**Email Phishing Demonstration Using Kali Linux and the Social-Engineer Toolkit (SET)**

Made By:

**Shahd Rafat Gamil**

**Raneen Abdelfattah Mabrouk Ibrahim**

**Esraa Salah Mohamed Helmy**

**Sondos Ali Abdelaziz**

**Haidy Sobhy AbdElghaffar**

**It is presented to:**

**Eng. Mariem Zaki**

**National Telecommunication Institute (NTI)**

**1.Introduction:**

Phishing attacks are a common and effective method used by cybercriminals to deceive individuals into revealing sensitive information, such as login credentials. In this cybersecurity project, we demonstrate a controlled email phishing simulation using Kali Linux, a specialized operating system for penetration testing, and the Social-Engineer Toolkit (SET), an open-source tool designed for social engineering exercises. The demonstration involves cloning a legitimate website (Google's login page), crafting a phishing email, and harvesting credentials in a virtual environment. This project highlights the ease with which such attacks can be executed, emphasizing the need for user awareness and robust security measures. All activities were performed in an isolated virtual machine (VM) setup using VMware Workstation.

**2.Objectives:**

**Objective 1: Illustrate the Phishing Attack Process**

- Provide a detailed, step-by-step walkthrough of executing an email phishing attack using open-source tools.

- Demonstrate the cloning of a legitimate website (e.g., http://www.google.com) to mimic its login interface.

- Show the configuration of a phishing server using a local IP (e.g., 192.168.184.128) to capture submitted credentials.

- Highlight the use of spoofed emails with a malicious link (e.g., "Reset your password: click here") to lure victims.

**Objective 2: Demonstrate SET Functionality in Kali Linux**

- Explore the Social-Engineer Toolkit (SET) within Kali Linux, focusing on the Credential Harvester method under Spear-Phishing Attack Vectors.

- Detail the setup process, including selecting menu options (e.g., option 2 for Spear-Phishing, option 3 for Credential Harvester).

- Showcase the tool’s ability to clone sites and start a web server, with terminal output confirming the process (e.g., "[\*] Credential Harvester is running on port 80").

- Capture and display harvested credentials (e.g., "raneenabdelfattah1@gmail.com" and "1234") as proof of concept.

**Objective 3: Educate on Phishing Risks**

- Emphasize the vulnerability of users to phishing by simulating the capture of sensitive data like email addresses and passwords.

- Use the demo to illustrate how attackers exploit trust, with the harvested data logged in files (e.g., /var/www/html or SET reports).

- Discuss the real-world implications, such as identity theft or unauthorized account access, based on the simulation outcomes.

- Encourage awareness of phishing indicators, such as suspicious links or unexpected emails, using the demo as a teaching tool.

**3.Scope:**

This project is limited to a simulated demonstration within a virtual machine environment running on VMware Workstation with Debian 10.x as the host operating system. It focuses on cloning the Google login page (http://www.google.com) and hosting it on a private IP address (192.168.184.128) for local network testing, ensuring no external exposure or connectivity to live internet services. The scope includes crafting a spoofed email with a phishing link (e.g., "Reset your password: click here"), capturing POST data upon login attempts, and redirecting users to the legitimate site (https://www.google.com) after submission to simulate a realistic attack flow. The demonstration excludes real-world targets, live internet exploitation, or advanced evasion techniques such as IP spoofing, encryption bypass, or multi-stage payloads. It also avoids integration with external databases or cloud services, relying solely on local resources within the VM, including the SET tool’s default configurations and a manually configured SMTP server for email delivery (e.g., smtp.gmail.com with appropriate security settings). The project focuses on technical execution, with all actions meticulously documented through terminal outputs, browser interactions, email interface screenshots, and log file reviews (e.g., /usr/share/set/reports/credentials.txt).

**Thread model & attack selection :**

Purpose: To identify the victim, attacker capabilities, and ethical/legal constraints of the simulation. The goal is not to execute a real attack, but rather to measure user behavior in an isolated environment for research and awareness purposes.

**Thread Model**: Step-by-Step Attack Flow , This is a simple guide to how the attack works,

1: Plan Pick a website to fake (e., Google login). Learn about your target. Use tools like the Harvester to find emails/domains

2: Build Copy the website and set up your fake version. Run SET, pick "1) Social-Engineering Attacks" → "2) Website Attack Vectors" → "3) Credential Harvester" → "1) Web Templates". Enter your Kali IP (e.g., 192.168.184.128). SET’s Site Cloner A web server starts on your VM.

3: Send Share the fake link with someone (e.g., via email).a tool like Gophish to send the link. then Check for a new website hosted on your IP.

4: Catch When someone logs in, SET grabs their username and password. Watch the SET screen for login details .Credential Harvester See usernames/passwords appear in the terminal

5: Use the captured info to log in to the real site .Open a browser and try the creds manually . Any browser Success if login works; fail if 2FA blocks it.

**Attack Selection** This guide follows what you did while using Kali Linux in a VMware VM.

Start Here (Main Menu): Type 1 and press Enter to choose "Social-Engineering Attacks". Next Step (Social-Engineering Attacks): Type 2 and press Enter to go to "Website Attack Vectors". This lets you create fake websites, which is what you used next. Pick Your Attack (Website Attack Vectors): Type 3 and press Enter to select "Credential Harvester". This is the option you chose to steal login details from fake sites.

Choose How to Make It (Credential Harvester): Type 1 and press Enter to pick "Web Templates". This gives you ready-made fake sites (like Facebook or Google) to use. Set It Up (Web Templates): Type your Kali IP address (e.g., 192.168.184.128) and press Enter. This tells SET where to send the login info it catches. Check your IP with ifconfig if you’re not sure.

**Tools that Used**

* **Credential Harvester:** This is the main tool you used in SET. It creates a fake website (like Google or Facebook) and grabs usernames and passwords when someone logs in.
* **Web Templates:** Part of the Credential Harvester, this gave you pre-made fake sites to use.
* **Apache Web Server:** SET uses this to host your fake website. It starts automatically when you set up the Harvester.
* **Terminal (Kali Shell):** You used the Kali terminal to run setoolkit as root and follow the menu. It shows all the action, like when you entered your IP

**Easy Labs to Practice the** steps:

**Lab 1: Test the Credential Harvester with a Fake Site**

What You Need: Your Kali VM + a second VM

Steps: Open Kali, type setoolkit and press Enter. Pick 1 → 2 → 3 → 1 (Social-Engineering → Website Attack Vectors → Credential Harvester → Web Templates). Enter your Kali IP (e.g., 192.168.184.128) when asked. Copy the fake website link SET gives you. Open the link on the second VM and type a fake username/password (e.g., testuser/testpass). Watch the Kali terminal to see if it catches the login.

**Goal: Learn how SET grabs logins from a fake site.**

**Lab 2: Clone a Site and Test It**

What You Need: Same as Lab 1, plus a website to clone (e.g., a test site like a blog).

Steps: Run setoolkit again. Go 1 → 2 → 3 → 2 (Social-Engineering → Website Attack Vectors → Credential Harvester → Site Cloner). Type the URL of the site to clone (e.g., http://example.com) and your IP. Get the new fake link and test it on the second VM. Check the terminal for captured data.

**Goal: Practice making a custom phishing page.**

**Lab 3: Check the Results**

What You Need: After Lab 1 or 2.

Steps: Look at the SET terminal for the login details it caught. Find the report files in /usr/share/set/reports/ using ls -l. Open a file (e.g., with cat report.html) to see the data.

**Goal: Learn where SET saves the info and how to read it.**

|  |  |  |
| --- | --- | --- |
| The Demo | | |
| **Explanation**: I started the project by running Kali Linux inside a VMware virtual machine. To gain full control of the system, I used the sudo -i command to become the administrator (root) of the machine and entered the password.  **Experiment**: Imagine you start your virtual machine, open a command window, type sudo -i, press Enter, and enter the password. The screen changes to "root@kali," and you're ready to go. | |  |
| **Attempting a Wrong Command**  **Explanation**: You accidentally tried to run a nonexistent tool (schoolkit), and you get the message "command not found," which means you need to type the correct command.  **Experiment**: Imagine you type "schoolkit," and a red error message appears. This indicates you need to correct the command for the SET tool. | |  |
| **Running the SET tool**  **Explanation**: You typed the correct command setoolkit to run the "Social-Engineer Toolkit." Strange numbers appear for a moment (possibly a temporary error), and then the welcome screen starts.  **Experiment**: Type setoolkit in the window, and imagine the screen displays numbers and then changes to a welcome message about the tool. | |  |
| **Detailed welcome screen**  **Explanation**: The welcome screen gives you information about the tool: it was created by David Kennedy, version 8.0.3, named "Maverick," with options to follow the tool on Twitter or visit the website.  **Experiment**: Imagine you see a text saying, "Welcome! This is SET 8.0.3. Follow us on Twitter or visit www.trustedsec.com," along with a list of options. | |  |
| **Selecting Social Attacks**  **Explanation**: You select option 1 ("Social-Engineering Attacks") from the main menu to start a social attack project.  Experiment: Press "1" on your keyboard, and imagine the menu changes to display options like "Spear-Phishing" or "Website Attack." | |  |
| **Exploring the Options**  **Explanation**: You tried option 2 ("Penetration Testing") but returned to option 1, meaning you're focusing on social attacks.  **Experiment**: Press "2," then press "1" again. Notice that the menu returns to the social attack options. | |  |
| **Selecting a Website Attack**  **Explanation**: I chose option 3 to focus on website attacks as part of the project.  **Experiment**: Press "3," and imagine seeing a new menu with options like "Credential Harvester" or "Web Jacking." | |  |
| **Selecting Website Templates**  **Explanation**: I chose option 1 ("Web Templates") to use pre-made templates, and I gave you a prompt to verify the IP address.  **Experiment**: Press "1," and imagine a message saying, "Verify the IP! Enter the website address now." |  | |
|  | |
| **Entering the Website Address**  **Explanation**: You entered a website address (such as google.com) to copy, which is the key step in the project to deceive users.  **Experiment**: Type "google.com" and imagine the screen begins processing the request. | |  |
| **Starting the Copy Process**  **Explanation**: The tool has started copying the website and gives you instructions to open the link later.  **Experiment**: Imagine a message "Copying..." followed by instructions: "Open http://192.168.1.10 for testing." | |  |
| **Explanation**: The site cloning process is complete, and the tool gives you a final URL (such as http://192.168.1.10) that you can use. This is the URL you will share to test the attack.  **Experiment**: Imagine the screen displays the URL "http://192.168.1.10," and open it in your browser to confirm that the copied site works. | |  |
| **Explanation**: You open the link in the browser inside the virtual machine, and the copied site appears (such as a copy of google.com). If someone enters login credentials here, you will be automatically logged in.  **Experiment**: Open your browser, type "http://192.168.1.10," and imagine you see a Google-like login page, ready for you to try entering fake credentials. | |  |
| **Explanation**: The tool returns you to the "Credential Harvester" screen, waiting for any login attempts. If someone logs in with credentials from the copied site, they will immediately appear here (such as the username and password).  **Experiment**: Imagine the screen displays "Waiting for login attempts...," and try entering fake credentials from the copied site to see how you log in. | | A screenshot of a computer  AI-generated content may be incorrect. |

A screenshot of a computer

AI-generated content may be incorrect.**8.Results:**

In this controlled simulation, a spear-phishing email was crafted to appear as an official Google security message, instructing the targeted subject to “Reset your password.” The email contained a hyperlink that, when clicked, directed the subject to a locally hosted cloned Google login page created using the Social-Engineer Toolkit (SET) inside a VMware virtual environment.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.Upon accessing the phishing page, the subject entered their email address and password, believing it to be the legitimate Google interface.

The SET credential harvester, running on the attacker’s Kali Linux machine, immediately intercepted and logged these credentials. The captured data was displayed in real time on the terminal and also stored in the SET reports directory (/usr/share/set/reports/).

To preserve realism and minimize suspicion, the phishing server was configured to redirect the subject seamlessly to the actual Google login page after submission. This ensured that the subject experienced no apparent disruption or warning signs.

The demonstration resulted in one verified credential capture, confirming the effectiveness of the phishing setup. The fake server operated locally on port 80 using the IP address assigned to the Kali machine (e.g., 192.168.184.128). Importantly, all activities were executed within an isolated VMware environment, with no interaction with external systems or real Google infrastructure.

**9.Analysis&Mitigation:**

*Root-cause analysis (why the attack succeeds in the demo):*

1. User trust & social engineering: The cloned page and a convincing email prompt bypassed user skepticism in the controlled test.
2. Lack of MFA on tested account: Simple password-only authentication allows captured credentials to be re-used.
3. Insufficient email authentication/spoof protections (in a real-world context): If the sender domain lacks SPF/DKIM/DMARC or the recipient’s mail filtering is weak, phishing emails are more likely to arrive in the inbox.
4. No URL inspection / link previewing: Users who click links without inspecting the destination are vulnerable.
5. Browser/endpoint warnings not heeded or not present: If the cloned site is hosted on HTTP or a certificate mismatch exists and the user ignores warnings, the attack succeeds more easily.

*Technical mitigations (defensive controls)*

* Enforce MFA / 2FA for all user accounts — this is the single most effective control to prevent credential-only compromises.
* Email authentication: Ensure sender domains publish and enforce SPF, DKIM, and DMARC to reduce successful spoofing.
* Advanced email filtering / URL rewriting: Deploy mail gateway solutions that rewrite sandbox links in emails, scan attachments, and block known malicious patterns.
* Browser and endpoint protections: Enable browser-based phishing protection lists, certificate validation enforcement, and OS/endpoint blocking of known phishing domains and HTTP-to-HTTPS mismatches.
* Network segmentation & host hardening: Limit lateral movement if credentials are compromised; isolated systems used for testing.
* Logging & monitoring: Centralize web server, mail gateway, and authentication logs. Alert on unusual logins (new IPs, impossible travel, atypical times).
* Simulated phishing campaigns and training: Regular, controlled phishing simulations combined with just-in-time training dramatically reduce click-through and credential submission rates.
* Account recovery hardening: Use additional verification for recovery flows to prevent attackers using harvested credentials plus simple recovery vectors.

*Operational controls & incident handling*

* Pre-authorized lab environment: Keep all security testing strictly within isolated lab networks and with documented scope/consent.
* Detection playbook: If credentials are suspected to be compromised, immediately revoke or reset credentials, force MFA re-enrollment, and perform access logs review.
* Forensic collection: Preserve SET report files, webserver logs, and VM snapshots for analysis while following chain-of-custody practices if needed.

**10. Ethics:**

*Legal & ethical constraints observed and required*

* Informed consent & scope: The demo was performed in an isolated VM environment with no real users targeted. This meets basic ethical requirements. Any real-user testing requires prior informed consent or an approved IRB/ethical review where applicable.
* Avoid real-world harm: Do not use harvested credentials against live accounts, attempt account takeover on real services, or send phishing emails to uninformed third parties.
* Data minimization & handling: Treat any captured credentials as sensitive data: store them securely, limit access, and securely delete them when the exercise concludes. Never publish raw captured credentials.
* Responsible disclosure: If the simulation finds vulnerabilities in production systems, notify responsible owners via an agreed disclosure process. Do not publicly disclose exploit details that would enable replication without mitigation guidance.
* Legal compliance: Ensure testing complies with local laws, organizational policies, and the Terms of Service of any tools/platforms used.

*Ethical best practices for this type of exercise*

* Always document authorization (who approved the test, scope, time limits).
* Use mock accounts and artificially generated credentials where possible.
* Debrief affected parties, share lessons learned, and provide training resources.
* Destroy or anonymize sensitive artifacts after the exercise.

**11. Conclusion:**

This controlled Kali/SET simulation successfully demonstrated how a credential-harvesting phishing page can be created, hosted, and used to capture user-submitted credentials inside a virtual lab. The results show the practical ease of such attacks and reinforce two central points:

1. *Human behavior is the primary attack vector.*

Training and regular phishing-resistant habits (checking sender domains, hovering over links, verifying unexpected requests) are essential.

1. *Technical control greatly reduces risk*.

Enforcing MFA, hardening email authentication (SPF/DKIM/DMARC), using strong email and endpoint protections, and improving logging/monitoring substantially mitigate the threat.