RELAZIONE VAPT

Breve Descrizione dell'Obiettivo del Test

Questa relazione descrive le fasi e i risultati di un penetration test condotto sulla web application “Juice Shop” ospitata su un container Docker.
L'obiettivo del test era identificare e sfruttare eventuali vulnerabilità per valutare la sicurezza del sistema.
Sono stati utilizzati vari strumenti e tecniche per raccogliere informazioni, valutare le vulnerabilità, tentare di sfruttarle e analizzare le conseguenze di un accesso non autorizzato.
La metodologia seguita includeva le fasi di Information Gathering, Vulnerability Assessment, Exploitation e Post-Exploitation, utilizzando strumenti come nmap, Burp Suite, wpscan, nikto, Joomscan e ffuf.
Burp Suite è stato impiegato come proxy locale tra il client (noi) e il server (ospitato su Docker) per intercettare e gestire tutte le richieste, permettendoci di analizzarle dettagliatamente.

Information gathering

1. Abbiamo utilizzato il comando “`docker ps"` per ottenere l'ID del container e il numero di porta del Juice Shop, in seguito tramite “`docker inspect <IDContainer>`” abbiamo ricavato anche l’indirizzo ip della webapp.
2. Tramite il tool nmap abbiamo utilizzato il comando “`sudo nmap -p3000 -sV -O 172.17.0.2`” per ottenere la versione del web server sulla porta 3000/tcp, ma non è stata rilevata alcuna versione specifica.
3. Abbiamo eseguito una scansione della web application con nikto, generando un output con informazioni su vulnerabilità e percorsi nascosti.
   L’output generato da questa scannerizzazione è visibile nel file allegato **“ScanNikto.txt”**, in particolare abbiamo utilizzato questa parte di output per iniziare la nostra ricerca di vulnerabilità:
   “`+ GET /ftp/: This might be interesting.`
   `+ GET /public/: This might be interesting.`”
   In seguito abbiamo analizzato i contenuti dei percorsi citati sopra.
4. Eseguendo una scansione con lo strumento Joomscan, fornendo sempre l'URL della webapp, abbiamo rilevato molti percorsi che sono risultati falsi positivi e quindi inesistenti.
   Tuttavia, è stato individuato anche un percorso contenente informazioni sensibili: “`http://172.17.0.2:3000/security.txt`”.

5. Tramite Joomscan abbiamo trovato anche un percorso "administrator", inizialmente sembrava un percorso falso positivo, così abbiamo provato parole simili e abbiamo trovato una corrispondenza con “administration”. Tuttavia non potevamo ancora visualizzare la pagina, in quanto non avevamo i permessi dell’admin.
6. Tramite l’utilizzo del tool ffuf, abbiamo esaminato tutte le directory esistenti, ignorando i falsi positivi grazie ad un filtro per la size delle risposte.
   L'output della scansione è stato salvato nel file allegato **"FfufScan.txt"**.
   Siamo riusciti a identificare un link rotto:
   “http://172.17.0.2:3000/assets/”

Vulnerability Assessment

1. Abbiamo inserito l'URL “http://172.17.0.2:3000/ftp” nella barra di ricerca per visualizzare i file contenuti all’interno della cartella trovata dalla scansione con Nikto.
   La directory contiene i seguenti file:

   a. Una cartella "quarantine" con al suo interno 4 malware riposti in quarantena dagli sviluppatori del sito.
   b. Un file "acquisizione", con al suo interno riportate informazioni sensibili che abbiamo riportato nel file **"acquisitions.txt."**
   c. La versione del framework utilizzata dal client, `"Express ^4.17.1"`, che ci impedisce di visualizzare file che non siano in formato .md o .pdf.
      Potendo risalire alla versione del framework utilizzata dalla webapp, gli attaccanti potrebbero cercare vulnerabilità specifiche sulla rete per quella versione, avendo un punto d’inizio per i loro attacchi.
      Consiglierei di rendere privata questa informazione e di mantenere aggiornata la versione il più possibile.
   d. Un file criptato "at-rest" chiamato "announcement_encrypted.md", che non dovrebbe essere accessibile agli utenti.
2. All’interno del percorso “http://172.17.0.2:3000/security.txt” individuato tramite una scansione con il tool Joomscan, abbiamo trovato:
   a. Un contatto: mailto:donotreply@owasp-juice.shop
   b. Un percorso secondario “https://keybase.io/bkimminich/pgp_keys.asc?fingerprint=19c01cb7157e4645e9e2c863062a85a8cbfbdcda” con la chiave pubblica e la sua versione, visibile nel file riportato **"PublicKey.txt"**, che potrebbe essere sfruttata per decifrare informazioni criptate riservate.

    c. La data di scadenza: `Thu, 12 Jun 2025 09:24:05 GMT`.
3. L'URL rotto "`http://172.17.0.2:3000/assets/`" rappresenta una vulnerabilità in quanto punta ad una pagina inesistente.
   Un attaccante potrebbe caricare un file con codice malevolo sul link, così quando l'utente naviga al suo interno potrebbe diventare vittima di un attacco di tipo "takeover".
4. Intercettando le richieste tramite il proxy di Burp, siamo riusciti ad identificare la query e la versione del database (SQL) utilizzato per la fase di login, inserendo nei dati di login parametri non attesi generando un errore.
   Da qui abbiamo provato un attacco di tipo SQL-injection, i risultati sono visibili nella reportistica dell'exploitation.
5. Provando le password più comuni, siamo riusciti ad identificare la password dell'admin, "admin123."
   Questo rappresenta un problema, in quanto il sito permette la configurazione e l'utilizzo di password molto deboli, e non a norma, per preservare i dati privati degli utenti.
6. Una volta ottenuti i privilegi dell'utente admin, siamo riusciti a visualizzare i contenuti del percorso:"`http://172.17.0.2:3000/administration`".
   Tramite questo percorso siamo riusciti a visualizzare le mail di tutte le persone con un account al Juice Shop.
   Tutti gli utenti sono visibili nel file allegato **"Users.txt"**.
7. Nel sito sono presenti molti path falsi positivi che intralciano l'accessibilità dello stesso sito, consigliamo di eliminare questi percorsi.

Exploitation

Tentativi di Sfruttamento delle Vulnerabilità:

1. Abbiamo provato a loggarci in un account della webapp utilizzando una SQL injection, in particolare inserendo un'espressione che ritorni sempre il valore TRUE nel campo "Email," e iniziando un commento in modo da evitare la validazione della password.
   Facendo ciò siamo riusciti a loggarci tramite l'account "admin@juice-sh.op."
   Accedendo al sito con l'account admin, possiamo visualizzare tutte le informazioni private dello user come:
    a. Lo storico degli acquisti.
    b. L'indirizzo di casa dello user, salvato in chiaro: `0815 Test Street, Test, Test, 4711`.
    c. Lo stato del portafoglio dello user riuscendo a cambiarlo utilizzando le 2 carte di pagamento impostate come predefinite.
    d. Possiamo inoltre scrivere recensioni e fare acquisti con i soldi dello user.

2. Una volta entrati nell'account dell'admin, abbiamo ottenuto i permessi per accedere al percorso `http://172.17.0.2:3000/administration`. Grazie alle mail trovate, siamo riusciti tramite una SQL injection ad entrare su ogni account potendo visualizzare tutte le loro informazioni personali, come quelle riportate precedentemente (nel punto n.1) per l'account dell'admin.

3. La password dell'amministratore è stata identificata come "admin123" utilizzando un attacco di forza bruta, un metodo che consiste nel tentativo sistematico di tutte le possibili combinazioni alfanumeriche fino al raggiungimento della corretta corrispondenza.

Post Exploitation

- **Azioni Compiute Dopo l'Accesso Iniziale:**
    a. Visualizzazione e modifica delle informazioni degli utenti, inclusi gli storici degli acquisti e gli indirizzi di casa.
    b. Modifica dello stato del portafoglio degli utenti e utilizzo delle carte di pagamento salvate.
    c. Scrittura di recensioni e acquisti con i fondi degli utenti.
    d. Cambio delle password degli utenti, rendendo gli account inaccessibili ai legittimi proprietari.
    e. Eseguire ulteriori attacchi di phishing utilizzando le mail trovate.
    f. Usare i dati degli utenti per furto di identità.

Strumenti Utilizzati

1. **BURP SUITE:**
    - **Motivo dell'Utilizzo**: Per intercettare e analizzare il traffico HTTP.
    - **Obiettivo della Scansione**: Monitorare le richieste tra client e server e identificare informazioni sensibili negli header HTTP.
    - **Spiegazione del Funzionamento**: Funziona come proxy, intercettando e permettendo la modifica delle richieste e delle risposte HTTP.
2. **NMAP:**
    - **Motivo dell'Utilizzo**: Per la scansione delle porte e dei servizi attivi.
    - **Obiettivo della Scansione**: Identificare la versione del web server e i servizi esposti.
    - **Spiegazione del Funzionamento**: Esegue una scansione delle porte specificate, rilevando i servizi attivi e le loro versioni.
3. **NIKTO:**
    - **Motivo dell'Utilizzo**: Per la scansione di vulnerabilità comuni nelle web application.
    - **Obiettivo della Scansione**: Identificare vulnerabilità conosciute e percorsi nascosti.

- **Spiegazione del Funzionamento**: Scansiona la web application utilizzando un database di vulnerabilità note, identificando potenziali minacce come file e directory sensibili, configurazioni errate e versioni software vulnerabili.

4. **JOOMSCAN:**
   - **Motivo dell'Utilizzo**: Per la scansione di vulnerabilità specifiche di Joomla.
   - **Obiettivo della Scansione**: Identificare vulnerabilità specifiche e percorsi sensibili.
   - **Spiegazione del Funzionamento**: Esegue una scansione del sito web per rilevare la presenza di Joomla e le sue vulnerabilità note.
5. **FFUF:**
   - **Motivo dell'Utilizzo**: Per l'analisi delle directory della web application.
   - **Obiettivo della Scansione**: Identificare directory nascoste e potenziali punti di ingresso.
   - **Spiegazione del Funzionamento**: Esegue una scansione delle directory del sito web utilizzando una lista di nomi comuni.

Analisi delle Vulnerabilità

1. **Vulnerabilità: Directory Accessibile /ftp e /security.txt.**
   - **Descrizione della Vulnerabilità**: Accesso non autorizzato alla directory /ftp e /security.txt contenente file di quarantena e informazioni sensibili.
   - **Riproducibilità**: Accedere all'URL ”http://172.17.0.2:3000/ftp” e “http://172.17.0.2:3000/security.txt”.
   - **Prova della Rilevazione**: Screenshot del contenuto della directory: **“Screenshot_Directory_ftp.png”** e **“Screenshot_Directory_SecurityTxt.png”**.
   - **Classificazione OWASP TOP 10**: A04 - Insecure design, si riferisce a un’inefficace progettazione di sicurezza che rende vulnerabile un sistema o un'applicazione.
   - **Requisiti dell'Attaccante**: Accesso al network e conoscenza dell'URL.
   - **Gravità e Impatti**: Alto. Possibile accesso a file sensibili e malware.
   - **Score CVSS**: 7.5, riflette l'elevata gravità della vulnerabilità, dovuta alla possibilità di accesso non autorizzato a informazioni sensibili e file tramite la rete senza richiedere privilegi.
   - **Fix del Codice**: Implementare una configurazione del server web che neghi l'accesso pubblico a questa directory, consentendolo solo a utenti autorizzati.

2. **Vulnerabilità: takeover.**
    - **Descrizione della Vulnerabilità**: Il link rotto rappresenta una vulnerabilità in quanto punta ad una pagina inesistente.
    - **Riproducibilità**: Accedere all'URL ″http://172.17.0.2:3000/assets/″
    - **Prova della Rilevazione**: Screenshot della pagina **“Screenshot_Assets.png”**.
    - **Classificazione OWASP TOP 10**: A06: Security Misconfiguration: Il link rotto può rientrare in questa categoria se dovuto a una cattiva configurazione del server o del sistema.
    - **Requisiti dell'Attaccante**: Accesso al server o al sistema di gestione DNS, conoscenza della vulnerabilità e dei metodi per sfruttarla, capacità di caricare contenuti sul server.
    - **Gravità e Impatti**: Alta. Un attaccante può sfruttare il link rotto per caricare file malevoli, condurre attacchi di phishing o distribuire malware, compromettendo la sicurezza degli utenti e l'integrità del sistema.
    - **Score CVSS**: 8.8, impatta notevolmente sulla riservatezza, integrità e disponibilità del sistema, e dei dati dell’utente stesso.
    - **Fix del Codice**: Rimuovere o correggere i link rotti nel codice sorgente. Implementare reindirizzamenti 404 personalizzati per gestire i link non validi. Monitorare regolarmente i link del sito web per rilevare e correggere link rotti. Assicurarsi che le risorse e le pagine siano correttamente configurate e disponibili. Utilizzare strumenti di gestione dei contenuti per monitorare e aggiornare automaticamente i link obsoleti o non validi.

3. **Vulnerabilità: SQL Injection**
    - **Descrizione della Vulnerabilità**: Possibilità di bypassare l'autenticazione utilizzando una SQL Injection.
    - **Riproducibilità**: Inserire “`admin' OR 1=1-`” nel campo "Email" e qualsiasi password.
    - **Prova della Rilevazione**: Screenshot del login di errore, e di quello riuscito come admin: **“Screenshot_SQL_Dati.png”** e **“Screenshot_SQL_Accesso.png”**.
    - **Classificazione OWASP TOP 10**: A01 - Injection: è la prima nella lista OWASP Top 10. Le vulnerabilità di injection si verificano quando dati non affidabili vengono inviati a un interprete come parte di un comando o query. Gli attaccanti sfruttano queste vulnerabilità per eseguire comandi non autorizzati o accedere a dati non consentiti.
    - **Requisiti dell'Attaccante**: Accesso al form di login.
    - **Gravità e Impatti**: Critico. Accesso completo al database e alle informazioni degli utenti.

- **CVSS:** ha ottenuto un punteggio di 9,8 poiché si tratta di una vulnerabilità estremamente critica, che consente all'attaccante di accedere all'intero database e visualizzare tutti i dati sensibili degli utenti attaccati.
- **Fix del Codice**:Implementare l'uso di query SQL preparate con parametri di bind (prepared statements) è essenziale per gestire in modo sicuro gli input degli utenti. Questo approccio impedisce l'esecuzione di codice SQL arbitrario, migliorando significativamente la sicurezza del sistema. È inoltre consigliabile utilizzare tecniche di sanitizzazione e validazione degli input SQL per rafforzare ulteriormente le difese contro potenziali vulnerabilità.

4. **Vulnerabilità: possibilità di impostare password troppo semplici.**
   - **Descrizione della Vulnerabilità**: Questa vulnerabilità riguarda la possibilità per gli utenti di impostare password troppo semplici che potrebbero essere facilmente indovinate o forzate da attaccanti malintenzionati. Le password deboli aumentano il rischio di accessi non autorizzati e compromettono la sicurezza complessiva del sistema.
   - **Riproducibilità**: Accedere all'interfaccia di modifica della password, inserire una password semplice, come "password" o "123456" e verificare se il sistema accetta la password senza notificare alcun errore.
   - **Prova della Rilevazione**: login nell’account.
   - **Classificazione OWASP TOP 10**: A02: Password Cracking - se la password è troppo semplice e facilmente indovinabile.
   - **Requisiti dell'Attaccante**: Un attaccante avrebbe bisogno di conoscere il processo di registrazione o di modifica della password e potrebbe utilizzare tecniche di forza bruta o dizionario per indovinare o forzare una password debole.
   - **Gravità e Impatti**: La gravità di questa vulnerabilità è alta, poiché le password deboli possono compromettere la sicurezza dell'account utente e del sistema nel suo complesso. Gli impatti potenziali includono: Accesso non autorizzato agli account degli utenti, possibile compromissione dei dati sensibili, minaccia alla reputazione dell'organizzazione.
   - **CVSS:** il punteggio di 9.8 è stato attribuito a questa vulnerabilità a causa della sua gravità estrema, poiché consente a un attaccante di ottenere accesso non autorizzato agli account degli utenti mediante password deboli, potenzialmente compromettendo dati sensibili e causando danni significativi al sistema.
   - **Fix del Codice:** Per correggere questa vulnerabilità, è consigliabile implementare le seguenti misure:

a. Politiche di password robuste: Imporre criteri per le password che includano lunghezza minima, caratteri speciali, lettere maiuscole e numeri.
b. Validazione lato client e server: Verificare la complessità delle password sia lato client che server per evitare l'accettazione di password troppo semplici.
c. Educazione degli utenti: Informare gli utenti sull'importanza di scegliere password sicure e promuovere l'uso di password manager per gestire password complesse.

Implementare queste correzioni aiuterà a mitigare il rischio associato alla vulnerabilità delle password troppo semplici e migliorare complessivamente la sicurezza del sistema.

Conclusioni

Il penetration test ha rivelato diverse vulnerabilità critiche nella web application Juice Shop, evidenziando la necessità di miglioramenti significativi nella sicurezza.
Sono state fornite raccomandazioni specifiche per la mitigazione delle vulnerabilità identificate.

”