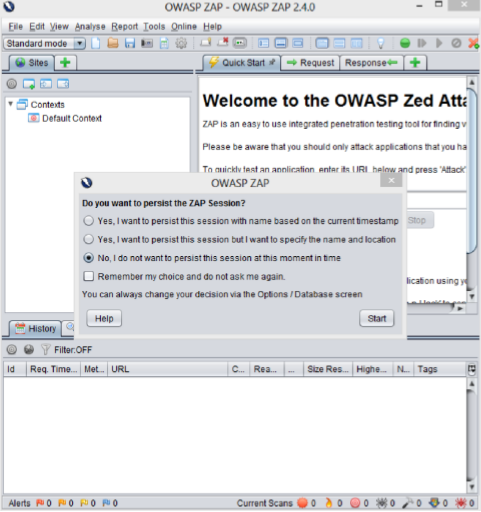
Nama: Limas Baginta

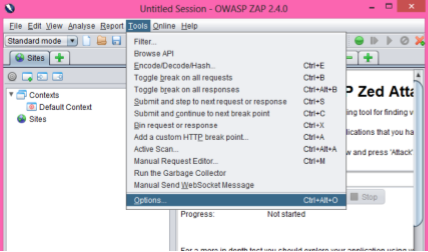
NPM: 1306368690

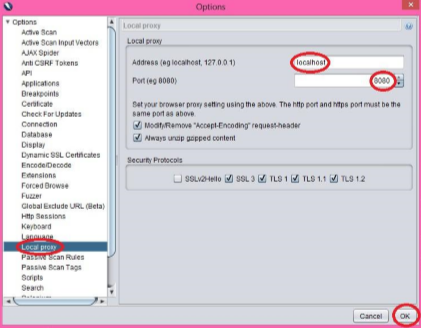
Jurusan: Teknik Komputer

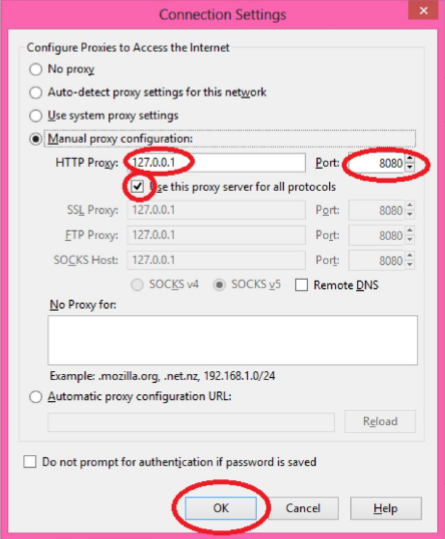
Praktikum: Keamanan Jaringan Komputer

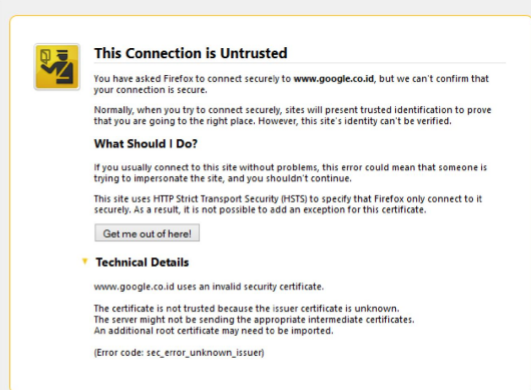
Tugas Tambahan Modul 6  
 **3. Capture web vulnerability assessment menggunakan OWASP ZAP (Boleh menggunakan selain OWASP ZAP , jika anda merasa output yang diberikan sama atau lebih baik) ! tuliskan tahap tahap , seperti yang dicontohkan pada modul, atau lebih lengkap ! (40)**  
  
Jawab:  
  
Tahap-tahap penetration test:  
  
1.Install OWASP ZAP.  
  
2. Buka OWASP ZAP lalu klik radio button pada ‘No, I do not want to persist this session at this moment in time’ lalu klik tombol Start.

  
3. Setting Proxy OWASP ZAP o Setting proxy OWASP ZAP melalui menu Tools  Option atau dengan menekan tombol Ctrl+Alt+O pada keyboard.

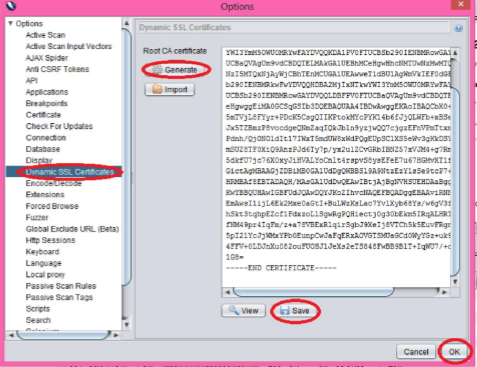
  
  
o Pilih menu Local Proxy, pastikan kolom Address berisi ‘localhost’ dan kolom Port berisi ‘8080’. Lalu klik OK

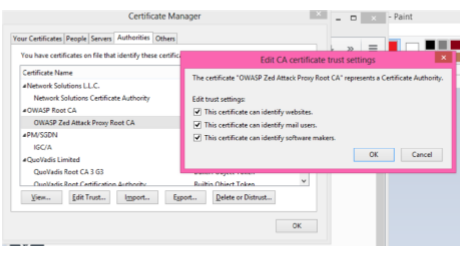


4. Setting Proxy pada Browser o Setting proxy pada browser agar terintegrasi dengan OWASP ZAP.  
  
  
  
5. Kunjungi halaman website yang diinginkan. OWASP ZAP akan memberikan notifikasi bahwa sertifikat dari halaman https yang diakses tidak terpercaya.

  
  
  
o Untuk membuat protocol https dapat diakses dengan proxy OWASP ZAP, adalah dengan cara sebagai berikut:

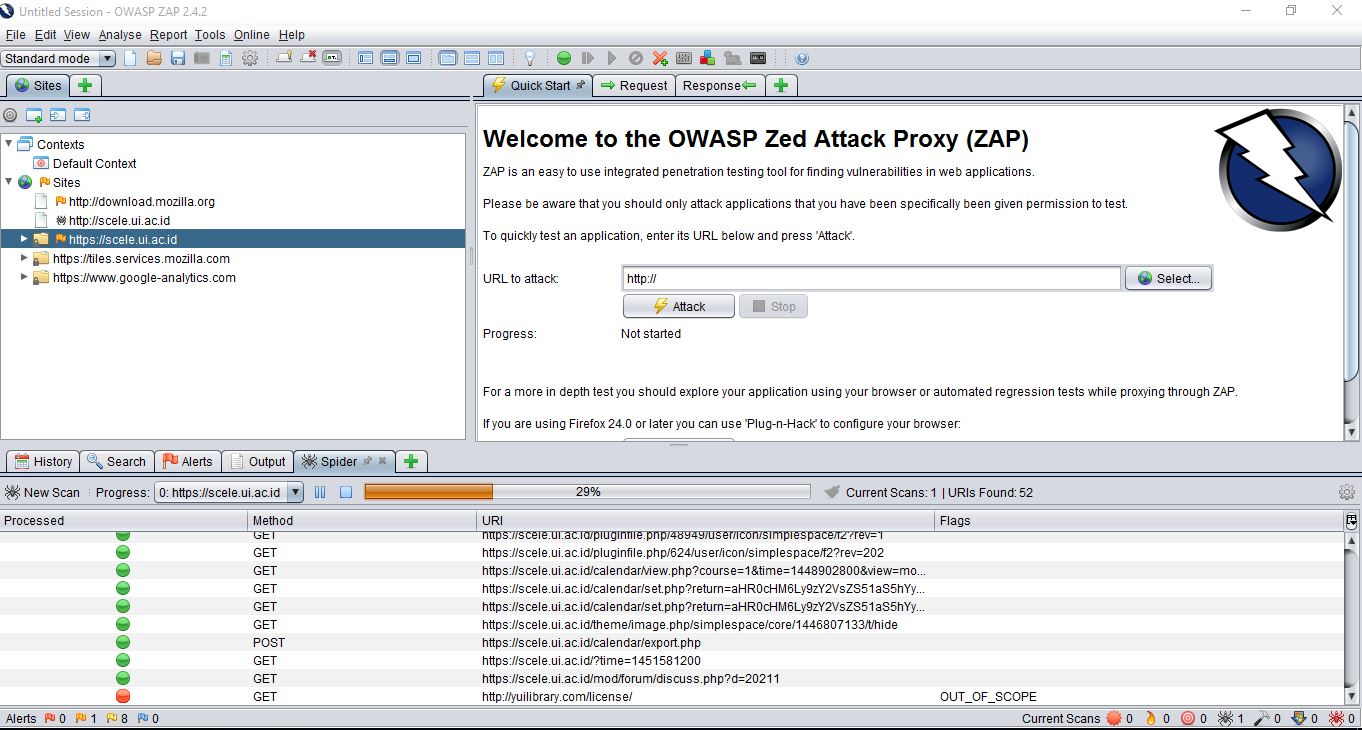
• Buka OWASP ZAP, dari menu Tools > Option > Dynamic SSL Certificate > Generate > file akan bernama owasp\_zap\_root\_ca.cer > Save pada direktori yang diinginkan > klik OK.

  
  
• Buka Web Browser, pilih Advance > View Certificate > pilih tab Authorities > lalu import file owasp\_zap\_root\_ca.cer > Edit Trusted > checklist pada box yang diinginkan > Klik OK.



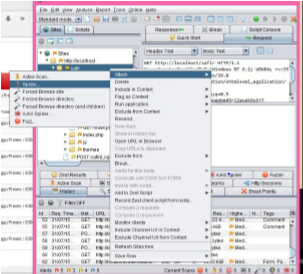
• Jika tahapan ini telah dilakukan maka situs dengan protocol https dapat diakses tanpa adanya notifikasi untrusted certification. Selain itu, semua situs yang dikunjungi pada browser yang telah terintegrasi dengan proxy OWASP ZAP dapat dipantau melalui OWASP ZAP

7. Buka halaman website yang diinginkan. Pada bagian kiri OWASP ZAP terdapat daftar halaman website yang sedang dikunjungi. (NOTE: Web browser hanya dapat membuka halaman website apabila OWASP ZAP telah diaktifkan).

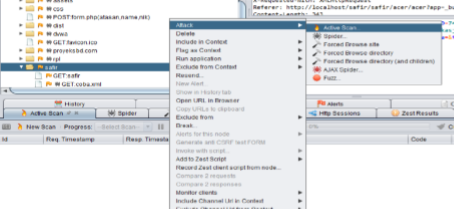


8. Lakukan Vulnerability Assessment

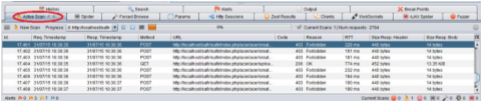
o Vulnerability assessment pada web dapat dilakukan dengan cara klik kanan pada web yang ingin dilakukan vulnerability assessment yang terdapat pada sebelah kiri OWASP ZAP > klik Attack > pilih Spider > klik Start Scan.



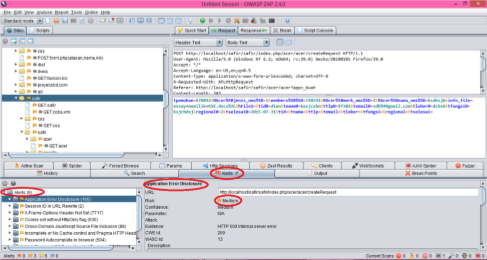
9. Melakukan Penetration Testing o Lakukan Active Scan dengan cara klik kanan pada web yang sama > klik Attack >Active Scan.



o Proses Active Scan dapat dilihat pada tab Active Scan

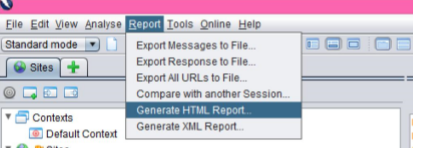


o Active Scan akan menemukan potential vulnerability pada target dengan melakukan serangan terhadap target yang dipilih dan eksploitasi terhadap kelemahan pada web lalu dilakukan pengamatan terhadap hasilnya. Hal ini termasuk dalam langkah penetration testing. Daftar dari potential vulnerability dapat terlihat pada tab Alerts. Setiap potential vulnerability dapat dilihat deskripsi secara details serta tingkat resikonya apakah High, Medium, Low, atau Informational.

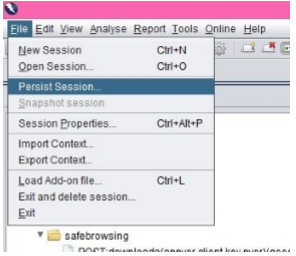


10. Membuat Laporan

o Laporan hasil penetration testing sangat penting untuk dilakukan agar setiap vulnerability yang ditemukan dalam suatu web application dapat dikaji kembali oleh web developer dan diperbaiki. Pada OWASP ZAP, laporan dari semua vulnerability yang telah ditemukan dapat langsung di-generate menjadi file HTML dengan cara melalui tab Report  Generate HTML report  tentukan directory yang diinginkan.



11. Simpan Session Session yang telah dilakukan dapat disimpan dengan cara melalui Menu File > Persist Session > Pilih direktori yang diinginkan > Klik Save.



12. Analisis Terhadap Hasil dari Penetration Testing Setelah mendapatkan laporan dari penetration testing, hal selanjutnya yang harus dilakukan adalah menganalisa terhadap vulnerability yang ditemukan. Setelah dianalisa maka akan didapatkan solusi dari vulnerability yang telah ditemukan sehingga web akan memiliki kemanan yang lebih baik.

**4. Analisa dan berikan solusi seperti yang anda kerjakan pada no.1 dan 2 ! (min 8) (60)**  
  
Jawab:

a. 

Analisa:

Header dari X-Frame-Options tidak tercakup dalam HTTP response untuk melindungi dari serangan 'ClickJacking' yang mana 'ClickJacking' adalah suatu metode untuk menipu suatu Web user untuk melakukan click sesuatu yang berbeda dari yang pernah mereka click pada halaman Web user sehingga Web user tertipu dan confidential dari halaman Web user tersebut terbaca dan bahkan dapat dikontrol oleh penyerang halaman Web user tersebut.

Solusi:

Beberapa Web browser yang modern men-*support* X-Frame-Options HTTP header. Yakinkan bahwa telah dilakukan set pada semua halaman web (jika ingin diakukan frame dengan server sendiri (contoh FRAMESET) gunakan SAMEORIGIN, sebaliknya jika tidak ingin dilakukan frame gunakan DENY. ALLOW-FROM memperbolehkan spesifik web melakukan frame pada halaman web dengan menggunakan web browsers yang dapat men-support X-Frame-Options HTTP header).

b. 

Analisa:

Halaman ini mengandung sebuah pesan error/warning yang mungkin tidak menampilkan informasi penting seperti lokasi file yang memproduksi suatu pengecualian yang tidak biasanya ditangani (unhanded exception). Informasiini dapat digunakan untuk melakukan serangan terhadap aplikasi web. Alert dapat menjadi false positive jika pesan eror ditemukan dalam halaman dokumentasi.

Solusi:

Melakukan review source code pada halaman yang terdeteksi. Implementasi halaman custom error. Implementasi juga sebuah mekanisme untuk menyediakan sebuah referensi eror yang bersifat unik/identifier menuju client (browser) saat logging pada server dan tidak menampilkannya pada user.

c. 

Analisa:

Halaman pada URL mencakup satu atau lebih file script dari sebuah third-party domain yang mana third-party domain adalah domain yang disediakan penyedia provider yang diperuntukkan bagi penjelajah akses di internet melalui browser.

Solusi:

Memastikan file JavaScript dibuka hanya dari trusted sources, dan source tersebut tidak dapat dikendalikan oleh end users dari aplikasi.

Melakukan **CSP** atau Content Security Policy juga dapat menjadi solusi yang mana dapat membuat suatu inline JavaScript (JavaScript yang tidak seharusnya berada dalam script) menjadi disabled secara default dan harus secara eksplisit diperbolehkan dalam kebijakan (policy).

Content Security Policy adalah suatu header standard yang digunakan pada Chrome versi 25 dan berikutnya, Firefox versi 23 dan berikutnya, serta Oper versi 19 dan berikutnya.



(<https://www.owasp.org/index.php/List_of_useful_HTTP_headers> ) (<https://www.owasp.org/index.php/Content_Security_Policy> )

d. 

Analisa:

Web Browser XSS belum di-enabled atau telah disabled pada konfigurasi header X-XSSProtection' HTTP response pada suatu Web server.

Solusi:

Memastikan bahwa filter XSS web browser telah dilakukan enabled dengan cara melakukan setting X-XSS-Protection HTTP response header menjadi '1'

e. 

Analisa:

Header Anti-MIME-Sniffing pada X-Content-Type-Options tidak di lakukan set menuju 'nosniff'. Ini memperbolehkan versi lama Internet Explorer dan Chrome untuk melakukan MIMEsniffing pada response body yang mana MIME-Sniffing adalah suatu cara untuk melakukan inspect terhadap konten berupa byte stream untuk mengetahui file format dari suatu data. Response body akan terintepretasi dan ditampilkan sebagai suatu tipe konten.

Solusi:

Melakukan set pada X-Content-Type-Options menuju 'nosniff' untuk mencegah versi lama Internet Explorer dan Chrome melakukan MIME-sniffing suatu response dari sebuah tipe konten (content-type).



(<https://www.owasp.org/index.php/List_of_useful_HTTP_headers> )

f. 

Analisa:

Web Browser XSS belum di-enabled atau telah disabled pada konfigurasi header X-XSSProtection' HTTP response pada suatu Web server.

Solusi:

Memastikan cache-control HTTP header telah dilakukan set menjadi no-cache, no-store, must-revalidate, private; dan juga pragma HTTP header dilakukan set dengan no-cache.

g. 

Analisa:

Halaman mencakup konten mixed, contohnya konten diakses melalui http bukan dari https.  
Ini membuat konten menjadi tidak aman (not secure) apabila diakses melalui http.

Solusi:

Suatu halamanyang memiliki TLS harus mencakup secara komplit konten yang ditransmisi melalui TLS.

Halaman tidak harus mengandung konten melalui HTTP.

Ini mencakup konten dari situs third party yang tidak aman.

h. 

Analisa:

Atribut AUTOCOMPLETE tidak diakukan disabled dalam elemen HTML FORM/INPUT yang mengandung tipe masukan password. Password mungkin tersimpan dalam browser and diterima.

Solusi:

Melakukan turn off pada atribut AUTOCOMPLETE dalam bentuk element tipe masukan yang mengandung password menggunakan AUTOCOMPLETE='OFF'