Security Awareness Infrastructure

Creazione di un'infrastruttura di rete per la security awareness aziendale

Titolo del progetto:	Security Awareness Infrastructure
Alunno/a:	Pascal Galli
Classe:	Info 4B
Anno scolastico:	2023/2024
Formatore:	Federico Bidoggia
Azienda:	InTheCyber Group S.A.
Perito:	Enea Macrini

1 Introduzione

1.1 Informazioni sul progetto

Le informazioni principali del progetto sono le seguenti:

- **Progetto:** LPI SPAI 2023/2024
- **Nome del progetto:** Security Awareness Infrastructure
- **Allievo coinvolto nel progetto:** Pascal Galli
- **Classe:** I4B – Scuola Professionale Arti Industriali di Locarno
- **Formatore responsabile:** Federico Bidoggia
- **Perito:** Enea Macrini
- **Data inizio:** 02.04.2024
- **Data di fine:** 15.04.2024

1.2 Abstract

Nonostante l'alfabetizzazione informatica abbia raggiunto livelli molto alti soprattutto grazie all'esplosione di internet negli ultimi venti anni, l'educazione alla navigazione nello stesso spazio virtuale e la consapevolezza dei rischi che si possono incorrere utilizzando questo strumento quotidianamente è molto bassa. Molte persone sottovalutano ancora i pericoli che si nascondono in ogni angolo di internet. Questi pericoli possono avere diversi tassi di rischio, valutati in base alla situazione e all'ambiente in cui avvengono. Gli spazi di lavoro nelle aziende sono gli ambienti dove i malintenzionati si concentrano maggiormente, per sfruttare l'ingenuità dei dipendenti e rubare informazioni vitali e preziose, se vendute al giusto prezzo e al giusto acquirente. Questi attacchi avvengono sfruttando soprattutto il social engineering e diverse tecniche informatiche, tra cui il phishing. Dall'altra parte ci sono invece aziende che operano nel campo della sicurezza informatica, che vendono servizi per portare attacchi fittizi alle aziende che chiedono queste operazioni, il tutto per testare l'integrità del proprio sistema informatico e del proprio personale. Ed è proprio su quest'ultimo che il progetto si concentra, andando a creare un'infrastruttura capace di portare attacchi di phishing all'interno dell'azienda che ha richiesto il servizio. I risultati saranno quello di poter constatare la consapevolezza dei rischi del personale e soprattutto la creazione di un prodotto pronto all'uso in pochi secondi grazie ad un sistema ottimizzato.

Even though computer literacy has reached very high levels, mainly due to the explosion of the Internet over the last twenty years, education on navigating in virtual space and awareness of the risks one can incur by using this tool daily is very low. Many people still underestimate the dangers that lurk in every corner of the Internet. These dangers can have different rates of risk, depending on the situation and the environment in which they occur. Workplaces in companies are the environments where malicious attackers are most concentrated, to exploit the naivety of employees and steal vital and valuable information, if sold at the right price and to the right buyer. These attacks take place mainly by exploiting social engineering and various IT techniques, including phishing. On the other hand, there are companies operating in the field of computer security, which sell services to bring fictitious attacks to the companies requesting these operations, all to test the integrity of their computer systems and personnel. And it is precisely on the latter that the project focuses, going to create an infrastructure capable of bringing phishing attacks inside the company that requested the service. The results will be the staff's risk awareness and, above all, the creation of a ready-to-use product within seconds thanks to an optimised system.

1.3 Scopo

Il progetto si propone di realizzare e configurare un'infrastruttura di rete specializzata per la pianificazione e gestione di campagne di phishing, facendo uso del framework Gophish. Questa infrastruttura comprenderà un sistema di mailing integrato e sarà implementata utilizzando la tecnologia di containerizzazione, garantendo così una gestione efficiente, scalabile e sicura delle risorse necessarie per condurre le operazioni di phishing in modo controllato e monitorato. In questo modo questo sistema può diventare portabile e utilizzabile su qualsiasi piattaforma compatibile.

Lavorando in un'azienda che opera nel campo della sicurezza informatica, è stato proposto questo progetto per rimanere pertinenti all'ambiente in cui è stata condotta la formazione dello studente, e per dare un percorso stimolante che evitasse di dare problemi di disinteresse da parte dello studente.

Questo progetto può essere un punto di partenza per lo studente per un proseguimento accademico proprio nella sicurezza informatica. Per la scuola può diventare uno spunto interessante per approcciarsi ad un campo, ovvero la cyber security, che ancora rimane poco considerato a livello scolastico.

1.4 Obiettivi

Gli obiettivi del progetto sono stati stabiliti per garantire una procedura coerente e stabile nel corso dell'implementazione dell'infrastruttura. Il risultato finale del prodotto sarà, a tutti gli effetti, il raggiungimento degli obiettivi. Gli obiettivi, provenienti dal QdC, sono nell'ordine:

1. **Correttezza della configurazione delle tecnologie utilizzate:** L'utilizzo di tecnologie simili per aumentare in primis l'awareness, soprattutto, degli impiegati aziendali, deve essere pensato con criterio ma soprattutto durante l'implementazione di quest'ultime deve essere coerente e il più corretta possibile, di modo da poter anche portare a termine gli altri obiettivi con facilità.

2. **Conformità alle best-practices di sicurezza:** Trattandosi un progetto che opera nel campo della sicurezza informatica, è ideale anche una particolare attenzione nei confronti della sicurezza del prodotto.

3. **Organizzazione ed estendibilità delle configurazioni:** Il prodotto deve poter avere una certa elasticità nelle sue configurazioni, di modo da potere venire incontro a tutte le necessità del caso, nel momento in cui il sistema viene utilizzato per il suo scopo.

4. **Riproducibilità del deployment:** Una parte importante del progetto sarà quella di poter riuscire a consegnare un prodotto che possa essere riprodotto in qualsiasi modo e in qualsiasi contesto, seguendo la documentazione scritta, come nel punto 7.

5. **Resilienza e tolleranza ai guasti:** A progetto finito, il prodotto dovrà essere il più completo possibile in termini di vendibilità, e questo riguarda anche la resilienza e tolleranza ai guasti. Nel caso più grave che ci sia un errore, il sistema deve essere in grado di essere riparato senza grossi problemi.

6. **Test esaustivo delle funzionalità principali:** Al termine delle implementazioni e configurazioni principali, lo svolgimento dei test è una parte fondamentale per capire se effettivamente si è svolto tutto correttamente.

7. **Produzione di documentazione chiara e completa:** Una documentazione redatta correttamente può servire per capire al meglio l'idea e la sua implementazione, ma soprattutto fornisce una spiegazione chiara per tutto ciò che riguarda il progetto nella sua interezza.

I vari obiettivi sono stati estrapolati direttamente dal QdC dove, grazie ad una attenta analisi preliminare, non è stato necessario modificare in seguito durante la trascrizione nella documentazione.

1.5 Pianificazione iniziale

La pianificazione del progetto si divide essenzialmente in quattro macro-fasi che compongono l'iter per il completamento del programma. Di seguito vengono elencate le fasi, seguite da una breve descrizione e la percentuale del tempo che queste fasi occupano nel progetto. Queste informazioni provengono dal QdC.

- **Analisi:** La prima parte dove avviene la pianificazione iniziale, le analisi preliminari, la raccolta delle informazioni e del materiale. *20% - 2gg*
- **Realizzazione:** La fase dove avviene l'implementazione effettiva. La parte più rischiosa ed è quella che potrebbe non far coincidere la pianificazione iniziale con quella finale. *30% - 3gg*
- **Test:** La terza parte del progetto, dove avvengono i test di funzionamento e dove vengono applicate le soluzioni nel caso si presentino eventuali errori. *20% - 2gg*
- **Documentazione:** L'ultima parte del progetto, ovvero tutto ciò che riguarda le documentazioni, che include anche il diario di lavoro e i vari manuali d'uso. *30% - 3gg*

Queste fasi poi sono suddivise in attività più precise e descrittive, mostrate attraverso il diagramma di Gantt nel capitolo *2.5 Pianificazione*, dove vengono visualizzate per intero le tempistiche del progetto.

2 Analisi

2.1 Analisi del dominio

In un panorama digitale sempre più complesso, le infrastrutture dedicate al phishing sono diventate un elemento comune tra gli attori malevoli. Tuttavia, mentre esistono varie soluzioni di questo genere, il progetto che viene presentato si distingue per un'innovativa caratteristica: l'infrastruttura tascabile.

L'obiettivo principale di questo progetto è introdurre un concetto rivoluzionario nell'ambito delle campagne di phishing: la portabilità. Mentre le infrastrutture esistenti spesso richiedono risorse significative e configurazioni complesse, la proposta mira a offrire un sistema leggero e flessibile che può essere facilmente trasportato e utilizzato su qualsiasi piattaforma compatibile. Questa infrastruttura tascabile è progettata per adattarsi alle esigenze degli utenti moderni che necessitano di flessibilità e mobilità nelle loro operazioni di phishing. Che si tratti di testare la sicurezza di una rete aziendale, condurre un'analisi di vulnerabilità su un sito web o effettuare una dimostrazione di sicurezza, la nostra soluzione offre la libertà di farlo ovunque e in qualsiasi momento. Inoltre, la portabilità non compromette la potenza o la sicurezza dell'infrastruttura. Utilizzando la tecnologia di containerizzazione, il sistema é in grado di garantire che tutte le componenti essenziali per il phishing, compresi i sistemi di mailing e il framework Gophish, siano isolati e protetti, garantendo la sicurezza delle operazioni condotte.

In sintesi, il progetto mira a ridefinire il modo in cui le infrastrutture di phishing sono concepite e utilizzate. Con questa infrastruttura tascabile, si vuole offrire agli utenti la possibilità di condurre le loro operazioni di phishing in modo agile, efficace e sicuro, ovunque si trovino.

2.2 Concetto

Il concetto di fondo del progetto è la creazione di uno strumento indipendente per l’esecuzione di scenari per aumentare la security awareness dei dipendenti di una qualsiasi azienda attraverso scenari specifici di phishing.

Il sistema si differenzia di tre container Docker che ospitano i tre software utili per la realizzazione, tutti internati in una virtual machine, che funge da host. E tra i container del web server e Gophish vi sono delle regole proxy-pass verso l’IP di Gophish. Lo schema di seguito rappresenta l’idea generale della struttura del prodotto.

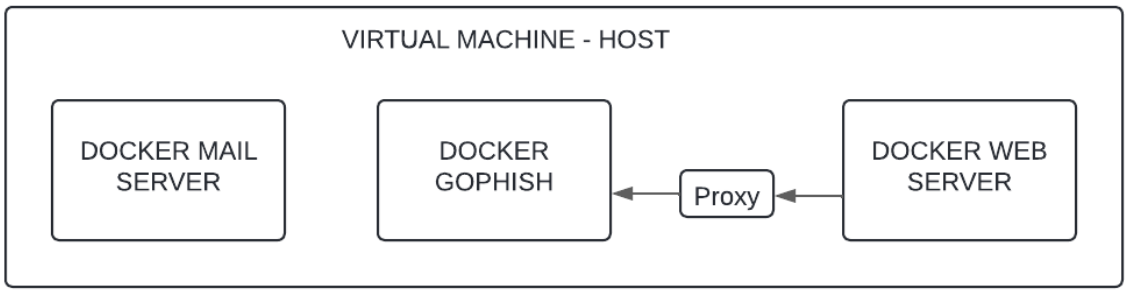

Figura 1: Flow Chart Generico

Dunque, il concetto del progetto è semplice quanto solido, che viene mantenuto per l’intera durata della realizzazione del prodotto, senza cambi significativi dell’idea iniziale, mostrata attraverso questi schemi.

2.3 Analisi e specifica dei requisiti

L'infrastruttura del progetto deve comprendere i seguenti requisiti, di modo da poter funzionare in modo ottimizzato e senza errori.

Requisito	REQ-01	**Priorità**	1	**Versione**	1.0
Nome	Configurazione file *docker-compose.yml*				
Note	La configurazione del *docker-compose.yml* è fondamentale per concedere all'infrastruttura di attivarsi con tutte le funzionalità opportune al caso.				

Requisito	REQ-02	**Priorità**	1	**Versione**	1.0
Nome	Invio di e-mail				
Note	L'infrastruttura deve essere in grado di inviare e-mail durante le campagne di phihing alle *vittime* dell'attacco				
Sotto requisito	REQ-02-01	**Descrizione**	L'infrastruttura deve contenere un mail server propriamente configurato.		

Requisito	REQ-03	**Priorità**	1	**Versione**	1.0
Nome	Capacità di restare sempre attiva				
Note	L'infrastruttura, almeno durante l'intera campagna di phishing, deve essere in grado di rimanere attiva o avere un *down-time* il più piccolo possibile.				

Requisito	REQ-04	**Priorità**	1	**Versione**	1.0
Nome	Creazione di campagne di phishing				
Note	Deve essere in grado di creare campagne di phishing da poter essere lanciate ai target selezionati.				

Requisito	REQ-05	**Priorità**	1	**Versione**	1.0
Nome	Monitoraggio delle risposte				
Note	Deve avere la capacità di poter monitorare le risposte una volta inizializzata la campagna di phishing, di modo da poter tenere traccia dell'andamento della campagna.				

Requisito	REQ-06	**Priorità**	1	**Versione**	1.0
Nome	Analisi dei dati				
Note	L'infrastruttura deve avere la capacità di analizzare i dati, una volta raccolti, e poterli visualizzare in modo che l'attaccante possa utilizzarli a suo piacimento.				

Requisito	REQ-07	**Priorità**	1	**Versione**	1.0
Nome	Funzionalità di proxy-pass				
Note	-				

2.4 Use case

Normalmente l'esecuzione di una campagna di phishing avviene pressoché nella stessa maniera, dunque lo schema *use-case* sottostante rappresenta un comportamento che ci si aspetta generalmente dall'attaccante e dalla vittima.

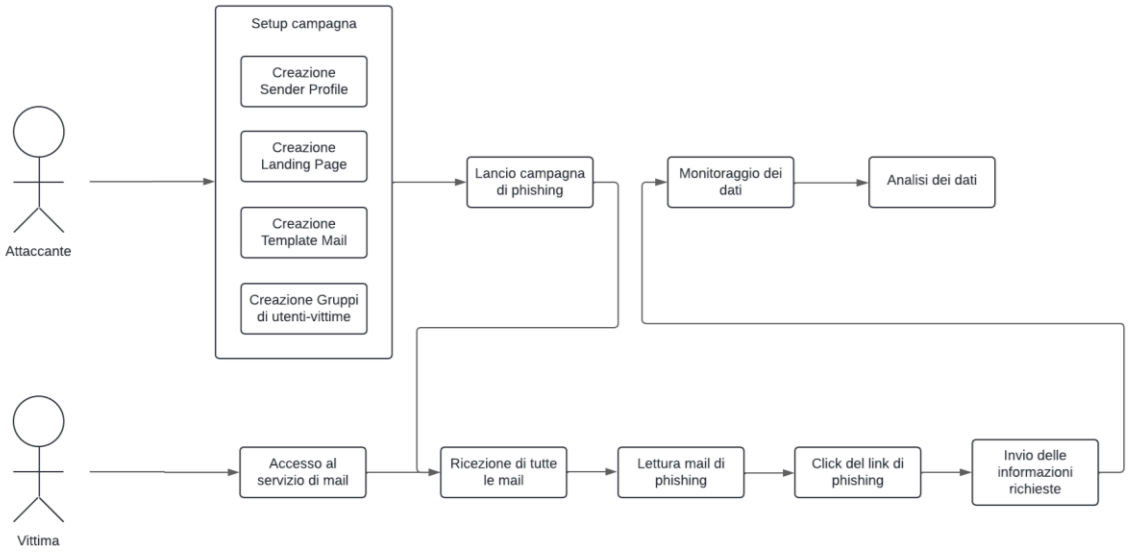

Figura 2: Use Case - Campagna di phishing

Come visibile nello schema, la campagna di phishing inizia dopo il setup della stessa, quando avviene il lancio ci si aspetta che la vittima risponda alla mail inserendo le informazioni da noi richieste tramite la landing page. In caso la vittima venga ingannata, nell'analisi dei dati l'attaccante è in grado di recuperare le informazioni che fanno al caso suo. In caso contrario non verrà inviato nessun dato e l'attaccante dovrà aspettare la risposta di un'altra eventuale vittima.

2.5 Pianificazione

Nel capitolo *1.5 Pianificazione iniziale* vi è la pianificazione pensata e approvata dal formatore e dal perito, mentre in questo capitolo vi è una rappresentazione più chiara e precisa delle attività che vengono condotte nel progetto. Di seguito è rappresentato un diagramma di Gantt con tutte le attività che verranno condotte nel progetto.

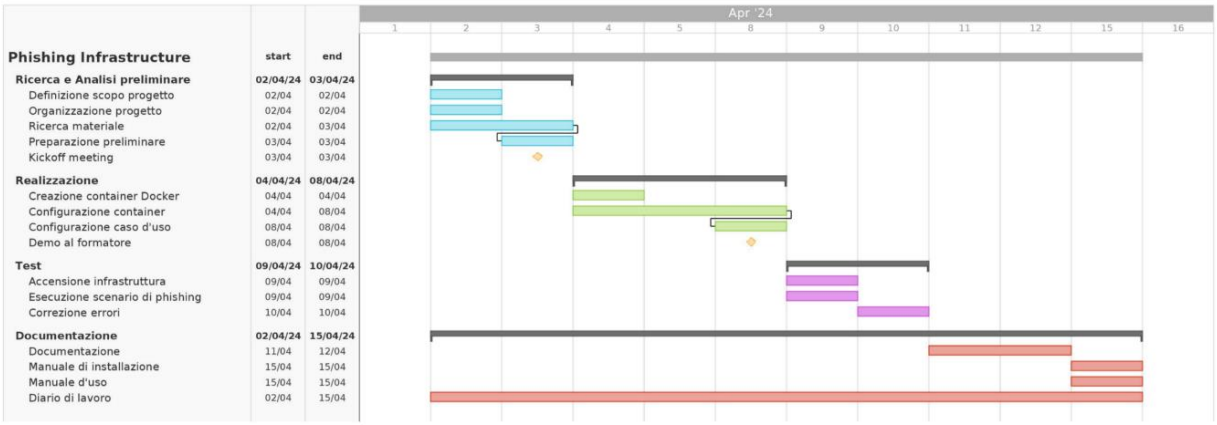

Figura 3: Diagramma di Gantt preventivo

Dal diagramma di Gantt si nota una pianificazione simile a quella dichiarata sul QdC, questo perché, dopo una breve discussione con il formatore, si è deciso di seguire la pianificazione consigliata inizialmente, con le diverse attività allocate nel giusto contesto e con le giuste tempistiche.
Chiaramente il diagramma non contiene tutte le attività specifiche (p.e. setup delle VM, configurazione file Docker, etc) a causa del fatto che, se fosse venuto fuori un imprevisto troppo grave, tale da far cambiare strategia di implementazione, il Gantt consuntivo sarebbe stato totalmente differente non solo nelle tempistiche ma anche nella descrizione delle attività.

Nel caso poi le varie attività venissero svolte, sono già state messe in conto eventuali task che andrebbero a migliorare e rendere più completo il progetto. È possibile anche che il tempo permetta di effettuare maggiori test per poter portare il livello di errori a zero.

2.6 Analisi dei mezzi

L'analisi dei mezzi rappresenta il punto di partenza essenziale per la realizzazione del progetto, offrendo una base solida su cui costruire e sviluppare le varie fasi successive. La corretta scelta e integrazione di software e hardware sono fondamentali per garantire l'efficienza, l'affidabilità e il successo complessivo dell'implementazione progettuale.

2.6.1 Software

I software utilizzati sono i seguenti:

- Oracle VirtualBox v6.1.36
- Linux Ubuntu v22.04.4
- Apache (Docker image) v2.4.59
  - Modulo: mod_proxy
  - Modulo: mod_http
- Gophish (cisagov) v0.0.8
- Postfix (cisagov) v0.0.6
- Docker v26.0.0
- VS Code v1.88.0
- Proton Mail v5.0.37.10

2.6.2 Hardware

Durante l'intera realizzazione del prodotto viene usato il PC aziendale, con le specifiche sufficienti per portare a termine il progetto. Specifiche che sono le seguenti:

HP Elitebook 845 G8

- **CPU:** AMD Ryzen 7 Pro 5850U
  - Clock base: 1.9 GHz
  - N. Cores: 8
  - N. Threads: 16

- **RAM:** 16 GB
  - Disponibile: 15.3 GB

- **Storage:** 500 GB
  - VM used: 20 GB

2.7 Rischi tecnici

Indubbiamente ci sono diversi aspetti tecnici che compongono dei rischi durante la realizzazione del progetto. Nessuno di questi dovrà comportare particolari problemi, se trattati e risolti con criterio, prima di iniziare l'implementazione del prodotto.

- **Docker:** Uno degli strumenti principali del progetto. Sono già state fatte delle attività in passato usando questo sistema. Attività riuscite con il difetto di avere poca documentazione prodotta; dunque, è fondamentale riprendere le nozioni base (comandi, funzionamento, etc) tramite anche la documentazione ufficiale del prodotto.

- **Proxy:** Una tipologia di strumento mai approfondita. È un elemento importante nell'implementazione; dunque, è importante assumere il maggior numero di competenze a proposito di questo sistema.

- **SMTP:** Protocollo fondamentale nell'ambito delle e-mail. Non comporta un rischio tecnico alto, ma sapere bene con che cosa si sta lavorando può aiutare con la risoluzione di parecchi errori.

A livello prettamente tecnico, di sistema, non ci sono particolari limiti. Il progetto in questione non presenta un livello di complessità elevato, dunque risolte le lacune elencate, il prodotto rappresenta una percentuale di fattibilità molto alto.

3 Progettazione

Questo capitolo descrive esaustivamente come deve essere realizzato il prodotto fin nei suoi dettagli. Una buona progettazione permette all'esecutore di evitare fraintendimenti e imprecisioni nell'implementazione del prodotto.

3.1 Design dell'architettura del sistema

Con le informazioni raccolte durante l'analisi preliminare è possibile stendere una bozza di quello che potrebbe essere la forma (concettuale) dell'infrastruttura del sistema. Di seguito un breve schema che rappresenta l'infrastruttura più nel dettaglio, ovvero l'evoluzione dello schema presente nel capitolo *2.2 Concetto*.

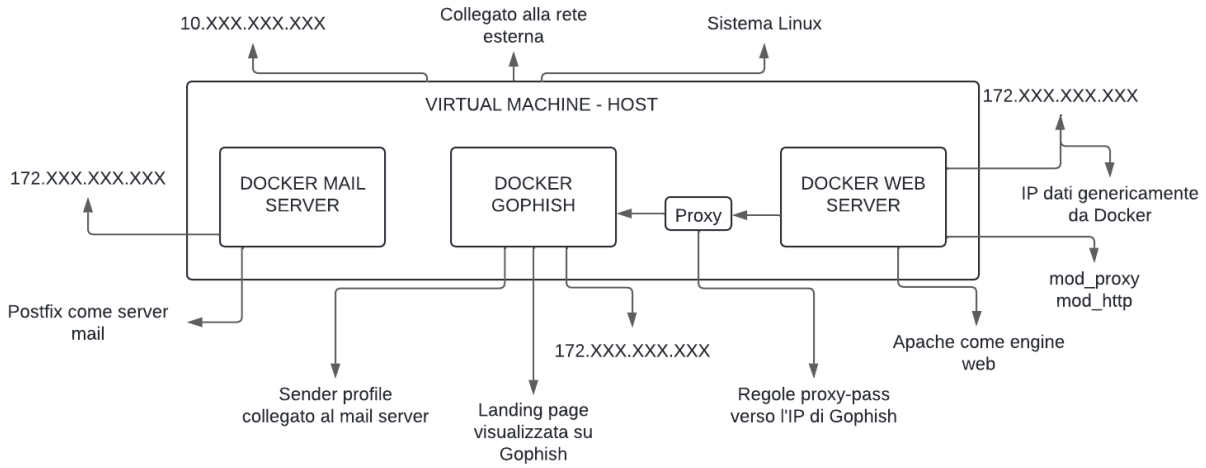

Figura 4: Flow Chart specifico

Questo schema rappresenta quello che è l''infrastruttura del sistema con i vari container Docker incubati in una macchina virtuale che assume il compito di fare da host. La VM è attiva con un sistema operativo Linux Ubuntu. I vari container attivi dallo stesso `docker-compose.yml` viene assegnato un indirizzo IP della stessa sottorete di default di Docker Engine, che è 172.18.0.0/24, oppure è possibile inserire una pool di indirizzi IP che poi il DHCP di Docker Engine si occupa di distribuire tra i vari container attivi. Di seguito una breve spiegazione dei vari container del sistema con le loro componenti principali.

- **Docker Mail:** Configurato con il nome *postfixenv*, è il container abilitato come server mail, per permettere al servizio di Gophish di appoggiarsi al servizio di mailing per inviare le proprie campagne di phishing. Nel container è inizializzata un'immagine di Postfix.

- **Docker Gophish:** Configurato con il nome *gophishenv*, è il container che ospita il servizio di gestione delle campagne di phishing. È il container principale a cui gli altri fanno riferimento e a cui si basa l'intero sistema. Al suo interno sarà collegata anche la landing page, impostata come destinazione finale del percorso di phishing alle vittime.

- **Docker WebServer:** Configurato con il nome *apachenv*, è il container che si occupa di ospitare il servizio web Apache dove vengono configurati i vari domini che servono per lanciare le varie campagne di phishing. Inoltre, sono abilitati i moduli *mod_proxy* e *mod_http* per configurare una sorta di proxy-pass che si collega con il servizio di Gophish.

3.2 Schema procedurale

Prima di iniziare effettivamente l'implementazione del progetto e durante l'analisi preliminare vi è una fase di progettazione dove viene deciso il percorso e la logica di implementazione del sistema, in questo caso l'infrastruttura di security awareness tramite phishing. Di seguito uno schema riassuntivo della procedura di implementazione.

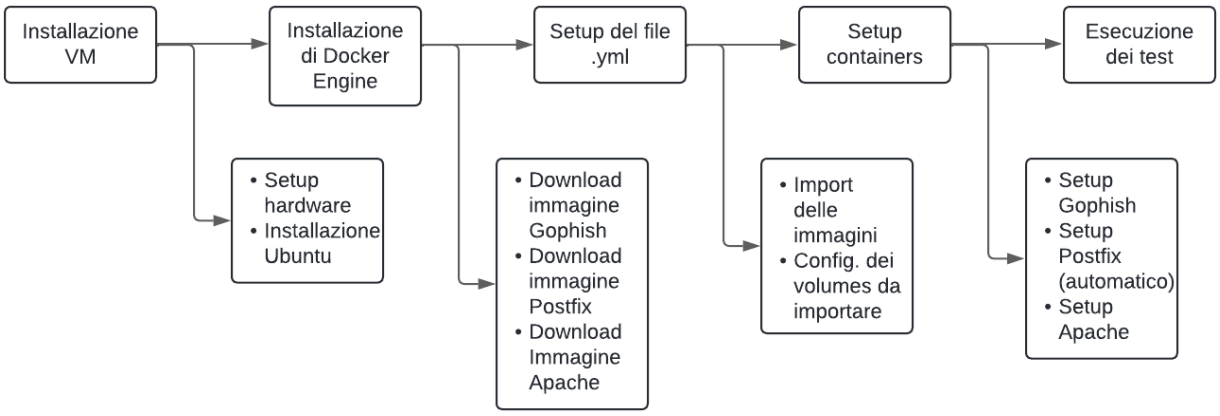

Figura 5: Schema procedurale

Avere un flusso di lavoro da seguire durante il progetto permette di non perdere tempo ad ogni azione per capire quale sia il passaggio successivo. Dunque, durante le analisi preliminari c'è stato modo di verificare e prevedere i vari step da prendere in considerazione per implementare l'intero sistema. Lo schema procedurale rappresenta dunque i vari passaggi. I blocchi collegati orizzontalmente sono i macro-step che si deve seguire durante l'implementazione, mentre i blocchi nel livello inferiore sono i passaggi più specifici dell'implementazione.

Inoltre, nonostante la maggior parte dell'implementazione è di tipo sistemistico, esistono dei file da configurare per assicurarsi la piena funzionalità dell'infrastruttura. Di seguito è mostrata una tabella con i file principali che vengono configurati.

ID	Nome file	Nome	Descrizione
01	docker-compose.yml	File di composizione	File di composizione principale, da dove vengono importate le varie immagini dei container che vengono inizializzati. Sono presenti anche tutte le
02	config.json	File di configurazione	File di configurazione per Gophish dove vengono configurate le varie porte utilizzate per i due servizi di *admin* e *listening*.
03	httpd.conf	File di configurazione	File di configurazione di Apache dove si possono trovare tutte le impostazioni e i moduli (come mod_proxy) da abilitare.
04	httpd-vhosts.conf	File di configurazione	File di configurazione utile a creare dei virtual-hosts per ospitare dei domini locali
05	/etc/hosts	File di configurazione	File di configurazione presente sulla macchina host per configurare i vari indirizzi che portano al dominio locale

Quando si è condotta la fase di analisi preliminare e si è conclusa la fase successiva, ovvero quella di progettazione, le basi del progetto sono pronte per essere utilizzate per l'implementazione che si può trovare nel capitolo successivo: *4. Implementazione*.

4 Implementazione

Nei capitoli precedenti, è stata delineata con precisione la pianificazione dettagliata del progetto, evidenziando gli obiettivi chiave, le risorse necessarie e le tappe critiche per la finalizzazione del sistema. Ora, è il momento di tradurre questa visione in azione attraverso l'implementazione pratica.

Questo capitolo si focalizza sulle componenti principali che hanno contribuito alla creazione e alla stabilità dell'infrastruttura del progetto. Anziché dettagliare ogni passaggio tecnico, viene mantenuto il *focus* sui concetti e sugli strumenti fondamentali che hanno guidato il processo di realizzazione. Sebbene la documentazione completa del processo sia indirizzata principalmente agli specialisti informatici, questo capitolo mira a fornire una panoramica comprensibile e accessibile delle principali fasi e decisioni che hanno guidato la creazione dell'infrastruttura del progetto.

4.1 Setup Virtual Machine

È possibile usare diversi virtualizzatori (ESXI, VirtualBox, etc) e per questo progetto è stato utilizzato VirtualBox dunque tutte le impostazioni si riferiscono a VirtualBox, nonostante concettualmente si possa traslare agli altri virtualizzatori.
Le impostazioni dell'hardware sono elencate di seguito, sono le impostazioni minime consigliate per avviare l'infrastruttura in totale sicurezza.

- **Sistema operativo:** Linux Ubuntu 24.04.4
- **Memoria RAM:** 4096 byte (4 GB)
- **vCPU:** 2
- **Storage:** 25 GB

Le impostazioni hardware utilizzate effettivamente nel progetto sono invece le seguenti.

- **Sistema operativo:** Linux Ubuntu 24.04.4
- **Memoria RAM:** 6144 byte (6 GB)
- **vCPU:** 4
- **Storage:** 25 GB

4.2 Installazione software e immagini

I software installati per l'utilizzo di quest

- **Docker Engine** Motore di containerizzazione
- **Vim** Editor di testo

Mentre le immagini che sono servite all'implementazione del progetto, attraverso l'avvio dei vari container adibiti al loro specifico scopo, sono le seguenti.

- httpd Servizio Apache - *web server*
- cisagov/gophish:0.0.8 Servizio Gophish – *phishing*
- cisagov/postfix:0.0.6 Servizoi Postfix – *mail server*

Per scaricare e installare le varie immagini, è consultabile il Manuale d'installazione in allegato a questa documentazione.

4.3 Configurazione file

Una parte importante dell'implementazione del sistema non è tanto la parte hardware, quanto la parte di configurazione attraverso dei file specifici. File descritti di seguito.

4.3.1 docker-compose.yml

Il file più importante per erigere in maniera funzionale l'intera infrastruttura di container è il file `docker-compose.yml`, ovvero il file dove si scrivono tutte le impostazioni di partenza per far si che i vari container si avviino in maniera corretta e con le configurazioni giuste. Da notare che il nome del file non può essere cambiato, deve rimanere `docker-compose.yml`.
La prima riga di codice riguarda la versione del file, non prettamente significativa in quanto indica solamente la versione del docker-compose ad ogni sua modifica.

```
version: "3.7"
```

Vi sono poi delle configurazioni da fare all'inizio del file, chiamati *secrets* che permettono di importare diverse impostazioni, come scritto nello snippet, `config.json` che viene trattato nel capitolo 4.3.2 ed altri file all'interno di Gophish in questo caso.

```
secrets:
  gophish_config_json:
    file: ./secrets/gophish/config.json
```

È necessario importare anche diverse configurazioni per il server mail Postfix. Nella configurazione base che è stata usata per il progetto non è stato necessario importare configurazioni importanti, ma un file di testo con all'interno gli utenti utili è necessario in caso di problemi al server mail.

```
secrets:
  postfix_users_txt:
    file: ./secrets/postfix/users.txt
```

Una volta preparati i *secrets*, è necessario preparare i vari container. Il primo passo é configurare le imposatzioni di base come l'immagine da utilizzare, il nome del container e nel caso il container dovesse chiudersi, come farlo ripartire. In questo caso è consigliato mantenerlo sempre acceso in modo che le varie campagne di phishing possano rimanere attive fino alla loro chiusura.

```
services:
  gophish:
    image: cisagov/gophish:0.0.8
    container_name: gophishenv
    init: true
    restart: always
```

Sono importanti anche le impostazioni delle porte da aprire per i servizi che si utilizzano. In questo caso sono state aperte la porta `3333` e la porta `3380`. La prima è necessaria per accedere al portare amministrativo di Gophish dove avviene tutta la configurazione della campagna. La porta `3380` è aperta per il *listener* di Gophish, che serve per ricevere i vari dati che vengono inviati a campagna avviata. Il parametro `target` indica la porta aperta sul container, mentre il parametro `published` è la porta da cui si accede dall'host, normalmente con `http://localhost:3333`

```
ports:
  - target: 3333
    published: 3333
    protocol: tcp
```

```
    mode: host
  - target: 3380
    published: 3380
    protocol: tcp
    mode: host
```

Subito dopo sono necessari i vari *secrets* che sono stati impostati in precedenza. Semplicemente impostare come di seguito, dato che questi sono impostazioni non modificabili di Gophish.

```
secrets:
  - source: gophish_config_json
    target: config.json
```

Per il container di Postfix le configurazioni iniziali sono leggermente diverse. Da notare il parametro `environment` che viene riempito con due costanti, ovver `PRIMARY_DOMAIN` ovvero il dominio primario che viene utilizzato per la campagna di phishing, dove è stato inserito `mircosoft.com` per assomigliare a microsoft.com. Nella costante di `RELAY_IP` è necessario inserire la subnet dove vengono avviati i vari container, in questo caso `172.18.0.0/24`. È importante questo range di IP perché rappresenta tutti gli IP che sono autenticati a mandare mail dal server di Postfix.

```
postfix:
    image: cisagov/postfix:0.0.6
    container_name: postfixenv
    init: false
    restart: always
    environment:
      - PRIMARY_DOMAIN=mircosoft.com
      - RELAY_IP=172.18.0.0/24
```

Impostando le varie porte del server mail ci si assicura di avere Postfix pienamente funzionante. La porta `1025` è la porta SMTP di Postfix (mentre quella interna rimane come da protocollo `25`).

```
ports:
  - target: 25
    published: 1025
    protocol: tcp
    mode: host
  - target: 587
    published: 1587
    protocol: tcp
    mode: host
```

Sono importanti anche i *secrets* impostati in precedenza di Postfix, importabili allo stesso modo come è stato fatto con Gophish.

Nell'ultimo container, quello di Apache le configurazioni sono differenti dagli altri due, sebbene le prime righe siano uguali.

```
apache:
    image: httpd
    container_name: apachenv
    init: false
    restart: always
```

Le porte a disposizione del container di Apache sono la `80` e la `8080`, ma è stata impostata solo la prima (sia in `target` che `published`) per non farla vedere poi nell'URL completo a campagna avviata.

```
ports:
      - target: 80
      - published: 80
      - protocol: tcp
      - mode: host
```

L'ultima configurazione da fare riguarda i `volumes`, che sono file fisici da importare nelle posizioni corrette del container di Apache. I file importati sono `httpd.conf` e `httpd-vhosts.conf`.

```
volumes:
  - ./conf/httpd.conf:/usr/local/apache2/conf/
  - ./conf/httpd-vhosts:/usr/local/apache2/conf/extra/
```

Una volta definito il file `docker-compose.up`, è possibile avviare i container con il comando seguente.

```
> sudo docker compose up
```

4.3.2 config.json

Uno dei file che si possono trovare all'interno del docker-compose.yml è il file di configurazione `config.json`. Il file è diviso in due parti principali, ovvero `admin_server` e `phish_server`. `admin_server` serve a chi conduce la campagna di phishing per accedere alla dashboard di Gophish. Ci sono da impostare dei certificati e delle chiavi attraverso dei file `.pem`, per inizializzare correttamente il servizio e utilizzarlo in totale sicurezza.

```
"admin_server": {
   "cert_path": "/run/secrets/admin_fullchain.pem",
   "key_path": "/run/secrets/admin_privkey.pem",
```

Alla stessa maniera, sempre in `admin_server`, è necessario impostare il `listen_url` che è l'URL da dove generalmente si può accedere alla dashboard di Gophish. Impostato come `0.0.0.0:3333` vuol dire che l'amministratore è abilitato all'accesso da qualsiasi parte, sempre sapendo l'indirizzo IP della macchina host che ospita i vari container. Disabilitare l'uso dei certificati TLS è importante se non si è in possesso di tali certificati, altrimenti l'accesso diventa impossibile.

```
"listen_url": "0.0.0.0:3333",
"use_tls": true
```

Esattamente come in `admin_server`, anche in `phish_server` (che è l'indirizzo di ascolto delle varie risposte delle vittime della campagna di phishing) sono necessari i vari certificati e chiavi private.

```
"phish_server": {
   "cert_path": "/run/secrets/phish_fullchain.pem",
   "key_path": "/run/secrets/phish_privkey.pem",
```

Come visto anche in `admin_server` è necessario abilitare l'ascolto del server attraverso l'indirizzo `0.0.0.0:3380` che ne permette l'accesso da parte delle vittime. Disabilitare l'uso dei certificati TLS è importante se non si è in possesso di tali certificati, altrimenti l'accesso diventa impossibile.

```
"listen_url": "0.0.0.0:3380",
"use_tls": false
```

4.3.3 users.txt

In caso di problemi al server mail di Postfix è necessario un intervento diretto al container, e per accedere al servizio in funzione al suo interno è consigliato impostare degli utenti da usare per queste situazioni. Utenti da configurare nel file di testo `users.txt` come di seguito.

```
# define other users below as needed
secadmin secadm124
```

4.3.4 httpd.conf

Un file importante da importare, nonostante sia già presente in Apache ma con le nostre condizioni, è `httpd.conf`, uno dei file più importanti nel servizio di web server di Apache. Per permettere il proxy-pass tra il container di Apache e Gophish è necessario abilitare questi due moduli di Apache `mod_proxy.so` e `mod_proxy_http.so`, come mostrato nello snippet di codice sotto.

```
#LoadModule version_module libexec/apache2/mod_version.so
LoadModule proxy_module libexec/apache2/mod_proxy.so
LoadModule proxy_http_module libexec/apache2/mod_proxy_http.so
#LoadModule proxy_connect_module libexec/apache2/mod_proxy_connect.so
```

Alla fine del file `httpd.conf` è necessario aggiungere due righe per comunicare al web server qual è l'IP, l'indirizzo o comunque l'host dove si desidera lasciar passare determinate richiesta. Come visibile sotto in entrambe le righe di ProxyPass e ProxyPassReverse è stato preso come host di partenza quello di Apache, dove alla sua richiesta viene passato verso il container di Gophish, `gophishenv`, alla porta 3380.

```
ProxyPass / http://gophishenv:3380/
ProxyPassReverse / http://gophishenv:3380/
```

Non é necessario abilitare o aggiungere ulteriori modifiche al file di configurazione, in quanto il restante file basta per avviare il server nelle condizioni che servono.

4.3.5 httpd-vhosts.conf

Insieme al file `httpd.conf`, bisogna configurare anche i vari virtual hosts, in questo caso solo uno (quello del dominio fittizio di phishing). Le quattro configurazioni sotto, sono necessarie per garantire che il servizio di Apache sia in grado di capire quando direzionare le richieste dal momento in cui la vittima, o l'admin, voglia volgersi sul dominio adibito a dominio fittizio di phishing. Il parametro `DocumentRoot` serve per impostare la directory dove sono posizionati i vari file web.

```
<VirtualHost *:80>
    ServerAdmin admin@mircosoft.com
    DocumentRoot "/usr/local/apache2/htdocs/"
    ServerName mircosoft.com
    ServerAlias www.mircosoft.com
```

Il tag del `VirtualHost` si può chiudere inserendo anche le configurazioni per i vari file `.log`, inseriti in questo modo mettendo in luce il dominio che si sta utilizzando per facilitare la ricerca poi del file `.log` in futuro.

```
    ErrorLog "logs/mircosoft-error.log"
    CustomLog "logs/mircosoft-custom.log" combined
</VirtualHost>
```

4.3.6 /etc/hosts

Sull'host dove si stanno avviando i vari container è necessario impostare anche il dominio che si intende usare per la campagna di phishing, all'interno del file `/etc/hosts`, come visibile nell'ultima riga.

```
127.0.0.1   localhost
127.0.1.1   secawaenv
127.0.0.1   mircosoft.com
```

4.3.7 mail.html

Gophish permette di inserire nelle mail inviate del codice HTML per personalizzare la mail, di modo che risulti più credibile. Facendo ciò si possono inserire anche dei parametri personalizzati di Gophish, esterni a HTML come i seguenti che si vedono. `{{.FirstName}}` e `{{.LastName}}` fanno riferimento al nome e cognome della vittima inseriti nei vari gruppi di utenti a cui verranno inviate le mail di phishing.

```
<p>Gentile {{.FirstName}} {{.LastName}},</p>
```

Seguito poi dal breve testo di richiesta di login per verificare l'identità dell'utente Microsoft, portandolo a cliccare il bottone. Notare bene che nei template delle email è possibile inserire lo stile del tag solo in modalità *inline*.

```
<p style="font-family: 'Segoe UI'; font-weight: 400;"> Abbiamo rilevato un'attività
sospetta sul tuo account Microsoft associato all'indirizzo e-mail fornito. Per
garantire la massima sicurezza dei tuoi dati e della tua privacy, ti chiediamo
cortesemente di procedere al login del tuo account Microsoft, di modo da poter
verificare la tua identità.</p>
```

Una volta letta la mail, la vittima andrà a cliccare il bottone a fine mail, che dirigerà l'attacco via browser aprendo la *landing page* dedicata all'attacco. Il link della *landing page* viene fornito attraverso il parametro `{{.URL}}` inserito nel classico parametro `href`, adibito al conservare i link nei tag.

```
<button>
   <a href="{{.URL}}">Login</a>
</button>
```

Una volta cliccato il bottone *Login*, la vittima viene trasportata sulla landing page del sito con il dominio fittizio (mircosoft.) dove penserà di accedere al proprio account inserendo le credenziali che vengono rubate.

4.3.8 landing.html

Per creare un form di login per ottenere i dati della vittima è necessario creare tutto all'interno del tag `<form>` che fornisce diverse funzioni per inviare i dati inseriti. Per rendere tutto stilisticamente più credibile si crea un sezione container con il tag `<div>` per inserire tutti gli input necessari e fornire uno stile quanto più realistico possibile. All'interno dei "…" ci sono i vari input che servono all'attaccante per raccogliere dati e alla vittima per inserire i valori che crede di dover inserire.

```
<form class="modal-content" method="post">
   <div class="container">
      ...
   </div>
</form>
```

All'interno del tag `<div class="container">` si inseriscono due tag `<input>` per raccogliere le varie informazioni. Il primo chiaramente è la richiesta di inserimento della mail; dunque, si chiede un input di tipo testo.

```
<label for="uname"><b>Email</b></label>
<input type="text" placeholder="Email" name="uname" required>
```

Dopo avere inserito la propria mail, manca la password. Per fare ciò si usa sempre un tag `<input>` e per rendere più credibile la situazione si configura il parametro `type=""` con all'interno password, di modo che i valori digitati siano oscurati come un vero form di login. Il parametro `required` richiede che la password venga fornita forzatamente, di modo che all'attaccante non arrivino dati vuoti, con solo la mail della vittima.

```
<label for="psw"><b>Password</b></label>
<input type="password" placeholder="Password" name="psw" required>
```

4.4 Creazione campagna

Una volta configurati i vari file di configurazioni è possibile avviare i container e accedere ai servizi installati. Per creare e avviare una campagna consultare il Manuale d'uso disponibile in allegato.

5 Test

5.1 Protocollo di test

In questo capitolo si possono trovare tutti i test condotti, con i loro protocolli e procedure. I test fungono da garanzia di qualità del prodotto. Ogni test deve essere ripetibile alle stesse condizioni dell'implementazione del progetto. Di seguito le varie tabelle:

Test Case	TC-001	**Nome**	Accesso remoto
Descrizione	Verifica la configurazione corretta delle regole del firewall per consentire l'accesso remoto alla virtual machine host		
Prerequisiti	La Virtual Machine accesa		
Procedura	1. Aprire il terminale sulla macchina 2. Eseguire il seguente comando: `ssh <username>@<IP_Address>`		
Risultati attesi	Connessione SSH accettata e accesso alla shell virtuale di Linux		

Test Case	TC-002	**Nome**	Isolamento dei container
Descrizione	Verifica che i container Docker ospitati sulla virtual machine host siano isolati correttamente l'uno dall'altro		
Prerequisiti	Esecuzione del file *docker-compose.yml* e i container in stato *Running*		

Procedura	1. Eseguire il seguente comando:
	`sudo docker ps`
	E verificare gli ID dei vari container attivi
	
	2. Eseguire il seguente comando:
	`sudo docker inspect <ID_container>`
	E verificare l'indirizzo IP del container		

Risultati attesi	Tre indirizzi IP diversi, facenti parte della stessa sottorete 172.18.0.0/24, uno per ogni container attivo		

Test Case	TC-003	**Nome**	Accesso ai container
Descrizione	Verifica che i container siano accessibili tramite accesso remoto di modo da potere entrarci per risolvere/modificare dei parametri		
Prerequisiti	Esecuzione del file *docker-compose.yml* e i container in stato *Running*		

Procedura	1. Eseguire il seguente comando:
	`sudo docker ps`
	E verificare gli ID dei container attivi
	
	2. Eseguire il seguente comando:
	`sudo docker exec -it <ID_container> /bin/bash`		

Risultati attesi	Accesso alla shell del container selezionato tramite il suo ID		

Test Case	TC-004	**Nome**	Invio e ricezione e-mail
Descrizione	Verifica che il Docker mail server sia in grado di inviare e ricevere e-mail correttamente		
Prerequisiti	Container di Postfix e Gophish in stato attivo		
Procedura	1. Digitare sul browser: http://localhost:3333 2. Eseguire il login al profilo di Gophish con le proprie credenziali 3. Recarsi sulla pagina “Sender profile” 4. Creare un nuovo sender profile Inserire le varie informazioni richieste 5. Cliccare il pulsante *Send test e-mail* e inserire DD **Non** è necessario inserire la posizione, è un dato superfluo 6. Verificare nell’inbox del servizio di mail (Proton) se la mail di test inviata da Gophish sia arrivata		
Risultati attesi	Invio corretto e ricezione nella casella di posta del servizio di mail utilizzato (Proton Mail nel caso di questo test).		

Test Case	TC-005	**Nome**	Creazione campagne di phishing
Descrizione	Verifica che Gophish sia in grado di creare correttamente campagne di phishing con i tipi di mail e domini impostati		
Prerequisiti	In Gophish: sender profile, landing page, gruppi e utenti e-mail template già impostati		
Procedura	1. Dirigersi nell’apposita sezione della pagina di amministrazione di Gophish 2. Aprire un nuovo form di creazione di una campagna di phishing e inserire le informazioni necessarie 3. Cliccare *Start campaign* e verificare la presenza di eventuali errori		
Risultati attesi	Creazione di una campagna di phishing legittima senza errori		

Una volta che i test TC-001, TC-002, TC-003, TC-004 e TC-005 sono stati confermati, bisogna eseguirli. Se il risultato atteso è quello compilato nelle varie tabelle, allora si può procedere con la fase di test, eseguendo i test case TC-006, TC-007 e TC-008.

Test Case	TC-006	**Nome**	Invio di e-mail di phishing
Descrizione	Verifica che Gophish sia in grado di inviare correttamente e-mail di phishing agli utenti di test		
Prerequisiti	Dominio di test (mircosoft.com) inserito nella *whitelist* del servizio di mail. Il template dell'e-mail in HTML deve esser pronto		
Procedura	1. Inserire tutte le informazioni richieste dal form di creazione della campagna 2. Inizializzare la campagna di phishing da Gophish 3. Verificare nell'inbox del mail service (Proton) se la mail di phishing sia arrivata correttamente		
Risultati attesi	Mail in arrivo sulla posta principale e non nella cartella di spam con all'interno la mail visualizzata in modalità grafica, renderizzata dall'HTML inserito in fase di preparazione		

Test Case	TC-007	**Nome**	Monitoraggio delle risposte
Descrizione	Verifica che Gophish monitori le risposte degli utenti alle e-mail di phishing e registri le informazioni pertinenti		
Prerequisiti	Campagna di phishing da Gophish già avviata		
Procedura	1. Una volta avviata la campagna di phishing attendere eventuali risposte delle vittime 2. Con l'utente mail di test rispondere alla mail inserendo i vari dati richiesti (in modo fittizio) 3. Tornare sulla dashboard della campagna di phishing di Gophish e verificare che i dati di monitoraggio si siano aggiornati		
Risultati attesi	Dati aggiornati in tempo reale. Mail inviate, lette e risposte inviate dalle vittime		

Test Case	TC-008	**Nome**	Proxy-pass
Descrizione	Verifica che le regole implementate funzionino in modo corretto		
Prerequisiti	Campagna di phishing da Gophish già avviata e dominio di test (mircosoft.com) inserito nella *whitelist* del servizio di mail		
Procedura	1. Aprile il servizio di mail In questo caso è utilizzato **Proton Mail** 2. Controllare che sia arrivata la mail di phishing Con le configurazioni attuali la mail non finisce nello spam 3. Aprire la mail di phishing Controllare se è presente un link o un bottone da cliccare 4. Cliccare il link all'interno della mail		
Risultati attesi	Una volta cliccato il link dall'email, si apre il browser mostrando la landing page che dove la vittima inserisce i suoi dati		

5.2 Risultati test

Dopo avere pianificato i vari test, sono stati eseguiti in ordine di compilazione in modo da non avere mancanze e errori causati da test fatti in modo disordinato. Di seguito una tabella riassuntiva dei risultati dei test.

Test Case	Esito	Data ultimo test	Risultato	Commenti
TC-001	Fallito	10.04.2024	Nessuno	A causa dell'utilizzo di una scheda di rete collegata al WiFi non è stato possibile impostare la scheda VM in bridge e connettersi in remoto
TC-002	Passato	10.04.2024	Indirizzi IP dei container diversi	-
TC-003	Passato	10.04.2024	Accesso abilitato alla shell dei vari container	-
TC-004	Passato	10.04.2024	Mail ricevuta nell'inbox del servizio di mail	-
TC-005	Passato	10.04.2024	Campagna di phishing creata con successo con tutte le impostazioni del caso	-
TC-006	Parzialmente passato	10.04.2024	Mail ricevuta nell'inbox del servizio di mail	Invio della mail di phishing passato ma qualche problema nella visualizzazione dell'HTML
TC-007	Passato	10.04.2024	Monitoraggio delle risposte attivo	-
TC-008	Passato	10.04.2024	Proxy-pass attivo e funzionante nella sua configurazione	-

5.3 Mancanze/limitazioni conosciute

Non ci sono mancanze nell'implementazione dell'infrastruttura di security awareness. Tutta l'implementazione che è stata pianificata è stata eseguita con successo. Non ci sono particolari limitazioni di questo progetto, perché come scritto nei vari concetti e *abstract* è un progetto che mira anche alla scalabilità e alla compatibilità con più sistemi possibili. Le limitazioni personali elencate in precedenza non hanno intaccato l'implementazione del sistema in alcun modo.

C'è da considerare solo il fallimento del test sull'accesso da remoto. Ma l'errore è comunque arginabile semplicemente collegando la macchina fisica che ospita l'host virtuale ad un cavo ethernet in grado di fornire l'impostazione corrette della scheda da impostare in modalità *bridge*. In questo modo è possibile accedere alla VM in remoto attraverso, per esempio, il protocollo SSH.

6 Consuntivo

Nel capitolo 2.5 Pianificazione è presente il diagramma di Gantt preventivo, ovvero il diagramma che segue la pianificazione iniziale. Il diagramma preventivo, come dice il nome, prevede un certo numero di giorni per eseguire le diverse attività programmate. Chiaramente seguire perfettamente il programma non è mai possibile, salvo casi eccezionali.

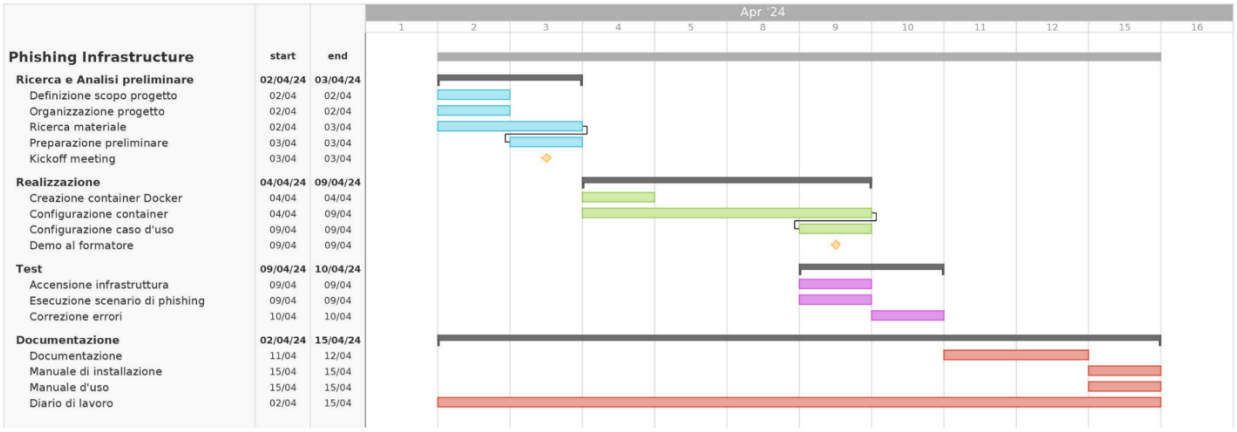

Figura 6: Diagramma di Gantt consuntivo

L'immagine rappresenta il diagramma di Gantt consuntivo che dista di qualche dettaglio rispetto al diagramma preventivo. Nonostante l'analisi preliminare, programmata per i primi due giorni del progetto LPI, sia stata coperta correttamente, la realizzazione dell'infrastruttura ha causato qualche problema, segnalato anche nella compilazione del diario di lavoro consegnato quotidianamente, che ha allungato di mezza giornata il programma, andando a sovrapporsi con la fase di test. Fortunatamente è stato organizzato il tempo rimanente di modo da potersi rimettere in linea con la programmazione iniziale. Infatti, dalla fase di testing in poi il diagramma rispetta quello visto nel capitolo della pianificazione.

Di seguito una tabella riassuntiva delle attività principali

ID	Attività	Giorni pianificati	Giorni effettivi	Note
01	Analisi	2 giorni	2 giorni	Nessuna nota
02	Realizzazione	3 giorni	3.5 giorni	A causa di alcuni problemi (segnalati nel Diario di lavoro, la fase di realizzazione si è allungata di mezza giornata andando a incrociarsi con il programma della fase di test
03	Test	2 giorni	1.5 giorni	Riduzione di mezza giornata nella fase di testing a causa dell'allungo della realizzazione. Test eseguiti senza fretta e con abbastanza tempo per correggiere gli eventuali errori
04	Documentazione	3 giorni	3 gionri	Nessuna nota

7 Conclusioni

La soluzione implementata durante questo progetto è perfettamente funzionante all'interno dello scopo che ci si è prefissati. Nonostante sia piena di strade per migliorare il sistema, rimane comunque un progetto andato a buon fine.

La soluzione implementata con Docker nel progetto di phishing offre una solida base per sviluppi futuri, come evidenziato nel capitolo *7.1 Sviluppi futuri*. Inizialmente, non ci sono particolari implicazioni significative da considerare. Tuttavia, è importante riconoscere il potenziale per miglioramenti e iterazioni future. L'impatto della soluzione implementata con Docker nel panorama più ampio potrebbe non essere rivoluzionario, non prevedendo un cambiamento radicale nel futuro della sicurezza informatica. Esistono già soluzioni consolidate per il phishing, ma questa implementazione ha sicuramente fornito una base solida per lo sviluppo di soluzioni personalizzate e mirate, che potrebbero dimostrarsi più vantaggiose in determinate situazioni.

Sebbene non sia previsto che questa soluzione cambi radicalmente il corso del futuro, ha comunque un significativo valore personale e di apprendimento. Il completamento del progetto in modo soddisfacente rappresenta un successo personale. Il processo ha offerto l'opportunità di acquisire conoscenze approfondite sulle tecniche di phishing e di riavvicinarsi alla tecnologia della containerizzazione attraverso Docker. Questo, in sé, rappresenta un successo importante nell'ambito della crescita personale e professionale.

7.1 Sviluppi futuri

Il progetto di infrastruttura per la consapevolezza della sicurezza ha fatto notevoli progressi nell'ultimo periodo, con l'obiettivo principale di educare e sensibilizzare le persone sui rischi della sicurezza informatica. Tuttavia, per garantire un impatto ottimale, è fondamentale affrontare alcune sfide chiave e implementare sviluppi futuri strategici.

Uno degli ostacoli principali riscontrati è stato il rischio che le e-mail inviate alle vittime finiscano nella cartella dello spam. Questo può compromettere l'efficacia della simulazione e ridurre l'attenzione degli utenti. Per superare questo problema, è necessario implementare tecniche avanzate di autenticazione e monitoraggio delle e-mail per garantire che vengano recapitate correttamente e siano visualizzate come legittime dagli utenti. Inoltre, per migliorare ulteriormente l'efficacia delle simulazioni, è stato considerato l'acquisto e la configurazione di domini dedicati. Questa strategia consentirebbe di condurre attacchi anche a distanza, senza la necessità di essere fisicamente presenti sul sito dell'attacco. L'implementazione di questa soluzione richiede un'attenta pianificazione e una gestione accurata dei record DNS per garantire l'efficacia e la sicurezza delle operazioni.

Guardando al futuro, c'è un potenziale per sviluppare un prodotto simile a Gophish, ma con una gamma più ampia di funzionalità e integrazioni. Questo nuovo strumento potrebbe essere parte di una suite di servizi progettati per l'hacking benevolo al servizio delle aziende. Tuttavia, va sottolineato che questa visione rappresenta l'ultima frontiera dello sviluppo del progetto e potrebbe essere considerata quasi irrealizzabile data la complessità e le implicazioni coinvolte. Dunque, il progetto di infrastruttura per la consapevolezza della sicurezza continua a evolversi per affrontare le sfide emergenti e sfruttare le opportunità di miglioramento. Con un approccio strategico e innovativo, è possibile continuare a promuovere la consapevolezza della sicurezza informatica e migliorare la difesa contro le minacce online.

8 Glossario

Di seguito è presente un glossario in ordine alfabetico dalla A alla Z dei termini utilizzati e/o utili alla comprensione di alcuni passaggi dell'implementazione dell'infrastruttura di security awareness.

Termine	Descrizione
Container	Un **container** Docker è un'istanza eseguibile di un'applicazione, insieme ai relativi file di sistema, librerie e dipendenze, isolata dal resto del sistema operativo in cui viene eseguita. I container Docker consentono di distribuire e gestire applicazioni in modo rapido, efficiente e riproducibile, fornendo un ambiente isolato e standardizzato per l'esecuzione delle applicazioni.
CSS	**Cascading Style Sheets**: un linguaggio di stile utilizzato per definire l'aspetto e la formattazione di documenti HTML, XHTML e XML. Attraverso la definizione di regole di stile, CSS consente di separare il contenuto strutturale di una pagina web dalla sua presentazione, consentendo una maggiore flessibilità e controllo sul design e la disposizione degli elementi sulla pagina.
Docker	**Docker** è una piattaforma open-source che semplifica la distribuzione e la gestione di applicazioni utilizzando container. Docker consente agli sviluppatori di creare, distribuire e eseguire applicazioni in un ambiente isolato, garantendo la portabilità e la consistenza tra diversi ambienti di sviluppo, test e produzione.
HTML	**HyperText Markup Language:** è il linguaggio di markup standard utilizzato per la creazione e la strutturazione dei contenuti delle pagine web. Attraverso l'utilizzo di tag e attributi, HTML definisce la struttura logica dei documenti web, inclusi testo, immagini, collegamenti ipertestuali e altri elementi multimediali.
JSON	**JavaScript Object Notation:** è un formato di dati leggero e basato su testo utilizzato per lo scambio di dati tra applicazioni. Grazie alla sua semplicità e alla sua struttura chiara, JSON è ampiamente utilizzato per rappresentare oggetti e dati strutturati in diversi contesti, come ad esempio le richieste e le risposte delle API web.
YML	**YAML:** è un formato di serializzazione dei dati leggibile dall'uomo utilizzato per rappresentare dati strutturati in modo chiaro e conciso. Nell'ambito di Docker, il file docker-compose.yml è utilizzato per definire i servizi, le reti e i volumi di un'applicazione multi-container, specificando le configurazioni e le dipendenze necessarie per orchestrare il deployment e la gestione dei container Docker.

9 Bibliografia

9.1 Sitografia

Ubuntu
- https://ubuntu.com/download/desktop

Docker
- https://docs.docker.com/engine/install/ubuntu/
- https://docs.docker.com/get-started/docker_cheatsheet.pdf
- https://docs.docker.com/compose/compose-application-model/
- https://docs.docker.com/compose/environment-variables/
- https://docs.docker.com/compose/networking/

Apache
- https://phoenixnap.com/kb/docker-apache
- https://httpd.apache.org/docs/2.4/mod/mod_proxy.html
- https://www.theserverside.com/blog/Coffee-Talk-Java-News-Stories-and-Opinions/How-to-configure-Apache-as-a-reverse-proxy-example

Postifx
- https://www.postfix.org/BASIC_CONFIGURATION_README.html
- https://www.postfix.org/DEBUG_README.html
- https://www.postfix.org/SMTPD_ACCESS_README.html
- https://www.postfix.org/LINUX_README.html

Gophish
- https://hailbytes.com/landing-pages-in-gophish/#:~:text=One%20of%20the%20easiest%20ways,URL%20appear%20in%20your%20editor
- https://docs.getgophish.com/user-guide/documentation/landing-pages
- https://docs.getgophish.com/user-guide/template-reference
- https://docs.getgophish.com/user-guide/additional-references
- https://docs.getgophish.com/user-guide/installation
- https://docs.getgophish.com/user-guide

HTML&CSS
- https://www.w3schools.com/tags/tag_meta.asp
- https://www.w3schools.com/html/html_links.asp
- https://www.w3schools.com/css/css_align.asp
- https://www.w3schools.com/howto/howto_css_login_form.asp

ALTRO
- https://github.com/cisagov/pca-gophish-composition
- https://www.unomaha.edu/college-of-information-science-and-technology/computer-science-learning-center/_files/resources/CSLC-Helpdocs-Vim.pdf
- https://eldermoraes.com/docker-basics-how-to-start-and-stop-containers/#:~:text=To%20stop%20one%20or%20more,or%20more%20containers%20to%20stop.
- https://proton.me/support/spam-filtering
- https://proton.me/support/what-is-difference-between-proton-domains#:~:text=%40proton.me%20is%20the%20default,yourusername%40protonmail.com)
- https://www.unomaha.edu/college-of-information-science-and-technology/computer-science-learning-center/_files/resources/CSLC-Helpdocs-Vim.pdf

9.2 Indice delle immagini

9.3 Indice delle tabelle

10 Allegati

Elenco completo degli allegati

- Documento Abstract PDF
- Manuale di installazione PDF
- Manuale d'uso PDF
- Diario di lavoro PDF