LocknShop: Documento di sicurezza

Autori:
Adjetey Isabelle
Andreetti Luca
Carillo Vincenzo
Gandolfi Matteo
Masa Biniam Abraha
Pinto Sabrina
Polzoni David

Data:
27 giugno 2025

Abstract

Questo documento analizza i principali rischi di sicurezza relativi all'applicazione **LocknShop**, facendo riferimento al framework **OWASP Top 10** per la classificazione delle vulnerabilità più critiche nei sistemi web. In particolare, vengono approfondite tematiche come l'esposizione non controllata delle API (*API Security*), la gestione delle credenziali (*Broken Access Control*, *Cryptographic Failures*), e la protezione contro attacchi lato client, come XSS (*Injection*). Sono proposte misure di mitigazione concrete, basate su funzionalità offerte da **Microsoft Azure**, tra cui *Key Vault* per la protezione delle informazioni sensibili, *Application Insights* per il tracciamento sicuro delle attività e *Azure Monitor* per l'identificazione di anomalie. Viene inoltre affrontato il tema della scalabilità e resilienza dell'infrastruttura, con un piano di miglioramento che include l'adozione di meccanismi avanzati di autenticazione (es. MFA, token rotation) e gestione sicura delle sessioni (*Identification and Authentication Failures*).

Capitolo 1

ASP: Application Security Policy

1.1. Esposizione accidentale delle API (es. path traversal + IDOR)

Rischio: le API backend potrebbero essere accessibili pubblicamente anche senza un'autenticazione adeguata (es. endpoint non protetti da JWT).

Mitigazione:

- Controlli di autorizzazione su ogni endpoint;
- Middleware di validazione JWT sempre attivo;
- CORS ristretto solo al dominio frontend.

1.2. Gestione delle credenziali (es. data breach)

Rischio: se la chiave segreta del JWT è salvata in chiaro nel codice o nel repository GitHub, può essere estratta e usata per generare token falsi.

Mitigazione:

- Uso di **Azure Key Vault** per salvare la chiave;
- Esclusione di file .env, secrets.json, appsettings.json dal controllo versione;
- Logging degli accessi anomali.

1.3. Mancanza di rate limiting (es. DDoS) - fuori scope

Rischio: assenza di protezione contro attacchi brute-force sul login, spam sulle API (es. creazione ripetuta di carrelli) e/o richieste automatizzate da bot.

Mitigazione:

- Middleware per throttling/rate limiting;
- Blocchi temporanei dopo N tentativi falliti;
- Alert automatici tramite **Application Insights**.

1.4. Logging insufficiente o eccessivo (es. SQLi, user enumeration)

Rischio: se i log non sono abbastanza informativi, è difficile rilevare attacchi o bug. Se troppo dettagliati, possono esporre dati sensibili.

Mitigazione:

- Log e input sanificati e crittografati (es. nessuna email o token);
- Logging centralizzato con **Azure Monitor**;
- Alert per login falliti multipli o attività sospette.

1.5. Errori di configurazione in produzione (misconfiguration)

Rischio: ambienti mal configurati (es. debugging attivo, connessione pubblica al database, CORS aperto, messaggi di errore troppo dettagliati).

Mitigazione:

- Ambiente di produzione separato da quello di sviluppo;
- Logging attivo ma senza *stacktrace* dettagliati;
- Accesso al database limitato a IP interni o frontend autorizzato.

1.6. Scalabilità limitata

Rischio: con l'aumento degli utenti o del traffico, la piattaforma potrebbe rallentare se non predisposta per scalare orizzontalmente (es. tier troppo basso di **App Service** o database).

Mitigazione:

- Monitoraggio attivo per identificare i colli di bottiglia;
- Possibilità di upgrade automatico del piano;
- Cache e ottimizzazione delle query lato database.

1.7. Attacchi lato client (XSS reflected o persistente)

Rischio: se l'output Angular non è correttamente sanificato, un attaccante può iniettare script nel DOM (Cross-Site Scripting).

Mitigazione:

- Uso corretto del binding ([innerHTML] vietato o sanificato);
- Escaping sistematico nelle view dinamiche;
- Intestazioni HTTP con Content-Security-Policy (CSP).

1.8. Mancanza di gestione avanzata della sessione

Rischio: se il token JWT non può essere invalidato o rigenerato, l'utente non può eseguire un logout effettivo o revocare un token compromesso.

Mitigazione:

- JWT con scadenza breve (es. 1 ora);
- Pianificazione dell'uso di refresh token;
- Blacklist lato backend per i token compromessi.

1.9. Upload futuri non controllati (storage)

Rischio: in caso di upload futuri (es. immagini prodotto o file), potrebbero essere sfruttati per caricare malware.

Mitigazione:

- Validazione di estensione e mime-type lato backend;
- Scansione antivirus su **Azure Storage**;
- Quarantena o approvazione manuale dei file.

1.10. Mancanza di MFA / autenticazione forte

Rischio: l'accesso con sola email e password è oggi considerato debole.

Mitigazione:

- Integrazione futura del MFA tramite **MS Entra ID** o altri provider;
- Aggiunta di conferma email o codice OTP.

Il sistema è stato progettato con attenzione alla sicurezza sin dalle prime fasi, tenendo conto delle minacce comuni a livello frontend, backend e infrastrutturale. Le contromisure adottate, tra cui autenticazione robusta, protezione delle API, logging sicuro e meccanismi di rate limiting, contribuiscono a ridurre significativamente la superficie di attacco. La seguente tabella riassume in modo sintetico i principali rischi identificati e le relative mitigazioni applicate.

Rischio	**Mitigazione**
API esposte	Autorizzazione per endpoint, JWT obbligatorio, CORS limitato
Credenziali compromesse	Azure Key Vault, esclusione file sensibili, logging accessi
Assenza rate limiting	Throttling, blocchi temporanei, alert automatici
Logging inadeguato	Log crittografati e sanificati, Azure Monitor, alert sospetti
Errori di configurazione	Ambiente separato, accessi limitati, debug disattivato
Scarsa scalabilità	Monitoraggio risorse, upgrade automatico, cache e query ottimizzate
XSS client-side	Binding sicuro, escaping, Content-Security-Policy
Sessioni non gestite	Token a breve scadenza, refresh token, blacklist JWT
Upload pericolosi	Validazione MIME, antivirus, quarantena file
Autenticazione debole	MFA, OTP, verifica email

Tabella 1.1: Tabella riepilogativa delle vulnerabilità e contromisure adottate.

Capitolo 2

Riepilogo e note aggiuntive

Durante la progettazione del sistema **LocknShop**, è stata posta particolare attenzione alla sicurezza, alla scalabilità e alla corretta gestione dei dati utente e delle operazioni di business. In particolare, si è proceduto a una valutazione sistematica dei principali rischi che possono compromettere la disponibilità, la riservatezza e l'integrità del sistema. Tra le minacce considerate vi sono attacchi noti come *path traversal*, *IDOR*, DDoS, Cross-Site Scripting, accessi non autorizzati e configurazioni errate, sia lato backend che frontend. Ognuno di questi rischi è stato analizzato e affrontato con misure tecniche e organizzative adeguate, che seguono le best practice suggerite dall'OWASP e dalle linee guida Microsoft per applicazioni cloud-native.

Le principali contromisure adottate includono:

- Autenticazione robusta basata su token JWT firmati;
- Gestione centralizzata delle chiavi/connection strings tramite **Azure Key Vault**;
- Rate limiting e controllo degli accessi su ogni endpoint API;
- Logging centralizzato e monitoraggio continuo tramite **Azure Monitor** e **Application Insights**;
- Protezione contro attacchi client-side mediante CSP e sanificazione dei contenuti;
- Separazione netta tra ambienti di sviluppo e produzione, con configurazioni sicure e restrizioni di rete;
- Approccio *security by design*, includendo validazione dell'input e dei file caricati, auditing e controllo delle dipendenze.

L'infrastruttura è stata progettata con un'ottica di scalabilità futura: l'impiego di **App Service** plans adattabili, database con livelli di prestazioni configurabili e servizi gestiti contribuisce a minimizzare i rischi operativi e ad assicurare un'elevata affidabilità in ambienti di produzione.
Il sistema si presenta in una fase matura dal punto di vista della sicurezza e dell'architettura. Tuttavia, alcune funzionalità avanzate (non essenziali per una prima versione ma utili per la sicurezza e l'esperienza utente) sono state previste nella roadmap del progetto. Tra queste:

- Gestione avanzata della sessione con *refresh token* e revoca;
- Integrazione della **MFA** tramite **Microsoft Entra ID** o equivalenti;
- Sistema di prevenzione sull'identificazione di anomalie;
- Dashboard di amministrazione con tracciamento degli eventi critici.

Il grafico di seguito è un radar chart (o grafico a ragnatela) che rappresenta il livello attuale di copertura delle principali misure di sicurezza implementate nel sistema **LocknShop**. Ogni asse del grafico corrisponde a un'area specifica della sicurezza informatica, mentre il valore radiale (da 1 a 5) indica il grado di maturità o di copertura di quella misura. Le aree più robuste includono **autenticazione**, **gestione credenziali** e **logging**, mentre aspetti come **MFA** o **Upload** (gestione sicura dei file caricati) risultano ancora parzialmente coperti e sono previsti nella roadmap. Il radar chart fornisce una panoramica chiara dei punti di forza e delle aree da migliorare in termini di sicurezza.

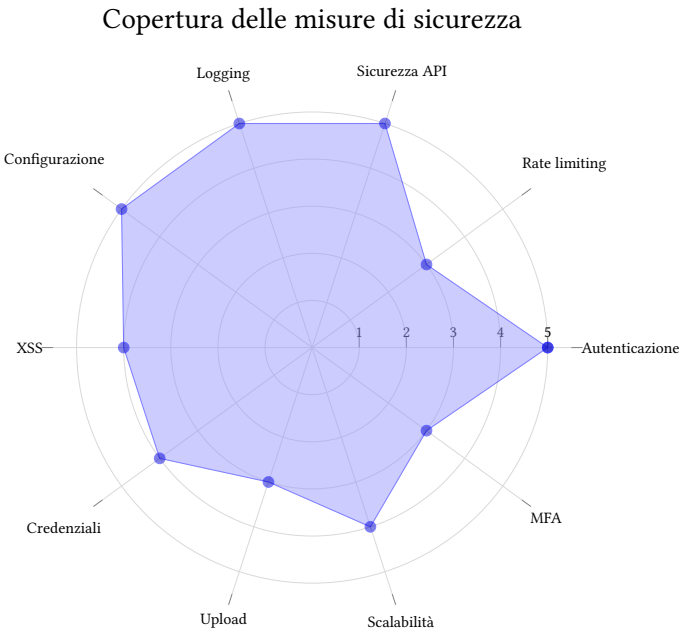

Figura 2.1: Copertura attuale delle principali aree di sicurezza.