UNIVERSITÀ DEGLI STUDI DI SALERNO

Penetration Testing Report

FOXHOLE1.0.1.

Carmine Citro | Corso di PTEH | A.A. 2023/2024

UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA
DIPARTIMENTO DI ECCELLENZA

Sommario

1 Executive Summary

Nell'ambito del corso su **Penetration Testing and Ethical Hacking**, è stata svolta un'attività progettuale che prevedeva l'esecuzione di un penetration test sulla macchina target **FOXHOLE1.0.1.** Lo scopo di questa attività era analizzare la sicurezza della macchina target e identificare eventuali misure correttive necessarie per le vulnerabilità individuate.

Abbiamo adottato un approccio di tipo *Black Box*, poiché non erano disponibili informazioni rilevanti sulla configurazione della macchina target e sulla sua rete di supporto. Il test è stato condotto dalla stessa rete locale della macchina sotto esame, simulando il comportamento di un attaccante con accesso a tale rete.

Le vulnerabilità individuate potrebbero consentire a un potenziale aggressore di ottenere il controllo completo della macchina, con conseguenti gravi danni al sistema e agli utenti che utilizzano i servizi erogati da essa. Questo potrebbe compromettere i principi fondamentali della sicurezza informatica, ossia *disponibilità*, *integrità* e *confidenzialità* (Triade C.I.A.).

Di conseguenza, possiamo affermare che il livello di sicurezza della macchina è valutato come **BASSO**, mentre il rischio di compromissione è considerato **ALTO**. È quindi necessario intervenire per modificare il sistema ed eliminare le vulnerabilità individuate, al fine di riportare il rischio a livelli accettabili.

Tutte le vulnerabilità identificate, insieme alle relative misure correttive, verranno elencate e descritte dettagliatamente nelle sezioni successive di questo documento.

2 Engagement Highlights

Le regole di ingaggio, in questo caso, non sono state contestualizzate facendo questo parte di un attività progettuale di tipo accademica e quindi non soggetta ad accordi di non divulgazione (**NDA**). In particolare non sono stati presentati limiti riguardo agli strumenti e alle tecniche consentite a patto di non sconfinare la rete **NAT** creata appositamente per l'analisi di questa macchina. Di seguito riportate le sezioni delle regole d'ingaggio comuni. Usualmente tutto ciò che non viene definito non è consentito, in questo caso non vengono apposti limiti e l'analista ha dichiarato le tecniche e gli strumenti nell'apposito documento nella sezione **Strumenti utilizzati**.

2.1 Accordo di riservatezza

Non è stato stipulato alcun accordo di riservatezza con il docente del corso o con l'istituzione universitaria di Salerno, pertanto l'analisi può essere divulgata liberamente senza alcun obbligo contrattuale di non divulgazione.

2.2 Tempo stimato di consegna

È stato previsto un periodo di 30 giorni lavorativi per completare l'analisi e redigere i documenti relativi agli strumenti, alle metodologie utilizzate e alla relazione di *Penetration Testing*.

2.3 Tecniche e strumenti ammessi

Non ci sono restrizioni riguardanti gli strumenti e le tecniche da utilizzare nell'analisi, quindi agli studenti è stata concessa piena libertà di scelta. Tuttavia, sono state definite responsabilità legali nel caso in cui l'analisi possa compromettere macchine o servizi esterni alla macchina target o alla rete **NAT** designata per l'analisi.

2.4 Ambito dell'analisi

L'analisi si concentra esclusivamente sulla macchina target, pertanto non è consentito raccogliere informazioni tramite tecniche di **Intelligence**: Human Intelligence e Signal Intelligence. L'analisi non deve coinvolgere terze parti, come ad esempio il creatore della macchina.

2.5 Procedure di analisi

Durante l'analisi, le vulnerabilità gravi non verranno segnalate. Queste informazioni saranno divulgate solo al termine dell'analisi, poiché si tratta di un esercizio didattico condotto su una macchina volutamente vulnerabile. Inoltre, poiché la macchina non offre servizi accessibili pubblicamente, non è necessario segnalare tempestivamente le vulnerabilità più gravi."

3 Vulnerability Report

Da un'analisi della macchina sono emerse alcune vulnerabilità che la espongono ad attacchi da parte di utenti maliziosi:

- All'interno di file di tipo immagine sono presenti credenziali d'accesso nascoste mediante tecniche di steganografia. È possibile estrapolare il contenuto non cifrato o con scarsa cifratura dall'immagine. Queste stringhe pseudo-cifrate non sono protette da password, quindi utilizzando un semplice tool è possibile prelevare l'informazione, fornendo così le credenziali di un utente del sistema. Tali credenziali consentono l'accesso tramite OpenSSH come utente *fox*, da cui è possibile accedere ai file presenti sul sistema.

- Il web server permette la navigazione e la visualizzazione di file che non dovrebbero essere accessibili a tutti gli utenti perché non fanno parte della parte funzionale del sito web, ma spesso sono file di configurazione o simili utili al funzionamento.

- All'interno della macchina è presente un file eseguibile chiamato *GiveMeRootPlz* che ha il bit SUID attivato, il che significa che possiede privilegi elevati. Tuttavia, questo eseguibile non effettua alcun controllo sull'input che riceve e utilizza una funzione ormai deprecata e vulnerabile, permettendo l'inserimento di dati più grandi di quanto il programma si aspetti. Di conseguenza, è possibile sfruttare una vulnerabilità di buffer overflow e, approfittando del bit SUID, ottenere privilegi di root.

4 Remediation Report

Date le problematiche di sicurezza riscontrate durante l'attività di penetration testing dovrebbero essere messe in atto le seguenti strategie al fine di migliorare la sicurezza del sistema:

- Eliminare tutti i dati sensibili non cifrati presenti nei file.
- Gestire l'accesso ai file e le directory relativi al web server e autorizzare la navigazione ai file agli utenti solo nel caso in cui la progettazione del web server prevede che un utente possa visualizzare determinati tipi di file.
- Non utilizzare tecniche di occultamento delle informazioni all'interno di altri file.
- Se necessario utilizzare tecniche di steganografia, utilizzare password per la lettura del contenuto molto robuste, in modo che un attacco a dizionario/crittoanalisi non abbia successo. Quindi utilizzare password lunghe più di 16 caratteri, alfanumeriche, con caratteri speciali e non utilizzare passowrd con nomi comuni o di breve lunghezza.
- Effettuare un Security Audit in modo frequente e programmato
- Programmare un Penetration testing, come quello effettuato, fissando un numero di volte per il quale effettuarlo durante l'anno.

Si consiglia di risolvere tutte le vulnerabilità presentate in questo documento seguendo un ordine decrescente in base alla gravità: è consigliato dunque risolvere tempestivamente le vulnerabilità critiche e procedere successivamente alla correzione delle vulnerabilità con criticità più bassa.

5 Findings Summary

La tabella seguente mostra il numero di vulnerabilità individuate per categoria:

Severity	Info	Low	Medium	High	Critical
# Vulnerabilità	25	2	2	2	3

Il grafico seguente mostra la distribuzione delle vulnerabilità per categoria:

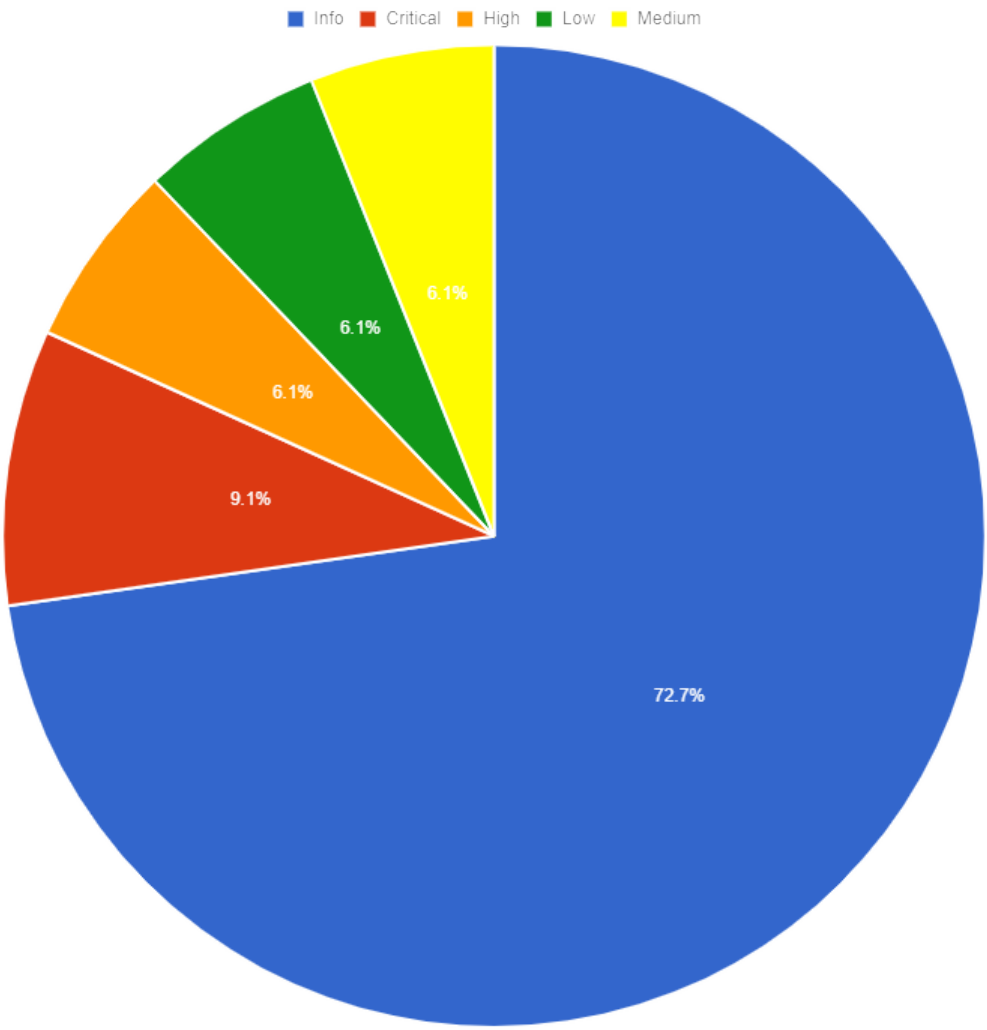

Figure 1: Grafico a torta delle vulnerabilità

6 Detailed Summary

Di seguito verranno riportate e descritte dettagliatamente le vulnerabilità individuate, partendo da quelle più critiche fino a quelle meno critiche. Inoltre per ognuna di essa verranno fornite alcune raccomandazioni su come mitigarle.

Da un'analisi automatica utilizzando strumenti *Nessus* e *Openvas* e *OWASP ZAP* sono state

riscontrate delle vulnerabilità, di seguito riportate le più rilevanti:

- Mancanza dell'header **Content Security Policy (CSP)** che riduce i rischi dovuti ad attacchi di tipo XSS.
- Mancanza di header Anti-clickjacking che può comportare attacchi inducendo un utente a eseguire altre funzionalità in modo da dirottare l'utente su altre pagine web.
- La mancanza dell'header **X-Content-Type-Option** che può portare a problemi di sicurezza come attacchi XSS, se l'header non è presente il browser tenta di capire da solo qual è il tipo del contenuto della risposta e come gestirla
- Possibilità di lasciar trasparire informazioni riguardo la versione tramite l'header delle risposte HTTP.

Le vulnerabilità evidenziate non hanno un impatto rilevante in questa fase di penetration testing ma sono quelle più rilevanti dal punto di vista degli strumenti di scansione, in particolare di **OWASP ZAP**.

6.1 Critical

Credenziali presenti all'interno di immagini in chiaro senza utilizzo di password

Descrizione

All'interno di un immagine navigabile dal web server sono presenti le credenziali dell'utente di sistema

Comando utilizzato: *steghide –extract -sf Pictures/foxy1.jpeg*

Impatto

Sfruttando questa password è possibile accedere alla macchina target tramite ssh.

Soluzione

Rimuovere la password in chiaro dal file, eliminare il file stesso oppure mitigare le vulnerabilità di *full path disclosure, directory listing* e *information leakage*. In alternativa, applicare un hash alla password per garantire una crittografia robusta.

6.2 Critical

Eseguibile con privilegi elevati

Descrizione

Presenza dell'eseguibile **GiveMeRootPlz** con privilegi elevati, ovvero con il bit setuid attivato.

Impatto

Lo sfruttamento di questo eseguibile tramite buffer overflow permette di ottenere una shell di root e quindi l'ottenimento dei privilegi di root.

Soluzione

Disattivazione del bit set user (SUID) tramite il comando sudo chmod u-s GiveMeRoot-Plz

6.3 Critical

Funzioni deprecate e mancato controllo di un input

Descrizione

All'interno dell'eseguibile GiveMeRootPlz viene utilizzata una funzione deprecata gets(), la quale è obsoleta poiché accetta un numero di byte superiore a quello definito per la variabile. Inoltre, il programma non effettua alcun controllo sull'input, pertanto è possibile inserire altri caratteri non consentiti. Ad esempio, nel nostro caso, una sequenza di caratteri trash concatenati all'indirizzo di memoria di una funzione (0xad 0x62 0x55 0x56) per effettuare il buffer overflow.

Impatto

Sfruttando questa vulnerabilità si riescono ad ottenere i privilegi di root, quindi si riesce ad effettuare una privilege escalation.

Soluzione

Sostituire la funzione gets() con una funzione che effettui un controllo sulla dimensione dell'input e controllare i caratteri in ingresso. Di seguito è riportato un esempio di funzione con l'implementazione delle mitigazioni:

```c
void funzioneSicura() {
    char buffer[100];

    // Leggere l'input dall'utente
    if (fgets(buffer, sizeof(buffer), stdin) != NULL) {
        // Rimuovere il carattere di newline finale se presente
        buffer[strcspn(buffer, "\n")] = '\0';

        // Definire una regex per accettare solo caratteri alfabetici e numerici
        regex_t regex;
        int risultato;

        // Compilare la regex per accettare solo stringhe alfabetiche e numeriche
        risultato = regcomp(&regex, "^[a-zA-Z0-9]+$", REG_EXTENDED);
        if (risultato) {  // Se risultato è diverso da 0, la compilazione della regex è fallita
            fprintf(stderr, "Non è stato possibile compilare la regex\n");
            return;
        }

        // Verificare se l'input rispetta la regex
        risultato = regexec(&regex, buffer, 0, NULL, 0);
        if (!risultato) {  // Se risultato è 0, l'input è valido
            printf("Input valido: %s\n", buffer);
        } else if (risultato == REG_NOMATCH) {  // Se risultato è REG_NOMATCH, l'input non è valido
            printf("Input non valido: contiene caratteri non consentiti\n");
        } else {  // Se risultato è un altro valore, si è verificato un errore nell'esecuzione della regex
            char msgbuf[100];
            regerror(risultato, &regex, msgbuf, sizeof(msgbuf));
            fprintf(stderr, "Errore nella verifica della regex: %s\n", msgbuf);
        }

        // Liberare la memoria allocata per la regex
        regfree(&regex);
    } else {
        printf("Errore nella lettura dell'input.\n");
    }

```

Figure 2: Mitigazioni

La funzione fgets legge l'input da stdin in modo sicuro, evitando il rischio di buffer overflow. Questo sostituisce l'uso della funzione deprecata e pericolosa gets. La funzione funzioneSicura utilizza una regex per accettare solo stringhe composte da caratteri alfabetici e numerici. Questo assicura che l'input contenga solo caratteri sicuri e previene l'inserimento di caratteri speciali pericolosi. Nel caso in cui la regex non riesca a compilarsi o a trovare una corrispondenza nell'input, vengono stampati messaggi di errore appropriati. Ciò aiuta a identificare e gestire eventuali problemi con l'input fornito dall'utente.

6.4 High

Limitazione non corretta di un percorso a un file o una directory limitata.
Path Traversal

Descrizione

Il web server espone file e directory a utenti non esplicitamente autorizzati ad accedervi ma non prevede protezioni .

Impatto

Sfruttando questa debolezza si può accedere a file e directory del web server e, in questo caso, trovare alcune credenziali di sistema.

Soluzione

Convalidare gli input, quando si convalidano nomi di file o percorsi, utilizzare liste consentite e, se possibile, non consentire mai la sequenza ".." per evitare **Relative Path Traversal** ed escludere "/" per evitare **Absolute Path Traversal**

6.5 High

Esposizione di informazioni attraverso **Directory Listing**

Descrizione

Il directory listing fornisce a un utente malintenzionato l'indice completo di tutte le risorse che si trovano all'interno della directory. I rischi e le conseguenze specifici variano a seconda dei file elencati e accessibili

Impatto

Sfruttando questa debolezza si può accedere a file e directory del web server e a seconda dei file trovati è più o meno grave l'impatto.

Soluzione

Le raccomandazioni includono la limitazione dell'accesso a directory o file importanti, adottando un requisito di necessità di conoscenza sia per il documento che per la radice del server, e la disattivazione di funzioni come l'elenco automatico delle directory che potrebbero esporre file privati e fornire informazioni che potrebbero essere utilizzate da un aggressore quando formula o conduce un attacco.

6.6 Medium

Content Security Policy (CSP) Header Not Set

Descrizione

Questa problematica si verifica quando un server web non imposta l'header HTTP Content-Security-Policy.

Impatto

Quando l'header CSP non è impostato, le pagine web sono vulnerabili a: Cross-Site Scripting (XSS): Gli attaccanti possono iniettare script dannosi, rubando informazioni sensibili come cookie di sessione. Clickjacking: Gli utenti possono essere ingannati a cliccare su elementi nascosti, portando a operazioni indesiderate. Iniezioni di contenuto: Gli attaccanti possono caricare contenuti non autorizzati, compromettendo l'integrità del sito.

Soluzione

La soluzione consiste nel definire una politica CSP appropriata, configurare il server web per includere questa politica nelle risposte HTTP e testare attentamente il sito per garantire la corretta applicazione della politica senza compromettere la funzionalità.

6.7 Medium

Missing Anti-clickjacking Header

Descrizione

header X-Frame-Options è utilizzato in HTTP per indicare al web browser se consentire il caricamento di contenuti in frame, iframe, embed o object

Impatto

La mancanza dell'header X-Frame-Options nella risposta HTTP del web server espone la macchina ad attacchi di tipo "ClickJacking", ovvero la creazione di un overlay sulle pagine dell'applicativo web con lo scopo di dirottare gli utenti del sistema su altri siti e di rubare informazioni.

Soluzione

I browser Web moderni supportano le intestazioni HTTP Content-Security-Policy e X-Frame-Options. È necessario che uno di essi sia impostato su tutte le pagine Web restituite dalla web app.

6.8 Low

TCP timestamps

Descrizione

L'host remoto implementa i timestamp TCP e quindi consente di calcolare il tempo di attività.

Impatto

Un effetto collaterale di questa funzionalità è che il tempo di attività dell'host remoto può talvolta essere calcolato.

Soluzione

Disabilitare i TCP timestamps all'interno del sistema: aggiungere la riga "net.ipv4.tcp_timestamps=0" in /etc/sysctl.conf. Lanciare successivamente il comando sysctl -p

6.9 Low

ICMP Timestamp Reply Information Disclosure

Descrizione

Il Timestamp Reply è un messaggio ICMP che risponde a un messaggio di Timestamp. È composto dal timestamp di origine inviato dal mittente del Timestamp, nonché da un timestamp di ricezione e da un timestamp di trasmissione.

Impatto

Queste informazioni potrebbero teoricamente essere utilizzate per sfruttare i generatori di numeri casuali deboli basati sul tempo in altri servizi.

Soluzione

Disattivare completamente il supporto per il timestamp ICMP sull'host remoto. Proteggere l'host remoto con un firewall e bloccare i pacchetti ICMP che passano attraverso il firewall in entrambe le direzioni (completamente o solo per le reti non attendibili).

6.10 Informative

Le vulnerabilità informative sono quelle individuate in maggioranza dagli strumenti di scansione automatica delle vulnerabilità, tuttavia non sono state riportate dato che non hanno rilevanza nei confronti di potenziali attaccanti.