**PENGEMBANGAN WEB (TEORI)**

**LAPORAN EKSPERIMEN MENGENAI PROBLEM SERANGAN SESSION FIXATION DENGAN PENYELESAIAN MENGGUNAKAN SOLUSI SESSION EXPIRATION TIME**

*Laporan ini disusun untuk memenuhi tugas 1 mata kuliah Pengembangan Web (Teori)*



Disusun oleh kelompok B4:

Asri Husnul Rosadi 221524035

Faris Abulkhoir 221524040

Mahardika Pratama 221524044

Muhamad Fahri Yuwan 221524047

Najib Alimudin Fajri 221524053

Sarah 221524059

Septyana Agustina 221524060

Dosen Pengampu:

Joe Lian Min, M.Eng.

**JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA**

**PROGRAM STUDI D4 TEKNIK INFORMATIKA**

**POLITEKNIK NEGERI BANDUNG**

**2024**

# **DAFTAR ISI**

[**DAFTAR ISI** i](#_Toc175601025)

[**A.** **IDENTIFIKASI PROBLEM** 1](#_Toc175601026)

[**B.** **DESKRIPSI PROBLEM** 1](#_Toc175601027)

[**C.** **METODOLOGI EKSPERIMEN** 1](#_Toc175601028)

[**D.** **PELAKSANAAN EKSPERIMEN** 3](#_Toc175601029)

[**E.** **ANALISIS HASIL EKSPERIMEN** 4](#_Toc175601030)

# **IDENTIFIKASI PROBLEM**

Dalam pengembangan aplikasi web, sesi pengguna yang tidak memiliki waktu kedaluwarsa (expiration time) dapat menyebabkan risiko keamanan, seperti sesi yang tetap aktif terlalu lama bahkan setelah pengguna meninggalkan aplikasi. Ini dapat menyebabkan potensi penyalahgunaan oleh pihak yang tidak sah, terutama jika perangkat pengguna ditinggalkan tanpa logout atau perangkat hilang.

# **DESKRIPSI PROBLEM**

Sesi pengguna biasanya dikelola menggunakan cookie yang disimpan di sisi klien. Cookie ini menyimpan ID sesi yang digunakan oleh server untuk mengidentifikasi pengguna yang sedang aktif. Tanpa expiration time, cookie sesi akan tetap ada sampai pengguna menutup browser atau sesi secara manual diakhiri. Hal ini dapat menyebabkan berbagai masalah, seperti:

* Penyalahgunaan sesi: Jika sesi tetap aktif, orang lain yang mengakses perangkat tersebut dapat menggunakan sesi yang sama tanpa perlu otentikasi ulang.
* Serangan XSS (Cross-Site Scripting): Penyerang dapat mencuri cookie sesi dan menggunakannya untuk mendapatkan akses tidak sah ke akun pengguna.

Penerapan expiration time pada cookie sesi memastikan bahwa sesi hanya akan aktif dalam waktu tertentu, mengurangi risiko tersebut.

# **METODOLOGI EKSPERIMEN**

1. **Desain Eksperimen**

Eksperimen ini menggunakan desain eksperimental sederhana yang berfokus pada penerapan expiration time pada cookie sesi pengguna. Eksperimen ini melibatkan pengujian aplikasi web sebelum dan sesudah penerapan expiration time untuk mengamati dampaknya terhadap durasi sesi pengguna.

1. **Variabel Eksperimen**

* Variabel Bebas (Independent Variable):
* Penerapan Expiration Time: Dalam eksperimen ini, variabel bebasnya adalah apakah cookie sesi diatur dengan atau tanpa expiration time.
* Variabel Terikat (Dependent Variable):
* Durasi Sesi Pengguna: Durasi sesi pengguna diukur sebagai variabel terikat. Eksperimen ini akan mengamati berapa lama sesi pengguna tetap aktif setelah login, baik sebelum maupun sesudah expiration time diterapkan.
* Variabel Terkendali (Controlled Variables):
* Lingkungan Pengujian: Server aplikasi web dijalankan pada konfigurasi yang sama di kedua kondisi, kecuali pada pengaturan expiration time.
* Atribut Cookie Lainnya: Semua atribut cookie lainnya (seperti httpOnly dan secure) akan dipertahankan sama untuk memastikan bahwa perbedaan yang diamati hanya disebabkan oleh perubahan pada expiration time.

1. **Prosedur Eksperimen**
2. Tahap Persiapan:

* Menyiapkan server aplikasi web menggunakan Express.js dengan sesi yang dikelola oleh middleware express-session.
* Membuat dua versi server: satu tanpa expiration time pada cookie sesi, dan satu dengan expiration time yang diatur pada 1 menit (60.000 milidetik).

1. Pengujian Tanpa Expiration Time:

* Akses endpoint /login untuk membuat sesi pengguna.
* Akses endpoint /profile untuk memverifikasi apakah sesi pengguna masih aktif.
* Periksa apakah sesi tetap aktif tanpa batas waktu hingga browser ditutup atau server di-restart.

1. Penerapan Expiration Time:

* Modifikasi server untuk menambahkan expiration time pada cookie sesi.
* Ulangi langkah yang sama dengan mengakses endpoint /login dan /profile.
* Amati apakah sesi kedaluwarsa setelah 1 menit sesuai dengan pengaturan expiration time.

1. **Alat dan Bahan**
2. Alat:

* Server Aplikasi: Server Express.js untuk menjalankan aplikasi web.
* Browser atau Alat Uji API: Digunakan untuk mengakses endpoint aplikasi dan memverifikasi status sesi.

1. Bahan:

* Kode Sumber: Kode JavaScript untuk aplikasi web dengan dan tanpa penerapan expiration time.

1. **Pengukuran dan Pengamatan**
2. Pengukuran:

* Waktu yang dihabiskan dari saat login hingga sesi kedaluwarsa atau hingga server menolak akses ke endpoint /profile.

1. Pengamatan:

* Perbedaan dalam durasi sesi sebelum dan sesudah penerapan expiration time.
* Respon aplikasi ketika sesi kedaluwarsa, termasuk apakah pengguna diminta untuk login ulang.

1. **Analisi Data**

Data yang dikumpulkan akan dianalisis untuk mengevaluasi dampak penerapan expiration time terhadap durasi sesi pengguna. Analisis ini akan mencakup perbandingan antara durasi sesi sebelum dan sesudah penerapan expiration time serta penilaian terhadap peningkatan keamanan yang dihasilkan oleh penerapan tersebut.

# **PELAKSANAAN EKSPERIMEN**

1. **Tahap Persiapan**

Eksperimen dilakukan dalam dua tahap utama: pengujian sebelum penerapan expiration time dan pengujian setelah penerapan expiration time. Server aplikasi dijalankan pada port yang sama (port 3000) untuk kedua tahap pengujian, dengan satu-satunya perbedaan adalah konfigurasi expiration time pada cookie sesi.

1. **Pengujian Tanpa Expiration Time**
2. Menjalankan Server:

* Server aplikasi dijalankan menggunakan kode tanpa expiration time yang telah disiapkan sebelumnya.

node server-before.js

1. Login Pengguna

* Akses endpoint /login melalui browser atau alat uji API untuk membuat sesi pengguna. Sesi ini tidak memiliki waktu kedaluwarsa, sehingga akan tetap aktif hingga browser ditutup atau sesi secara manual diakhiri.
* URL: http://localhost:3000/login
* Metode: GET

1. Pengujian Durasi Sesi

* Akses endpoint /profile beberapa kali dalam interval waktu yang berbeda (misalnya, 5 menit, 10 menit, dan 30 menit setelah login) untuk memverifikasi bahwa sesi tetap aktif.
* URL: http://localhost:3000/profile
* Metode: GET

1. **Pengujian Setelah Penerapan Expiration Time**
2. Menjalankan Server dengan Expiration Time:

* Server aplikasi dimodifikasi untuk menambahkan expiration time (1 menit) pada cookie sesi dan dijalankan dengan konfigurasi baru.

node server-after.js

1. Login Pengguna

* Akses endpoint /login untuk membuat sesi pengguna dengan cookie yang memiliki waktu kedaluwarsa 1 menit.
* URL: http://localhost:3000/login
* Metode: GET

1. Pengujian Durasi Sesi

* Akses endpoint /profile dalam waktu kurang dari 1 menit setelah login untuk memverifikasi bahwa sesi masih aktif.
* Tunggu lebih dari 1 menit, lalu akses endpoint /profile kembali untuk memeriksa apakah sesi telah kedaluwarsa.

Hasil:

* Sesi pengguna tetap aktif selama 1 menit setelah login, kemudian kedaluwarsa, dan akses ke /profile ditolak dengan status 401.

1. **Dokumentasi dan Pengumpulan Data**

Selama pelaksanaan eksperimen, hasil-hasil pengujian dicatat, termasuk waktu akses endpoint /profile dan status respons yang diterima dari server. Data ini digunakan untuk mengevaluasi efektivitas penerapan expiration time pada cookie sesi.

# **ANALISIS HASIL EKSPERIMEN**

1. **Hasil Pengamatan**

* Pengujian Tanpa Expiration Time:
* Sesi pengguna tetap aktif tanpa batas waktu, memungkinkan akses terus-menerus ke halaman profil (/profile) selama browser tetap terbuka.
* Tidak ada mekanisme yang membatasi durasi sesi, sehingga sesi dapat disalahgunakan jika perangkat pengguna ditinggalkan tanpa logout.
* Pengujian Setelah Penerapan Expiration Time:
* Sesi pengguna kedaluwarsa tepat setelah 1 menit seperti yang diharapkan.
* Setelah waktu kedaluwarsa, akses ke halaman profil (/profile) ditolak, dan pengguna harus login ulang untuk mendapatkan sesi baru.

1. **Analisis Pros dan Cons**

**Pros:**

1. Keamanan Meningkat:

* Penerapan expiration time pada cookie sesi berhasil membatasi durasi sesi, mengurangi risiko penyalahgunaan sesi oleh pihak yang tidak sah.
* Hal ini sangat penting dalam aplikasi yang menangani data sensitif atau digunakan di lingkungan publik.

1. Kontrol Sesi yang Lebih Baik:

* Dengan adanya expiration time, administrator aplikasi memiliki kontrol yang lebih baik terhadap durasi sesi pengguna, memungkinkan pengelolaan keamanan yang lebih ketat.

**Cons:**

1. Potensi Ketidaknyamanan Pengguna:

* Pengguna mungkin merasa terganggu jika sesi kedaluwarsa terlalu cepat, terutama jika mereka sedang bekerja atau melakukan aktivitas penting dalam aplikasi.
* Ini dapat menyebabkan pengalaman pengguna yang kurang optimal jika waktu kedaluwarsa tidak diatur dengan bijaksana.

1. Kebutuhan untuk Penanganan Ulang Sesi:

* Jika pengguna melakukan aktivitas yang membutuhkan waktu lebih lama dari durasi sesi, mereka harus login ulang, yang bisa menjadi kurang nyaman.
* Ini membutuhkan strategi tambahan untuk menangani kasus di mana pengguna berada di tengah aktivitas penting saat sesi kedaluwarsa (misalnya, memberi peringatan sebelum sesi berakhir).

1. **Kesimpulan**

Eksperimen ini menunjukkan bahwa penerapan expiration time pada cookie sesi secara efektif membatasi durasi sesi pengguna, meningkatkan keamanan aplikasi web dengan mengurangi risiko penyalahgunaan sesi yang terlalu lama aktif. Meskipun penerapan expiration time dapat menyebabkan sedikit ketidaknyamanan bagi pengguna, manfaat keamanannya jauh lebih besar, terutama dalam aplikasi yang menangani informasi sensitif atau yang digunakan di lingkungan publik. Penting untuk menyeimbangkan antara keamanan dan kenyamanan pengguna dengan menetapkan waktu kedaluwarsa yang sesuai berdasarkan jenis aplikasi dan kebiasaan pengguna.