Muhamad Hidayat Follow
Feb 2, 2022 · 5 min read

VulnLab XSS Reflected GET & POST (Basic Reflected)— Dynamic Application Security Testing #2

Assalamualaikum Wr.Wb

Vulnlab adalah sebuah web aplikasi yang di design vulnerable sebagai lab untuk praktik offensive security serta memiliki 10 kategori kerentanan dan lebih dari 30 lab siap testing, aplikasi tersebut dibuat oleh Yavuzlar, Tim Yavuzlar adalah tim keamanan cyber yang dibentuk dalam lingkup proyek Cyber Vatan dan bekerja dengan fokus pada keamanan web. Misinya adalah melaksanakan berbagai proyek untuk memastikan bahwa setiap anggota tim menjadi kompeten di bidangnya dan untuk meningkatkan jumlah proyek domestik di sektor keamanan siber negara turkey.

Lab tersebut memiliki 10 kategori kerentanan:

- SQL Injection
- Cross-site Scripting

- Insecure Direct Object Reference
- Command Injection
- XML External Entity Attack (XXE)
- File Inclusion
- Unrestricted File Upload
- Cross Site Request Forgery
- Insecure Deserialization
- Broken Authentication

Download: Yavuzlar/VulnLab (github.com)

Pada article kali ini saya ingin menjelaskan tentang lab dengan kategori **Cross-Site Scripting**, biasa disingkat XSS, vulnerability tersebut sedang menjadi sorotan banyak penggiat web application security dikarenakan menjadi vulnerability yang menempati urutan ketiga dalam kategori yang dibuat oleh **OWASP** yaitu **OWASP TOP 10.**

Owasp Top 10 Series — A3 (Injection Flaw) [Indonesia] | by Muhamad Hidayat | Jan, 2022 | Medium

Apa itu XSS?

Cross-Site Scripting (XSS) adalah sebuah kerentanan yang disebabkan karena aplikasi tidak melakukan encoding pada inputan yang di-input oleh user sehingga aplikasi & browser akan menjalankan kode javascript yang di-inputkan oleh user.

Payload XSS sendiri berisi kode javascript dengan perintah jahat / malicious intent, yang dapat melakukan pencurian data pada browser.

Terdapat beberapa jenis XSS berdasarkan letak input & output user:

Reflected XSS

Terjadinya XSS ketika kode javascript yang di input attacker ter-refleksikan langsung oleh browser, sifatnya temporer (sementara), karena hanya ter-refleksikan ketika payload tersebut di-inputkan saja.

Stored XSS

Merupakan berlawanan dari Reflected XSS, yaitu kondisi dimana attacker dapat menginput sebuah kode javascript pada aplikasi lalu akan tersimpan pada database, yang nanti nya dapat terus menerus dipanggil berulang.

DOM XSS

DOM XSS terjadi ketika inputan attacker di-refleksikan melalui fungsi DOM pada browser yang digunakan oleh Javascript.

Apa itu DOM?
Document Object Model (DOM) adalah sebuah API (Application Programming Interface) yang disediakan web browser kepada programmer untuk memanipulasi konten yang akan ditampilkan pada browser.

Bagaimana Reflected XSS Terjadi?

```
22  <?php
23
24  if (isset($_GET['q'])) {
25    $q = $_GET['q'];
26    echo '<div class="alert alert-danger" style="margin-top: 30vh;" role="alert" >';
27    echo '' . $strings['text'] . ' <b>' . $q . '</b> ';
28    echo '<a href="index.php" ">' . $strings['try'] . '</a>';
29    echo "</div>";
30  } else {
31    echo '<form method="GET" action="#" style="margin-top: 30vh;" class="row g-3 col-md-6 row justify-content-center mx-auto">';
32    echo '<input class="form-control" type="text" placeholder="' . $strings['search'] . '" name="q">';
33    echo '<button type="submit" class="col-md-3 btn btn-primary mb-3">' . $strings['s_button'] . '</button>';
34    echo '</form>';
35  }
36
37  ?>
```

Code Vulnerable terhadap XSS

Pada line ke 24, developer mencoba membuat pengkondisian apakah data dari inputan **$_GET[‘q’]** telah terinput atau tidak.

Pada line tersebut developer menerima inputan dari user dengan input user metode GET request, namun saat melakukan request inputan user tersebut tidak di sanitasi

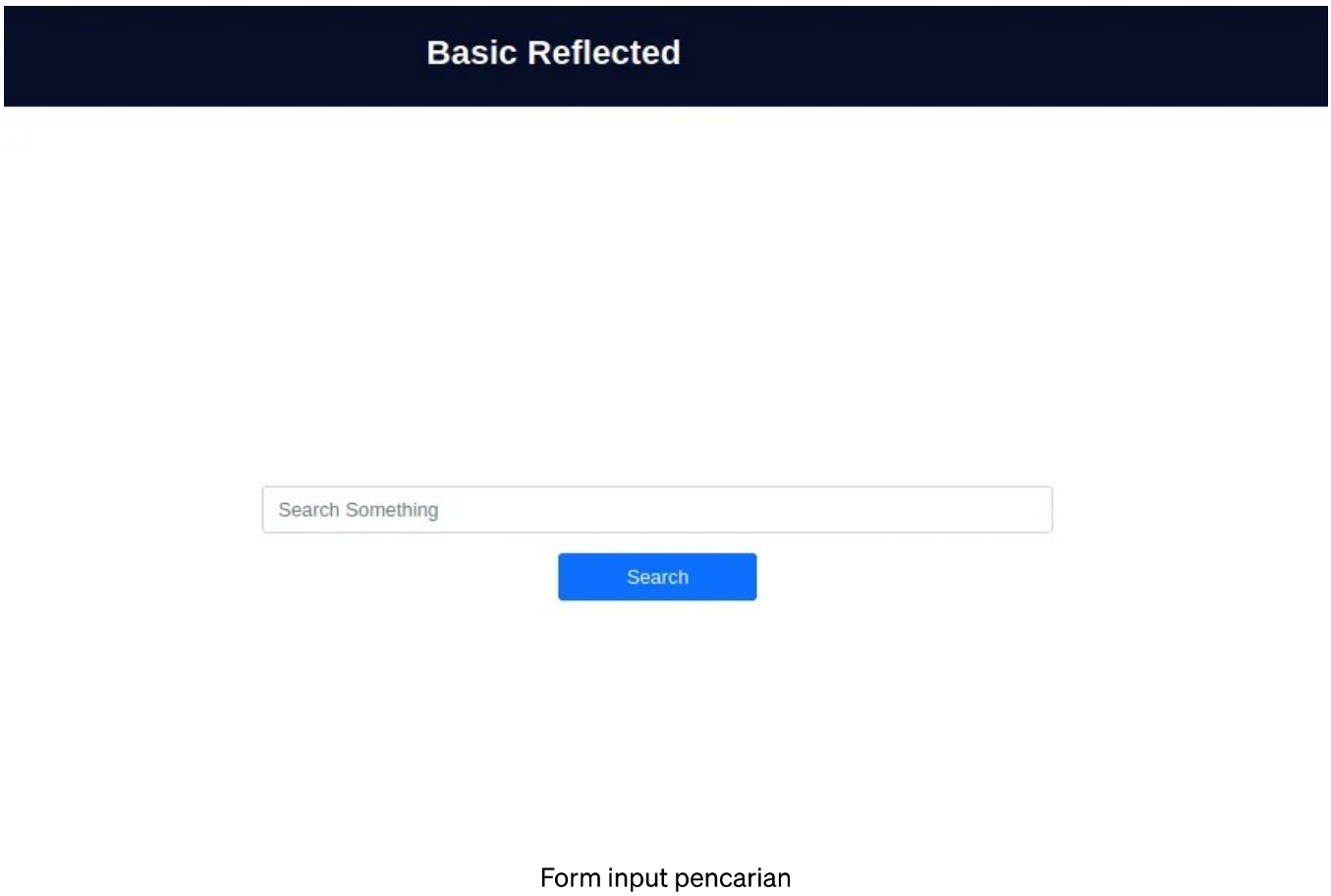
setelahnya, namun hanya diterima mentah-mentah.

Pada line ke 25, inputan tersebut di assign pada variable **$q.**
Pada line ke 27, variable **$q** direfleksikan oleh aplikasi.

Pada line tersebut developer merefleksikan langsung hasil inputan user tanpa disanitasi terlebih dahulu, yang nantinya jika attacker menginput html tag <script> berisi javascript code, maka kode javascript tersebut akan terbaca, karena tag <script> tidak tersanitasi akan dibaca sebagai HTML tag oleh browser, dan <script> tersebut akan menjalankan javascript kode attacker.

Basic Reflected (Search Form)

Pada halaman awal dari lab Basic Reflected terdapat sebuah form input pencarian, disini terdapat sebuah button yang ada tepat dibawah form input pencarian.



Form input pencarian

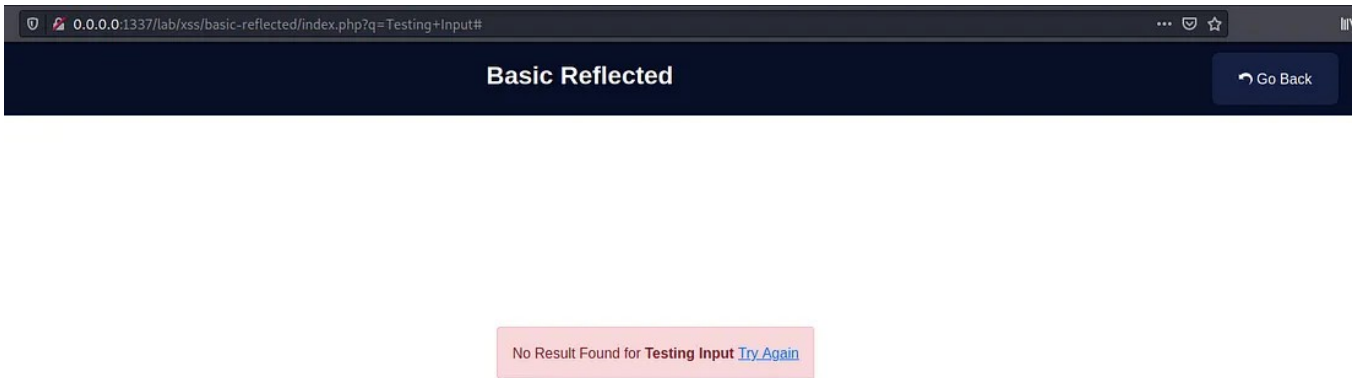
Sebelumnya kita diberikan sebuah objektif pada lab tersebut.

Yaitu “Find a way to trigger XSS via search bar” yang berarti kita diharuskan untuk mencari cara untuk mentrigger XSS dengan form input pencarian.

Disini kita mencoba untuk normal test terlebih dahulu untuk mengetahui flow dari fungsi form input pencarian tersebut.

Setelah kita lakukan input pada kolom pencarian, seperti yang terlihat terdapat sebuah parameter GET pada url halaman Basic Reflected berisi value inputan yang kita input sebelumnya.

Dan terlihat juga pada body content halaman tersebut merefleksikan inputan yang kita input sebelumnya.



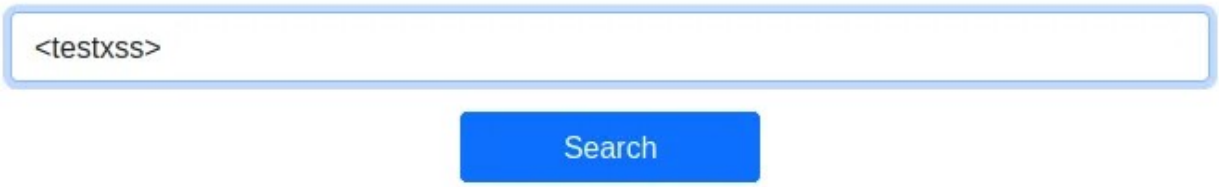
Eksploitasi

Setelah kita melakukan enumarasi, dan observasi kita mendapatkan sebuah flow dari fungsi tersebut, ternyata apapun yang kita input kan pada form input pencarian akan ter-refleksikan ke body konten yang ada pada halaman Basic Reflected.

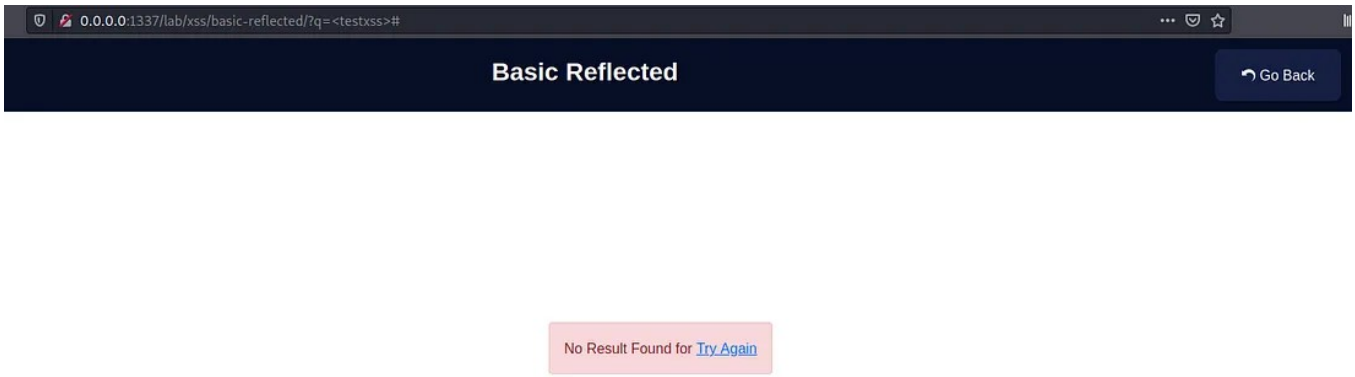
Saya mencoba melihat dengan inspect element, dan terlihat metode request yang dipakai adalah GET, itu penyebab mengapa inputan kita muncul pada parameter GET

?q=.

Disini kita mencoba memasukan sebuah tag html pada input form pencarian.

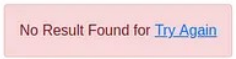


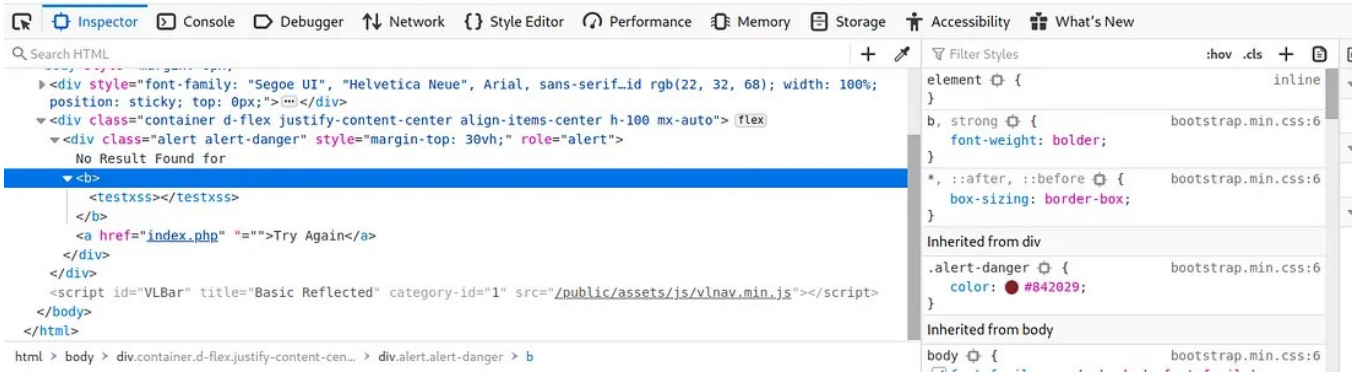
Lalu kita mendapatkan sebuah response seperti ini.



Ada yang aneh pada konten web tersebut, seperti yang terlihat kita telah menginput html tag pada input form pencarian tersebut, namun tidak terefleksi seperti inputan sebelumnya.

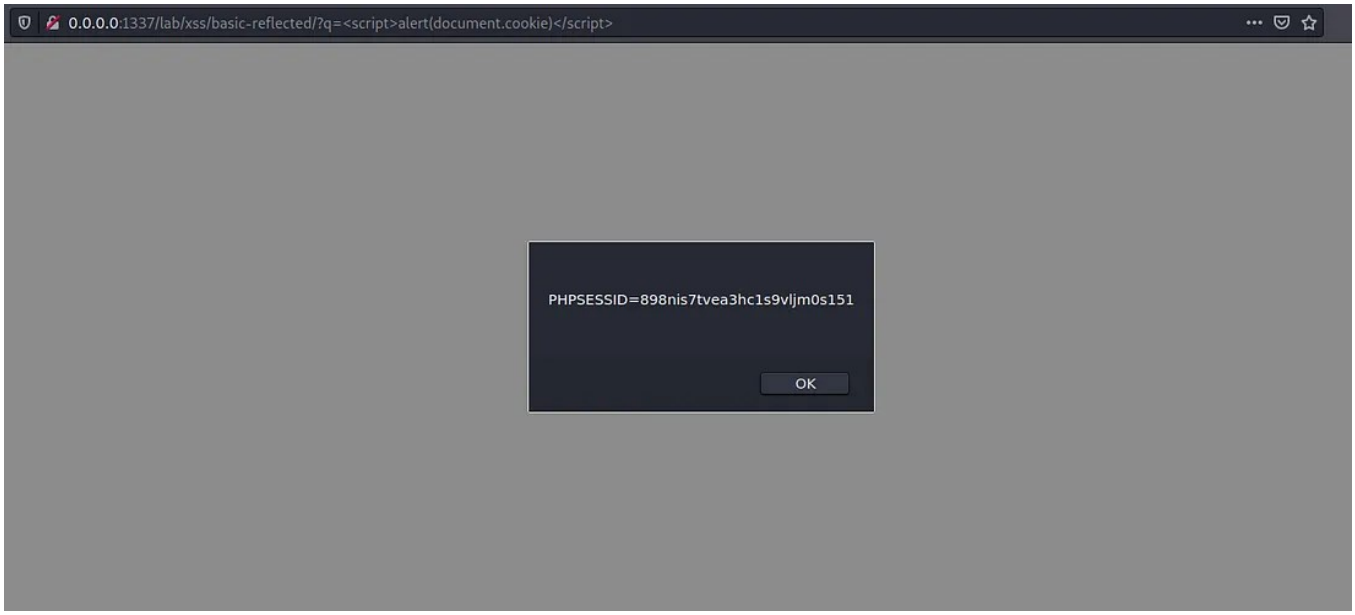
Lalu saya mencoba untuk melihat inputan tersebut dengan inspect element.





Ternyata inputan saya sebelumnya terefleksikan, namun dikarenakan browser & aplikasi membaca sebagai html, maka tag tersebut ter-render sebagai tag html yang sah.

Disini kita bisa langsung menginput function javascript berisi perintah alert popup untuk menampilkan data cookie yang tersimpan pada browser.



Disini kita berhasil melakukan eksploitasi XSS reflected dengan GET parameter.

Reflected XSS tidak hanya menggunakan metode GET, ada beberapa aplikasi yang menggunakan POST metode untuk mengirimkan request dari user.

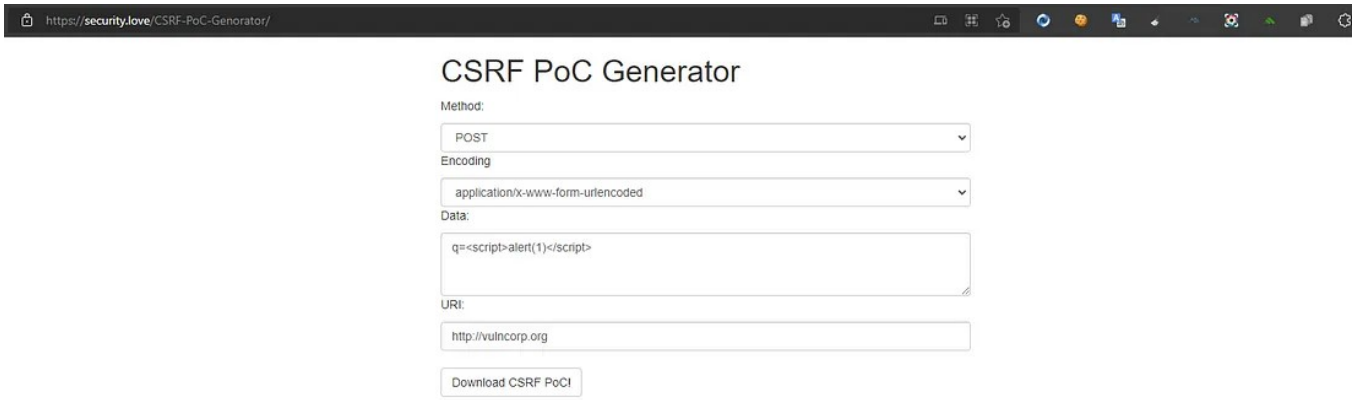
Untuk Reflected XSS POST kita dapat mengkombinasikanya dengan kerentanan Insecure CSRF Token.

Untuk membuat script Insecure CSRF, kita dapat menggunakan sebuah website untuk mempermudah kita.

CSRF PoC Generator

Edit description

security.love



Exploit Reflected XSS POST Chaining dengan Insecure CSRF Token.

```
<html>
<form enctype="application/x-www-form-urlencoded" method="POST"
action="http://vulncorp.org">
<tr>
<td>q</td>
<td>
<input type="text" value="<script>alert(1)</script>" name="q"></td>
</tr>
</table>
<input type="submit" value="http://vulncorp.org"></form>
</html>
```

Owasp Owasp Top 10 Xss Attack Webapplicationpentest Web App Security

About Help Terms Privacy

Get the Medium app

Download on the App Store GET IT ON Google Play

Pengertian Core JavaScript dan DOM (Document Object Model) | Duniailkom

A03 Injection — OWASP Top 10:2021