Report

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

Tool to identify the type of devices used in communication

(Computer/Mobile)

Mini Project Report

CSE3162 COMPUTER NETWORKS LAB

Student: Aditi Shrivastava

Registration No: 210905244

Student: Vrindavaneshwari Subramanian

Registration No: 210905198

Student: Soumya Sahu

Registration No: 210905196

Course: CSE3162

17-Nov-2023

Abstract

This project presents a tool designed for identifying diverse devices, such as mobile phones and desktop computers, through network traffic analysis using Wireshark. The tool extracts essential details, including User-Agent information, from raw traffic data in pcap files, associating them with specific device types. With a focus on aiding developers in optimizing content, layout, and applications for varied devices, the program employs a user-friendly interface and provides insights into device properties like screen size, browser type, and support for technologies such as CSS, HTML, and JavaScript. The tool also detects outdated devices and monitors bandwidth consumption, offering a holistic approach to network analysis.

Contents

1. Introduction

In today's world with lots of different gadgets, developers need tools to make sure websites and apps work well on all kinds of devices, like phones and computers. This project is all about creating a tool that uses Wireshark, a special program that looks at internet traffic, to figure out what kind of devices are using a network. The tool focuses on grabbing important information, like the type of browser and device size, from the internet traffic data. This info helps developers make their stuff look and work better on different devices. The tool also checks if a device is kind of old-fashioned and may need updates. It's like giving developers a special pair of glasses to see and understand the devices using their creations, making their job easier in today's fast-changing digital world.

2. Background

Device detection is vital for developers to optimize content and applications based on specific device properties. This project utilizes Wireshark, a robust network protocol analyzer, to capture raw network traffic and extract User-Agent information, providing insights into device characteristics.

3. Data

The project involves the collection of raw network traffic using Wireshark and the creation of pcap files for comprehensive software analysis. The program will analyse the pcap files.

3.1 Data preparation

Connecting wireshark with local network and using different devices connected to the same network to request http GET packets which will be captured in the pacp files.

4. Approach

The software captures and analyzes network traffic, categorizes devices into mobile or desktop types based on User-Agent strings, and provides insights into device security by identifying outdated User-Agents. The user interface enhances user experience by visualizing detected devices, including relevant details.

5. Results

In the culmination of our data analysis, our tool effectively extracted and categorized device information from Wireshark-captured network traffic. The User-Agent details uncovered a diverse range of devices, including mobile phones, tablets, desktop computers, and unidentified devices. A notable observation was the prevalence of outdated User-Agents marked by keywords such as "MSIE 6.0," "MSIE 7.0," and "Android 2," underscoring the critical importance of device security and the need for timely updates.

The user-friendly interface's visual representation of device types proved to be a standout feature, aiding developers in optimizing content and layouts for specific device characteristics. Additionally, the tool's capacity to monitor bandwidth utilization offered valuable insights into network efficiency, allowing for the identification of high-traffic devices and potential bottlenecks. Overall, the results highlight the tool's significance in providing actionable insights for content optimization and network management, empowering developers to create more adaptive and efficient digital experiences.

6. Discussion

The results hold significant implications for developers seeking to enhance user experiences across a spectrum of devices. The categorization of devices, identification of outdated User-Agents, and monitoring of bandwidth utilization collectively empower developers to optimize content and layouts. This has direct relevance to web developers, application designers, and IT administrators, providing them with actionable insights for creating adaptive and efficient digital environments.

The prevalence of outdated User-Agents raises awareness of potential security vulnerabilities, prompting the need for users to update their browsers and operating systems. The visual representation of device types within the user-friendly interface aids developers in making informed decisions regarding content optimization.

6.1 Limitations and Challenges

Implement capturing packets through the code itself instead of using pcap files, there was some error which we were unable to figure out in the code and the data was not being displayed on the interface.

7. Conclusion

In conclusion, this project provided valuable insights into device detection using Wireshark-based network traffic analysis. The tool successfully categorized devices, emphasizing the prevalence of outdated User-Agents and their security implications. Effective data collection, management, and analysis, coupled with a user-friendly interface, empowered developers with actionable insights for content optimization.

Limitations include potential inaccuracies in device categorization relying solely on User-Agent strings. Future investigations should explore advanced machine learning techniques to enhance accuracy. Additionally, evaluating the tool's performance in dynamic network environments would strengthen its robustness. This project highlights the importance of continuous refinement and identifies avenues for future research to solidify the tool's effectiveness in addressing device detection challenges and providing a reliable solution for developers seeking to optimize digital content across diverse devices.

8. Code Snippet

```
import os
import sys
import pyshark
from user_agents import parse
from tkinter import *
from tkinter import filedialog
from PIL import Image, ImageTk
import tkinter.font as tkFont

heading = 'Device Type Detection'
root = Tk(className=heading.title())
fontStyle = tkFont.Font(family="Lucida Grande", size=10)
fontStyle1 = tkFont.Font(family="Lucida Grande", size=25, weight="bold
    ")
root.geometry("2000x800")
root.title(heading)

# create background image bg.jpg
image = Image.open("bg.jpg")
photo = ImageTk.PhotoImage(image)
label = Label(root, image=photo)
label.place(x=0, y=0, relheight=1, relwidth=1)

def isMobileDevice(useragent):
    user_agent = parse(useragent)
    if user_agent.is_mobile:
        return True
    else:
        return False

def isTabletDevice(useragent):
    user_agent = parse(useragent)
    if user_agent.is_tablet:
        return True
    else:
        return False

def isPC(useragent):
    user_agent = parse(useragent)
    if user_agent.is_pc:
        return True
    else:
```

```
        return False

def isOutdated(useragent):
    outdated_keywords = ["MSIE 6.0", "MSIE 7.0", "MSIE 8.0", "Android 2.", "iOS 9_"]
    for keyword in outdated_keywords:
        if keyword in useragent:
            return True
    return False

def getUserAgent():
    root.filename = filedialog.askopenfilename(
        initialdir="/", title="Select file")

    if root.filename == '':
        print("No file selected")
        sys.exit()
    else:
        useragents = []
        cap = pyshark.FileCapture(
            root.filename, display_filter='frame contains "GET"')
        for packet in cap:
            user_agent = packet['http'].user_agent
            useragents.append(user_agent)
            i = len(useragents)
            myLabel = Label(
                root, text=f"Packet {i}: {user_agent}\n", font=fontStyle, fg="white", bg="#042592")
            myLabel.pack()

            if isMobileDevice(user_agent):
                device_type = "Mobile "
            elif isTabletDevice(user_agent):
                device_type = "Tablet"
            elif isPC(user_agent):
                device_type = "Desktop "
            else:
                device_type = "Unknown"

            myLabel = Label(
                root, text=f"Device Type: {device_type}", font=fontStyle1, fg="white", bg="#042592")
            myLabel.pack()

            if isOutdated(user_agent):
                security_label = Label(
                    root, text="Outdated or Vulnerable User-Agent!", font=fontStyle1, fg="red", bg="#042592")
                security_label.pack()

        cap.close()

fileInputBtn = Button(root, text="Choose file", font=tkFont.Font(family="Lucida Grande", size=15),
                      command=getUserAgent).pack(
    side=TOP, pady=20, padx=20)

```

```
94 root.mainloop()
```

9. Reflections on Own Work

- **Improvements and Changes:**
  - Regularly updated the tool to accommodate new device types, browsers, and technologies.
  - Considered incorporating machine learning algorithms for improved device identification.
  - Enhanced the tool's reporting capabilities to provide more detailed insights to developers.
  - Using only the code to analyse the http packets being captured by the network.