UNIVERSITÀ DEGLI STUDI DI CATANIA

Dipartimento di Matematica e Informatica

Corso di Laurea Triennale in Informatica

Sophia Grazia Catalano

Security Logging and Monitoring Failures

Relazione Internet Security

Anno Accademico 2023 - 2024

Indice

Capitolo 1

Introduzione

Nell'era digitale odierna, la sicurezza informatica è una priorità fondamentale per le organizzazioni di ogni settore. Le minacce informatiche sono in costante evoluzione e diventano sempre più sofisticate, rendendo indispensabile l'adozione di misure di sicurezza proattive e reattive. Due dei **pilastri fondamentali** della sicurezza informatica sono il logging e il monitoraggio della sicurezza. Questi processi giocano un ruolo cruciale nel proteggere le risorse informatiche, prevenire violazioni di dati e garantire la conformità alle normative vigenti.
Il **logging** consiste nella registrazione sistematica degli eventi che si verificano all'interno di un sistema informatico. Questa attività permette di creare un archivio dettagliato delle operazioni eseguite da utenti e applicazioni, fornendo una traccia storica indispensabile per l'analisi forense, la risoluzione dei problemi e la verifica della conformità.
Il **monitoraggio della sicurezza**, d'altra parte, è il processo continuo di osservazione e analisi di tale log. Questo approccio proattivo consente di identificare tempestivamente attività sospette, prevenire potenziali attacchi e rispondere efficacemente agli incidenti di sicurezza. Strumenti avanzati come i sistemi di rilevamento delle intrusioni (IDS), i sistemi di prevenzione delle intrusioni (*IPS*) e le piattaforme di gestione delle informazioni e degli eventi di sicurezza (*SIEM*) sono fondamentali per un monitoraggio efficace.
Tuttavia, nonostante l'importanza del logging e del monitoraggio, esistono numerose sfide e **potenziali fallimenti** associati alla loro implementazione e gestione.

Capitolo 2

A09:2021

Il progetto **Open Web Application Security Project** (*OWASP*) è un'iniziativa no-profit nata nel 2001 con l'obiettivo di fornire linee guida e strumenti pratici per lo sviluppo sicuro di applicazioni web. I principi e le best practices definiti da OWASP sono riconosciuti a livello internazionale e vengono utilizzati per prevenire vulnerabilità e migliorare la sicurezza delle applicazioni web. Sebbene queste indicazioni siano facoltative, molti contesti stanno lavorando per renderle obbligatorie nello sviluppo web.
Una delle risorse più conosciute di OWASP è l'**OWASP Top 10**, una lista periodicamente aggiornata che elenca i rischi più critici per la sicurezza delle applicazioni web. La versione 2021 di questa lista include una problematica cruciale, nota appunto come **"A09:2021 – Security Logging and Monitoring Failures"**. Questa problematica era già presente nella versione 2017 come **"A10:2017 – Insufficient Logging and Monitoring"**.

2.1 Definizione

I **fallimenti nel logging e nel monitoraggio** della sicurezza si riferiscono all'incapacità di registrare e monitorare adeguatamente le attività all'interno di un sistema informatico. Questo può includere l'assenza di log sufficientemente dettagliati, la mancata analisi dei log generati, o l'incapacità di rispondere tempestivamente agli eventi di sicurezza. Questi fallimenti possono avere gravi conseguenze, poiché impediscono la rilevazione e la risposta efficace agli attacchi informatici, lasciando le applicazioni vulnerabili e aumentando il rischio di danni significativi.

2.2 Errori comuni

Gli errori di registrazione e monitoraggio della sicurezza non rappresentano **vulnerabilità dirette** che possono essere sfruttate in modo immediato, ma piuttosto un punto critico che può esporre il sistema o l'applicazione a rischi significativi di compromissione della sicurezza. Questi errori possono permettere agli aggressori di operare inosservati, ottenendo accessi non autorizzati senza essere rilevati. Di seguito vengono descritti alcuni dei principali errori di logging e monitoraggio.

- **Mancata registrazione di eventi di sicurezza importanti**. La mancata registrazione di tentativi di accesso non riusciti, modifiche ai privilegi utente, o altri eventi critici di sicurezza può lasciare i sistemi vulnerabili. Senza questi log, è impossibile rilevare e rispondere tempestivamente a tentativi di intrusione o abusi di accesso.

- **Mancato monitoraggio dei registri** Anche quando gli eventi di sicurezza vengono registrati, è essenziale monitorare attivamente questi log. Ad esempio, ripetuti tentativi di accesso non riusciti possono indicare un tentativo di attacco brute force. Senza una revisione continua, le anomalie e i comportamenti sospetti possono rimanere inosservati, aumentando il rischio di danni estesi.

- **Non archiviare i registri per un periodo sufficientemente lungo**. Conservare i log per un periodo insufficiente può ostacolare le indagini post-incidente. Molti attacchi non vengono scoperti immediatamente, e la mancanza di log storici può impedire la ricostruzione degli eventi e l'identificazione delle vulnerabilità sfruttate.

- **Sistemi di registrazione e monitoraggio non Sicuri**. Se i sistemi che gestiscono i log non sono adeguatamente protetti, gli aggressori possono accedere e modificare i log stessi, cancellando tracce delle loro attività o inserendo informazioni fuorvianti. È fondamentale assicurarsi che i sistemi di logging siano sicuri e resistenti alle manipolazioni.

2.3 Progettazione sicura di sistemi di logging

Per **mitigare i rischi** appena visti è essenziale progettare soluzioni di logging che siano non solo efficienti ma anche sicure. Questo implica l'adozione di pratiche che garantiscano l'integrità, la confidenzialità e la disponibilità dei log, rendendo difficile per gli aggressori alterare o manipolare le informazioni registrate.

- **Codifica e convalida dei dati**. È fondamentale codificare e convalidare tutti i dati prima di registrarli. Questo implica la sanitizzazione dei dati di input e l'escape dei caratteri speciali che potrebbero essere utilizzati per manipolare i log.
- **Non registraziore informazioni aensibili**. Evitare di registrare informazioni sensibili come password, ID di sessione, numeri di carte di credito e altre informazioni riservate. Queste informazioni non devono essere conservate nei log per proteggere la privacy degli utenti e per ridurre il rischio di esposizione in caso di violazione dei dati.
- **Protezione dell'integrità del registro**. Proteggere l'integrità dei log è cruciale per prevenire manipolazioni da parte di utenti malevoli che potrebbero tentare di alterare o cancellare le loro tracce.
- **Archiviazione sicura dei log**. Assicurarsi che i log siano archiviati in un ambiente sicuro e resiliente.
- **Monitoraggio continuo**. Implementare un sistema di monitoraggio continuo per rilevare eventuali anomalie o tentativi di manipolazione dei log.

Capitolo 3

CWE

Una **CWE** (*Common Weakness Enumeration*) è una categorizzazione standardizzata delle debolezze di sicurezza dei software. Gestita e mantenuta dalla **MITRE Corporation**, la CWE fornisce un linguaggio comune per identificare, discutere e affrontare le vulnerabilità e le debolezze nel codice sorgente dei programmi, nelle configurazioni dei sistemi e nelle pratiche di sviluppo. Di seguito vengono riportate tutte le CWE correlate al problema della sicurezza e monitoraggio dei log. Tutte le weakness che verranno viste si riferiscono ad una progettazione errata relativa ad una tattica di sicurezza architetturale.

3.1 CWE-117: Improper Output Neutralization for Logs

"*Il prodotto non neutralizza o neutralizza in modo errato l'output scritto nei registri.*"
Tra le varie CWE, la presente è quella che verrà principalmente utilizzata per la dimostrazione dell'attacco. Questa specifica debolezza si riferisce alla mancata neutralizzazione corretta degli output nei log, che potrebbe permettere ad un attaccante di falsificare le voci di registro o inserire contenuti dannosi nei log, attacco altresì noto come "**log forging**" o "**log injection**".

3.1.1 Log Forging

Il log forging è apputno una tecnica utilizzata dagli attaccanti per **manipolare** i file di log di un sistema. Quando un'applicazione registra eventi o informazioni senza neutralizzare adeguatamente i dati in ingresso, un attaccante può iniettare caratteri o stringhe speciali che alterano l'integrità del registro. Le **conseguenze** di questa vulnerabilità possono essere gravi, in quanto posso essere compromessi vari aspetti della sicurezza del sistema.

- **Integrità**. Un attaccante può modificare/aggiungere voci false nei log, facendo apparire attività che non sono mai avvenute o modificando i dettagli di eventi esistenti.
- **Riservatezza**. Se i log contengono informazioni sensibili e un attaccante riesce a inserirsi nei registri, potrebbe accedere a dati che non dovrebbe visualizzare, come credenziali, numeri di carte di credito o altre informazioni private.
- **Disponibilità**. L'iniezione di caratteri o stringhe malformate può corrompere i file di log, rendendoli inutilizzabili e impedendo l'analisi forense in caso di incidenti di sicurezza. Inoltre manipolando i log, un attaccante potrebbe provocare errori che portano al crash del sistema di logging, compromettendo la disponibilità dei servizi dipendenti dai log.
- **Non ripudio**. La possibilità di alterare i log compromette il principio del non ripudio, poiché un attaccante potrebbe rimuovere o modificare le tracce delle proprie azioni, rendendo difficile provare che un'azione è stata compiuta da un determinato utente.

3.2 CWE-223: Omission of Security-relevant Information

"*Il prodotto non registra nè visualizza informazioni che sarebbero importanti per identificare la natura di un attacco o per determinare se un'azione è sicura.*"
La CWE-223 si riferisce alla mancanza di inclusione di informazioni cruciali per la sicurezza nei log, il che potrebbe compromettere la capacità di rilevare e quindi rispondere a eventuali incidenti di sicurezza. Tale vulnerabilità compromette un importante aspetto della sicurezza, quale il **Non ripudio**, poiché un eventuale attacco sarà difficile o impossibile da determinare. Ciò può consentire che gli attacchi al sistema continuino senza alcun preavviso.

3.3 CWE-532: Insertion of Sensitive Information into Log File

"*Le informazioni scritte nei file di registro possono essere di natura riservata e fornire indicazioni preziose a un utente malintenzionato o esporre informazioni sensibili dell'utente.*"
La CWE-532 si verifica quando un sistema registra informazioni sensibili nei file di log in modo non sicuro. Queste informazioni possono includere dati personali, credenziali di autenticazione, dettagli delle sessioni o altre informazioni che, se esposte, potrebbero compromettere la sicurezza e la privacy degli utenti. Quindi in questo caso l'aspetto della sicurezza che viene compromesso è la **Segretezza**.

3.4 CWE-778: Insufficient Logging

"*Quando si verifica un evento critico per la sicurezza, il prodotto non registra l'evento oppure omette dettagli importanti sull'evento durante la registrazione.*"
La CWE-778 si verifica quando eventi critici per la sicurezza non vengono registrati correttamente, come un tentativo di accesso fallito. Anche in questo caso, come per la CWE-223, viene compromesso il **Non ripudio**, in quanto non ci sarebbe alcuna traccia per l'analisi forense, e quindi non si potrebbe scoprire la causa dei problemi o l'origine degli attacchi.

Capitolo 4

Demo

4.1 Strumenti utilizzati

Per la dimostrazione di questo attacco sono stati utilizzati 2 strumenti:

- **Xampp**, software libero multipiattaforma utilizzato per creare un ambiente di web hosting locale sul proprio computer. Tale ambiente è particolarmente utile per scopi di testing nella programmazione web. XAMPP include alcuni componenti essenziali.:
  - **Apache HTTP Server**, web server utilizzato per servire le pagine web. È il componente principale e permette di ospitare siti web e applicazioni web in locale.
  - **MariaDB e SQLite**, ossia i database server inclusi.
  - **Perl e PHP**, linguaggi di programmazione, il primo è utilizzato per lo sviluppo di applicazioni web dinamiche, mentre il secondo viene solitamente utilizzato per script di automazione e amministrazione di sistema.
- **Burp Suite**, strumento molto potente che viene principalmente utilizzato per testare la sicurezza delle applicazioni web. Si tratta di una suite che include un gran numero di strumenti, ma lo strumento specifico che verrà utilizzato sarà il componente proxy.
  Burp Proxy funziona come un server proxy web tra il browser e le applicazioni di destinazione, consentendo di intercettare, ispezionare e modificare il traffico che passa in entrambe le direzioni. Funziona attraverso l'utilizzo del browser di Burp, e tutto il traffico che avverrà su tale browser verrà automaticamente inoltrato a Burp.

Attraverso la combinazione di questi due strumenti non solo è possibile identificare vulnerabilità, ma anche sperimentare soluzioni e mitigazioni in un ambiente controllato. Si possono implementare patch e aggiornamenti, testare la loro efficacia, e assicurarsi che le misure di sicurezza adottate siano sufficientemente robuste contro possibili attacchi.

4.2 Scenario

Consideriamo il seguente scenario: l'applicazione Web, implementata dalla sattoscritta, registra i tentativi di accesso non riusciti da parte degli utenti. Di seguito è riportato il codice utilizzato dall'applicazione:

```
1 if ($_SERVER["REQUEST_METHOD"] == "POST") {
2     $nome_utente = $_POST["nome_utente"];
3     $password = $_POST['password'];
4 
5     $login_sql = "SELECT * FROM utenti WHERE nome_utente = '$nome_utente'";
6     $login_result = $conn->query($login_sql);
7 
8     if ($login_result->num_rows > 0) {
9         $row = $login_result->fetch_assoc();
10        if (password_verify($password, $row['password'])) {
11            $_SESSION['logged_in'] = true;
12            $_SESSION["nome_utente"] = $nome_utente;
13            logMessage("INFO", "Accesso riuscito", $nome_utente);
14            header("Location: forum.php");
15            exit;
16        }else {
17            logMessage("ERROR", "Accesso negato", $nome_utente);
18            echo "Credenziali errate.<a href='login.php'>Riprova</a> oppure <a
                  href='register.php'>Registrati</a>.";
19        }
20    } else {
21        logMessage("ERROR", "Accesso negato", $nome_utente);
22        echo "Questo account non esiste.<a href='login.php'>Riprova</a> oppure
               <a href='register.php'>Registrati</a>.";
23    }
24 }
```

Per assicurare chiarezza e coerenza nel codice, viene sotto riportata anche l'implementazione della fuznione '*logMessage*', la quale registra i messaggi di errore/informazione nel sistema di logging.

```php
function logMessage($level, $message, $nome_utente, $ip = null) {
    global $logFile;
    if ($ip === null) {
        $ip = getUserIP();
    }

    $date = date('Y-m-d H:i:s');
    $logEntry = "$date | $level | $message | User: $nome_utente | IP: $ip" .
        PHP_EOL;

    $fileHandle = fopen($logFile, 'a');

    if ($fileHandle) {
        fwrite($fileHandle, $logEntry);
        fclose($fileHandle);
    } else {
        echo "Errore: impossibile aprire il file.";
    }
}
```

Attraverso questa implementazione, l'aspetto del file di log risulterà essere il seguente:

```
2024-06-26 11:07:05 | INFO | Registrazione avvenuta con successo | User: sophi | IP: ::1
2024-06-26 11:07:08 | INFO | Logout effettuato | User: sophi | IP: ::1
2024-06-26 11:07:27 | INFO | Registrazione avvenuta con successo | User: jade | IP: ::1
2024-06-26 11:07:30 | INFO | Logout effettuato | User: jade | IP: ::1
2024-06-26 11:08:40 | INFO | Registrazione avvenuta con successo | User: pippo | IP: ::1
2024-06-26 11:08:41 | INFO | Logout effettuato | User: pippo | IP: ::1
2024-06-26 11:09:12 | INFO | Registrazione avvenuta con successo | User: mary | IP: ::1
2024-06-26 11:09:15 | INFO | Logout effettuato | User: mary | IP: ::1
2024-06-26 11:13:36 | INFO | Registrazione avvenuta con successo | User: chiara | IP: 127.0.0.1
2024-06-26 11:13:38 | INFO | Logout effettuato | User: chiara | IP: 127.0.0.1
2024-06-26 11:13:44 | INFO | Accesso riuscito | User: chiara | IP: 127.0.0.1
2024-06-26 11:13:53 | INFO | Account rimosso | User: chiara | IP: 127.0.0.1
2024-06-26 11:14:37 | ERROR | Accesso negato | User: sophi | IP: ::1
```

Figura 4.1: File di log

È stato inoltre implementato un **sistema di monitoraggio** che blocca l'IP di un utente che ha superato il limite di tentativi falliti, per quindi prevenire eventuali attacchi di Brute force. Il sistema si reimposta ogni qual volta si verifica un accesso riuscito prima di raggiungere il limite dei tentativi, in questo caso specifico si è scelto di impostare una soglia di 5 tentativi.

Di seguito è riportato il codice che analizza il file di log:

```
function analizzaLog($nuoveRighe, $tentativiMassimi) {
    static $tentativiFallitiPerIP = [];

    if (file_exists($GLOBALS['fileTentativi'])) {
        $tentativiFallitiPerIP = json_decode(file_get_contents($GLOBALS['
            fileTentativi']), true);
    }

    $ipDaBloccare = null;

    foreach ($nuoveRighe as $linea) {
        $ip = estraiIP($linea);

        if ($ip && strpos($linea, "Accesso negato") !== false) {
            if (!isset($tentativiFallitiPerIP[$ip])) {
                $tentativiFallitiPerIP[$ip] = 1;
            } else {
                $tentativiFallitiPerIP[$ip]++;
            }
            if ($tentativiFallitiPerIP[$ip] >= $tentativiMassimi) {
                bloccaIP($ip, $GLOBALS['htaccessFile']);
                $ipDaBloccare = $ip;
                unset($tentativiFallitiPerIP[$ip]);
                break;
            }
        } elseif ($ip && strpos($linea, "Accesso riuscito") !== false) {
            if (isset($tentativiFallitiPerIP[$ip])) {
                unset($tentativiFallitiPerIP[$ip]);
            }
        }
    }
    salvaTentativi($tentativiFallitiPerIP);
    return $ipDaBloccare;
}
```

4.3 Attacco

Prima di procedere con la dimostrazione dell'attacco, è importante notare che in HTML i campi di input come <input> o <label> non interpretano i caratteri speciali, quali `'\n'` o `'\r\n'`; infatti, quando il form viene inviato, i dati vengono trasmessi come stringhe di testo esattamente come sono ricevuti, e non come dei **caratteri di escape**. È necessario fare questa precisazione, poichè il ritorno a capo sarà fondamentale per il successo dell'attacco.

Per superare questa problematica verrà utilizzato un Proxy. Come è gia stato precisato, il Proxy di Burp Suite funziona attraverso l'utilizzo del browser di Burp, per questo la demo avverrà all'interno di tale browser. Una volta arrivati nella pagina di login dell'applicazione web presa in esame, e una volta inserite delle credenziali qualsiasi, occorrerà cliccare sul pulsante "*Intercept is off*" presente nell'interfaccia del software in questione. Ci troveremo davanti a tale schermata:

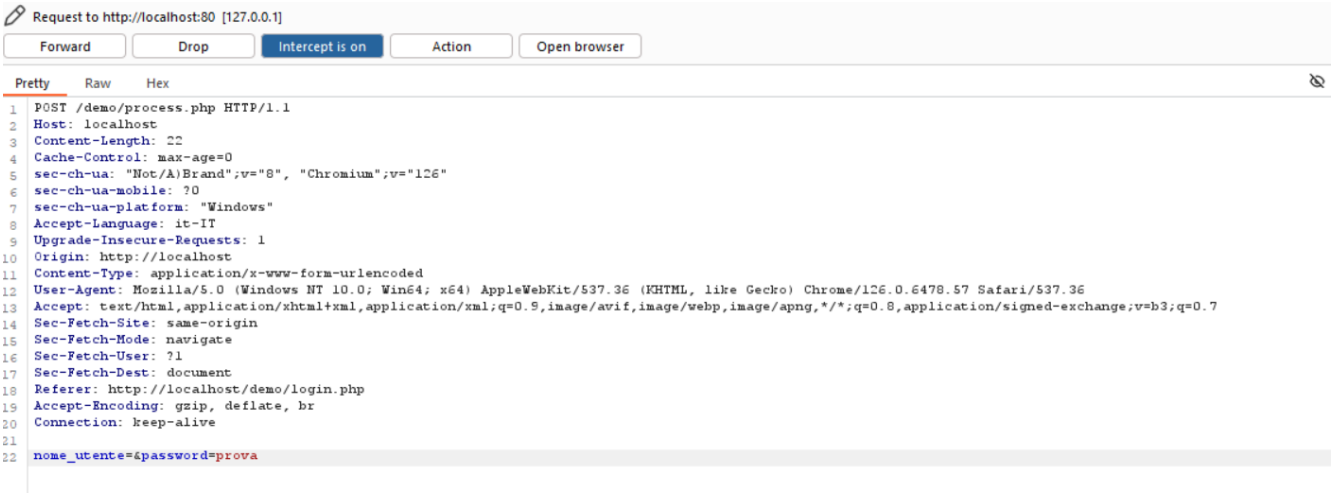

Figura 4.2: Burp Proxy

A questo punto, saremo in grado di inserire i caratteri di ritorno a capo direttamente dalla tastiera. Precisamente, andremo a digitare la seguente riga nella riga 12 all'interno campo "nome utente", quindi inserire una password desiderata e poi premere sul pulsante "*Forward*".

```
1 jade | IP: 127.0.0.1
2 2024-06-27 22:09:55 | INFO | Accesso riuscito | User: jade
```

Una volta iniettato tale input, ciò che apparirà all'interno del file di log sarà:

```
2024-06-27 22:07:28 | ERROR | Accesso negato | User: jade | IP: 127.0.0.1
2024-06-27 22:07:33 | ERROR | Accesso negato | User: jade | IP: 127.0.0.1
2024-06-27 22:07:37 | ERROR | Accesso negato | User: jade | IP: 127.0.0.1
2024-06-27 22:08:28 | ERROR | Accesso negato | User: jade | IP: 127.0.0.1
2024-06-27 22:09:55 | INFO | Accesso riuscito | User: jade | IP: 127.0.0.1
```

Figura 4.3: File di log 2

Grazie a questa tecnica, siamo riusciti a inserire una voce falsa nei log dell'applicazione. Tale manipolazione fa in modo che si azzeri il contatore dei tentativi di accesso falliti. Questo significa che un attaccante può condurre attacchi di brute force senza attivare gli allarmi di sicurezza. Questa dimostrazione rappresenta naturalmente un **fallimento critico** nel monitoraggio del registro di log.

4.4 Difesa

Ma cosa ha permesso la riuscita di quest'attacco? Essenzialmente sono due le vulnerabilità che sono state sfruttate: l'accesso al file di log e la mancata neutralizzazione dell'input.

4.4.1 Limitazione dell'accesso ai log

Inanzitutto l'attacco è stato possibile grazie a una grave vulnerabilità presente nell'applicazione web: la **visibilità pubblica dei log**. In particolare, i log dell'applicazione sono accessibili a tutti gli utenti tramite un URL, poiché il file dei log è situato all'interno della directory del sito web. Questa configurazione errata espone i log pubblicamente, consentendo a chiunque di visualizzarli semplicemente conoscendo l'URL specifico. Questa esposizione ci ha permesso di studiare il formato dei log e comprendere come manipolarli al meglio.

Per far fronte a tale problematica, è stata introdotta una configurazione all'interno del file '*.htaccess*'. Questo file è posizionato nella directory del sito web e contiene le seguenti istruzioni:

```
1 <Files "log.log">
2     Order deny,allow
3     Deny from All
4 </Files>
```

Tale configurazione blocca l'accesso diretto al file di log attraverso il web per tutti gli utenti, incluso il server stesso.

4.4.2 Neutralizzazione dell'input

La seconda vulnerabilità sfruttata è stata la mancata neutralizzazione dell'input dell'utente. Questo significa che l'applicazione web **non filtra o valida** adeguatamente i dati inseriti dagli utenti prima di registrarli nei log, rendendo possibile l'inserimento di informazioni dannose o manipolate.

Per mitigare questa vulnerabilità, è stata implementata una funzione nel codice del sito che si occupa di sanificare l'input dell'utente per renderlo sicuro e adatto all'inserimento nei file di log dell'applicazione. Questa funzione sarà chiamata ogni volta che un input utente dovrà essere inserito all'interno di un messaggio di log. Di seguito è riportato il codice:

```
1 function sanitizeForLog($input) {
2
3     $input = str_replace(["\r\n", "\r", "\n"], '\\n', $input);
4     $input = trim($input);
5     $input = preg_replace('/[^\w\s@.-\\\]/', '', $input);
6
7     return $input;
8 }
```

Questa funzione esegue tre operazioni cruciali:

- **Sostituzione delle sequenze di newline**: tutte le sequenze di newline vengono convertite in un'escape sequence `'\n'`. In questo modo l'utente non potrà inserire linee aggiuntive nei log.
- **Trimming dell'input**: ogni spazio vuoto e carattere di spaziatura all'inizio e alla fine dell'input viene rimosso.
- **Filtraggio**: i caratteri che non appartengono alla lista di caratteri accettati vengono rimossi.

Conclusione

In conclusione, la sicurezza del logging e del monitoraggio emerge come un pilastro cruciale nella difesa degli applicativi e dei sistemi informatici contro le crescenti minacce informatiche sofisticate. Come evidenziato in questa relazione, i fallimenti nella corretta registrazione e nel monitoraggio dei log possono compromettere gravemente la capacità di individuare e rispondere agli attacchi, esponendo le organizzazioni a rischi significativi di violazioni della sicurezza e perdite di dati sensibili.
È imperativo adottare le best practices, inclusa la registrazione accurata degli eventi critici, il monitoraggio proattivo dei log in tempo reale, e l'archiviazione sicura a lungo termine dei dati. .
Solo attraverso un approccio integrato e una vigilanza costante è possibile mitigare le vulnerabilità e proteggere in modo adeguato le risorse digitali dalle minacce sempre più sofisticate che caratterizzano il panorama della cybersecurity odierno. Proteggere e monitorare i log non è solo una pratica tecnica, ma un investimento cruciale per la sicurezza e la resilienza dei sistemi informatici contro le minacce in continua evoluzione presenti nella sicurezza informatica.

Bibliografia

In questa sezione sono presentate le fonti principali utilizzate come base per l'analisi e le considerazioni esposte nella presente relazione.

1. https://owasp.org/Top10/A09_2021-Security_Logging_and_Monitoring_Failures/.
2. https://cwe.mitre.org/data/definitions/117.html
3. https://cwe.mitre.org/data/definitions/223.html
4. https://cwe.mitre.org/data/definitions/532.html
5. https://cwe.mitre.org/data/definitions/778.html
6. https://owasp.org/www-community/attacks/Log_Injection
7. https://owasp.org/www-project-proactive-controls/v3/en/c9-security-logging.html