How to Set Up a Phishing Campaign

Jordan C. Lanning

5/25/2025

Before any phishing campaign is conducted, it is essential to have written approval from both internal leadership and any applicable clients. This ensures that all parties are aware of the nature and intent of the campaign and that it aligns with organizational policy and legal boundaries. The approval process should also include detailed information regarding the scope of the campaign, including who will be targeted, the timing of the campaign, potential impacts, any inclusion or exclusion lists, cost considerations, and any additional information required by stakeholders. Without this documentation in place, a phishing simulation could be misinterpreted as an actual threat, leading to confusion or reputational damage.

For the technical setup, I acquired a Virtual Private Server (VPS) that did not block port 587 by default. Vultr was chosen as the provider, and I opted to run Ubuntu 22.04 as the operating system. After deploying the VPS, I downloaded GoPhish directly from GitHub using the curl command, following all necessary zip extraction and file modifications required to run the software effectively. The server's firewall (UFW) needed to be configured as well—specifically, enabling traffic on ports 80 and 3333 to allow for the landing page and GoPhish dashboard access, respectively. Once the environment was ready, I logged into the GoPhish interface using the default credentials provided.

To send phishing emails, a sending profile was created. I used an Outlook email account because it does not currently block SMTP traffic the way Gmail does. Additionally, I registered for a SendGrid account, which is commonly used to improve email deliverability and avoid blacklisting. After validating the email address with SendGrid, I generated an API key, which was then used in the GoPhish sending profile configuration. When using services like SendGrid, the username field is typically set to "apikey," and the API key itself serves as the password. Other details such as the SMTP host and port were standard, derived directly from the provider.

The landing page for the campaign was modeled after a CompTIA sign-in page. I selected this page primarily for demonstration purposes, as I wanted something recognizable and professional. After copying the URL of the real CompTIA login page, I used ChatGPT to generate a corresponding HTML file that closely mimicked the look and feel of the original. This file was then uploaded to GoPhish and configured to serve as the campaign's landing page.

For the email template, I replicated a legitimate CompTIA notification email that I had previously received. I maintained the general format and styling but customized the text content to align with the goals of my phishing simulation.

After completing the campaign, a report should be generated that summarizes the results. This report must include metrics on email open rates, link clicks, credential submissions, and any

other user interactions. Additionally, it should provide training resources to help educate users on phishing awareness and include security recommendations based on observed behavior. This post-campaign analysis ensures that the simulation serves its ultimate purpose: strengthening the organization's overall cybersecurity posture. In the future, I will be looking to spoof email addresses.

Appendix

Commands:

```
# Update system and install unzip
    sudo apt update && sudo apt install unzip

# Download GoPhish
    curl -O https://github.com/gophish/gophish/releases/download/v0.12.1/gophish-v0.12.1-linux-64bit.zip

# Unzip GoPhish archive
    unzip gophish-v0.12.1-linux-64bit.zip
    cd gophish

# Allow required ports
    sudo ufw allow 80
    sudo ufw allow 3333

# Run GoPhish
    sudo ./gophish
```

Images:

Email (Spam)

Stolen Credentials (Sample)

Fake Landing Site