WhatsApp QRLJacking Tool
A professional red team tool designed for demonstrating WhatsApp session hijacking through QR code phishing (QRLJacking). Built with a modern Streamlit GUI, Flask-based phishing server, Selenium for QR code extraction, and Nginx/Certbot for production-grade HTTPS deployment, this tool mimics WhatsApp's login page to lure targets into scanning a QR code, capturing session data for authorized testing.
Disclaimer: This tool is for ethical hacking and authorized red team operations only. Unauthorized use is illegal and unethical. Always obtain explicit consent and comply with applicable laws.
Features

Modern GUI: Cyberpunk-themed Streamlit interface with custom logo, real-time logging, and intuitive controls.
Realistic Phishing: HTTPS-enabled phishing page mimicking WhatsApp's official login.
Social Engineering: Customizable lures (group invites, channel promos, urgent logins) with obfuscated URLs.
Robust Deployment: Nginx reverse proxy with Certbot SSL for production-grade security.
Modular Design: Clean separation of concerns (QR extraction, phishing server, lure generation).

Prerequisites

System: Ubuntu 20.04+ (or similar Linux distribution).
Dependencies:
Python 3.8+
Tesseract OCR
Chromedriver (compatible with installed Chrome version)
Nginx
Certbot

Domain: A registered domain with DNS pointing to the server’s public IP.
Access: Root or sudo privileges for deployment.
Logo: Place whatsappqr_logo.png in static/ for GUI branding.

Installation

Clone the repository:git clone https://github.com/your-repo/whatsapp_qrljacker.git
cd whatsapp_qrljacker

Install system dependencies:sudo apt update
sudo apt install -y python3 python3-venv nginx certbot python3-certbot-nginx tesseract-ocr

Install Python dependencies:python3 -m venv venv
source venv/bin/activate
pip install -r requirements.txt

Install Chromedriver:
Download from https://chromedriver.chromium.org/downloads.
Ensure it matches your Chrome version and is in your PATH.

Add logo:
Place whatsappqr_logo.png in the static/ directory.

Configuration

Update config/config.yaml:
Set phishing_url to your domain (e.g., https://yourdomain.com).
Adjust lure_type (options: group_invite, channel_promo, urgent_login).

Update config/nginx.conf:
Replace yourdomain.com with your domain.

Update scripts/deploy.sh:
Replace yourdomain.com and your.email@example.com with your details.

Deployment

Make the deployment script executable:chmod +x scripts/deploy.sh

Run the deployment script:./scripts/deploy.sh

This sets up the virtual environment, starts the Flask server, configures Nginx, and obtains SSL certificates via Certbot.

Usage

Access the GUI:
Navigate to http://localhost:8501 in your browser.
The sidebar displays the custom whatsappqr_logo.png for branding.
Select a lure type and generate a message.
Copy the lure message for distribution (email, SMS, social media).

Start Phishing Server:
Click "Start Phishing Server" to extract the QR code and host the phishing page at https://yourdomain.com.
The QR code is displayed in the GUI for reference.

Monitor Logs:
View real-time logs in the GUI or logs/qrl_jacker.log for session capture details.

Stop Server:
Click "Stop Phishing Server" to terminate the Flask server.

Red Team Tactics

Phishing Page: Hosted over HTTPS, mimicking WhatsApp's official login with a convincing lure (e.g., "Join our exclusive group!").
Lure Distribution: Use generated messages with short URLs (e.g., https://wa.me/abc123) to entice targets via targeted campaigns.
Stealth: Nginx security headers and HTTPS reduce detection risks.
Session Capture: Logs capture session data when targets scan the QR code.

Security Notes

Authorization: Use only with explicit permission from the target organization.
Deployment: Deploy on a controlled server with an anonymous domain to avoid traceability.
Maintenance: Regularly renew SSL certificates (sudo certbot renew).
Customization: Modify templates/phishing.html for campaign-specific pretexts.
Logo: Ensure whatsappqr_logo.png is a high-quality PNG (recommended: 150x150px) for optimal GUI display.

Project Structure

src/: Core Python scripts for GUI, QR extraction, phishing server, and lures.
templates/: HTML template for the phishing page.
config/: Configuration files (YAML and Nginx).
logs/: Session and error logs.
static/: QR code images and custom logo (whatsappqr_logo.png).
scripts/: Deployment script.
requirements.txt: Python dependencies.
run.sh: Local development script.

Troubleshooting

QR Extraction Fails: Ensure Chromedriver and Tesseract are correctly installed and configured.
Server Issues: Verify Nginx configuration (sudo nginx -t) and Flask port availability.
Logo Not Displaying: Confirm whatsappqr_logo.png is in static/ and accessible.
Logs: Check logs/qrl_jacker.log for detailed error messages.

License
For educational and authorized use only. No warranty provided. Use at your own risk.
Contact
For issues or contributions, contact your red team lead or submit a pull request to the repository.