KEAMANAN JARINGAN
(Laporan Instalasi OWASP)

Fifin Nur Rahmawati

Dosen : Dr. Ferry Astika Saputra ST, M.Sc

3122640040

D4 LJ-Teknik Informatika

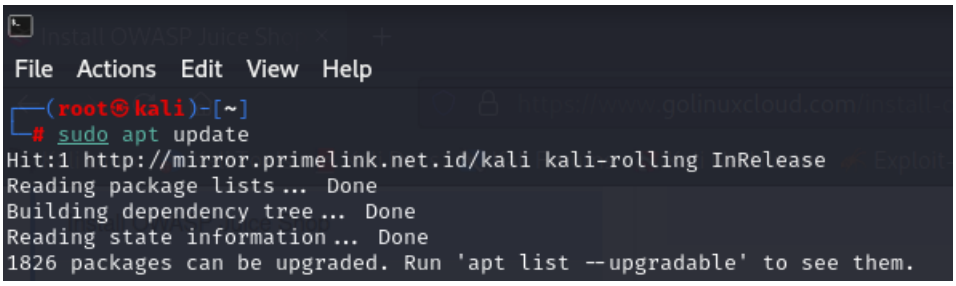
PROGRAM STUDI TEKNIK INFORMATIKA

DEEPARTEMEN TEKNIK INFORMATIKA DAN KOMPUTER

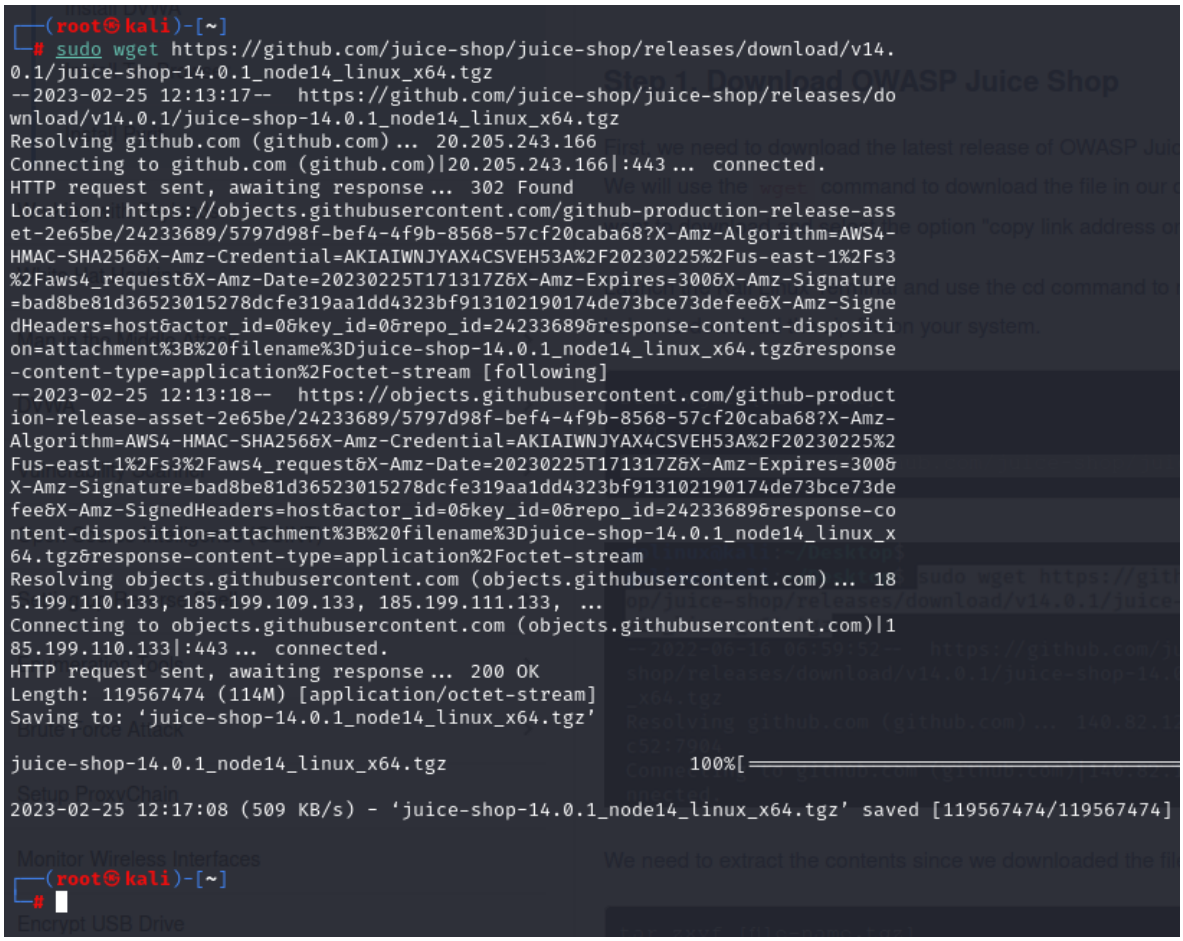
POLITEKNIK ELEKTRONIKA NEGERI SURABAYA

2023/2024

1. Perintah :
   Sudo apt update → **Memperbarui basis data paket**. Memperbarui daftar paket untuk peningkatan versi yang diperlukan.

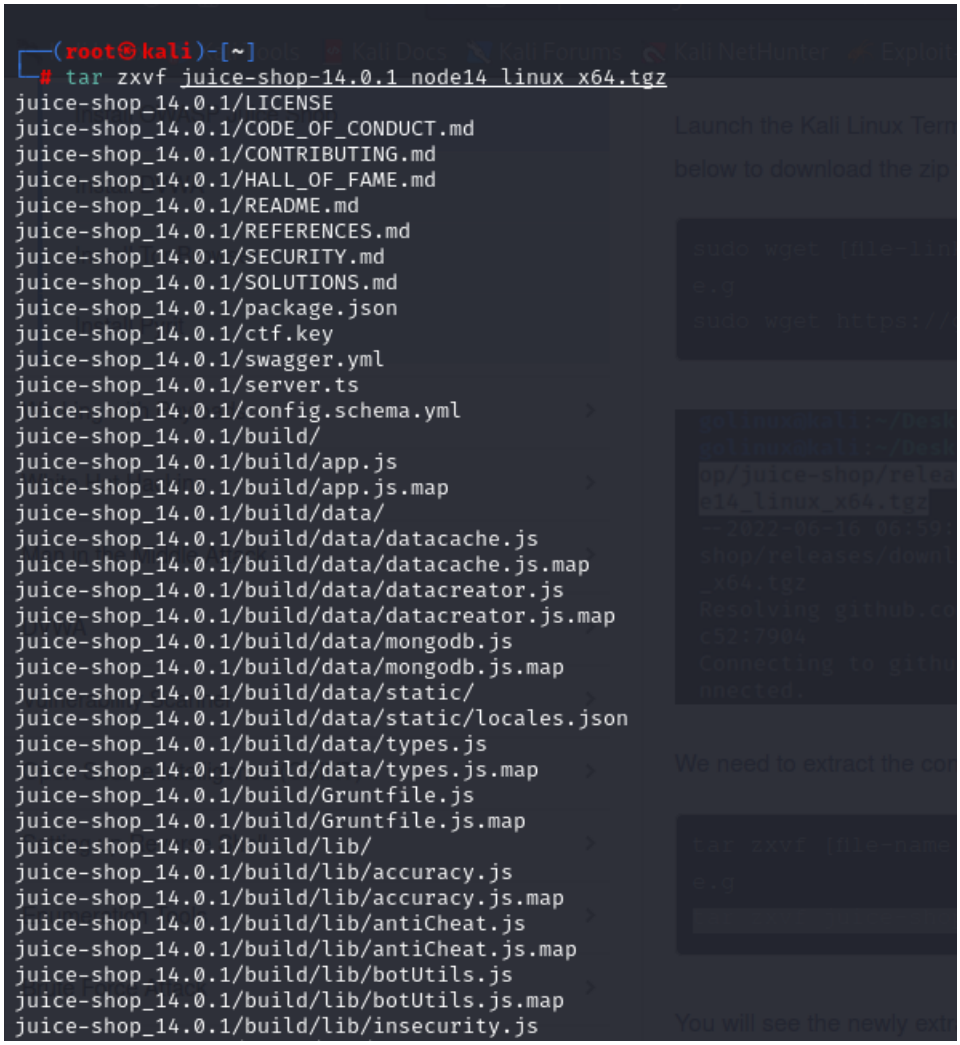


2. Perintah :
   sudo wget https://github.com/juice-shop/juice-shop/releases/download/v14.0.1/juice-shop-14.0.1_node14_linux_x64.tgz digunakan untuk pengunduhan OWASP versi terbaru 14.0.1 dan **wget** untuk pengunduhan file dilokasi yang diinginkan. **Wget** dapat digunakan untuk mengunduh banyak file sekaligus. Untuk dapat melakukannya, kita harus membuat dokumen teks dan menempatkan URL unduhan di di dalamnya. Maka hasil download akan ditampilkan seperti gambar dibawah ini :



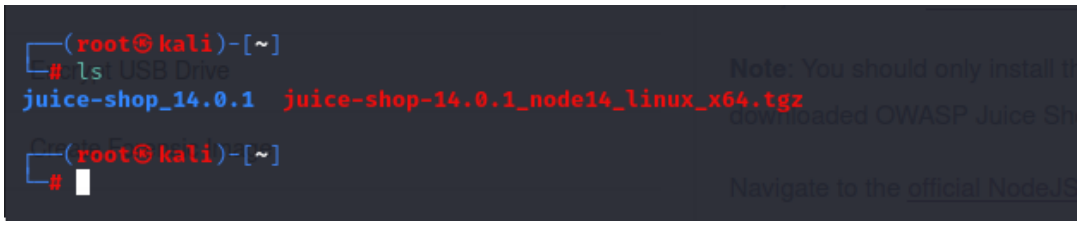
3. Perintah :

   tar zxvf juice-shop-14.0.1_node14_linux_x64.tgz → **Tape archive** digunakan untuk mengompres serangkaian file dan folder. Umumnya, file yang telah di-compress dengan menggunakan command tar akan tersimpan dalam bentuk file .tar.



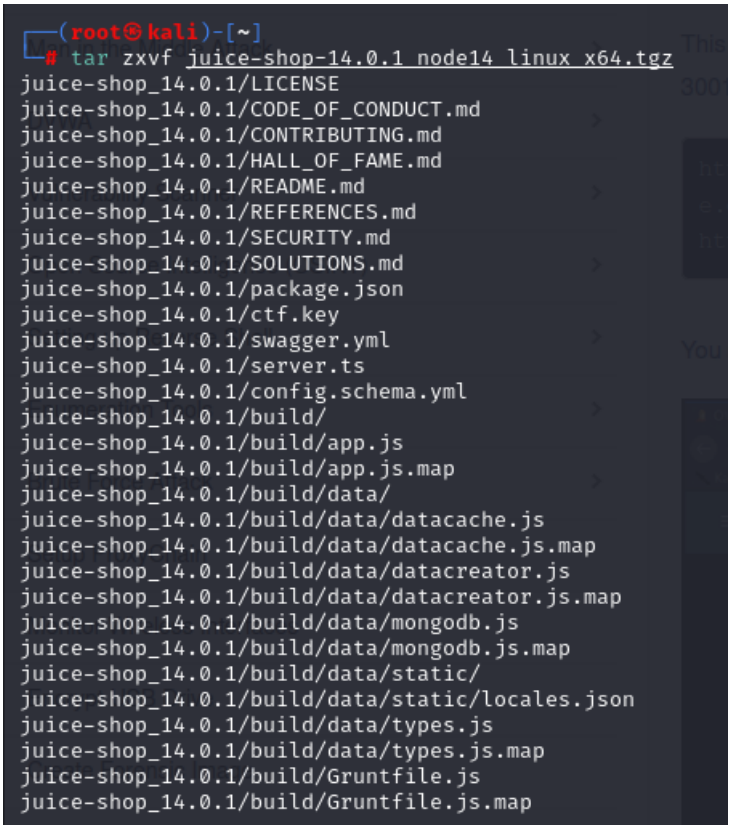
4. Perintah

   ls → perintah yang digunakan untuk melihat direktori pada linux. Perintah ini digunakan untuk melihat atau menampilkan/list isi dari folder/direktori di linux. Jika diketikan langsung maka akan menampilkan isi dari direktori Anda berada saat ini. Perintah ini digunakan untuk membuat folder kosong.



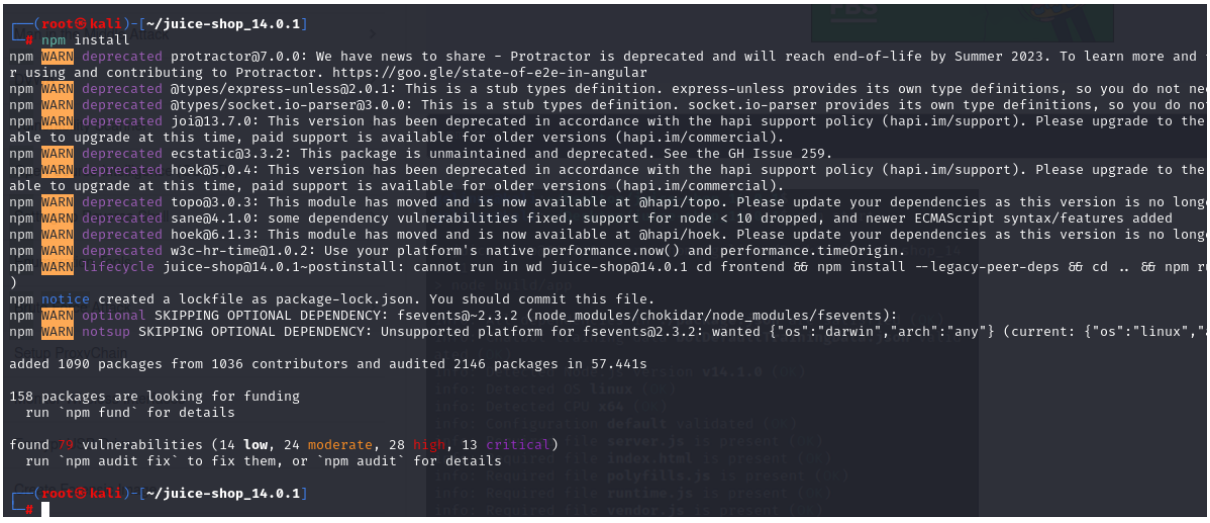
5. Perintah :
   sudo wget https://nodejs.org/download/release/v14.1.0/node-v14.1.0-linux-x64.tar.xz
   Penginstalan Nodejs.org disarankan menginstal versi NodeJS yang mirip dengan versi file setup OWASP Juice Shop yang diunduh. Misalnya, mengunduh OWASP Juice Shop versi 14.0.1. Oleh karena itu, kita perlu mengunduh NodeJS versi 14.

6. Perintah :
   sudo tar -xvf node-v14.1.0-linux-x64.tar.xz → mengompres serangkaian file dan folder.

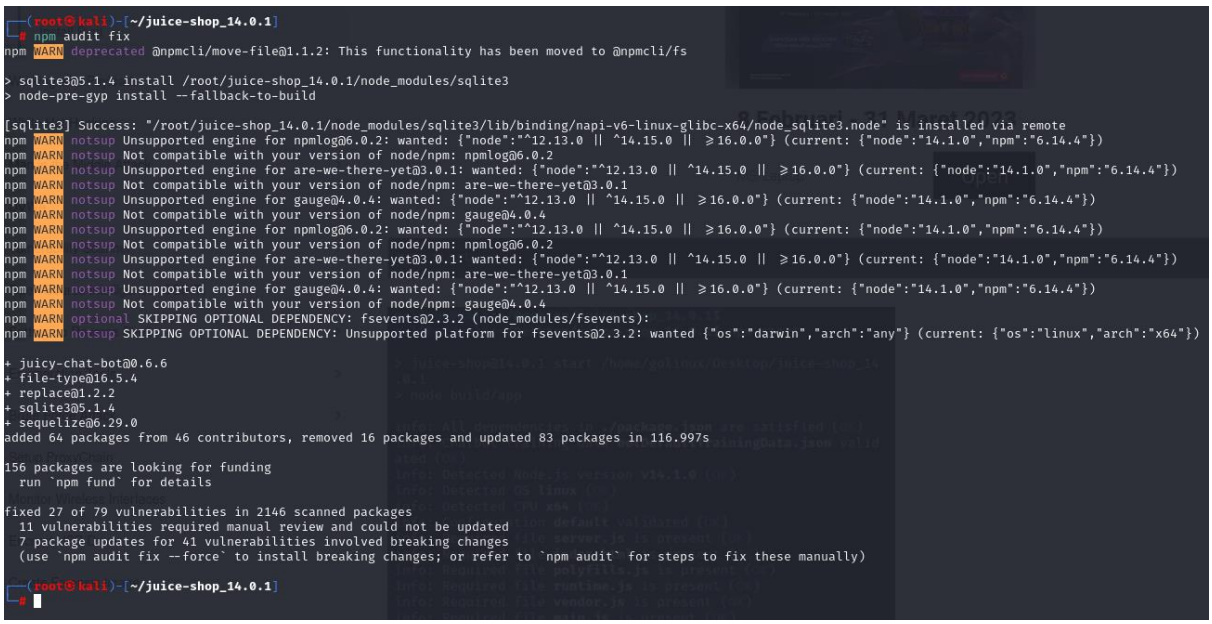


7. Perintah :
   sudo cp -r node-v14.1.0-linux-x64/{bin,include,lib,share} /usr/ akan melihat folder "Node" baru yang dibuat di sistem. Ada beberapa file yang perlu disalin dari folder yang baru diekstrak ini ke direktori /usr untuk menginstal NodeJS dan NPM

8. Perintah :
   Npm install → digunakan untuk menginstal

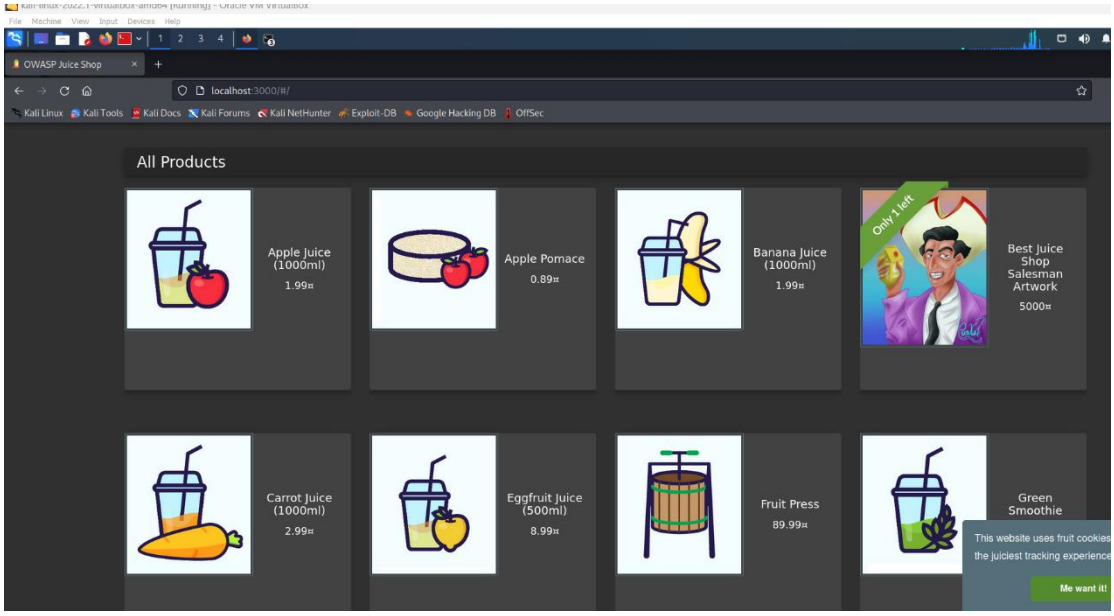


9. Perintah :
   Npm audit fix → Memperbaiki kerentanan ini dapat mencegah hal-hal seperti kehilangan data, pemadaman layanan, dan akses tidak sah ke informasi sensitif.



10. Perintah :
    Npm start → Perintah ini akan memulai aplikasi web di port 3000.

11. Perintah :
    http://localhost:3000/ OWASP Juice Shop telah berhasil diinstal pada Kali Linux.



Hubungan antara OWASP 10 Dengan Juice Shop

OWASP Top 10 adalah daftar yang memuat sepuluh celah keamanan aplikasi web yang paling umum terjadi. Daftar ini disusun oleh Open Web Application Security Project (OWASP), sebuah organisasi nirlaba yang fokus pada meningkatkan keamanan aplikasi web. Daftar OWASP Top 10 menjadi acuan bagi para pengembang aplikasi web dalam mengidentifikasi celah keamanan yang umum terjadi dan memperbaikinya.

Sementara itu, Juice Shop adalah sebuah aplikasi web yang dirancang khusus untuk tujuan pelatihan dan demonstrasi celah-celah keamanan aplikasi web. Aplikasi ini dikembangkan oleh OWASP dan menyediakan berbagai celah keamanan yang sering ditemukan pada aplikasi web, seperti injection, broken authentication, cross-site scripting (XSS), dan lain-lain. Juice Shop juga menyediakan beberapa fitur tambahan seperti leaderboard, badge, dan challenge yang membuat proses belajar lebih menarik dan interaktif.

Koneksi antara OWASP Top 10 dan Juice Shop adalah bahwa Juice Shop didesain untuk mencakup semua celah keamanan yang terdapat pada daftar OWASP Top 10, sehingga pengguna dapat menggunakan Juice Shop sebagai platform untuk mempelajari dan melatih keahlian dalam mengatasi celah-celah keamanan tersebut. Dalam hal ini, Juice Shop dapat digunakan sebagai alat praktis untuk memahami celah keamanan yang terdapat pada daftar OWASP Top 10, dan memperlihatkan secara langsung bagaimana celah tersebut dapat dimanfaatkan oleh para penyerang.

Melalui pelatihan menggunakan Juice Shop, pengguna dapat memperoleh pengalaman praktis dalam menangani celah keamanan aplikasi web dan meningkatkan kemampuan mereka dalam mengamankan aplikasi web. Dengan demikian, OWASP Top 10 dan Juice Shop saling mendukung dalam upaya meningkatkan keamanan aplikasi web.

Dalam hal ini, Juice Shop dapat digunakan sebagai alat praktis untuk memahami celah keamanan yang terdapat pada daftar OWASP Top 10, dan memperlihatkan secara langsung bagaimana celah tersebut dapat dimanfaatkan oleh para penyerang. Melalui pelatihan menggunakan Juice Shop, pengguna dapat memperoleh pengalaman praktis dalam menangani celah keamanan aplikasi web dan meningkatkan kemampuan mereka dalam mengamankan aplikasi web.

10 Kerentanan OWASP yang paling popular

1. **INJEKSI**
   Serangan injeksi dapat terjadi pada hal-hal seperti database (SQL, noSQL), sistem operasi, atau server (melalui protokol seperti LDAP). Mereka terjadi ketika data yang tidak bersahabat dikirim ke *interpreter* sebagai bagian dari kueri atau perintah. Data ini kemudian mengelabui *interpreter* agar menjalankan perintah yang seharusnya terlarang bagi orang luar. Itu juga dapat digunakan untuk mengakses data pribadi tanpa otentikasi yang tepat.
   Misalkan, menjalankan sebuah *e-commerc*, dan cara mengakses item tertentu adalah dengan memasukkan yang berikut ini ke bilah alamat browser :

- **http://www.yourstore.com/catalog/item.asp?itemid=999**
  Di mana "999" menghasilkan kueri SQL untuk menampilkan item apa pun yang sesuai dengan "999". Seorang penyerang dapat memanipulasi ini dengan memasukkan sesuatu seperti ini:

- **http://www.yourstore.com/items/item.asp?itemid=999 atau 1 = 1**
  Kueri SQL yang dihasilkan adalah:

```
SELECT ItemName, ItemDescription
FROM Items
WHERE ItemNumber = 999 OR 1=1
```

Tetapi karena 1 selalu sama dengan 1, setiap nama dan deskripsi produk akan dikembalikan (bahkan yang mungkin tidak ingin Anda lihat oleh pemiliknya).
Anda dapat melangkah lebih jauh dengan memasukkan:

- http://www.estore.com/items/iteam.asp?itemid=999; **DROP TABLE**
  Kueri SQL sekarang menjadi:

```
SELECT ItemName, ItemDescription
FROM Items
WHERE ItemNumber = 999; DROP TABLE USERS
```

Hasil akhirnya? Penghapusan table Anda.
Bagaimana cara untuk melindunginya?

- Validasi dan / atau bersihkan data yang dikirimkan pengguna (validasi menolak entri. yang mencurigakan, dan sanitasi membersihkan bagian yang mencurigakan).
- Tetapkan kontrol untuk meminimalkan jumlah informasi yang terungkap.
- Gunakan API yang aman yang menghindari penggunaan penerjemah.
- Gunakan kontrol SQL seperti LIMIT dan lainnya untuk mencegah pengungkapan data secara massal

2. **Broken Authentication and Session Management (KERUSAKAN AUTENTIKASI)**
   Autentikasi yang rusak mengacu pada contoh ketika fungsi autentikasi dan sesi manajemen diterapkan secara tidak benar. Kredensial seperti kata sandi, kunci, atau token sesi dapat dicegat dan mengasumsikan identitas pengguna lain. Serangan bahkan dapat memperoleh akses ke akun admin yang dapat membahayakan seluruh sistem.
   Credential Stuffing adalah contoh serangan autentikasi yang rusak. Ini terjadi ketika penyerang menggunakan daftar kata sandi yang diketahui (memperolehnya dari pelanggaran data) untuk mencoba dan mendapatkan akses dengan aplikasi bertindak sebagai mekanisme validasi untuk setiap upaya kata sandi.
   Bagaimana melindungi situs ataupun web aplikasi dari ancamana ini?
   - Gunakan autentikasi multi-faktor (2F Authentication)
   - Menerapkan pembatasan tingkat untuk membatasi jumlah upaya login yang gagal.
   - Jangan gunakan kredensial default sistem.
   - Pilih kata sandi berdasarkan standar NIST 800-63Bbagian 5.1.1.
   - Gunakan manajer sesi sisi server *built-in*.

3. **CROSS-SITE SCRIPTING (XXS)**
   Jenis kerentanan ini merupakan hasil dari sesi manajemen yang lemah dan terjadi saat aplikasi web memungkinkan pengguna menambahkan kode khusus ke URL atau situs yang akan ditampilkan kepada orang lain. Kode JavaScript berbahaya kemudian dapat dijalankan di browser mereka. Seorang peretas yang berpura-pura dari bank tepercaya dapat mengirim email kepada seseorang, termasuk tautan ke situs web bank di dalam email. Tautan tersebut, bagaimanapun, mungkin memiliki kode berbahaya yang ditambahkan ke akhir URL. Jika situs bank tidak diamankan dengan baik, maka kode berbahaya akan dijalankan di browser korban setelah mereka mengkliknya.
   Bagaimana untuk mencegahnya?
   - Gunakan sebuah Web Application Web (WAF), yang akan menggunakan pemfilteran berbasis tanda tangan untuk mengidentifikasi dan memblokir permintaan dari penyerang.
   - Gunakan kerangka kerja yang lolos dari XSS dengan desain dan pelajari batasan perlindungan XSS mereka sehingga Anda dapat menangani kasus yang tidak tercakup.
   - Terapkan pengkodean peka konteks saat melakukan modifikasi dokumen browser sisi klien.

4. **Broken Access Control**

   kerentanan ini terjadi ketika aplikasi web tidak menerapkan kontrol akses dengan benar, sehingga pengguna yang tidak berhak dapat mengakses sumber daya yang seharusnya tidak dapat diakses.

   Ini terjadi ketika pembatasan tentang apa yang diizinkan/tidak diizinkan oleh pengguna yang diberlakukan autentikasi secara tidak benar. Serangan kemudian dapat memanfaatkan untuk mendapatkan fungsionalitas yang tidak sah, termasuk mengakses dan mengubah akun pengguna, file sensitif, data pengguna, hak akses, dan banyak lagi.

   Apa yang harus dilakukan untuk mencegahnya?

   - Tolak akses secara default untuk semua hal kecuali sumber daya publik
   - Buat mekanisme kontrol akses yang kuat dan gunakan di mana saja
   - Jangan izinkan pengguna membuat, membaca, atau menghapus rekaman apa pun
   - Nonaktifkan daftar direktori server, dan jangan simpan metadata di root folder
   - Catat upaya akses yang gagal dan buat peringatan
   - Nilai batas akses API

5. **Security Misconfiguration**

   Kerentanan ini terjadi ketika aplikasi web dikonfigurasi dengan tidak benar, seperti pengaturan sandi yang lemah, izin file yang tidak tepat, atau tidak memperbarui versi aplikasi dengan patch keamanan terbaru.

   Ini adalah kerentanan paling umum pada Daftar OWASP Top Ten dan biasanya disebabkan oleh penggunaan konfigurasi/kredensial default atau menampilkan pesan kesalahan yang panjang dan tidak perlu. Pesan-pesan ini berpotensi mengungkapkan kerentanan dalam aplikasi.

   Contohnya akan menampilkan pesan atau notifikasi error seperti di bawah ini:

   Seperti yang Anda lihat, detail kode aplikasi terungkap, yang dapat dimanfaatkan oleh pihak ketiga yang tidak bertanggung jawab.

   Perbaiki kesalahan tersebut dengan:

   - Hapus semua fitur yang tidak digunakan dalam kode.
   - Hanya tampilkan pesan kesalahan umum yang tidak mengungkapkan terlalu banyak informasi.
   - Gunakan program Static Application Security Testing (SAST) untuk mengidentifikasi risiko paparan informasi dari pesan atau notifikais error.

6. **Insecure Cryptographic Storage**

kerentanan ini terjadi ketika aplikasi web menyimpan informasi sensitif dalam bentuk yang tidak aman, seperti menyimpan kata sandi dalam format teks yang tidak terenkripsi.

Eksposur data sensitif terjadi ketika aplikasi web dan API gagal melindungi data sensitif seperti informasi keuangan atau perawatan kesehatan. Data yang terlindungi secara lemah ini dapat dengan mudah dicuri oleh penyerang untuk melakukan penipuan, pencurian identitas, dan kejahatan lainnya.
Jika situs web tidak menggunakan sertifikat SSL/TLS berkualitas untuk semua halaman, maka penyerang dapat memantau lalu lintas, mengubah koneksi dari HTTPS ke HTTP, dan kemudian mencuri sesi *cookie* untuk mendapatkan akses. Contoh lainnya adalah hash yang kurang. Jika hash sederhana digunakan untuk menyimpan kata sandi dan penyerang memperoleh akses ke database, hash dapat dengan mudah dirusak.
Bagaimana untuk mengamankan data dari aksi eskposur?

- Identifikasi data sensitif dan terapkan kontrol yang sesuai.
- Enkripsi semua data sensitif, baik saat transit maupun saat istirahat dengan sertifikat SSL/TLS.
- Nonaktifkan *cache* informasi sensitif apa pun dan jangan simpan semua itu secara tidak perlu.
- Simpan kata sandi menggunakan fungsi hashing yang kuat dan akurat seperti scrypt, bcrypt, dan Argon2.

7. **Insufficient Logging and Monitoring**

kerentanan ini terjadi ketika aplikasi web tidak mencatat aktivitas yang terjadi dengan benar atau tidak memonitor aktivitas aplikasi web secara teratur, sehingga sulit untuk mendeteksi serangan atau celah keamanan.

Pencatatan (*logging*) dan pemantauan harus dilakukan secara rutin untuk membantu memastikan keamanan web bekerja maksimal. Kegagalan mampu meningkatkan risiko serangan yang dapat terjadi dan menghambat waktu respons situs Anda. Hal ini pada gilirannya memberi penyerang cukup banyak waktu untuk merusak, mengekstrak, atau menghancurkan data, beralih ke sistem lain, dan mengacak-acak semua yang ada di dalamnya.
Seperti aksi *credential stuffing*, di mana penyerang berulang kali mencoba menggunakan nama pengguna dan pasangan kata sandi yang bocor. Katakanlah setelah 100 percobaan, mereka akhirnya mencapai kombinasi yang tepat untuk akun tertentu. Karena tidak ada *logging* atau pemantauan di tempat, tidak ada yang pernah diberitahu tentang tingginya jumlah upaya login pada akun tersebut. Jika tidak, aktivitas tersebut akan dianggap mencurigakan dan pelanggaran dapat dengan mudah dicegah.
Apa tindakan terbaik untuk masalah ini?

- Menerapkan logging dan pemantauan untuk semua aspek aplikasi web Anda
- Buat rencana respons insiden yang menyertakan peringatan sehingga Anda segera mengetahui adanya serangan

- Pastikan log Anda dalam format yang dapat dengan mudah digunakan oleh solusi manajemen log terpusat
- Siapkan log Anda agar berisi konteks yang memadai untuk mengidentifikasi akun yang mencurigakan

8. Insecure Communication

kerentanan ini terjadi ketika aplikasi web tidak mengenkripsi data yang dikirim antara server dan klien, sehingga memungkinkan penyerang untuk membaca data yang dikirim.Ini terjadi ketika pembatasan tentang apa yang diizinkan/tidak diizinkan oleh pengguna yang diberlakukan autentikasi secara tidak benar. Serangan kemudian dapat memanfaatkan untuk mendapatkan fungsionalitas yang tidak sah, termasuk mengakses dan mengubah akun pengguna, file sensitif, data pengguna, hak akses, dan banyak lagi.

Apa yang harus dilakukan untuk mencegahnya?

- Tolak akses secara default untuk semua hal kecuali sumber daya publik
- Buat mekanisme kontrol akses yang kuat dan gunakan di mana saja
- Jangan izinkan pengguna membuat, membaca, atau menghapus rekaman apa pun
- Nonaktifkan daftar direktori server, dan jangan simpan metadata di root folder
- Catat upaya akses yang gagal dan buat peringatan
- Nilai batas akses API

9. Using Components with Known Vulnerabilities

kerentanan ini terjadi ketika aplikasi web menggunakan komponen atau perpustakaan dengan kerentanan yang diketahui, sehingga memungkinkan penyerang untuk mengeksploitasi kerentanan tersebut. *Web developer* sering menggunakan komponen yang ada dalam aplikasi untuk menghindari pekerjaan yang berlebihan sambil menyediakan fungsionalitas yang dibutuhkan. Penyerang akan mencari kerentanan di dalam komponen ini yang dapat mereka manfaatkan untuk melakukan serangan pada aplikasi itu sendiri. Komponen populer dapat digunakan di ratusan ribu situs, dan satu kerentanan dapat membuat semuanya berisiko.

Pelanggaran Equifax pada tahun 2017 adalah contoh sempurna dari jenis kerentanan ini. Disebabkan oleh penggunaan versi Apache Struts yang memiliki kerentanan yang ditemukan enam bulan sebelum serangan. Andai saja mereka membaca OWASP Top 10 List terlebih dahulu, maka 700 juta USD mereka bisa diselamatkan.
Amankan kompones web aplikasi Anda dengan menerapkan tindakan-tindakan berikut:

- Selalu pastikan untuk memiliki patch keamanan terbaru dan pembaruan untuk komponen Anda.
- Hapus komponen yang tidak digunakan dari proyek Anda.
- Hanya dapatkan komponen dari sumber tepercaya.

- Gunakan Software Composition Analysis (SCA) untuk mengidentifikasi komponen yang sudah usang atau tidak aman.

10. Server-Side Request Forgery (SSRF)

kerentanan ini terjadi ketika aplikasi web menerima permintaan dari pengguna untuk mengakses sumber daya yang tidak aman atau tidak terpercaya, sehingga penyerang dapat memanipulasi permintaan tersebut untuk mengakses sumber daya yang seharusnya tidak dapat diakses. Deserialization adalah kebalikannya, dan ini memungkinkan peretas untuk mengeksekusi kode berbahaya di server. Meskipun kerentanan tidak mengakibatkan eksekusi kode jarak jauh, penyerang masih dapat menggunakannya untuk melakukan tindakan seperti serangan berulang, serangan injeksi, dan serangan eskalasi hak istimewa.

Lakukan tindakan preventif dari deserialisasi ini untuk mengamankan web aplikasi Anda:

- Pantau deserialisasi.
- Menerapkan tipe pemeriksaan.
- Larang deserialisasi data dari sumber yang tidak tepercaya.