dikarenakan hasil dari *encode* *BASE64* berupa *plain* *text*, maka data ini akan jauh lebih mudah dikirim, dibandingkan dengan format data yang berupa *binary*. Dalam implementasinya beberapa contoh transformasi *BASE64* yang antara lain sebagai berikut :

#### *PEM (Privacy-Enhanced Mail)* adalah protokol pertama dengan teknik *BASE64* yang didasarkan pada *RFC* *989* pada tahun 1987, yang terdiri dari 7 karakter (7-bit) yang digunakan pada *SMTP (Simple Mail Transfer Protocol)* dalam transfer data tapi untuk sekarang *PEM* sudah tidak menggunakan *RFC 989* tapi sudah diganti dengan *RFC 1421* yang menggunakan karakter A-Z, a-z, 0..9.

#### *MIME (Multi Purpose Mail Extension)* menggunakan "*BASE64*" sebagai salah satu dari dua skema pengkodean *binary to-text*. MIME didasarkan pada *PEM* versi *RFC* *1421*. Menggunakan alfabet 64-karakter adalah mekanisme penyandian yang sama dengan *PEM*, dan menggunakan simbol "=" untuk bantalan *output* dengan cara yang sama, seperti dijelaskan dalam *RFC 1521*.

#### *UTF-7* didasarkan pada *RFC 2152* yang umumnya disebut dengan *“MODIFICATION BASE”. UTF-7* menggunakan karakter *MIME*, tidak menggunakan *padding* “=”, karakter “=” digunakan sebagai *escape* untuk *encoding*.

## *URL (Uniform Resource Locator)*

### **Definisi *URL***

Pengertian *URL* menurut Harminingtyas (2014:43) :

“*URL* adalah alamat unik di dunia internet yang digunakan untuk mengidentifikasi sebuah *website*, atau dengan kata lain *domain name* adalah alamat yang digunakan untuk menemukan sebuah *website* pada dunia internet.”

Menurut Sebesta (2014:11) *URL* digunakan untuk mengidentifikasi dokumen (sumber) yang terletak pada Internet. Dan banyak jenis untuk sumber, identifikasi dengan banyak bentuk pada *URL*.

### **Ekstensi Domain *URL***

Contoh nama *domain* ber-ekstensi lokasi Negara Indonesia menurut Harminingtyas (2014:43) adalah:

#### *.co.id* : untuk badan usaha yang mempunyai badan hukum sah.

#### *.ac.id* : untuk lembaga pendidikan.

#### *.go.id* : khusus untuk Lembaga Pemerintahan Republik Indonesia.

#### *.mil.id* : khusus untuk Lembaga Militer Republik Indonesia.

#### *.or.id* : untuk segala macam organisasi yang tidak termasuk dalam kategori “ac.id”,”co.id”, “go.id”, “mil.id”, dan lain-lain.

#### .*war.net.id* : untuk industri warung internet di Indonesia.

#### .*sch.id* : khusus untuk Lembaga Pendidikan Indonesia yang menyelenggarakan pendidikan seperti SD, SMP, dan atau SMU.

#### *.web.id* : ditunjukkan untuk badan usaha, organisasi ataupun perseorangan yang melakukan kegiatannya di *World Wide Web*.

### **Format *URL***

Menurut Sebesta (2014:11) sebagian besar *URL* memiliki bentuk yang sama :

*scheme:object-address*

Skema sering digunakan sebagai protokol komunikasi. Kebanyakan skema terdiri sebagai berikut : *http, ftp, gopher, telnet, file, mailto*, dan *news*. Skema yang berbeda menggunakan alamat dari objek yang memiliki bentuk yang berbeda. Ketertarikan pada kali ini terhadap protokol *HTTP*, yang mana mendukung Web. Dan protokol ini meminta dan mengirimkan dokumen *HTML*.

*//fully-qualified-domain-name/path-to-document*

Skema yang digunakan juga berupa *file*. Protokol *file* mengartikan letak di mana dokumen itu berada pada mesin yang menjalankan *browser*. Pendekatan ini berguna untuk menguji dokumen yang tersedia pada Web tanpa membuat *file* tersebut tersedia untuk *browser* yang lain. Ketika *file* adalah protokol. Nama kualifikasi domain dihilangkan, membuat bentuk dari URL menjadi :

[*File://path-to-document*](File://path-to-document)

*URL* tidak dapat memiliki ruang yang disematkan. juga, ada kumpulan karakter khusus, termasuk titik koma, titik dua, dan tanda bintang (&), yang tidak dapat muncul di *URL*. Untuk memasukkan spasi atau salah satu karakter khusus yang tidak diizinkan, karakter tersebut harus dikodekan sebagai tanda persen diikuti oleh kode *ASCII* heksadesimal dua digit untuk karakter tersebut.

### ***Paths URL***

Path dokumen dari protokol *HTTP* mirip dengan path ke suatu *file* atau direktori pada sistem *file* dari sebuah sistem operasi dan diberikan oleh urutan nama direktori dan nama *file*, semua dibagi berdasarkan pemisah karakter pada sistem operasi yang digunakan.

*http://www.gumboco.com/files/f99/storefront.html*

Path dokumen dapat berbeda dari path untuk mengakses suatu *file* dikarenakan sebuah *URL* tidak perlu memasukkan semua direktori ke *Pathnya*. Sebuah *path* yang memasukkan semua direktori disebut sebagai *complete path.* Dalam sebagian besar kasus, *path* dari dokumen relatif terhadap beberapa jalur dasar yang spesifikasi terhadap konfigurasi *file* pada server. Sebagian *path* disebut sebagai partial *paths*.

*http://www.gumboco.com/storefront.html*

Jika spesifik dokumen adalah sebuah direktori daripada sebuah dokumen, nama direktori diikuti oleh sebuah *slash*.

[*http://www.gumboco.com/departments/*](http://www.gumboco.com/departments/)

Terkadang, sebuah direktori disimplikasikan tapi namanya tidak diberikan

[*http://www.gumboco.com/*](http://www.gumboco.com/)

## *SQL (Structured Query Language)*

### **Definisi *SQL***

Menurut Zam (2012):

“*SQL* adalah kependekan dari *Structured Query Language*, yang merupakan sebuah bahasa yang digunakan untuk mengakses data dalam *database* relasional.”

Sedangkan *database* realsional itu sendiri adalah sebuah *database* yang bekerja dengan menghubungakan data pada file-file yang berbeda menggunakan kunci atau elemen data yang umum. Sederhananya, data yang diolah oleh *SQL* berbentuk informasi yang disimpan dalam bentuk tabel. Setiap tabel memiliki *field-field* (kolom) tertentu untuk menyimpan data atau informasi.

*SQL* juga dikenal sebagai bahasa standar yang digunakan untuk mengakses *database* server. Bahkan saat ini hampir semua server *database* yang ada mendukung bahasa ini untuk melakukan manajemen datanya. *SQL* banyak diterapkan pada pemrograman berbasis *client-server*, seperti *PHP, ASP*, *Java J2EE*, dan sebagainya. Contoh *software SQL* yang terkenal adalah *MySQL, MsSQL (Microsoft), Oracle SQL, Postgre SQL (OpenSource).* Perintah dasar *SQL* pada dasarnya hampir sama, baik *MySQL, Postgre SQL,* *MsSQL*, atau *Oracle SQL.*

### **Perintah *SQL***

Secara umum, perintah-perintah *SQL* dibagi menjadi dua kelompok, yaitu sebagai berikut:

#### *DDL (Data Definition Language)*

Merupakan bagian dari *SQL* yang digunakan untuk mendefinisikan data dan objek *database*. Perintah tersebut berupa membuat, mengubah, menghapus *database* beserta objek lain dalam *database*, seperti tabel. Beberapa contoh perintah *SQL* yang termasuk *Data Definition Language*, seperti tabel 2.1 berikut ini.

**Tabel 2.1  
Perintah DDL**

|  |  |
| --- | --- |
| Perintah | Kegunaan |
| *Create Database* | Membuat *database* |
| *Create Table* | Membuat *tabel* |
| *Create Index* | Membuat *index* |
| *Alter Table* | Mengubah atau menyisipkan ke dalam tabel |
| *Drop Database* | Menghapus *database* |
| *Drop Table* | Menghapus tabel |
| *Drop Index* | Menghapus *index* |

Sumber : *SQL Injection*

#### *DML (Data Manipulation Language)*

Merupakan bagian dari perintah *SQL* yang digunakan untuk memanipulasi data yang ada dalam suatu tabel. Beberapa perintah *SQL* yang termasuk *Data Manipulation Language*, seperti terlihat pada tabel 2.2

**Tabel 2.2  
Perintah DML**

|  |  |
| --- | --- |
| Perintah | Kegunaan |
| *Select* | Menampilkan data |
| *Insert* | Menyisipkan / memasukkan data |
| *Delete* | Menghapus isi data |
| *Update* | Mengubah isi data yang sudah ada |

Sumber : *SQL Injection*

### **Penggunaan phpMyAdmin & MySQL dalam phpMyAdmin**

**Tabel 2.3  
Fitur *phpMyAdmin* & *MySQL* dalam *phpMyAdmin***

|  |  |
| --- | --- |
| phpMyAdmin | MySQL dalam phpMyAdmin |
| Membuat *Database* | Membuat *Database* |
| Membuat Tabel | Membuat Tabel |
| Memasukkan Data | Memasukkan Data |
| Melihat Isi Tabel | Memodifikasi Data |
| *Export & Import Database* | Menghapus Tabel |
| Menghapus Tabel | Menghapus *Database* |

Sumber : *SQL Injection*

Pada tabel 2.3, terdapat perbedaan antara fitur *phpMyAdmin* dengan *MySQL* dalam *phpMyAdmin*.

## *SQL Injection*

*SQL injection* terdiri dari dua kata, yaitu *SQL* merupakan sebuah bahasa yang digunakan untuk mengakses suatu basis data, sedangkan kata *injection* jika diterjemahkan memiliki arti menyuntik.

### **Definisi *SQL Injection***

Menurut Zam (2012) :

“*SQL injection* adalah sebuah metode untuk memasukan perintah *SQL* sebagai input melalui sebuah situs web guna mendapatkan akses basis data.”

Menurut Clarke (2012) :

“*SQL injection* adalah serangan di mana kode *SQL* dimasukkan atau ditambahkan ke parameter input aplikasi / pengguna yang kemudian diteruskan ke *back-end* server *SQL* untuk *parsing* dan eksekusi.”

Setiap prosedur yang membangun pernyataan *SQL* berpotensi menjadi rentan, karena sifat beragam *SQL* dan metode yang tersedia untuk membangunnya memberikan banyak opsi pengkodean. Bentuk utama dari *SQL Injection* ini terdiri dari penyisipan langsung kode ke dalam parameter yang digabungkan dengan perintah *SQL* dan dieksekusi. Serangan yang tidak langsung menyuntikkan kode berbahaya ke *string* yang ditujukan untuk penyimpanan di sebuah tabel atau sebagai *metadata*. Ketika *string* yang disimpan kemudian digabungkan menjadi perintah *SQL* dinamis, kode berbahaya dieksekusi. Ketika aplikasi web gagal membersihkan parameter yang dilewatkan ke pernyataan *SQL* yang dibuat secara dinamis (bahkan ketika menggunakan teknik parameterisasi) dimungkinkan bagi penyerang untuk mengubah konstruksi pernyataan *back-end SQL*. Ketika seorang penyerang dapat memodifikasi pernyataan *SQL*, pernyataan itu akan dieksekusi dengan hak yang sama dengan pengguna aplikasi. Saat menggunakan *SQL server* untuk menjalankan perintah yang berinteraksi dengan sistem operasi, proses akan berjalan dengan izin yang sama dengan komponen yang mengeksekusi perintah (misalnya server *database*, server aplikasi, atau server web), yang seringkali sangat istimewa.

### **Jenis S*QL Injection***

Pada dasarnya ada tiga jenis *SQL Injection*, yaitu sebagai berikut :

#### *Error Based SQL Injection*

*Error Based SQL Injection* dalam aksinya akan memberikan sebuah perintah ke *database* sehingga menampilkan pesan *error*. Dari pesan *error* tersebut dapat diperoleh informasi yang bisa dimanfaatkan.

Dari sisi metode yang digunakan, pada aksi *Error* *Based* *SQL* *Injection* dapat dikelompokkan menjadi dua metode, yaitu :

##### *GET Method*

Inilah yang umum atau banyak digunakan untuk memasukkan perintah *SQL* *Injection* melalui *URL* sebuah situs web pada *address* *bar*, sehingga metode ini dikenal juga dengan nama *URL* *Based* *SQL* *Injection*. Disebut metode *GET* karena menggunakan parameter *GET* untuk memasukkan atau menggunakan perintah *SQL* *Injection*.

##### *POST Method*

Metode *POST* berasal dari variabel yang ada pada *form*, contohnya seperti *form* *login* atau *form* pencarian. Cara untuk mengidentifikasinya juga sama, dengan memberikan karakter ‘ atau – atau karakter *ASCII* yang bisa menghasilkan *error*. Hal ini yang sering digunakan untuk mem-*bypass* sebuah halaman *login* tanpa mengetahui *username* maupun *password* yang digunakan.

#### *Union Based SQL Injection*

*Union Based SQL Injection* adalah metode *SQL Injection* perintah *UNION* untuk menggabungkan hasil dari dua atau lebih perintah *SELECT* menjadi sebuah hasil tunggal.

#### *Blind SQL Injection*

*Blind SQL Injection*, jenis ini tidak menampilkan pesan *error* dan tidak menampilkan data atau informasi yang ada, terkadang sedikit sulit untuk melakukan eksploitasi untuk jenis *Blind SQL Injection*. Hal ini karena prosesnya dengan memberikan pertanyaan pada *database* berupa kondisi *TRUE/FALSE* dan apakah dari halaman yang ditampilkan benar atau tidak.

## Penelitian Terdahulu

Terdapat penelitian terdahulu yang berkaitan dengan penerapan algoritma *BASE64*. Beberapa penelitian terdahulu yang telah dilakukan tersebut akan menjadi acuan dalam memperoleh informasi mengenai tema pembahasan penelitian yang dilakukan.

Pada tahun 2017, Annas Apreja dan Zulfadhli Syarif, Sistem Informasi Profesional dari Fakultas Ilmu Komputer Universitas Sriwijaya, serta Ali Ibrahim, Dosen Sistem Informasi Fakultas Ilmu Komputer Universitas Sriwijaya melakukan penelitian yang berjudul “Analisis Tingkat Keamanan Enkripsi Data Menggunakan Algoritma *BASE64* *Encode*”. Dalam penelitian tersebut, terdapat pembahasan yang menyatakan bahwa tingkat keamanan dari *BASE64* sendiri itu sudah cukup tinggi, dikarenakan harus melalui proses perubahan ke *binery*, dan menjadikannya ke sebuah *index* setelah *bit* dipecah.

Pada tahun 2018, Siswanto, dan M. Anif mahasiswa dari Universitas Budi Luhur dan Windu Gata dari STMIK Nusa Mandiri melakukan penelitian yang berjudul “Penerapan Algoritma Kriptografi *TEA* dan *BASE64* Untuk Mengamankan *Email*”. Dalam penelitian tersebut, peneliti membandingkan antara *Tiny Encryption Algorithm* dengan *BASE64* dari segi pengamanan, perbandingan ukuran file sebelum dan sesudah dienkripsi, sebelum dan sesudah didekripsi, dan faktor-faktor lainnya.

Penelitian tersebut akan penulis gunakan sebagai pedoman dalam penerapan sistem keamanan *SQL Injection* terhadap *URL Website* dengan menggunakan algoritma kriptografi *BASE64*.