

Architettura di sicurezza: identità, SSO, autorizzazione, audit e rilascio sicuro

Guida della lezione · Guida al lab / esame

 LearnWisej

Guida della lezione

Integrazione dell'identità, modello dei permessi, audit trail, HTML sicuro, configurazione sicura. Che cosa significa per un team Wisej.NET avanzato — e come affrontare questo modulo.



Focus di produzione

Integrazione dell'identità, modello dei permessi, audit trail, HTML sicuro, configurazione sicura.

1. Il problema in produzione

Un'architettura lato server riduce molte superfici di attacco del browser, ma i sistemi avanzati hanno comunque bisogno di integrazione con l'identità, autorizzazione a livello di servizio, audit trail, rendering sicuro e una configurazione operativa sicura.

2. Concetti chiave

Concetto	Perché è importante
confine OIDC/SSO	Il punto in cui un identity provider esterno consegna all'applicazione un'identità verificata. Tutto ciò che viene dopo quel confine è responsabilità dell'applicazione.
mappatura dei claims	Tradurre i claims specifici del provider nei ruoli e nei permessi propri dell'applicazione, così il codice di autorizzazione non dipende mai dal formato di un singolo identity provider.
servizio dei permessi	Un unico servizio che risponde se l'utente corrente può eseguire un'azione, così la regola vive in un solo punto invece che in ogni schermata.
autorizzazione a livello di servizio	Imporre i permessi dentro i servizi applicativi, non solo nella UI, così un pulsante nascosto non è mai l'unica cosa a proteggere un'operazione.
log di audit	Una registrazione a prova di manomissione di chi ha fatto che cosa e quando per le azioni importanti, così gli eventi di sicurezza si possono indagare e la conformità si può dimostrare.
revisione di AllowHtml	Verificare ogni punto in cui un controllo ammette HTML grezzo per assicurarsi che il contenuto fornito dall'utente sia sanificato, perché l'HTML non revisionato è un rischio di cross-site scripting.
gestione dei segreti	Tenere stringhe di connessione, chiavi e token in una configurazione sicura o in un vault, mai nel controllo del codice sorgente o in impostazioni visibili dal client.
header di sicurezza	Header di risposta HTTP come la content security policy e la strict transport security, che riducono la superficie di attacco del browser per l'applicazione.

3. Il filo della lezione

Autenticazione non significa autorizzazione

Un utente che ha fatto login non è automaticamente autorizzato ad approvare un ordine di lavoro, vedere un altro tenant, esportare dati o eseguire un'attività amministrativa. Il controllo dei permessi appartiene al servizio che esegue l'azione.

Il contesto di sessione fa da ponte tra identità e UI

Dopo l'autenticazione, l'applicazione dovrebbe mappare l'identità in un contesto di sessione: ID utente, nome visualizzato, tenant, ruoli, permessi, cultura, tema e correlation ID. Questo contesto aiuta le schermate ad adattarsi, mentre i servizi continuano a imporre le regole.

Un modello di identità e autorizzazione per l'enterprise

L'autorizzazione enterprise ha più dimensioni di un nome di ruolo. La stessa persona può essere dispatcher in un tenant, auditor in sola lettura in un altro, e autorizzata ad approvare ordini di lavoro solo sotto un limite di costo. Un modello in grado di esprimerlo parte da tre livelli: l'identità, che dice chi è l'utente;

l'appartenenza, che dice quali tenant e ruoli detiene; e i permessi, che sono le azioni granulari concesse da un ruolo, come `ApproveWorkOrder` o `ExportData`.

Il lab trasforma tutto questo in una matrice dei permessi: le righe sono i permessi, le colonne sono i ruoli, e ogni cella è concesso, negato o condizionale. La matrice è l'artefatto che product owner, revisori della sicurezza e sviluppatori possono leggere tutti, quindi dovrebbe esistere prima del codice. Anche i passi dei flussi di lavoro vi si mappano. Inviare un'escalation, approvarla e chiuderla possono richiedere ciascuno un permesso diverso, e la matrice lo rende esplicito.

Mantieni il modello incentrato sul tenant. Un permesso viene sempre valutato all'interno di un contesto del tenant, così un utente che ha un ruolo nel tenant A non ottiene nulla nel tenant B. La chiamata `PermissionService.Demand` del lab verifica prima la guardia del tenant e solo dopo consulta l'archivio dei ruoli.

Gate di autenticazione, contesto di sessione e servizi

Un identity provider esterno, raggiunto tramite un login OpenID Connect o SSO, dice all'applicazione chi è l'utente. È lì che finisce il compito del provider. L'applicazione mappa poi i claims in arrivo nel proprio contesto di sessione: ID utente, nome visualizzato, tenant, ruoli, permessi e un correlation ID. Poiché la mappatura viene eseguita una volta al login ed è memorizzata nella sessione, le schermate possono leggerla a basso costo e i servizi possono farvi affidamento senza ripetere il parsing dei claims.

Il gate stesso deve stare davanti a ogni punto di ingresso, non solo alla pagina principale. Job in background, chiamate WebMethod, gestori di postback e pagine di diagnostica devono tutti sapere chi sta chiamando. Un servizio non dovrebbe mai leggere l'identità da un controllo della UI, da una query string o da un campo nascosto, perché tutti questi possono essere modificati dall'utente. Legge il contesto di sessione, e il contesto di sessione è stato costruito da claims verificati.

Progetta il contesto in modo che possa essere aggiornato. I ruoli cambiano mentre le sessioni sono aperte, e una sessione di lunga durata deve poter ricaricare i permessi senza forzare un nuovo login. La domanda di revisione per ogni metodo di servizio è semplice: quale riga dimostra che il chiamante ha il diritto di essere qui.

Audit trail per i comandi sensibili

Un audit trail risponde a chi ha fatto che cosa, su quale record, in quale tenant, quando e con quale risultato. Viene scritto dal servizio che esegue il comando, all'interno della stessa operazione, così un'azione non può riuscire senza essere registrata. La schermata del log di audit del lab mostra ogni `Demand`, concessa o negata, e ogni comando sensibile come un'approvazione, un'esportazione o una modifica dei permessi.

I dinieghi sono importanti quanto i successi. Un pulsante abilitato per errore che permette a un utente di richiedere `ExportData` viene intercettato dal servizio, che rifiuta con un errore con un nome e scrive una voce di diniego. Dinieghi ripetuti da un account sono il segnale che qualcuno sta sondando l'applicazione, e sono visibili solo se i dinieghi vengono registrati. La voce dovrebbe portare il correlation ID, così si collega ai log strutturati del Modulo 11.

- Autore, tenant ed eventuale flag di impersonificazione.
- Nome del comando, record di destinazione e i valori prima e dopo che contano.

- Esito: concesso, negato o fallito, con il motivo con un nome.
- Timestamp e correlation ID, scritti nella stessa transazione della modifica.

Una checklist di sicurezza per l'intera superficie

Il rendering lato server elimina molti rischi lato browser, ma un'applicazione Wisej.NET invia comunque HTML, accetta upload, imposta cookie, scrive log, legge segreti e sta dietro a un proxy. Ognuno di questi è un punto in cui un piccolo errore diventa un incidente. La checklist di hardening del lab è il documento che trasforma questi punti in voci di revisione verificate a ogni rilascio, invece di ricordarle occasionalmente.

Comincia dall'HTML. Qualsiasi controllo che visualizza HTML grezzo proveniente dall'input dell'utente deve sanificarlo, e la revisione di `AllowHtml` del lab elenca ogni controllo di questo tipo. Gli upload hanno bisogno di limiti di dimensione, verifiche del tipo e archiviazione fuori dalla web root. I cookie dovrebbero essere marcati secure e HTTP-only, con una policy same-site. Una content security policy limita quali script possono essere eseguiti. I log non devono mai contenere password, token o record personali completi. I segreti vivono in un vault o in una configurazione protetta, mai nel controllo del codice sorgente. Un reverse proxy deve inoltrare lo schema e l'indirizzo client originali, così l'applicazione vede i valori reali.

Tratta la checklist come codice vivo: aggiungi una voce ogni volta che una revisione trova una lacuna, e registra chi ha verificato ogni voce per ogni rilascio.



La regola di produzione avanzata

Non fermarti quando la schermata sembra corretta. Prima di considerare completo il modulo, spiega a chi appartiene lo stato, le implicazioni di sicurezza, i percorsi di errore e il comportamento in produzione.



Obiettivi di apprendimento

Alla fine di questo modulo sarai in grado di:

- ✓ Progettare un modello di identità e autorizzazione per ruoli, tenant e flussi di lavoro enterprise.
- ✓ Integrare i gate di autenticazione con il contesto di sessione e i servizi applicativi.
- ✓ Costruire audit trail per i comandi sensibili.
- ✓ Creare checklist di sicurezza per HTML, upload, cookie, CSP, log, segreti e reverse proxy.

Guida al lab / esame

Che cosa costruirai nel lab pratico, l'approccio suggerito e i deliverable richiesti.



Lab pratico

Aggiungi una simulazione di autenticazione enterprise, la mappatura dei claims sui permessi, l'imposizione dei ruoli per tenant, l'autorizzazione a livello di servizio, la revisione dell'HTML sicuro, l'approvazione delle esportazioni e il log di audit per le azioni sensibili.

Approccio suggerito

- ✓ Leggi il problema in produzione e individua il rischio architetturale prima di toccare la UI.
- ✓ Crea o aggiorna per primo l'incapsulamento in un servizio, comando, modello o componente coinvolto.
- ✓ Costruisci la schermata o il componente Wisej.NET seguendo gli standard di progetto stabiliti.
- ✓ Aggiungi una dimostrazione del percorso di errore, non solo del percorso felice.
- ✓ Annota che cosa è cambiato nel registro di apprendimento e aggiorna la checklist relativa.

Deliverable richiesti

- ✓ Progetto della mappatura delle identità
- ✓ Matrice dei permessi
- ✓ Implementazione del servizio dei permessi
- ✓ Schermata del log di audit
- ✓ Checklist di hardening della sicurezza

Domande di revisione per lo studente

- ✓ Dove viene imposta l'autorizzazione?
- ✓ Quali elementi della UI visualizzano HTML?
- ✓ Un pulsante abilitato può aggirare le regole del servizio?



La regola di produzione avanzata

Non fermarti quando la schermata sembra corretta. Prima di considerare completo il modulo, spiega a chi appartiene lo stato, le implicazioni di sicurezza, i percorsi di errore e il comportamento in produzione.