# BAB III

# ANALISA DAN PERANCANGAN

## 3.1 Analisa Masalah

Era yang modern tentunya semakin banyak perusahaan atau instansi membutuhkan sebuah website yang akan dikelola guna untuk memajukan usaha dibidang apapun. Kerentanan *Cross-Site Scripting (XSS)* akan menjadi ancaman pada sebuah aplikasi web. Pengujian kerentanan *Cross-Site Scripting (XSS)* yang dilakukan menggunakan cara manual tidak akan efektif. Oleh karena itu dibutuhkan sebuah alat atau perangkat lunak yang bisa mendeteksi berbagai kerentanan *Cross-Site Scripting* (XSS) secara otomatis tanpa perlu menguji secara manual.

Kerentanan *Cross-Site Scripting* (XSS) perlu mendapatkan sebuah perhatian khusus dari sebuah perusahaan karena setiap tahunnya laporan kerentanan *Cross-Site Scripting* (XSS) ini kasusnya semakin meningkat. Jika kerentanan ini dibiarkan akan sangat beresiko pada sistem dari perusahaan itu sendiri.

## 3.2 Pemecahan Masalah

Berdasarkan analisa masalah tersebut, didapatkan pemecahan masalahnya yaitu dengan membuat sebuah tools atau perangkat lunak pemindai kerentanan *CrossSite Scripting (XSS)* secara otomatis untuk membantu pentester dalam melakukan pekerjaannya menemukan sebuah kerentanan *Cross Site Scripting (XSS).* Perangkat lunak yang dibuat berbasis *Command-Line Interface* (CLI) yang akan melakukan pemindaian secara otomatis dan dapat melakukan pencarian parameter pada website , membaca parameter, menemukan parameter yang valid, menemukan status WAF,

29

melakukan *bruteforce payload* Ketika dikirim ke website dan dapat menemukan payload yang valid. Dengan adanya solusi tersebut diharapkan dapat membantu pentester dalam melakukan tugasnya mencari kerentanan.

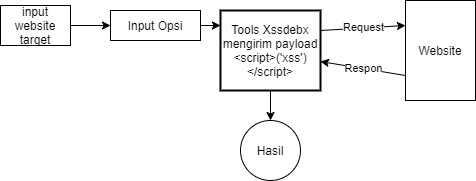
## 3.3 Perancangan Algoritma

Tahapan algoritma yang akan dilakukan untuk pembuatan *tools* pemindai kerentanan *Cross-Site Scripting* (XSS) berbasis *Command-Line Interface* (CLI) yaitu sebagai berikut.

#### 3.3.1 Perancangan Utama

Pembuatan tools pemindai kerentanan *Cross-Site Scripting* (XSS) berbasis *Command-Line Interface* (CLI) membutuhkan sebuah rancangan utama sebagai

berikut :



Gambar 3. 1 Perancangan Utama

1. Input website target

Input website target yang akan dilakukan pemindaian

1. Input Opsi

Input opsi berfungsi untuk melakukan pemindaian sesuai dengan opsi yang diinginkan

1. *Tools* Xssdebx

*Tools* akan melakukan pengiriman semua payload pada website

1. Website

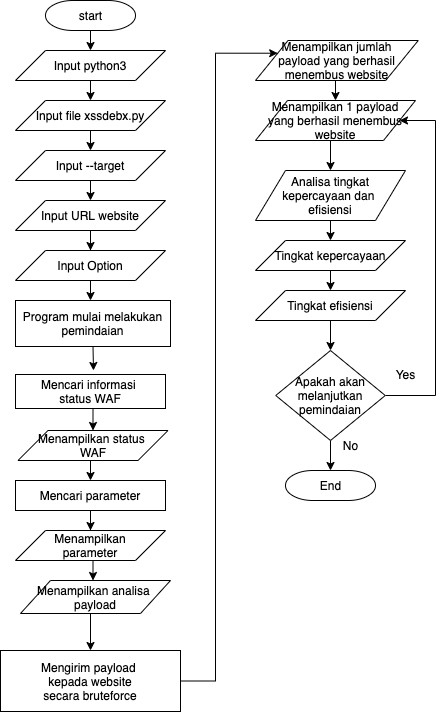
Website akan diinjeksi oleh payload yang akan dikirim dan respon dari website akan dikembalikan ke tools xssdebx.

1. Hasil

Hasil dari pemindaian pada fitur *relected* berupa status waf, parameter yang ditemukan, Analisa refleksi, Analisa *payload*, Analisa efisiensi. Lalu hasil untuk fitur *crawling* berupa nama *JQuery* yang ditemukan, versi *JQuery*, letak *JQuery* pada website tersebut, jumlah kerentanan dari *JQuery* yang ditemukan dan data CVE.

#### 3.3.1 *Flowchart*

• *Reflected*



###### Gambar 3.2 *Flowchart Reflected*

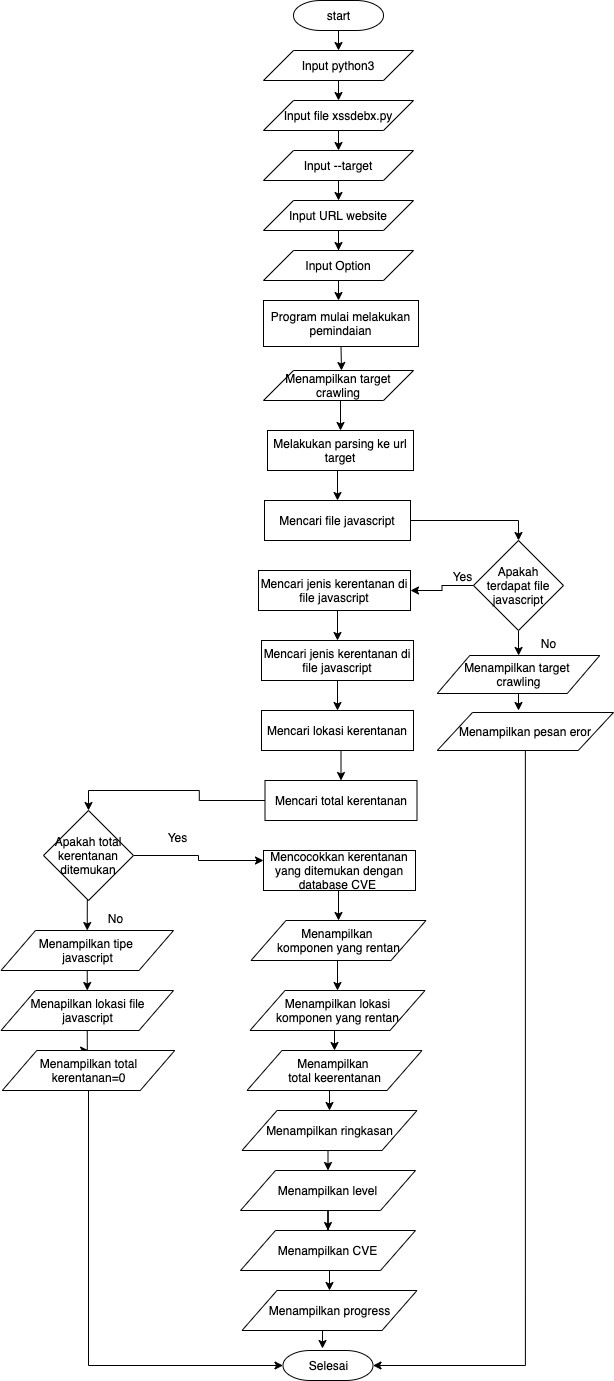
Keterangan :

*Pentester* menginputkan *python3* karena tools menggunakan bahasa pemprograman *python* dengan versi 3. Selanjutnya menginputkan xssdebx.py karena file yang akan dijalankan terlebih dahulu adalah xssdebx.py dan akan menghubungkan pada file-file lainnya. Lalu dilanjutkan dengan menginputkan - -target yang merupakan perintah untuk melakukan pemindaian berdasarkan opsi. Selanjutnya diinputkan url target yang akan dilakukan pemindaian. Terakhir inputkan opsi - -reflected lalu tekan enter.

Output dari hasil pemindaian yaitu menampilkan informasi waf menyala atau tidak.

Jika waf *offline* bukan berarti waf dari website target benar-benar tidak menyala. Kemungkinan tools tidak bisa mendeteksi waf dikarenakan waf tersebut sudah sangat aman sehingga tidak bisa diketahui status waf. Selanjutnya menampilkan pencarian parameter, jika parameter ditemukan maka akan menampilkan nama parameter tapi jika tidak maka akan menampilkan “ tidak ada parameter ditemukan”. Selanjutnya akan menampilkan Analisa refleksi yaitu menampilkan jumlah refleksi yang ditemukan dari suatu halaman website. Lalu dilanjutkan dengan menampilkan analisa payload yang terdiri dari total *payload* yang bisa digunakan, urutan *payload* yang ditampilkan, payload yang bisa digunakan. Lalu terakhir akan ditampilkan efisiensi dari payload yang akan digunakan untuk injeksi website.

• *CRAWLING*



**Gambar 3.3 *Flowchart Crawling***

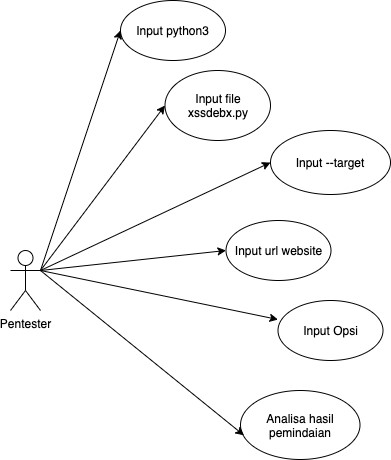
Keterangan:

*Pentester* menginputkan *python3* karena *tools* menggunakan bahasa pemprograman python dengan versi 3. Selanjutnya menginputkan xssdebx.py karena file yang akan dijalankan terlebih dahulu adalah xssdebx.py dan akan menghubungkan pada file-file lainnya. Lalu dilanjutkan dengan menginputkan - -target yang merupakan perintah untuk melakukan pemindaian berdasarkan opsi. Selanjutnya diinputkan url target yang akan dilakukan pemindaian. Inputkan opsi - -*crawling* untuk melakukan pemindaian *crawling*. Jika tidak terdapat file *JQuery* maka tools akan memunculkan “ *No JS Found*” jika terdapat file *JQuery* maka Selanjutnya *tools* akan mencari kerentanan berdasarkan versi *JQuery* yang digunakan lalu mencocokkan dengan database definitions.json. Setelah itu akan menampilkan elemen rentan seperti versi dari *JQuery* selanjutnya akan ditampilkan lokasi elemen rentan yaitu lokasi file *JQuery* dan dilanjutkan dengan total kerentanan dari *JQuery* yang telah ditemukan. Selanjutnya akan ditampilkan urutan kerentanan dari kerentanan yang telah ditemukan setelah itu akan ditampilkan ringkasan dari dari kerentanan yang telah ditemukan, dilanjutkan dengan level kerentanan dan yang terakhir menampilkan CVE dari kerentanan yang telah ditemukan.

## 3.4 Perancangan Sistem

Pada pembahasan kali ini akan dilakukan sebuah perancangan *tools* yang dikembangkan pada tugas akhir ini. Tujuan perancangan ini adalah untuk membuat sebuah alat pemindai kerentanan *Cross-Site Scripting (XSS)*. Sehingga *tools* yang dikembangkan dapat membantu pentester dalam melakukan pekerjaannya.

#### 3.4.1 *Use Case Diagram*



###### Gambar 3.4 *Use Case Diagram*

Keterangan gambar 10 Use case diagram adalah sebagai berikut :

* Input *python3*

Input *python3* berfungsi untuk memanggil bahasa pemprograman python karena tools yang akan dijalankan membutuhkan python3

* Input file xssdebx.py

Input file xssdebx.py berfungsi unutk memanggil file xssdebx.py karena inti dari tools terdapat pada file xssdebx.py

* Input -- target

Input -- target berfungsi untuk memanggil perintah pemindaian

* Input url website

Input url website diperlukan untuk target pemindaian

* Input opsi

Input opsi berfungsi untuk menjalankan pemindaian berdasarkan opsi masingmasing

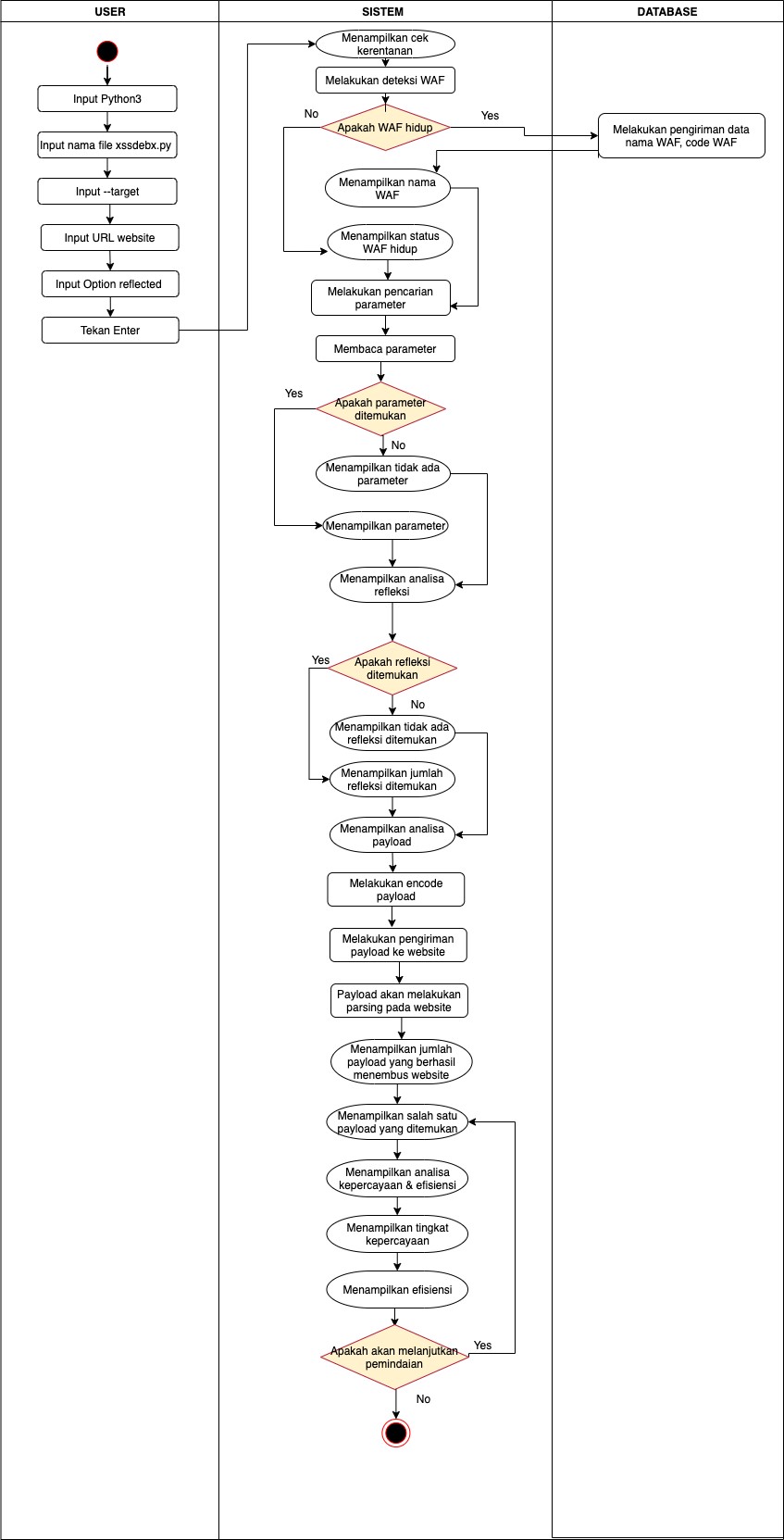
* Analisa Hasil Pemindaian

Setelah URL selesai dipindai, pentester menganalisa hasil pemindaian

#### 3.4.2 Activity Diagram

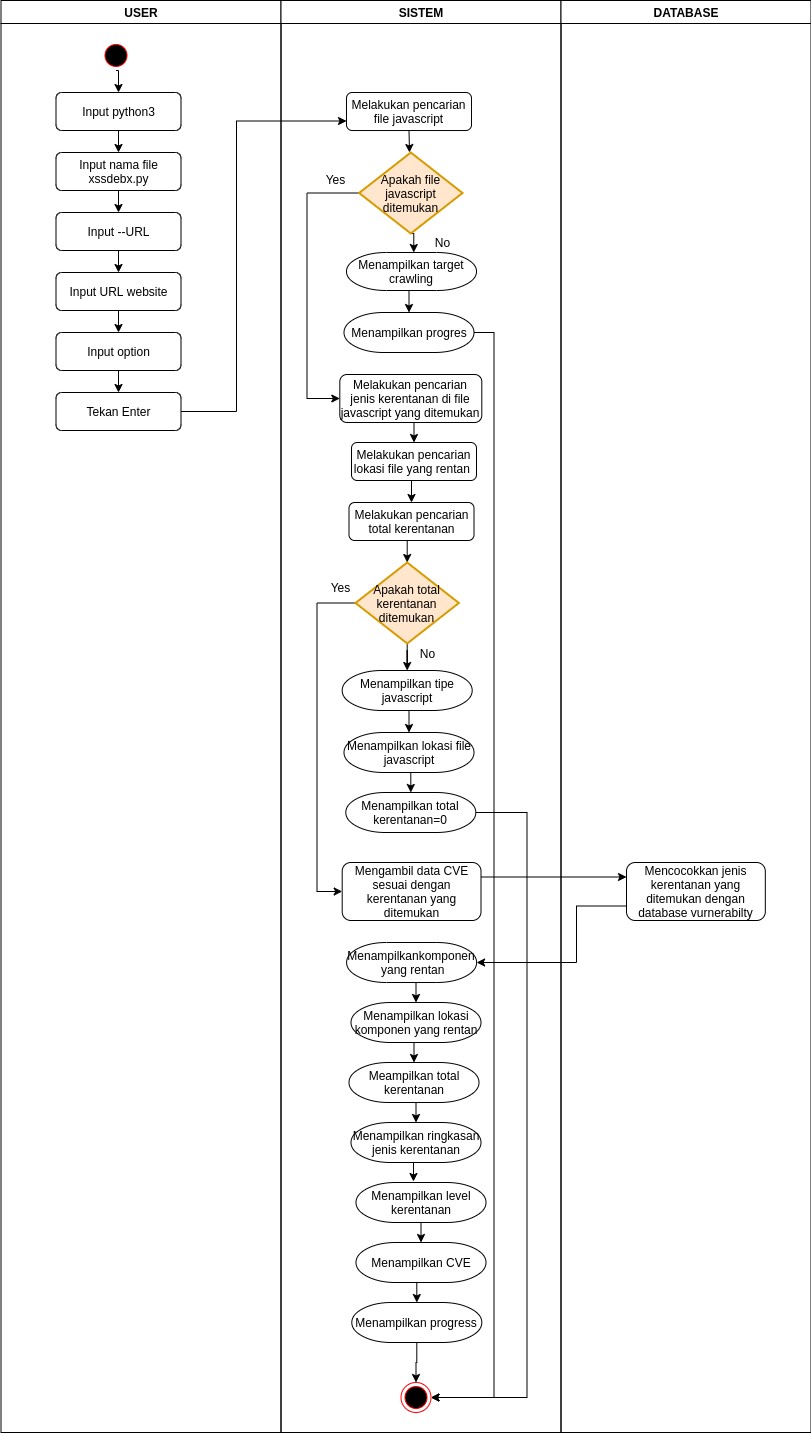
Terdapat activity diagram dari beberapa metode sebagai berikut

• *Reflected*



###### Gambar 3.5 *Activity Diagram Reflected*

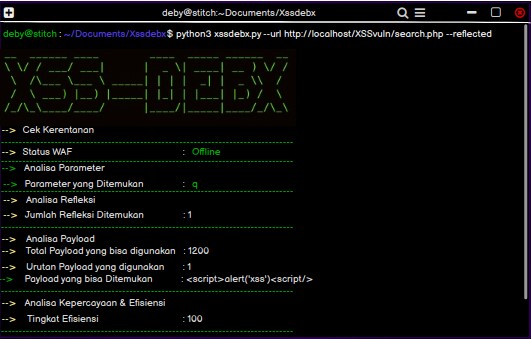
• *CRAWLING*



###### Gambar 3.6 *Activity Diagram Crawling*

## 3.5 *Desain Interface*

• *Reflected*



###### Gambar 3.7 *Desain Interface Reflected*

• *Crawling*



**Gambar 3.8 *Desain Interface Crawling***