Indice

1 — Introduzione

Questo documento presenta la progettazione di una piattaforma per la generazione di risorse di deception. In particolare, ci si concentrerà sulla generazione di RestApi server, con la possibilità di aggiungere, in futuro, altre tipologie di risorse, come provider di sicurezza o database relazionali.

La piattaforma sarà composta da due parti principali: un'interfaccia grafica per la configurazione e la generazione delle risorse, ed un backend per gestire la logica di business.
I dettagli architetturali verrano però esaminati in seguito dopo una precisa definizione ed analisi dei requisiti.

1.1 Core-business Features

Le funzionalità *core* che verranno fornite sono:

- creazione di un server RestApi, con la possibilità di configurare sotto ogni punto di vista gli endpoint esposti, le relative implementazioni e risposte.
- configurazione della sicurezza, implementabile in maniera automatica sugli endpoint di interesse;
- generazione e popolamento di database partendo da template fornito dall'utente (e.g. tabella sql o json).

La piattaforma che si vuole implementare ha inoltre l'obiettivo di coinvolgere una **community di utenti** che sfrutti i servizi e collabori al fine di generare componenti di deception sempre più complessi e realistici.

Entrando in un maggior livello di dettaglio ed integrando quanto appena detto, rimanendo nel contesto in cui le risorse di deception generate siano RestApi server, verranno messe a disposizione le seguenti funzionalità:

- forum
- realizzare un server RestApi completo e pronto all'uso
- realizzare, condividere ed utilizzare di altri:
  - esempi di configurazioni generali della specifica OpenApi che risultino realistiche ed affidabili
  - entità alla base della specifica, con dati realistici
  - endpoint e (opzionalmente) eventuale implementazione
  - configurazioni di sicurezza ed eventuale associazione agli endpoint definiti
  - database e popolamento con dati realistici
  - eventuali pagine restituite da un certo endpoint (e.g. pagina di *admin* o di *login*)
- possibilità di navigare le risorse condivise, salvarle, votarle, ...
- sezioni per impostazioni utente, istruzioni, faq, contatti, ...

Sarà presente una sezione per l'autenticazione e si dovra valutare la possibilità di definire diversi livelli per l'accesso alle funzionalità. Un'idea può essere la versione *"pro"* per entrare nella community e utilizzare le risorse condivise.

1.1.1 Funzionalità aggiuntive

Oltre ai topic principali appena visti, saranno presenti funzionalità aggiuntive per:

- testare gli endpoint generati;
- supportare il versioning delle API (e dei server generati);
- consultare logging e report degli attacchi;
- introdurre integrazioni con servizi di threat intelligence per arricchire le informazioni sui potenziali attacchi e le tecniche utilizzate dagli attaccanti;
- integrare sistemi di rilevamento delle minacce (come intrusion detection system, a livello di rete) per monitorare in tempo reale il traffico delle API e rilevare comportamenti sospetti o anomalie che potrebbero indicare un potenziale attacco.

1.1.2 Realisticità dei RestApi server generati

Il fatto che i RestApi server generati appaiono realistici corrisponde al requisito principale, in quanto, senza di esso, verrebbe a meno lo scopo principale per cui esistono: indurre gli utenti malevoli ad attaccare questi servizi, facendogli credere che siano risorse autentiche.
Per raggiungere questo obiettivo, oltre alle funzionalità già viste, vengono riportati alcuni concetti generali che saranno alla base di questi server:

- il server deve essere ben strutturato e sicuro
- gli endpoint presenti devono essere sensati e coerenti con lo scenario applicativo dell'azienda che li utilizza
- le risposte agli endpoint devono essere reali, coerenti a fronte di interazioni distinte; contestualmente, anche gli errori forniti in risposta devono essere sensati
- i dati su cui si basano le risposte degli endpoint devono apparire come realistici
- integrazione con servizi di terze parti (fittizi o meno), come database, servizi di autenticazione, sistemi di monitoraggio, ...

1.1.3 Funzionalità relative alla Deception

Di seguito, verranno riportate tutte le funzionalità offerte, relative ai solo aspetti di generazione componenti di Deception, che corrispondono al core-business della piattaforma:
Per definire questi requisiti mi sono messo nel contesto di un esperto di sicurezza che deve proteggere le risorse informatiche dell'azienda per cui lavora. L'idea è quella di poter, oltre a creare dei RestApi server generali e personalizzati, anche di generare dal Rest Api server esistenti, ulteriori che siano però fittizi, che quindi abbiano la stessa struttura di quelli esistenti, ma che coinbolgano dati e risposte fittizie.
Lo scopo deve essere *in primis* quello di far sprecare tempo e risorse agli attaccanti su un asset senza valore. ma deve essere anche possibile raccogliere informazioni sugli attacchi e sulle eventuali vulnerabilità trovate, che potrebbero essere presenti anche nelle risorse reali dell'azienda.

Detto ciò, saranno presenti le seguenti funzionalità:

-

1.1.4 Estensioni e sviluppi futuri

Vengono riportate di seguito alcune funzionalità avanzate che verranno eventualmente implementate se ve ne sarà la possibilità:

- Collaborazione in tempo reale per la configurazione e l'implementazione delle risorse;
- Supporto multi-lingua per migliorare l'accessibilità;
- Marketplace di plugin per estendere le funzionalità della piattaforma tramite plugin e add-ons;
- Integrazione con Ambiente di Simulazione per eseguire test di attacchi e difese in un ambiente controllato e sicuro.

Altre idee potrebbero sorgere durante gli sviluppi.
In ogni caso il software sarà strutturato in modo tale da poter aggiungere funzionalità ed estensioni in maniera comoda, senza stravolgimenti del codice già presente.

1.2 Obiettivi architetturali

Astraendo dalle funzionalità applicative legate ai servizi forniti, questo progetto si pone l'obiettivo di realizzare un'architettura all'avanguardia, che soddisfi requisiti di scalabilità, disponibilità, fault tolerance, resilienza e sicurezza.

L'architettura deve essere progettata per consentire sia la scalabilità orizzontale che quella verticale. Sarà necessario pensare in termini di load balancing, auto-scaling, e gestione del traffico, sfruttando tecniche di containerizzazione e orchestrazione di essi.

Ci si concentrerà sulla disponibilità e sulla fault-tolerance, con meccanismi di monitoraggio continuo e di rilevamento automatico dei guasti per garantire tempi minimi di inattività e massima disponibilità dei servizi. Verranno integrati anche meccanismi di ridondanza, failover e ripristino automatico dei servizi.

In generale, la piattaforma sarà conforme alle specifiche e ai requisiti di un **sistema enterpise**.