

Social Engineering Attacks deployed using Docker Containers

Author: Oghenetega Awaritefe

Abstract

Using containers to construct and build a contemporary cloud architecture makes it easier to create, deploy, and run applications have transformed the software development and operations era. Docker is a tool that allows you to build, ship, and operate any application from anywhere. In addition, Docker enables you to share the same resources and works flawlessly across multiple operating systems. But, no matter how much the world changes, one threat remains constant: Social Engineering. Security analysts are hired by businesses to conduct frequent assessments on their networks and, in some cases, employees. This paper discusses social engineering, attacks, techniques, and strategies that apply to everyone. It also demonstrates how to build a docker container with all of the dependencies needed to run the social engineering tools available to penetration testers and researchers step-by-step.

Table of Contents

Abstract	2
Listing of Figures.....	4
1. INTRODUCTION.....	5
2. Literature review	6
3. Social Engineering.....	10
3.1. Social Engineering Attacks	11
3.2. Social engineering strategy	11
3.3. Social Engineering techniques	12
4. Docker	14
4.1. Docker Architecture.....	15
4.2. Docker Engine	16
5. Kali Linux and Social Engineering Tools.....	16
5.1. Social Engineering toolkit (SET)	16
5.2. Metasploit MSF	17
5.3. Wifiphisher	18
5.4. MSFPC (MSFVenom)	19
6. Methodology	19
6.1. Setting up the base container – Docker + Kali Linux	19
6.2. Installing Metasploit Framework and MSFPC (MSFvenom)	24
6.3. Installing Social Engineering Toolkit (SET)	25
6.4. Installing Wifiphisher.....	26
6.5. Creating and pushing Docker Image to repository.....	27
7. CONCLUSION	30
8. Project Management	30
8.1. Project Scheduling.....	31
8.2. Risk Management.....	31
8.3. The Social, legal and Ethical Consideration	32
9. Bibliography	33
References.....	33
APPENDIX A – Interim Progress Report and Meeting Records	38

Listing of Figures

Figure 1 - Virtual machines vs Containers	14
Figure 2 - Docker installation check	20
Figure 3 - Docker Hub Repository	21
Figure 4 - docker pull kalilinux/kali-rolling	22
Figure 5 - Starting docker image in reverse shell	22
Figure 6 - python version installed	23
Figure 7 - An image of Metasploit running on container	24
Figure 8 - Installing SEToolkit requirement using pip3	25
Figure 9 - Social Engineering Toolkit Main Menu	26
Figure 10 - Installing Wifiphisher with Python3	27
Figure 11 - Wifiphisher installed	27
Figure 12 - Image highlighting the Container Id for the created container	28
Figure 13 - New container image name	28
Figure 14 - Pushing container to docker hub repository	29
Figure 15 - Kali Social Engineering Tools ready for use	29
Figure 16 - Gantt chart	31

1. INTRODUCTION

Cybercrime is a massive threat to the economy, personal safety, and even the broader population, as it is a primary means of terrorism (Breda et al., 2017). Before the Covid-19 pandemic in 2020, cybersecurity incidents have been increasing in number and severity at an exponential rate. Unfortunately, not all organizations and users have yet to implement defences to deter criminal intent to strike entirely. Significant corporations worldwide have traditionally devoted substantial resources and time in safeguarding their networks and staying current on industry trends.

They invest in software, hardware, and security protocol, all in the effort to fight cyber-crimes and protect the data they hold. However, this has not stopped hackers, as they often devise new cyberattack techniques every day that could involve a human user. Penetration attacks that are social rather than technical are among the most well-practiced and effective; they are so effective that these exploits play a critical role in supporting many cyber-attacks. This is because most sophisticated cyberattacks rely on human factors as they target the weakest link (Zhang et al., 2021).

Regardless of the extent to which machines and networks are automated today, there is not a single computer system in the world that is not reliant on human interaction at some point. For example, it may be to carry out maintenance or get information from the computer. A hacker with knowledge about social engineering will try to identify these weak links in companies and get information from them most of the time. Hackers typically begin by examining an organization's complete network infrastructure to gather as much information as possible and attack open ports or vulnerabilities (Aldawood & Skinner, 2020).

Given sufficient effort, any system can be compromised, so most organizations conduct penetration testing at least annually. This enables a penetration tester or security analyst to examine the functional features of a system to determine its vulnerability to network security and intrusion threats, as well as its protection mechanisms against these assaults (Vats et al. 2020). These tests ascertain the genuine risk, separating it from the scanner and assigning a business risk value to each asset and the organization's brand image. It's not about risk; it's about exposure (Velu & Beggs, 2019). Because the penetration test is conducted only after consent is obtained, it is considered ethical (Hatfield 2019).

While the majority of penetration testing is directed at systems, social engineering attempts are directed at individuals. Organizations can assess which employees are vulnerable to social engineering attacks and design a focused security awareness training program. Confident security analysts may choose to conduct social engineering as an add-on rather than part of the broader penetration test. This is because you risk upsetting some people and getting into problems if you begin attempting to obtain physical access to execs or sending trojans to them. Only a few penetration testers specialize in social engineering, and most of them work for large penetration testing firms (Edwards et al., 2017).

Containerisation is a technology used by a range of businesses, including Google, Amazon, and IBM, as well as for research endeavours to deliver a collection of services quickly and reliably. Docker is used to ensuring a container is built again and again to maintain compliance and is easy to update when changes occur in projects.

The scope is limited when you think about the people who know Social Engineering, the people who use Kali Linux as their day-to-day OS, and the people who have the knowledge and use the Kali Linux OS daily. This is because many people use Windows or macOS and will preferably run Kali from a virtual machine like VMware.

The paper's main contribution is to build a docker container bundled with all the dependencies required to run the social engineering tools available to penetration testers and researchers. This container can then be downloaded by anyone and further configured to suit their needs, be it additional tools. The reason for choosing to use a container is that Kali Linux has many tools we do not need for Social Engineering. We want a lightweight application with just limited tools. The beauty of using containers with just the specific tools you want is packaged nicely, and you can deploy it on whatever OS you use flawlessly. This makes our scope more significant as many people would benefit from a lightweight tool that will not dry up resources.

Organizations need people to carry out social engineering, and they mostly always go for people with the required skill set, which can be expensive to run regularly. It could lead to social consequences, but organizations can have their front desk security analyst go up a bit on hierarchy but learning and deploying the container created. Businesses tend to divide skill sets amongst employees to save costs.

The paper also sheds light on social engineering techniques used in this day and time. The information social engineering, attacks, techniques, and strategies n will be helpful to everyone as it provides a basic understanding of what to expect in such situations.

2. Literature review

There are many studies to show how penetration testing on an organization's network infrastructure is done and what tools to use to exploit the vulnerabilities found.

Gunawan et al. (2018) described Kali Linux history, setup, and installation and configured a vulnerable server for testing purposes. They used popular techniques used for the test, including SQLi, XSS, LFI, RFI, DDos, MITM, and zero-day vulnerabilities were reviewed. This article also discusses penetration testing, security analysis, and security auditing. Shebli & Beheshti (2018) examined the elements to consider when doing a penetration test, the process, and the most often used tools and software for conducting the test. In addition, the paper covered the function of an information security management system (ISMS) and the professional, ethical, and technical competencies required to conduct a penetration test.

Denis et al. (2016) conducted a thorough investigation using the tools within Kali to ascertain the various aspect of penetration testing, attack methodologies, and defence strategies. They performed traffic sniffing, hacking of smartphones, Bluetooth, WPA-protected WIFI, and many more. They clarified principles to aid in the comprehension of penetration testing and demonstrated several penetration tests utilizing private networks, devices, and traditional virtualized systems and tools. Vats et al. (2020); B L V Vinay Kumar et al. (2016); Antunes & Vieira (2014) all used tools to conduct penetration testing across networks or web services, discussed techniques, vulnerabilities, detection, and prevention.

A lot of researchers have spoken about the benefit and approaches to penetration testing. However, there seems to lack substantial research that utilizes social engineering to increase the output of penetration testing. This may be due to the extensive or costly thing that it takes to conduct these studies.

Edwards et al. (2017) interviewed penetration testers to understand better the types of social engineering tactics utilized and how open-source intelligence (OSINT) data aids these attacks. They further explained how employee profiles can be linked across numerous online social networks once they've been discovered, allowing us to gather additional valuable information for effective social engineering assaults on real critical organizations. Finally, they offered a series of countermeasures based on findings, including an automated social engineering vulnerability scanner that organisations may employ to assess their exposure to social engineering assaults originating from open-source intelligence. Syed (2020) discussed the concept of social engineering and delved more into the life cycle of an attack. This consists of information gathering, developing relationships, exploitation, and execution. Syed also explained modern techniques used in carrying out attacks and provided security countermeasures through education, awareness, training, and securing your devices and services. Abass (2018) did similar work to Syed because his paper examines the effect that contemporary social engineering has on organisations and individuals. It outlines the various categories of Social Engineering, which may involve technological or human deception and sometimes the combination of both and how the attacker exploits human ignorance and behaviour. Additionally, it discusses social engineering's direct and indirect attacks and the defense mechanism against them. A significant point in his conclusion is that since people are the weak point in most episodes, the same people will be the best tool to defend against attacks.

Dimkov et al. (2010) performed 14 live penetration tests over two years, trying to prove methodologies that tackle the problem of how to perform a physical penetration test using social engineering most respectfully. The goal was to gain physical assets from the premises of the organization. Other employees may stop trusting the victim of the attack when they let an intruder into the office. They underlined that personnel should not be agitated, uneasy, or exposed to dangers during the penetration test because they may get unhappy with the company or possibly begin legal action. The two methodologies proposed are Environment Focused (EF) and Custodian Focused (CF) methods. In the EF method, the security officer, the contact (the one who contacted the penetration tester), and the custodian (one who owns the asset) are made aware of the social engineering test been carried out on the other employees. The CF methodology involves the security officer, the contact, and the penetration tester. They concluded that the CF methodology is less considerate of the custodian than the EF methodology. The custodian is duped and may become disturbed when she discovers the asset has been stolen. The EF approach does not maintain confidence between employees, the organisation, and the custodian. The CF technique safeguards the relationship of trust between the custodian and the employees and the relationship between the employees and the organisation. The trust relationship between the custodian and the contact person, on the other hand, maybe harmed. However, these papers do fail to talk about ethics concerned with a social engineering penetration test. While technically feasible, there are ethical and moral considerations to make in this case. Individuals may be singled out in this manner to emphasise focused

mitigation methods and to alert an individual to difficulties they may be unaware of (Edwards et al., 2017).

Hatfield (2019) discussed the virtue ethics analysis of social engineering in penetration testing. Employees cannot be told they are being tested, reflecting on the accuracy assessment of a genuine penetration test. Therefore, penetration testing firms must cultivate a robust ethical training program that governs their use of social engineering. Hatfield argues that previous researchers like Fulton et al. (2013); Abass (2018) have either misconstrued or overlooked virtue ethics, a form of ethical reasoning that conceptualises moral action in terms of habits of behaviour acquired through and embedded in social relationships.

Fulton et al. (2013) mention the necessity of white hat hacking, including social engineering techniques, and the importance of ethical training for penetration testing, but does not examine social engineering ethics directly. Orchestrating a full penetration test is striking the best balance between the conflicting requirements. If the balance is not achieved, the test might either not fully assess the organization's security or might harm the employees (Dimkov et al., 2010).

Alharthi & Regan (2021) developed a taxonomy outlining the main social engineering objectives and defense mechanisms. Next, the paper offered a customizable and extensible model of formal SE-IPs for companies to use. To measure employee awareness of social engineering defense mechanisms, the authors designed two survey instruments. The first one is suitable for this purpose, while the second can measure SE-IPs incorporation level in an organisation. To collect data from employees in different industries, the authors conducted a survey, analysed the results, and presented them. The researchers and practitioners in the field of cybersecurity also have access to the dataset and others who could duplicate or enhance the work.

A survey carried out by Bob Watson (2019) shows that of the 32.6 million in employment, around 1.7 million people reported working mainly from home, with approximately 4.0 million working from home in the week before being interviewed for the survey. Things changed in significant ways due to COVID-19. Sarginson (2020) talked about organisations forced to work remotely within new environments, implement new security measures that can be unfamiliar to employees, and increase sophisticated threats like phishing attacks. His solution will be to implement multi-factor authentication (MFA) which provides a valuable additional layer of security compared to two-factor authentication (2FA) – through memorable words or SMS one-time passwords (OTPs). 2FA and OTPs can still be susceptible to phishing attacks. MFAs could be software-based MFA that texts or emails a code or a hardware-based MFA like a security key.

(Saleem & Hammoudeh, 2017) provided in-depth research of possible social engineering countermeasures found that could help secure enterprises from possible attacks. They reviewed various case studies, which demonstrate that security awareness is the most crucial tool in the fight against Social Engineering. The report features a thorough evaluation of previously published articles on the issue of Social Engineering prevention, including previously unpublished essays that have covered issues surrounding Social Engineering and

other published research that cover the topic of social engineering awareness. The researchers also discussed and assessed several options for ensuring that staff members who operate in an office setting receive security awareness training and reminders.

The various research all tries to provide methods for reducing the likelihood of social engineering attacks occurring. On the other hand, organizations will benefit by combining all the defense measures indicated in each report to achieve the highest possible level of protection and effectiveness.

Kolli et al. (2018) used reverse TCP payload to perform remote hacking on an older version of a popular operating system. They made use of an open-source tool called Armitage, which provides a graphical user interface for Metasploit. The attacker then uploads the payload generated into a server to carry out the attack. The link to the payload is sent via a legitimate-looking email and with the assistance of the Social Engineering Toolkit (SET). This attack establishes a backdoor. They ran two virtual computers concurrently: Kali Linux (the attacker's operating system) and Windows 7. (target). Kali already has Armitage and Metasploit installed. Their attack was described in detail. The experiment was conducted by exploiting a previously known vulnerability in the Windows 7 system firewall, which will no longer exist after the software has been upgraded.

Container technology has grown in prominence because of its ability to deliver near-native performance in cloud environments (Ruan et al., 2016). Containers can be categorised into application containers (e.g., Docker) and system containers based on their design objectives and underlying implementations (e.g., LXC).

The impact of various container platforms (Docker, LXD, and LXC) on the performance of various TCP services was investigated in this study Putri et al., (2020). They used overall performance measurements to evaluate the system performance of each container to that of the native system without any container solution. The study examines the three most often used PaaS: FTP server, web server, and mail server. Overall performance statistics show that LXD outperforms Docker and LXD by 90.5 percent. These findings suggest that adding LXD to LXC can improve overall Linux container system performance. LXD has improved its system performance and has shown that it can improve overall performance.

Felter et al. (2015) examined the performance of standard virtual machine (VM) deployments. They compared it to Linux containers, using KVM as a sample hypervisor and Docker as a container manager. Their findings indicate that containers perform as well as or better than virtual machines in almost all circumstances. However, for containers to be generally accepted, they must offer benefits in addition to steady-state performance. They found that the mix of convenience, speed of deployment, elasticity, and performance will likely become increasingly enticing in the near future.

Docker container technology has attracted academics' attention due to its ease of deployment and outstanding performance. Lu & Chen (2017) compared traditional virtualization (e.g., VMware, Xen, and KVM) to a docker architecture and component. They

also analysed the security of Docker quantitatively. They summarized the various penetration testing techniques in the docker environment focusing on the denial of service, container escape and side-channel attack. However, they realised that the security risk of Docker is higher than that of the traditional virtualization technology even though Docker has more performance advantages (Sharma et al., 2016), (Morabito et al., 2015).

Perrone and Romano (2017) built a Docker Security Playground (DPS). It was designed from the start as a hands-on tool for studying network security. They created a virtual network security lab, which provides a remote infrastructure for simulating susceptible networks via virtualization techniques. A user needs to install Docker and git in order to sync with the given Dockerlabs. This means they have a fully functional management platform at their disposal for conducting network security experiments. DPS is a continuous work in progress since they want to make it as rich as possible in terms of available scenarios and instances. Vats et al. (2020) wrote a paper that examines Docker from a security standpoint and examines how a penetration tester should evaluate the security of Docker-based systems. They introduced two attack models: container escapes and Docker daemon attacks. In container escapes, the attacker assumes the role of a process running within a container. In Docker daemon attacks, the attacker takes the perspective of a process operating on a host with Docker installed. Known vulnerabilities in Docker was looked upon, and they described how to identify relevant attack model during a penetration test. They end by offering a checklist that summarises the research in the form of questions that a penetration tester should ask during an evaluation of a target system that uses Docker.

3. Social Engineering

There have been many definitions of social engineering. Mouton et al., (2014b) defined The term "social engineering attack" refers to the practise of communicating directly or indirectly with a target, a channel, a goal, one or more regulatory principles, and one or more targets.

Hackers use social engineering attacks that heavily rely on human interaction and frequently entail manipulating people into violating standard security procedures and best practices to gain unauthorised access to systems, networks, or physical locations or obtain financial gain. The main target for hackers includes the elderly, children, and organizations. The elderly are frequently targeted by attackers, as many older adults struggle to adapt to modern technology, and they often live at home with not a lot of support.

Threat actors use social engineering techniques to conceal their true identities and motivations by posing as trusted individuals or information sources. The goal is to persuade, manipulate, or trick users into disclosing sensitive information or granting access to sensitive areas within an organisation. Numerous social engineering scams prey on people's willingness to assist or their fear of punishment. For instance, the attacker could pose as a co-worker experiencing an emergency requiring access to additional network resources. Social engineering is a frequently used attack technique by attackers because it is easier to exploit people than it is to discover a network or software vulnerability. It is a common first

step in a larger attack campaign, wherein hackers attempt to gain access to a system or network and to steal sensitive data or install malware.

3.1. Social Engineering Attacks

Physical, technical, and social-based approaches are three categories of attacks that can be used in social engineering.

Physical approaches: The attacker attempts to physically enter an organization's physical premises to access sensitive data and internal systems. Additionally, real-time evidence of a general lack of security best practises is gathered, such as clear desk policies, workstations that are left logged on and unattended, and confidential reports left out for anyone to read.

Technical approaches: The evolution of the internet and the growth of social media has led to many technical attacks been carried out over the internet. Attackers have a vast amount of knowledge on their victims using a simple search engine, and a majority of these victims are not aware that they are freely providing an abundance of personal information on their social networking sites.

Social approaches: Here, attackers make use of persuasion to manipulate their victims. These methods could include claimed authority. To increase the likelihood of success, perpetrators frequently attempt to develop a relationship with their potential victims. Because they involve human interaction, these attacks are the most dangerous and successful attacks (Patil & Devale, 2016).

3.2. Social engineering strategy

Social engineering attacks differ from one another, which is why Milosevic (2013) combined multiple processes of the social engineering attack consisting of different phases and summarized it as follow:

- **Pre-engagement interactions:** The first phase involves the attacker conducting surveillance to identify a victim who possesses the necessary information/knowledge to carry out an attack.
- **Intelligence gathering:** In this phase, the attacker tries to get as much information as possible to carry out an attack. This could be through the use of social media or information gathering tools and websites.
- **Pretexting and exploitation:** The attacker builds a relationship with the victim using the information attained from the previous phase to get even more information.
- **Post-exploitation and exit:** Return to a previous phase if necessary to continue the attack chain until the final information is obtained and evaluate the attack and information collected. The attacker then exits without a trace.

To carry out the social engineering strategy, an attacker may employ a combination of a few social engineering techniques and operators. Examples of these techniques include baiting, phishing, spear phishing, dumpster diving, pre-texting, tailgating, water-holing, honey trap, vishing, quid pro quo, scareware, rogue software, SMS phishing, and diversion theft.

3.3. Social Engineering techniques

To prevent this type of crime, it is essential to understand the various attack vectors that can be used and how they relate to the different attack categories.

- **Baiting:** Baiting is a form of physical social engineering attack that involves dangling something in front of a victim to compel them to act. It can be delivered through a peer-to-peer or social networking site in the form of a movie or application download, or it can be delivered through a USB drive labelled "Office staff pay increase 2021" that is left out in a public place for the victim to discover, among other things. Once the device is used, or a malicious file is downloaded, the victim's computer becomes infected, allowing the criminal to control the network and the victim's information.
- **Phishing:** It is the act of sending an entity bulk email that attempt to bypass spam filters and appear to be legitimate with the sole purpose of duping the victim and obtaining sensitive information such as usernames, credit card information, passwords, or installing malware.
- **Spear Phishing:** This is like phishing attack but is directed at a specific individual or organization. Spear phishing attacks require the attacker first to gather information about this potential victim. The success rate of a spear-phishing attack is high if done correctly.
- **Dumpster diving:** This is a social engineering attack in which the attacker sifts through the trash left by individuals or organizations in search of items containing sensitive information that can be used to infiltrate the organization's network and steal information. Sensitive information can be passwords or access codes written on sticky notes or scraps of paper.
- **Pretexting:** In a pretexting social engineering attack, the attacker creates a made-up scenario to engage a victim by posing as a legitimate and rusted identity as in a phishing attack. False motives are usually based on actual knowledge of the victim (such as their date of birth or national insurance number). They are used in an attempt to obtain even more information from the victim. Pretexting can also be used to discover security vulnerabilities in an organization's IT infrastructure or to gain unauthorised access to that organization's IT infrastructure.
- **Tailgating:** Tailgating attacks, also known as piggybacking attacks or physical access attacks, consist in gaining access to a restricted area or building by following someone who has been granted access to that area or building. This attack is predicated on the assumption that the person who has legitimate access to the building will be courteous enough to hold the door open for the person behind them, assuming they are permitted to be in the building in question.
- **Water-Holing:** This technique takes advantage of websites that people visit regularly and have come to trust and use regularly. The attacker will gather information about a targeted group of individuals to determine what websites they see and then test those websites for security flaws, as described above. When the victim arrives at the waterhole, the attackers wait for him.

- **Honey trap:** In this attack technique, the attacker pretends to be an attractive person and instigates a type of relationship. At the same time, try and gather information from the unsuspecting victim.
- **Vishing:** It is a telephone social engineering attack to gain access to sensitive information such as credit card numbers or authentication credentials and is also known as voice phishing.
- **Quid Pro Quo:** In Latin, this means "something for something." The attacker pretends to have something beneficial to the victim, which could help with some IT situation and, in return, get information from the victim or have them type in commands to run malware.
- **Scareware:** Pop-ups informing users that their machine is infected with viruses are a type of social engineering threat that commonly shows up, invoking fear in the victim. This tricks the victim into believing that the attacker is there to help and downloads
- **Rogue security software:** It is a type of computer malware that deceives or misleads users into paying for a phony or simulated malware removal.
- **SMS phishing:** is a technique in which scammers send text messages that mimic multi-factor authentication requests and direct victims to malicious web pages that collect their credentials or install malware on their phones. It is known as SMS phishing or SMS spoofing. Since the start of the COVID-19 pandemic, there have been an increase in this type of attack telling people to put in their information to access the COVID-19 grant from the UK government.
- **Diversion Theft:** The attacker convinces a delivery driver or courier to travel to an incorrect location or deliver a parcel to someone other than the intended recipient, thus intercepting the package.
- **Robocall attack:** These types of attacks are frequently carried out over phone calls. An automated voice down the other line calls and tells you your family was involved in an accident. It could be a computerized voice claiming to be from gov.co.uk contacting regarding your taxes.

A social engineering attack's originator (operator) can be a human or software-based attack (Krombholz et al., 2015). Because of the lower capacity of a person conducting an attack than a computer conducting an attack, a person conducting an attack is limited in the number of targets that can be attacked. Attacks can be carried out in the following ways:

- Emails are the most frequently used method of conducting phishing and reverse social engineering attacks.
- Social engineers frequently use the telephone and voice-over IP as attack channels to coerce their victims into providing sensitive information.
- Websites, if combines with an email, can be used to perform phishing attacks. Water-holing attacks are most frequently carried out through websites.
- Social media platforms provide numerous opportunities for social engineering attacks. They make it easy for attackers to conceal their identity and harvest sensitive information due to creating fake identities like on Facebook or Twitter. A lot of honey trap attack happens through social media which connects everybody to everybody. These platforms also make instant messaging applications, which are very popular among social engineers.

4. Docker

There has been an increase in the development and use of virtualization technologies in the past years. However, a virtual machine (VM) can only run one operating system at a time, so it will require a lot of resources and multiple operating systems and VMs to carry out a production test (e.g., building an application). Linux containers were then designed to solve this situation as they run remotely on an operating system's kernel.

Virtualization enables excellent resource optimization and creating several environments on a single computer (Perrone & Romano, 2017). It is also efficient in load balancing and multi-tenancy. In addition, virus, Worms, and other threats that could potentially destroy a real-world environment can be run inside a virtualized network infrastructure.

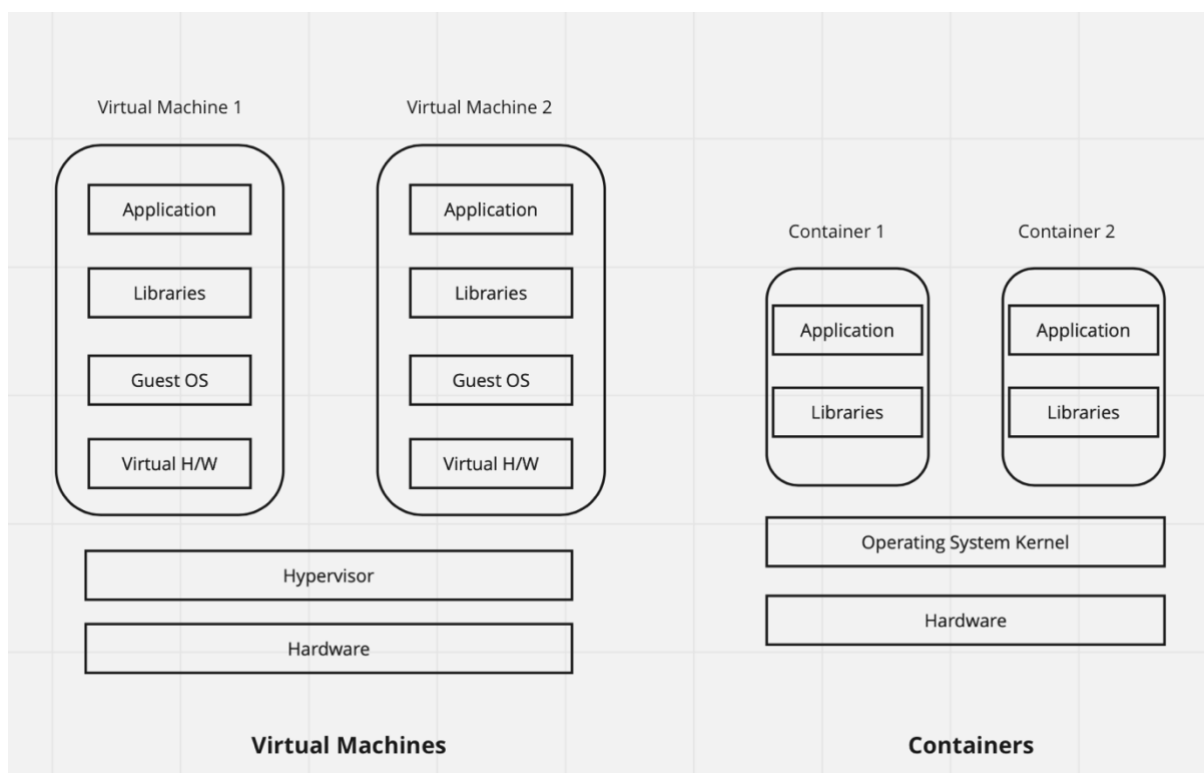


Figure 1 - Virtual machines vs Containers

Docker is a free and open-source platform for building, distributing, and executing apps. Docker enables the packaging and execution of applications within a loosely isolated environment known as a container. When it comes to software delivery, Docker is quite advantageous because it typically allows for the separation of the application from the existing infrastructure that is used to deliver the software (Patil, 2021). Thus, it becomes simple for one to transfer data securely in a container, and the data arrangement will not be disrupted. In addition, Docker is great for continuous deployment and testing as there is flexibility with patching. This is very useful for DevOps teams.

Docker is a project founded by Solomon Hykes, which he released in March 2013. Since then, it has surpassed all other technologies in its field and has established itself as the greatest.

4.1. Docker Architecture

Docker's architecture comprises four primary components: Docker client and server, Docker containers, Docker images, and Docker registries. A summary of each of these components is provided below.

- **Docker client and server:** Docker operates on a client-server architecture. The Docker client communicates with the Docker daemon, responsible for creating, executing, and distributing Docker containers. Docker client and daemon can coexist on the same system, or a Docker client can communicate with a remote Docker daemon.
- **Docker Image:** A Docker image is a file that contains instructions for running code within a Docker container. Docker images, like a template, serve as a set of instructions for building a Docker container. Additionally, Docker images serve as a starting point when using Docker. In virtual machine (VM) environments, an image is similar to a snapshot. A Docker image comprises the application's source code, libraries, tools, dependencies, and any other files required to run the application. When a user executes an image, it transforms into one or more instances of a container. Thus, it is a copy of a Docker container at a particular moment.
- **Docker Registry:** Docker Hub, Docker's public registry instance, is the default communication channel for the Docker engine. A Docker registry is a repository for storing and distributing Docker images with unique names. There may be different versions of the same image, each with its own set of tags. Docker users can download images locally and contribute new ones to the registry via the registry.
- **Docker container:** A Docker container image is a small, standalone software package that contains everything needed to run an application, including code, runtime, system tools, system libraries, and settings. When images run on Docker Engine, they become containers. The key distinction between a container and an image is the existence of a top writable layer. This writable layer stores all writes to the container that adds new or modifies existing data. When a container is removed, it also deletes the readable layer. However, the underlying docker image is not altered. Due to the fact that each container has its own thin writable container layer in which all changes are preserved, several containers can share the same underlying docker image while maintaining their data state.

The isolation and security features enable you to operate several containers concurrently on a single host. These containers are small and contain everything necessary to run the application, eliminating relying on the host's installed software. The apps contained within these containers can be used without modification in any location. Containers can access software resources with the host (such as libraries), minimising code duplication. These containers boot fast because they lack a kernel and some system libraries, making them extremely lightweight. Furthermore, the hard drive footprint is substantially smaller than the usual virtual machines because we start with a lightweight image and add the necessary tools and because containers do not create discs or require hardware virtualization.

4.2. Docker Engine

The Docker Engine enables the development, assembly, shipping, and execution of applications by utilising the following components:

- Docker Daemon: A background process handles Docker images, containers, networks, and storage volumes in the background. The Docker daemon is always on the lookout for Docker API queries to handle.
- Docker Engine REST API: An API (Application process interface) is used by applications to communicate with the Docker daemon; the Docker Engine API is a RESTful API that can be accessed via an HTTP client such as curl, or via the HTTP library included with most modern programming languages.
- Docker CLI: It is the Docker daemon's command-line interface. It significantly simplifies the management of container instances, which is one of the primary reasons developers love Docker.

5. Kali Linux and Social Engineering Tools

A broad spectrum of security experts uses Kali Linux, the world's most powerful and popular penetration testing platform and based on a Debian-Linux distribution. These professionals work in various disciplines, including penetration testing, digital forensics, reverse engineering, web applications, password attacks, and vulnerability assessment. Kali comes with over 600 pre-installed penetration testing tools, a lot of which may not ever be used depending on which career path you take. In addition, the Kali Linux distribution includes various metapackages that allow us to quickly and efficiently install subsets of utilities based on our specific requirements.

This section provides an overview of some social engineering penetration testing tools we will be using in our experiment. It also details their applications and the methods through which they attack vulnerabilities. The tools introduced in this paper are mentioned below. Each tool is discussed in detail for those with only fundamental knowledge of the subject matter.

- Social engineering toolkit (SET)
- Metasploit MSF
- Wifiphisher
- MSFPC (also called MSFVenom payload creator)

5.1. Social Engineering toolkit (SET)

SET is a python-driven open-source application that is intended for penetration testing in the context of social engineering. The toolkit's assaults are designed to be targeted and focused on a specific person or organisation during a penetration test. It was written and designed by Dave Kennedy, the founder of TrustdSec, and with a lot of help from the community, it integrates assaults that have never been seen in an exploitation toolset of this calibre. A variety of the attack vectors are predefined and customised. Additionally, SET integrates with the Metasploit framework, allowing for the delivery of Metasploit payloads via Social Engineering techniques. SET can run on different platforms (Windows, Linux, and

Unix). Furthermore, it can integrate with third-party modules and allowing for any configuration. modifications via the configuration menu. SET employs a variety of attack channels and techniques.

- **Infectious media creator:** One of the options available to SET is to infect a media device (USB/CD/DVD) with an autorun.inf file. If placed into any PC it automatically run a Metasploit payload if autorun is enabled. This is extremely advantageous when it comes to baiting technique of attack.
- **Phishing Attacks:** SET includes numerous phishing attack choices to assist you in determining the best method for your victim. You can send email messages containing harmful payloads to a small or big number of recipients. SET comes pre-formatted with phishing pages for famous websites like as Facebook, Twitter, Google, and Yahoo. Additionally, it allows you to impersonate someone else's email account by simply changing a few simple parameters, making it a very simple tool to employ.
- **Web Attack:** This module combines many attack methods to compromise the remote victim. It employs various attack techniques, including the Java Applet Attack and the Metasploit Browser Exploit, to deploy malicious payloads. In addition, Credential Harvester method is possible, which is helpful because it allows you to clone a website and harvest information from user and password fields. It also comes with TabNabbing, HTA Attack, Web-Jacking, and Multi-Attack techniques, all of which are aimed at tricking end-users into revealing their credentials.
- **Generate a Payload and Listener:** The fourth option on the main menu when you run the SET tool using your terminal allows you to create malicious payloads for Windows. Such payloads include Reverse TCP Meterpreter, Shell Reverse TCP, Shell Reverse TCP X64, and Meterpreter Reverse HTTPS. As the titles indicate, you'll be able to launch command shells, setup reverse connections, and tunnels, among other things.
- **Mass mailer attack:** This type of attack can be more effective if combined with a phishing attack. It lets you import a list of users and any person you wish to send an email to. It also allows you to use a Gmail account for your email attack, your own server, or an open relay to distribute large-scale emails.
- **QR Code Generator:** There are many websites online that helps you generate a QR Code. SET can also do this. Generating a QR code and redirecting whoever scans it back to a website we cloned using Credential Harvester Attack Method a be very effective.

SET also provide other options like Wireless Access Point and PowerShell Attack Vectors.

5.2. Metasploit MSF

Metasploit is a penetration testing platform for identifying and validating vulnerabilities and developing and executing exploit code over a remote target workstation.

This framework's advantage is that it enables the pairing of any attack with any payload. Furthermore, since it is an open-source framework, it is easily customizable and compatible with the majority of operating systems. Before Metasploit, pen testers were required to do

all queries manually, utilising a range of tools which may or may not have been supported by the system under test. They had to create their code by hand and manually introducing it into networks.

The following steps provide the framework for exploiting a system:

- Select and configure an exploit.
- Determine whether the target system is exploitable.
- Specify and configure the payload.
- Select an encoding strategy that prevents the intrusion prevention system from detecting the encoded payload.

Additionally, the Metasploit framework supports nearly 500 payloads, including the following:

- Dynamic payloads that enable testers to create distinctive payloads that bypass antivirus protection.
- Spawn shell payloads that allow users to execute programs or random commands against a host.
- Static payloads that enable port forwarding and inter-network communication.
- Meterpreter payloads enable users to seize control of device monitors via VNC, taking over sessions and uploading and downloading files.

Meterpreter makes it easy for an attacker to compromise a computer system by executing an invisible (or nondescript) shell on the compromised system and creating a communication channel between the attacker's system and the compromised system. However, Metasploit's multi/handler is required to intercept meterpreter payloads. Metasploit is frequently updated, and new exploits are included as soon as their creators make them available.

5.3. Wifiphisher

Wifiphisher is a rogue Access Point framework that enables lawful red team engagements and Wi-Fi security testing. It is a tool that attempts to obtain the password of personal accounts by tricking wireless networks. In this particular attack, brute force is not used. Instead, an ingenious way of bypassing security is employed. Captive portals and third-party login pages that use WPA/WPA2 passphrases as their entry requirements make it an easy option to gain credentials. In addition, the tool can scan neighbouring Wi-Fi access points and build a spoofed or cloned access point.

Anyone connecting to the evil twin-like open network is confronted with a seemingly legitimate phishing page requesting the Wi-Fi password to download a firmware upgrade, which is described as the cause for the Wi-Fi not working. This puts the attacker in a man-in-the-middle position.

Wifiphisher notifies targets immediately upon entering a password, allowing them to stall for time. After the captured password has been transferred, a misleading reboot timer will be displayed, as will a fake update page to give you extra time before you resend your captured password. It can be used to assess the security defences against Wi-Fi threats using social engineering.

5.4. MSFPC (MSFVenom)

Metasploit framework MSFvenom is a simple-to-use tool. MSFvenom streamlines the process of developing primary payloads and allows you to use your favourite payloads from the Metasploit Framework without having to go through the process of creating them yourself.

It is a mix of MSFpayload and MSFencode, and it combines the payload generator and encoding functionality into a single Framework instance. It was introduced on June 8th, 2015, and it has since replaced MSFpayload and MSFencode.

Now that we have the basic understanding of Docker, Kali, and the social engineering tools we want to use for attacks, this is how I will be utilizing a kali environment in a docker container within my studies. Bear in mind that not all Kali tools are easily installable on another operating system, which is why many utilise bootable USB sticks or virtual machines. This article discusses the use of the official Kali Linux container for Docker.

6. Methodology

All our tests were performed on a MacBook Pro running macOS Big Sur with 1.4 GHz Quad-Core Intel Core i5 and 16 GB of RAM.

The steps we took in creating and pushing our container can be summarized as follow:

1. Download docker to our system and install a Kali Linux operating system.
2. Install the necessary tools needed for social engineering attacks.
3. Save and give the created image a name.
4. Push image file to docker hub so it can be readily available for whoever needs it.

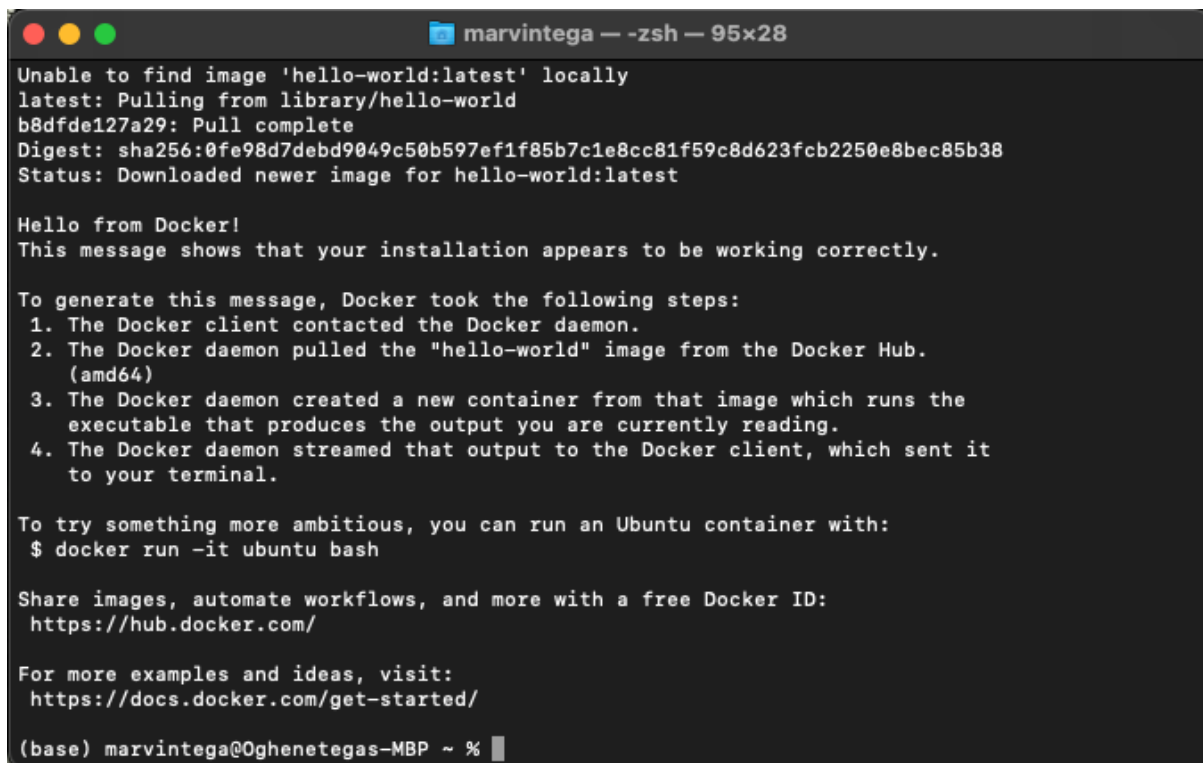
6.1. Setting up the base container – Docker + Kali Linux

1. The first step will be to download the docker for whichever operating system you will be using. Docker can be gotten from the official Docker website (<https://docs.docker.com/get-docker/>).

2. Once the installation is completed, we open our terminal. To check if our installation succeeded, we run the following command:

```
docker run hello-world
```

The following output will be revealed if everything goes well:

A screenshot of a terminal window titled 'marvintega - zsh - 95x28'. The terminal output shows the process of pulling the 'hello-world:latest' image from Docker Hub. It reports the image digest as 'sha256:0fe98d7debd9049c50b597ef1f85b7c1e8cc81f59c8d623fcb2250e8bec85b38' and confirms the download. The output then says 'Hello from Docker!' and 'This message shows that your installation appears to be working correctly.' It lists four steps: 1. Docker client contacted the daemon, 2. daemon pulled the 'hello-world' image, 3. daemon created a container from that image, and 4. daemon streamed the output to the client. It then suggests running an Ubuntu container with the command '\$ docker run -it ubuntu bash'. It also provides a link to Docker Hub for sharing images and a link to Docker documentation for more examples. The prompt at the bottom is '(base) marvintega@Oghenetegas-MBP ~ %'.

```
marvintega - zsh - 95x28
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
b8dfde127a29: Pull complete
Digest: sha256:0fe98d7debd9049c50b597ef1f85b7c1e8cc81f59c8d623fcb2250e8bec85b38
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

(base) marvintega@Oghenetegas-MBP ~ %
```

Figure 2 - Docker installation check

3. Next, we install our Kali Linux image file. It is a base image. Kali already provides an official Kali Linux Docker which is apt sources configured and with minimal base (i.e., it is free of tools, and you will need to download whatever tool you wish to run on the operating system). We go to docker hub (<https://hub.docker.com>) and search Kali Linux. The page also provides a command to pull this image directly from your docker terminal.

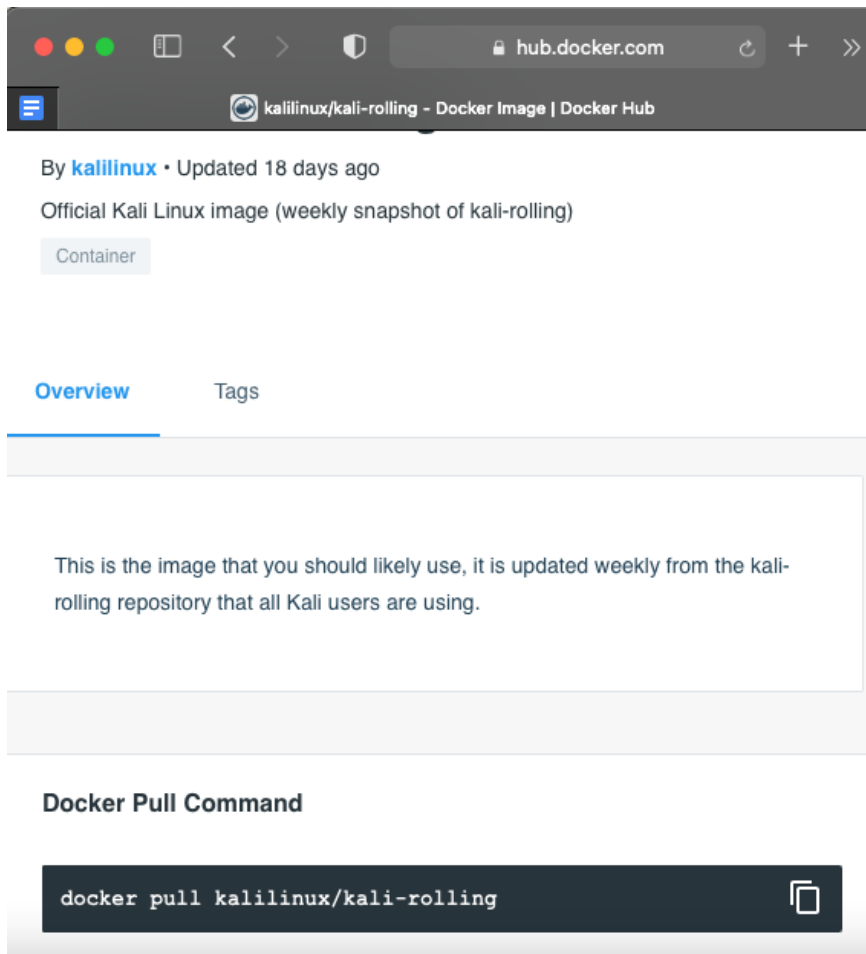


Figure 3 - Docker Hub Repository

Note that there are also other versions of Kali available on the hub. The term "base OS" refers to an image that contains a Linux distribution such as Ubuntu, CentOS, or Windows Server Core. When we build a new container, we add a new writable layer on top of the underlying stack of layers in the original docker image. All changes made to the running container, such as new files being created, old files being modified, or deleted, are written to this thin writable container layer.

4. Next, we go back to our terminal and pull the Kali Linux image with the following command:

```
docker pull kalilinux/kali-rolling
```

This is the output:

```

(base) marvintega@Oghenetegas-MBP ~ % docker pull kalilinux/kali-rolling
Using default tag: latest
latest: Pulling from kalilinux/kali-rolling
6423497f6670: Pull complete
Digest: sha256:328dffa42f3dc5424e87e21d68d6398fe1d9189d11e1c56102c7a07957c74c5
Status: Downloaded newer image for kalilinux/kali-rolling:latest
docker.io/kalilinux/kali-rolling:latest
(base) marvintega@Oghenetegas-MBP ~ %

```

Figure 4 - docker pull kalilinux/kali-rolling

5. We then run the Kali Linux image with the following command:

```
docker run -t -i kalilinux/kali-linux-docker /bin/bash
```

This will start a reverse shell with this image. and you will get the command line control of Kali Linux. The -t and -i options, which can alternatively be used in combination as '-ti', indicate that we want to use a tty and leave STDIN open for interactive processes.

```

(base) marvintega@Oghenetegas-MBP ~ % docker run -t -i kalilinux/kali-rolling /bin/bash
(root@5b045fa31718)-[/]
#

```

Figure 5 - Starting docker image in reverse shell

6. If there is a new version of packages available, now will be the best time to update it. We run the following command to update the packages database of Kali. If there is a new version of package itself, apt-get will download the information.

```
apt-get update
```

7. To upgrade all the packages that are already installed, we use the following command:

```
apt-get dist-upgrade
```

This will check for dependencies between the installed package's newer version and attempts to install new packages or uninstall existing ones on its own.

8. We also installed man-db and exploitdb packages using the following commands:

```

apt install man-db
apt install exploitdb

```

Man-db is utilised to display the user manual for each operation that may be executed via the terminal. Exploitdb is a repository for publicly available exploits and associated vulnerable software. It was created for penetration testers and vulnerability researchers to use. Installing man-db and exploitdb is optional .

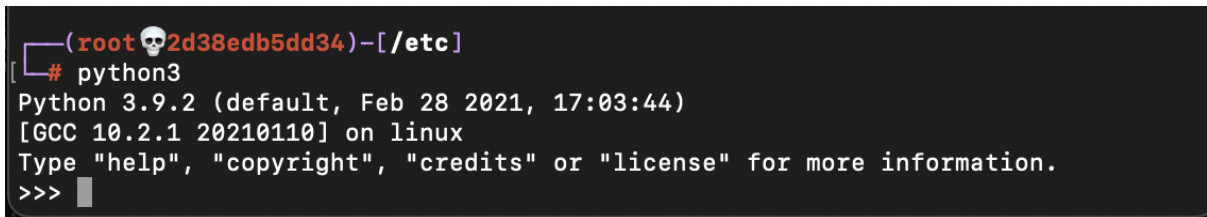
9. Some tools we will be installing will require python to run. For this we will be installing Python3 in our Kali Linux environment using the following command:

```
apt install python3-pip
```

It downloads the python3 toolkit and ask you to proceed with the installation when you type in 'y'. To check if python3 has been installed and what version, we run the next command:

```
python3
```

This is the result:

A terminal window with a dark background. The prompt is (root@kali:~) - [/etc]. The user has entered # python3. The output shows Python 3.9.2 (default, Feb 28 2021, 17:03:44) [GCC 10.2.1 20210110] on linux. It also displays instructions to type "help", "copyright", "credits" or "license" for more information. The prompt >>> is visible at the bottom.

```
(root@kali:~) - [ /etc ]
# python3
Python 3.9.2 (default, Feb 28 2021, 17:03:44)
[GCC 10.2.1 20210110] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Figure 6 - python version installed

10. The Curl command can be used for getting the source code of a website or download a content from a webpage. We use the following command to download the Curl library:

```
apt install curl
```

11. Numerous projects use Git to manage their files, and services such as GitHub, GitLab, and Bitbucket have made sharing and contributing code simple, beneficial, and effective.

Linus Torvalds created Git in 2005, he also developed the Linux kernels.

The git pull command is used to fetch new modifications from a remote repository. We use the following command to install Git:

```
apt install git
```

12. Other dependencies to be installed will be with this command:

```
apt-get install -y python3-setuptools libnl-3-dev libnl-genl-3-dev libssl-dev
```

This will be handy and help to avoid error when installing the wifiphisher tool.

13. We installed the systemctl command, which is a utility for inspecting and controlling the systemd and service manager systems. We used the following command:

```
apt-get install systemd
```

14. Another requirement mostly for the Metasploit framework will be to install the PostgreSQL as Metasploit relies on this database connection. PostgreSQL is a relational database that is designed to enable both relational queries (SQL) and non-relational (JSON) queries. PostgreSQL is an enterprise-class database that is free and open-source. This aids

developers in the development of fault-tolerant settings, as well as the protection of information integrity in general. We run the following command to download and install PostgreSQL:

```
apt install postgresql -y
```

You can proceed to start PostgreSQL on Kali Linux by executing the commands listed below:

```
service postgresql start
```

6.2. Installing Metasploit Framework and MSFPC (MSFvenom)

We now have a simple base version of Kali Linux running in Docker, which we can develop further by installing the social engineering tools we discussed previously. We will start by installing the Metasploit framework.

To install the Metasploit framework, we run the following command in terminal:

```
curl https://raw.githubusercontent.com/rapid7/metasploit-omnibus/master/config/templates/metasploit-framework-wrappers/msfupdate.erb > msfinstall && chmod 755 msfinstall && ./msfinstall
```

After the installation completes, we start Metasploit with the following command:

```
msfconsole
```

This is the output:

```
(root@2d38edb5dd34)-[/]
# msfconsole
[!] The following modules could not be loaded!..
[!] /opt/metasploit-framework/embedded/framework/modules/auxiliary/gather/office365userenum.py
[!] Please see /root/.msf4/logs/framework.log for details.

      .\$$$$L.,,==aaccaacc%#s$b.      d8,      d8P
      #$$$$$$$$$$$$$$$$$$$$$$$$$$$b.      `BP  d888888p
      '7$$$\'`"HHHH'IAA'^'.7$$$|D*"'`"      ?88'
d8bd8b.d8p d8888b ?88' d888b8b      d8P      ?8b 88P
88P'?'P'?'P d8b_,dP 88P d8P' ?88      .oaS##S*"'      d8P d8888b $whi?88b 88b
d88  d8 ?8 88b      88b 88b ,88b .os$$$$$* ?88,.d88b, d88 d8P' ?88 88P `?8b
d88' d88b 8b`?8888P' ?8b`?88P'.a$$$$$Q*"'`?88' ?88 ?88 88b d88 d88
      .a$$$$$Q*"'      88b d8P 88b`?8888P'
      ,s$$$$$Q*"'      888888P' 88n      _.,,ass;:
      .a$$$$$SP'      d88P'      .,ass%#$$$$$$$$$$$$$$$$$'
      .a####$SP'      _.,,-aqsc#$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$'
      ,a####$SP'      _.,,-ass#$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$'
      .a$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$'
      .a$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$'
      ,&$$$$$'
      11&$$$$$'
      .,;11&$$$'
      .,;1111&'
      .....;1111;.....
      `.....;?;?.....

      =[ metasploit v6.0.57-dev-
+ -- --=[ 2154 exploits - 1145 auxiliary - 367 post
+ -- --=[ 592 payloads - 45 encoders - 10 nops
+ -- --=[ 8 evasion

Metasploit tip: Adapter names can be used for IP params
set LHOST eth0
msf6 > |
```

Figure 7 - An image of Metasploit running on container

To install MSFvenom used to create payloads, we run the following command:

```
apt-get install -y msfpoc
```

or

```
git clone https://github.com/g0tmilk/mpc.git
```

6.3. Installing Social Engineering Toolkit (SET)

To install the Social Engineering toolkit we run the following command with git:

```
git clone https://github.com/trustedsec/social-engineer-toolkit setoolkit/
```

After we have pulled the file, we have to install it on our Kali. We start by changing directory to the downloaded folder and downloading the requirements needed for SEToolkit using the following command:

```
pip3 install -r requirements.txt
```

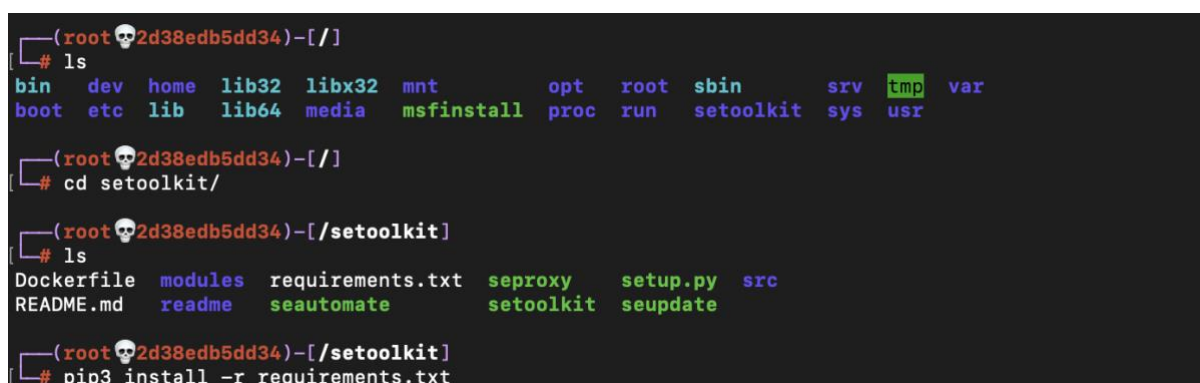
A terminal window screenshot showing the installation of SEToolkit requirements. The prompt is (root@kali:~) and the user is in the root directory. They run 'ls' and see a list of directories including bin, dev, home, lib32, libx32, mnt, opt, root, sbin, srv, tmp, var, boot, etc, lib, lib64, media, msfinstall, proc, run, setoolkit, sys, and usr. They then run 'cd setoolkit/' and are now in the /setoolkit directory. They run 'ls' again and see a list of files including Dockerfile, modules, requirements.txt, seproxy, setup.py, src, README.md, readme, seautomate, setoolkit, and seupdate. Finally, they run 'pip3 install -r requirements.txt'.

Figure 8 - Installing SEToolkit requirement using pip3

It's now time to install the requirements that was downloaded. We use this command:

```
python3 setup.py
```

This is how we install the SEToolkit on a Kali Linux operating system and this is the SET menu:

```

[---]      The Social-Engineer Toolkit (SET)      [---]
[---]      Created by: David Kennedy (ReL1K)      [---]
              Version: 8.0.3
              Codename: 'Maverick'
[---]      Follow us on Twitter: @TrustedSec      [---]
[---]      Follow me on Twitter: @HackingDave     [---]
[---]      Homepage: https://www.trustedsec.com    [---]
Welcome to the Social-Engineer Toolkit (SET).
The one stop shop for all of your SE needs.

The Social-Engineer Toolkit is a product of TrustedSec.

Visit: https://www.trustedsec.com

It's easy to update using the PenTesters Framework! (PTF)
Visit https://github.com/trustedsec/ptf to update all your tools!

Select from the menu:

1) Social-Engineering Attacks
2) Penetration Testing (Fast-Track)
3) Third Party Modules
4) Update the Social-Engineer Toolkit
5) Update SET configuration
6) Help, Credits, and About

99) Exit the Social-Engineer Toolkit

```

Figure 9 - Social Engineering Toolkit Main Menu

6.4. Installing Wifiphisher

To download Wifiphisher to our kali Linux, we will be using this command to clone:

```
git clone https://github.com/wifiphisher/wifiphisher.git
```

After downloading the tool, we change directory to that of Wifiphisher and use python3 to install the tool with this command:

```
python3 setup.py install
```

```
(root@kali:~/wifiphisher)-[  
# git clone https://github.com/wifiphisher/wifiphisher.git  
Cloning into 'wifiphisher'...  
remote: Enumerating objects: 3902, done.  
remote: Counting objects: 100% (12/12), done.  
remote: Compressing objects: 100% (10/10), done.  
remote: Total 3902 (delta 2), reused 7 (delta 2), pack-reused 3890  
Receiving objects: 100% (3902/3902), 12.46 MiB | 7.01 MiB/s, done.  
Resolving deltas: 100% (2473/2473), done.  
  
(root@kali:~/wifiphisher)-[  
# ls  
bin      dev      home     lib32    libx32   mnt      opt      root     sbin     srv      tmp      var  
boot     etc      lib      lib64    media    msfinstall  proc    run      setoolkit  sys     usr      wifiphisher  
  
(root@kali:~/wifiphisher)-[  
# cd wifiphisher/  
  
(root@kali:~/wifiphisher)-[  
# ls  
CHANGELOG          ISSUE_TEMPLATE.md  MANIFEST.in        bin      pylintrc    tests  
CODE_OF_CONDUCT.md LICENSE.txt         README.md          docs     setup.py    wifiphisher  
  
(root@kali:~/wifiphisher)-[  
# python3 setup.py install
```

```
Finished processing dependencies for wifiphisher==1.4
```

Figure 11 - Wifiphisher installed

At any point, you should be able to start the script by entering `wifiphisher` in a terminal window. It is vital to note that in order for this assault to be successful, you will require a wireless network adapter that is compatible with Kali Linux.

One technique to generate an image is to pull a Docker image, create a container from it, and then edit or change the container, such as installing our social engineering tools. It is essential you sign up on the docker website so you can push and pull any image. After adding all the tools to our base image, we exit Kali using the “exit” command.

```
docker ps -a
```

And this is the output:

```
(base) marvintega@Oghenetegas-MBP ~ % docker ps -a
CONTAINER ID   IMAGE                  COMMAND                  CREATED        STATUS              PORTS              NAMES
2d38edb5dd34   kalilinux/kali-rolling "/bin/bash"             5 hours ago   Exited (1) 12 seconds ago              funny_kilby
82d2bc062c2e   kalilinux/kali-rolling "/bin/bash"             6 hours ago   Exited (129) 5 hours ago              xenodochial_lalande
fae18d16ebbd   kalilinux/kali-rolling "/bin/bash"             6 hours ago   Exited (127) 6 hours ago              interesting_payne
950b53a0068d   kalilinux/kali-rolling "/bin/bash"             6 hours ago   Exited (129) 5 hours ago              silly_antonelli
b16e3675e831   hello-world           "/hello"                 7 hours ago   Exited (0) 7 hours ago              beautiful_clarke
5b045fa31718   kalilinux/kali-rolling "/bin/bash"             7 days ago    Exited (137) 5 hours ago              priceless_lumiere
8746873de2d4   docker101tutorial     "/docker-entrypoint..." 4 weeks ago   Exited (0) 4 weeks ago              docker-tutorial
f00601041852   alpine/git            "git clone https://g-"   4 weeks ago   Exited (0) 4 weeks ago              repo
```

Figure 12 - Image highlighting the Container Id for the created container

We obtained the container ID for the dicker image we are working with using the previous command.

Next, we use docker commit to create a new image of the edited container we initially downloaded with the following command:

```
docker commit 2d38edb5dd34 marvvv26/kali-social-engineering-tools
```

With the above command, we know what the container ID is. I created a docker account and my username is "marvvv26". Kali-social-engineering-tools is the name I would like the image to be called. Note that the image name must all be small letters.

To view all the locally stored docker images we use the following command:

```
docker images
```

Output:

```
(base) marvintega@Oghenetegas-MBP ~ % docker images
REPOSITORY          TAG         IMAGE ID      CREATED        SIZE
marvvv26/kali-social-engineering-tools   latest     bae66a5af5ea  8 minutes ago  2.07GB
```

Figure 13 - New container image name

For the final part, we push our created docker image to the docker hub repo. We first make sure we are logged into docker on the terminal using this command:

```
docker login
```

This asks you to put in your username and password. The reason for this is that the image will be posted under your account in the hub.

To push our docker image, we use this command:

```
docker push marvvv26/kali-social-engineering-tools
```

This may take a little while depending on the size of the image you created.

```
Create a new image from a container's changes
(base) marvintega@Oghenetegas-MBP ~ % docker push marvvv26/kali-social-engineering-tools
Using default tag: latest
The push refers to repository [docker.io/marvvv26/kali-social-engineering-tools]
44ca157d3f20: Pushing [=====>] 1.063GB/1.944GB
21f3fc525322: Mounted from kalilinux/kali-rolling
```

Figure 14 - Pushing container to docker hub repository

Make sure to use reliable internet when trying to push the container to the repository. After a couple of minutes, we successfully pushed our version of Kali Linux for Social Engineering on the docker hub.

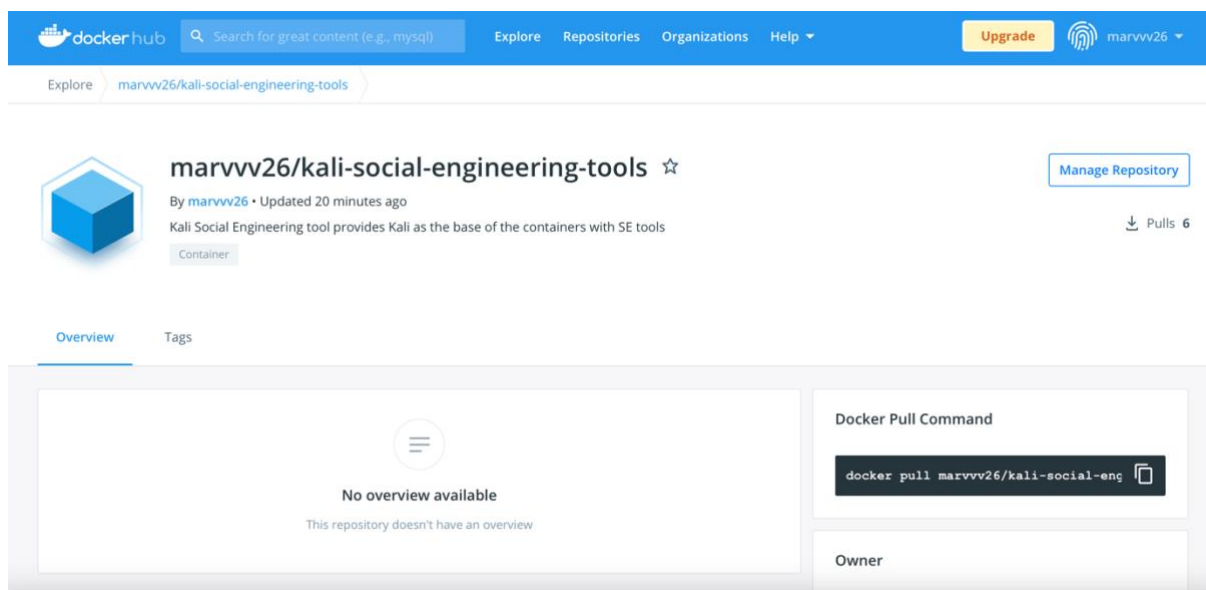


Figure 15 - Kali Social Engineering Tools ready for use

This container is readily available to anyone that wants to do anything with social engineering tools and try out techniques. To use our container, you just have to run the following command from your terminal:

```
docker pull marvvv26/kali-social-engineering-tools
```

7. CONCLUSION

This article discussed penetration testing, social engineering, attacks, techniques, and strategies. To assist an attacker in conducting a social engineering attack, we analysed the most commonly used techniques. These include phishing, spear phishing, baiting, tailgating, pre-texting, among others.

On the other hand, we built a Docker container from scratch with a base image. Docker has transformed the way programmes run in any environment. The base image used is Kali Linux. It is the most widely used penetration testing and security auditing platform, including comprehensive tools for detecting any vulnerabilities, discovered on the target machine.

Along with the setup and installation for each of the social engineering tools we employed, a brief history of Docker and Kali Linux was covered. In addition, this project gave me a chance to sharpen my python skills trying to understand payloads. One issue I came across was connecting the Metasploit framework to the database PostgreSQL.

Further study will focus on securing the docker container and simulating actual attacks with our tools running on the container.

8. Project Management

Project management is the application of procedures, methods, skills, knowledge, and experienced to accomplish specific project objectives within agreed-upon parameters, in accordance with the project acceptance criteria. The System Development Lifecycle (SDLC) was utilized in the sense that we changed existing systems (docker image) and create new systems (docker containers). The system development life cycle has 7 stages which was followed:

- Planning stage: Also called the feasibility stage is where I made plan for the upcoming project. I was able to set my goals for the coming weeks and what I plan to achieve with the project.
- Analysis stage: I gathered all the information required to create a docker container and made sure my system was up to par.
- Design stage: Here, I outlined the details of the container and what social engineering tools I plan to use. The operating system used which is the base for the container was also thought about.
- Development stage: This is the stage where the container was built and put together. Had some challenges which sharpened my python skills while I was trying to solve the issues that came up. The result of all the previous stages happened here.
- Testing stage: While we were not able to carry out an actual attack, we still tested to make sure that all the social engineering tools installed in the container were running and all prerequisites that they depend on was installed.

- Implementation stage: The whole reason for building the container is so people can have easy access to it and be able to build on it. After pushing the created docker container to the docker repository, the docker desktop engine was installed on a windows computer. We set it up as normal and pulled the docker container we created from the docker hub. No issues were experienced and all the tools work just fine.
- Maintenance stage: SDLC does not end when the container is made available to everyone. At the moment, everything is up to date and will be regularly updated when updates do pop up. Also, apt-get update ran after starting the container will look out for regular updates and update the container accordingly.

Through the course of my dissertation, I had regular meetings with my supervisor who came through for me a lot with the conversations we had. The original topic I had in mind to discuss was not possible due to unforeseen circumstances with the company.

8.1. Project Scheduling

My project management tool of choice is a Gantt chart, which provides me with a visual picture of my project and the actions I need to accomplish to ensure its successful completion. Among other things, the Gantt chart can be used to organise the timeline of a project and to highlight tasks that need to be performed, among other things.

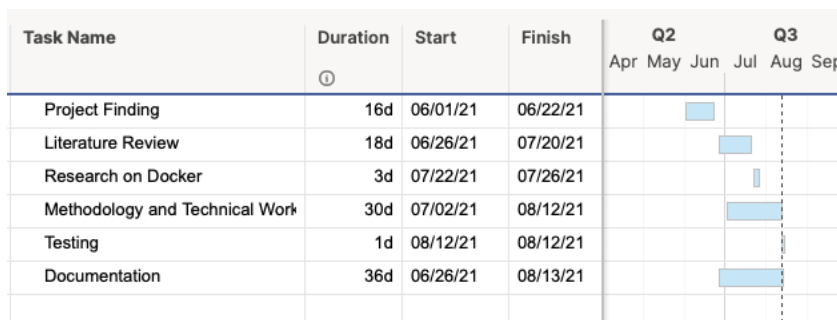


Figure 16 - Gantt chart

8.2. Risk Management

Risk management enables one to prepare for the unexpected by mitigating risks and additional costs in advance of their occurrence. Because I was not dealing with real-world user data, there was no risk associated with this project.

8.3. The Social, legal and Ethical Consideration

Legally, this programme is covered by section 3A of the Computer Misuse Act, as it is an article that facilitates the commission of other sections of the act's offences. This makes it criminal to produce, supply, or get this tool with the intent of using it to commit another offence. While this is not the project's intention, it is a possibility. Additionally, it is stated that the intended users of this project are needed to sign legally binding contracts in order to execute their activities, ensuring that the tool is not used maliciously.

The container produced is intended to be used ethically, for training and more research purposes. It can also be used by a certified penetration tester but not just anybody in the world. It is also not intended for social use as the tools in this arsenal can lead to drastic situations.

9. Bibliography

References

- Aldawood, H., & Skinner, G. (2020). Analysis and Findings of Social Engineering Industry Experts Explorative Interviews: Perspectives on Measures, Tools, and Solutions. *IEEE Access*, 8, 67321–67329. <https://doi.org/10.1109/access.2020.2983280>
- Alharthi, D., & Regan, A. (2021). *A Literature Survey and Analysis on Social Engineering Defense Mechanisms and INFOSEC Policies*. Papers.ssrn.com. <https://ssrn.com/abstract=3830208>
- BORGES, E. (2020). *SecurityTrails | The Social Engineering Toolkit*. Securitytrails.com. <https://securitytrails.com/blog/the-social-engineering-toolkit>
- Breda, F., Barbosa, H., & Morais, T. (2017). SOCIAL ENGINEERING AND CYBER SECURITY. *INTED2017 Proceedings*. https://www.academia.edu/33830548/SOCIAL_ENGINEERING_AND_CYBER_SECURITY?auto=citations&from=cover_page
- B L V Vinay Kumar, K Raja Kumar, & V Santhi. (2016). Penetration Testing using Linux Tools: Attacks and Defense Strategies. *International Journal of Engineering Research And*, V5(12). <https://doi.org/10.17577/ijertv5is120166>
- Denis, M., Zena, C., & Hayajneh, T. (2016). Penetration testing: Concepts, attack methods, and defense strategies. *2016 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*. <https://doi.org/10.1109/lisat.2016.7494156>
- Dimkov, T., Pieters, W., & Hartel, P. (2010). Two methodologies for physical penetration testing using social engineering. *Proceedings of the 26th Annual Computer Security Applications Conference on - ACSAC '10*. <https://doi.org/10.1145/1920261.1920319>
- Docker. (2020). *Docker overview*. Docker Documentation. <https://docs.docker.com/get-started/overview/>
- Docker. (2013). *What is a Container? | Docker*. Docker. <https://www.docker.com/resources/what-container>
- Edwards, M., Larson, R., Green, B., Rashid, A., & Baron, A. (2017). Panning for gold: Automatically analysing online social engineering attack surfaces. *Computers & Security*, 69, 18–34. <https://doi.org/10.1016/j.cose.2016.12.013>
- Felter, W., Ferreira, A., Rajamony, R., & Rubio, J. (2015). An updated performance comparison of virtual machines and Linux containers. *2015 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*. <https://doi.org/10.1109/ispass.2015.7095802>

Fulton, E., Lawrence, C., & Clouse, S. (2013). White Hats Chasing Black Hats: Careers in IT and the Skills Required to Get There. *Journal of Information Systems Education*, 24(1), 75–80. <https://aisel.aisnet.org/jise/vol24/iss1/8/>

Gunawan, T. S., Lim, M. K., Zulkurnain, N. F., & Kartiwi, M. (2018). On the Review and Setup of Security Audit using Kali Linux. *Indonesian Journal of Electrical Engineering and Computer Science*, 11(1), 51. <https://doi.org/10.11591/ijeecs.v11.i1.pp51-59>

Hatfield, J. M. (2019). Virtuous Human Hacking: The Ethics of Social Engineering in Cybersecurity. *Computers & Security*, 83, 1477. <https://doi.org/10.1016/j.cose.2019.02.012>

Jeremiah, J. (2019). Awareness Case Study for Understanding and Preventing Social Engineering Threats using Kali Linux Penetration Testing Toolkit. *ech Insig*, 43.

Krombholz, K., Hobel, H., Huber, M., & Weippl, E. (2015). Advanced social engineering attacks. *Journal of Information Security and Applications*, 22(2214-2126), 113–122. <https://doi.org/10.1016/j.jisa.2014.09.005>

Lu, T., & Chen, J. (2017). Research of Penetration Testing Technology in Docker Environment. *Proceedings of the 2017 5th International Conference on Mechatronics, Materials, Chemistry and Computer Engineering (ICMMCCE 2017)*. <https://doi.org/10.2991/icmmcce-17.2017.238>

Milošević, N. (2013). History of malware. *Arxiv.org*. <https://arxiv.org/abs/1302.5392>

Morabito, R., Kjällman, J., & Komu, M. (2015). *Hypervisors vs. Lightweight Virtualization: A Performance Comparison*. IEEE Xplore. <https://doi.org/10.1109/IC2E.2015.74>

Mouton, F., Malan, M. M., Leenen, L., & Venter, H. S. (2014). Social engineering attack framework. *2014 Information Security for South Africa*. <https://doi.org/10.1109/issa.2014.6950510>

Patil, P., & Devale, R. (2016). IJARCCCE A Literature Survey of Phishing Attack Technique. *International Journal of Advanced Research in Computer and Communication Engineering*, 5. <https://doi.org/10.17148/IJARCCCE.2016.5450>

Patil, S. (2021). Study of Container Technology with Docker. *International Journal of Advanced Research in Science, Communication and Technology*, 2581-9429, 504–509. <https://doi.org/10.48175/ijarsct-1283>

Perrone, G., & Romano, S. P. (2017). The Docker Security Playground: A hands-on approach to the study of network security. *2017 Principles, Systems and Applications of IP Telecommunications (IPTComm)*. <https://doi.org/10.1109/iptcomm.2017.8169747>

Please use Python3 to install Wifiphisher · Issue #1236 · wifiphisher/wifiphisher. (n.d.). GitHub. Retrieved August 12, 2021, from <https://github.com/wifiphisher/wifiphisher/issues/1236>

Putri, A. R., Munadi, R., & Negara, R. M. (2020). Performance analysis of multi services on container Docker, LXC, and LXD. *Bulletin of Electrical Engineering and Informatics*, 9(5). <https://doi.org/10.11591/eei.v9i5.1953>

Ruan, B., Huang, H., Wu, S., & Jin, H. (2016). A Performance Study of Containers in Cloud Environment. *Lecture Notes in Computer Science*, 10065, 343–356. https://doi.org/10.1007/978-3-319-49178-3_27

Saleem, J., & Hammoudeh, M. (2017). Defense Methods Against Social Engineering Attacks. *Computer and Network Security Essentials*, 603–618. https://doi.org/10.1007/978-3-319-58424-9_35

Sarginson, N. (2020). Securing your remote workforce against new phishing attacks. *Computer Fraud & Security*, 2020(9), 9–12. [https://doi.org/10.1016/s1361-3723\(20\)30096-8](https://doi.org/10.1016/s1361-3723(20)30096-8)

Sharma, P., Chaufournier, L., Shenoy, P., & Tay, Y. C. (2016). Containers and Virtual Machines at Scale. *Proceedings of the 17th International Middleware Conference*. <https://doi.org/10.1145/2988336.2988337>

Shebli, H. M. Z. A., & Beheshti, B. D. (2018). A study on penetration testing process and tools. *2018 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*. <https://doi.org/10.1109/lisat.2018.8378035>

Syed, A. M. (2021). Social engineering: Concepts, Techniques and Security Countermeasures. *Arxiv.org*. <https://arxiv.org/abs/2107.14082>

The Social-Engineer Toolkit (SET). (n.d.). TrustedSec. <https://www.trustedsec.com/tools/the-social-engineer-toolkit-set/>

Vats, P., Mandot, M., & Gosain, A. (2020). A Comprehensive Literature Review of Penetration Testing & Its Applications. *2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*. <https://doi.org/10.1109/icrito48877.2020.9197961>

Velu, V. K., & Beggs, R. (2019). Mastering Kali Linux for Advanced Penetration Testing: Secure your network with Kali Linux 2019.1 – the ultimate white hat hackers' toolkit, 3rd Edition. In *Google Books*. Packt Publishing Ltd. https://www.google.co.uk/books/edition/Mastering_Kali_Linux_for_Advanced_Penetr/kQGGDwAAQBAJ?hl=en&gbpv=1&dq=Velu

Virgillito, D. (2021). *Kali Linux: Top 5 tools for social engineering*. Infosec Resources. <https://resources.infosecinstitute.com/topic/kali-linux-top-5-tools-for-social-engineering/>

Watson, B. (2019). *Coronavirus and homeworking in the UK labour market* - Office for National Statistics. [Www.ons.gov.uk](http://www.ons.gov.uk).
<http://www.ons.gov.uk/employmentandlabourmarket/peopleinwork/employmentandemployeetypes/articles/coronavirusandhomeworkingintheuklabourmarket/2019>

wifiphisher - Penetration Testing Tools. (n.d.). Penetration Testing Tools. Retrieved August 11, 2021, from <https://en.kali.tools/?p=90>

Zhang, X., & Ghorbani, A. A. (2021). Human Factors in Cybersecurity. *Research Anthology on Privatizing and Securing Data*, 1695–1725. <https://doi.org/10.4018/978-1-7998-8954-0.ch082>