QR Code Secure Scan

(Link Verify Project)

John Saw

SAW HAN BENG
7/7/23

QR CODE SECURE SCAN

Objective

A Web App that scans QR Codes for insecure or malicious links.

Required Functions

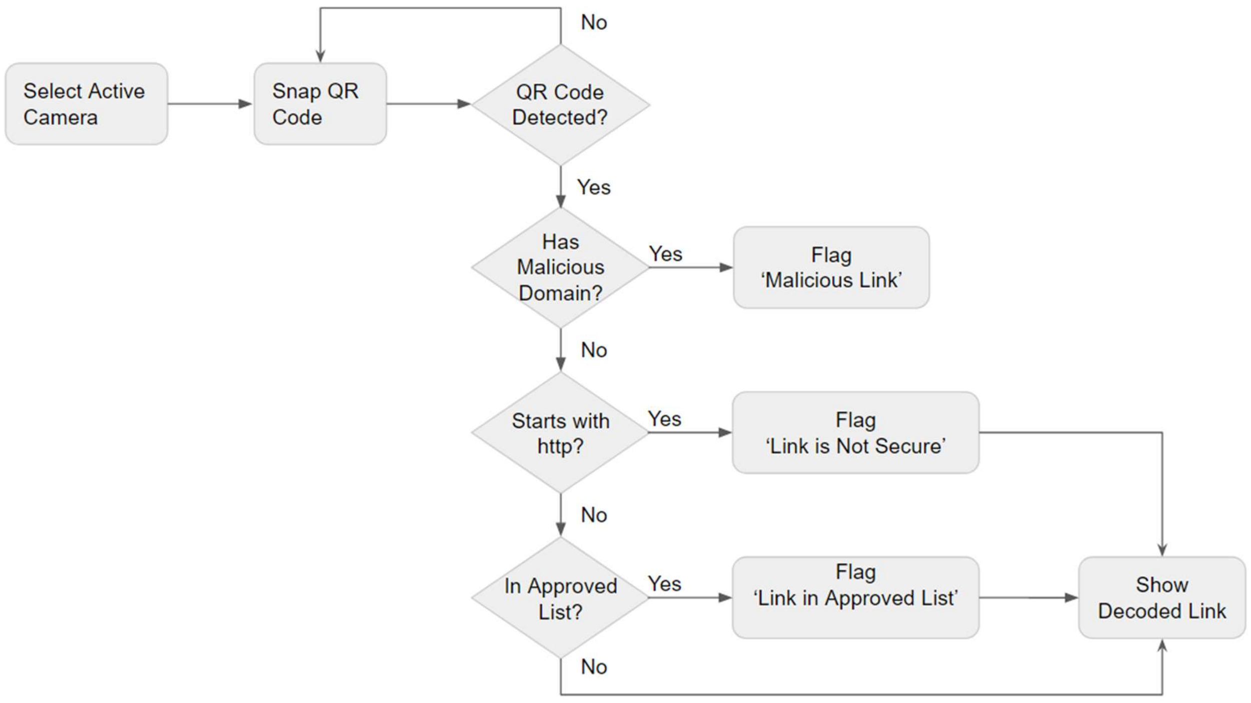
1. User to aim and scan at QR Codes.
2. Decoded data to check against known malicious domains.
3. Decoded data to check against approved list.
4. Reveal results and provide option for user to proceed.

Motivation

Enable security personnels and/or public users to check for compromised QR Code labels efficiently on mobile devices.

Web Application Setup

Flow map below explains the order of activities chained by checkpoints.



1. Select Active Camera to view QR Code.
2. Snap a photo of the QR Code.
3. With QR Code detected successfully, decoded data will be checked in a series of Yes or No filters.
4. Apart from Malicious Link, remaining Decoded Links will be visible and clickable to user.

Optional – Upload Approved List of URLs

Python Script

```
import cv2
import numpy as np
import streamlit as st
import requests
from PIL import Image
from pyzbar.pyzbar import decode
```

Script Explanation
Import required libraries.
streamlit to facilitate web application setup with components e.g. write, camera_input.
requests to deal retrieve web query status.
cv2, numpy, PIL to manage and manipulate image.
pyzbar to decode QR Code.

```
@st.cache_data
def check_string_contains_domains(string):
   url = "https://hole.cert.pl/domains/domains.txt"
   response = requests.get(url)

   if response.status_code == 200:
      domains = response.text.splitlines()

      for domain in domains:
         if domain in string:
            return True

   return False

@st.cache_data
def check_string_contains_approved_list(string, file):
   for link in file:
      if link.decode().strip() == string:
         return True
   return False
```

Script Explanation
st.cache_data to improve memory management.
check_string_contains_domains to check decoded string against ‘bad’ reference.
check_string_contains_approved_list to check decoded string against ‘good’ reference.

```
## Welcome Message
st.header("Welcome to My QRCode Scan Web App")

## File Uploader
file = st.sidebar.file_uploader("Upload Approved List", type = ['csv','txt'])

## Access Camera and Enable Photo Taking
image = st.camera_input("Snap a photo of QRCode to Check for Unsecure Links")
```

Script Explanation
st.header for welcome message.
st.sidebar.file_uploader to enable file upload and restricted to extension of csv or txt.
st.camera_input to enable connection to client camera.

```
## Decode QR Code and pass through nested ifs
if image:

    pil_image = Image.open(image)
    cv2_img = cv2.cvtColor(np.array(pil_image), cv2.COLOR_RGB2BGR)

    try: # Handle potential errors
        data = decode(cv2_img)[0].data.decode()
        if data and file:
            if check_string_contains_domains(data):
                st.write("Link is malicious")
            elif data.startswith(r"http://"):
                st.write("Link is not secure")
                st.write(f"Decoded data is {data}")
            elif file:
                if check_string_contains_approved_list(data, file):
                    st.write("Link is in Approved List")
                    st.write(f"Decoded data is {data}")
                    st.write("")
                else:
                    st.write(f"Decoded data is {data}")
            else:
                st.write(f"Decoded data is {data}")
        elif data:
            if check_string_contains_domains(data):
                st.write("Link is malicious")
            elif data.startswith(r"http://"):
                st.write("Link is not secure")
                st.write(f"Decoded data is {data}")
            else:
                st.write(f"Decoded data is {data}")
        else:
            st.write("QR code not found..")
    except:
        st.write("QR code not found..")
```

Script Explanation
Starts of with image captured exist
Followed by image formatting.

Nested IFs are grouped in terms of whether approved file upload exist.
Followed by decoded link check if contains malicious domains.
Followed by decoded link check if starts with ‘http://’.
Lastly reveal decoded link excluding malicious link.

Credit and Links

Web App Link:

https://qrcodesecurescan.streamlit.app/

Demonstration Link:

https://qrcodesecurescan.streamlit.app/demo

GitHub Link:
https://github.com/sawhanbeng/Cybersecurity/tree/main/QRCodeSecureScan