Augmented Reality Wireless Network Security Mapper

By

Orion R. Gonzales

A SENIOR RESEARCH PAPER PRESENTED TO THE DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE OF STETSON UNIVERSITY IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF BACHELOR OF SCIENCE

STETSON UNIVERSITY

2023

Acknowledgments

This research project was developed and programmed with a Meta-Quest 2 VR/Oculus Quest 2 VR which aims to demonstrate the applicability of this program on advanced equipment like the Windows Hololens or Google Glass. Notably, the project was developed using Unity version 2020.03, an intentional choice based on compatibility. The completion of this work included the utilization of external resources, such as the Unity Engine Wiki, ChatGPT, ChatGPT co-pilot, Git-Hub Advance Search, and various Android resources. These resources provided insights and support, contributing to the project's progression and outputs.

Contents

List of Tables

List of Figures

Abstract

Augmented Reality Wireless Network Security Mapper

By

Orion R. Gonzales

08-MAY-24

Advisors: Dr.Plante,

Department: Mathematics and Computer Science

The project's focus is a tool designed to map wireless connections to routers while uncovering security authentications in an AR setting to visualize the connected network structure. Engaging a diverse methodology, the project combined Unity's Augmented Reality/Virtual Reality capabilities, Unity's Android Engine, and the Mixed Reality Toolkit. Development growth involved building APK files on the Meta Quest 2/Oculus Quest 2 by using the Oculus Home App, Oculus Developer, and Side Quest. The research generated discoveries including the accessibility of beneficial network information such as BSSIDs, RSSI Signal Strength, and authentication data. Field testing that relies on the user's initial position, had the outcome of mapping the wireless connection as they move. This study displays the possibility of AR technology within cybersecurity creating features to physically allow users to see and touch the wireless network with only their hands with Unity's hand tracking.

1 INTRODUCTION

Exploring information on connected wireless networks on Windows or Linux machines involves utilizing various commands like 'Netsh WLAN show interfaces' or 'ipconfig,' which reveals essential or raw data. The outcome of your machine can retain details including SSID, BSSID, SIGNAL STRENGTH, and AUTHENTICATION TYPE, among others. In the Department of Cybersecurity, these components offer potential vulnerabilities for those who can connect to any wireless network or router. For instance, identifying a router using outdated security protocols such as WEP (Wired Equivalent Privacy) or missing WPA (Wi-Fi Protected Access) encryption exposes potential security hazards.

In September 1999, WEP occurred as the standard encryption protocol for wireless networks. However, now WEP networks are non-secure due to their low-bit encryption ranging from 64-bit to 256-bit. In 2003, WPA was introduced, featuring 256-bit Pre-Shared Keys, overcoming WEP networks and becoming common among most wireless configurations [1]. Such findings emphasize the consequence of not updating systems or skipping firewall configurations to minimize cybersecurity threats. WPAII was introduced in 2006 with AES algorithms and CCMP (Counter Cipher Mode with Block Chaining Message Authentication Code Protocol) [1]. This protocol has become mandatory for modern devices, ensuring improved security standards for wireless connections.

Modern devices incorporate these security types such as Virtual Reality (VR) and Augmented Reality (AR). VR equipment can range from headsets to full-body physical tracking that brings the user into a digital simulated environment. Headsets and equipment for VR can range to different techniques using a three-dimensional camera system or light and movement sensors. The feedback from these simulations can give visual, auditory, or sometimes a touch-base sensation for users to experience more than what a personal computer

can offer [2]. The most common companies that use VR equipment are Facebook and Valve. VR has proven valuable programs in the cybersecurity field, enabling the creation of testing environments for training or futuristic terminal interfaces.

Yet, while these VR simulations allow more visual representations, they often are not an alternative to operating a Linux machine terminal for cybersecurity applications. AR offers a different range of alternative capabilities that VR can't achieve. AR technology can bring in virtual objects while integrating the real-life environment around you. AR is commonly used in various fields of computer science and cybersecurity such as identifying real object-to-face scanning recognition [2]. AR functionality can usually be accessed using the same equipment as VR, while mobile phone camera capabilities have the potential to recreate AR experiences.

1.1 Challenges in Wireless Network Mapping

The command 'Netsh WLAN show interfaces,' used within Windows PowerShell, offers security information about the connected network. Yet, navigating through wireless networks presents various challenges, including signal interference, network congestion, and outdated security protocols. For example in an educational environment like school, signal strength might vary across different positions within a classroom, which could be solved with a time-consuming process of relocating to each spot individually, disconnecting at each position, and reconnecting to the same network two to three times while running the Netsh command. After each process, you could analyze the results of your position and signal strength and confirm the details of the command.

In the output of the command, the SSID displays the connected network name, the BSSID verifies the correct access point, crucial for network mapping across multiple routers

of the same network. Each physical router has its BSSID, however, when changed in the command could imply that you may have to recall the command and see how the output has been altered. Authentication and Cipher output is the network's security type of the network. Signal strength, varying from 1 to 100 percent, calculates the connection's availability in the position of your device that ran the Netsh command.

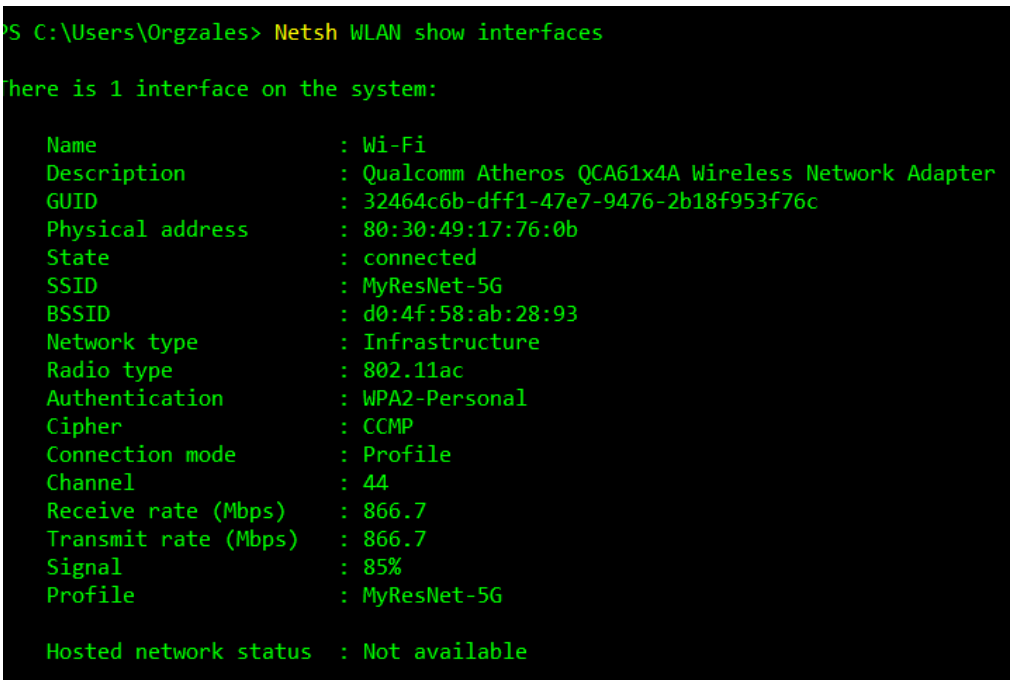

Figure 1: Netsh WLAN show interface Powershell Command Output

1.2 Chronicles of AR/VR Development (1956-2023)

Virtual Reality (VR) technology was introduced in 1956 by Cinematographer Morton Heilig's Sensorama - a booth that fitted up to four people [3]. This surrounded users with a multi-sensory experience, combining 3D colored video, vibrations, scents, audio, and environmental effects [3]. This was completed with primal technologies, including scent sprays, vibrating chairs, and stereo speakers, visualizing a futuristic cinematic experience at the time [3].

Figure 2: Morton Heilig's Equipment: Sensorama [3]

In 1968, the first Augmented Reality (AR) machine was the 'The Sword of Damocles,' a head-mounted display merging computer-generated graphics with the physical environment around the user [5]. The Sensorama and 'The Sword of Damocles' laid the foundation for VR/AR evolution, continually undergoing upgrades throughout the following decade. In 1979, VR surpassed its cinematic purposes, functioning as a pilot eye movement tracking for comparison to computer-generated images [3]. Later, in 1985, Jaron Lanier and Thomas Zimmerman's VPL Research, Inc. invested in and developed more user-adaptable VR hardware, introducing equipment like the DataGlove, EyePhone HMD, and the Audio Sphere [3].

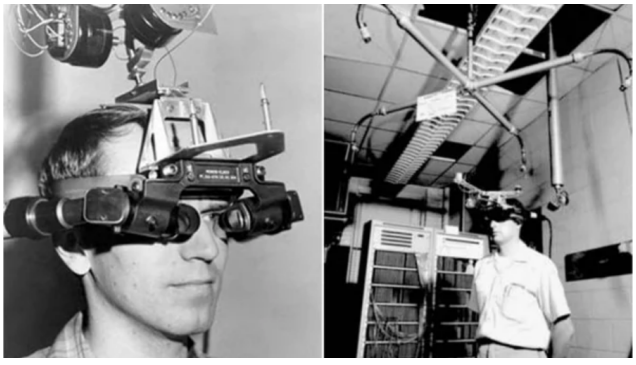

Figure 3: Ivan Sutherland's equipment: The Sword of Damocles [5]

Figure 4: Jaron Lanier and Thomas Zimmerman's VPL VR Equipment [3]

By 1989, NASA utilized astronaut training within simulated digital environments from VR equipment [4]. Nasa additionally included the integration of AR technology into their X-38 spacecraft's hybrid vision system in 1999, improving navigation during test flights [5]. The functionality of AR applications also made its way into vehicles in 2003 when Toyota launched an AR parking assistance backup cam in the Prius Hybrid model [6].

Figure 5: NASA's X-38 Spacecraft AR Haptics [5]

In 2012, A Facebook company called Oculus had a VR Kickstarter campaign for their new equipment called the Oculus Rift, leading to innovations from other rival companies like

PlayStation VR, Samsung Cardboard Phone VR, and the Steam VR headset in 2014 [3]. By 2016, VR was upgraded with better haptic interfaces and controllers, such as the HTC VIVE SteamVR headset, allowing sensor-based tracking for unrestricted free movement [3]. Near the same time, AR technologies, represented by devices like Google Glass and Windows Hololens, adopted compact mobile-friendly forms, offering improved travel and mobility [5].

Between 2017-2023, minor AR features were being placed into mobile device applications such as the popular Pokemon Go game and facial filters for most social media apps. [5]. However, in 2019 Oculus created the Oculus Quest, featuring complete sensor-based tracking within the controller handles and incorporated cameras into the headset itself for full environment tracking in the digital environment [7]. In 2020, Oculus Quest 2 (now Meta Quest 2), introduced AR Passthrough technology via cameras, enabling effortless navigation between VR and AR settings [7]. However, with the AR Passthrough feature being integrated with the Quest 2 by default, it was only capable without color.

Figure 6: Modified Photo of All Quest Equipment [9]

The recent unveiling on October 20, 2023, witnessed Oculus/Meta's launch of the Quest 3, containing enhanced processors and AR-capable colored cameras, a significant upgrade from the previous black-and-white AR Passthrough [8].

1.3 AR/VR/Mixed Reality of Unity Engines

Unity is a multi-platform game creation engine and developer environment used in the process of making video games, simulations, and interactive experiences across various devices and platforms by building based on their operating systems. Unity has allowed developers to build and create environments that evolve around 2D Dimensions, 3D Dimensions, augmented reality (AR), and virtual reality (VR) applications all in the same engines. Unity additionally provides an extensive amount of tools, features, and packages that are from its public asset store which allow users of the engine to create projects on a wide variety amount of devices such as websites, desktops, consoles, or Android/IOS devices. Among the Android devices, Unity has the feature to build to devices that are linked directly to the application such as the Quest, Quest 2, or other VR/AR equipment.

Numerous Unity versions are integrated with VR engines that are specialized for VR development such as access to realistic 3D environments, precise motion tracking kits and settings, VR hardware-specific packages, and Permission file and module access [10]. With developer kits such as Unity XR, projects can be optimized for VR headset compatibility and performance across many devices as long as they are supported with the specific installed version of Unity. Unity also contains AR development kits that have mobile phones or camera foundations to have control over an application that overlays virtual content over the physical environment the user is in. Unity's AR tools and features can contain object tracking, image recognition, and overlay or environment mapping [11]. With the combination of both VR and AR in a Unity building engine application, third-party toolkits have been created to be

imported into projects such as Mixed Reality.

Mixed reality applications use the resources of both VR and AR to craft a digital layer atop the physical world around the user [12]. For instance, they transform a physical table into a digital interface accessible within their device, allowing them to interact with it in a virtual space while it originates in the real world. There has been related work with Mixed reality tools such as creating and mapping a whole layout of the inner rooms of a home for gaming aspects.

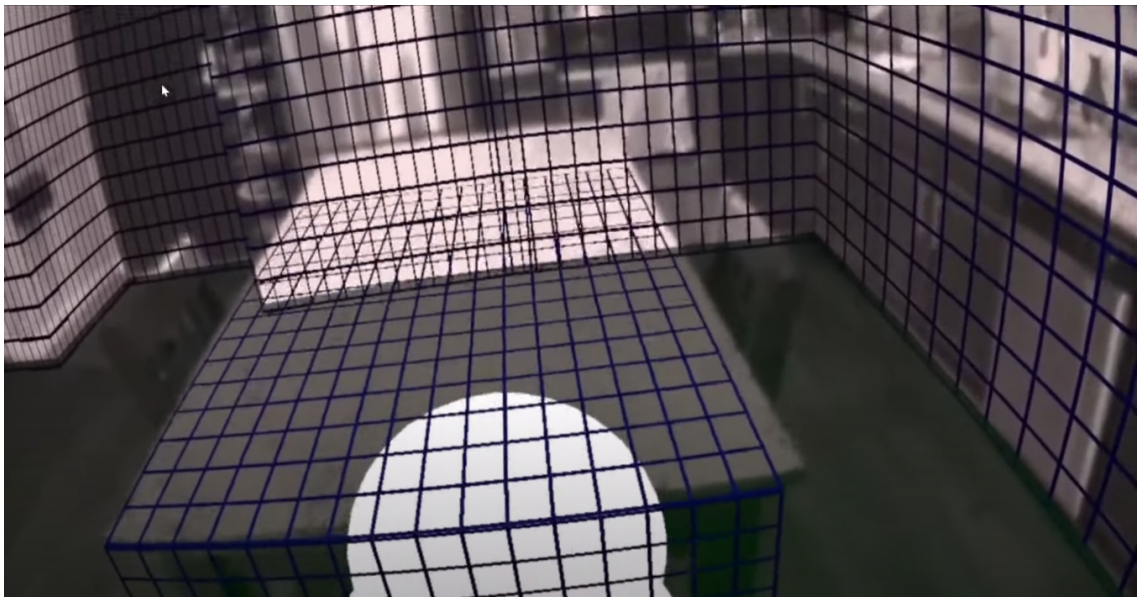

Figure 7: Clip Screenshot of Mapping a House with Mixed Reality [13]

2 Related Work

This research within the Department of Cybersecurity focuses on AR Mapping with wireless networks. Other projects within the same department could share common features such as security aspects, Unity development, or AR/VR resources this research has its differentials. Some related work sources are from Unity, GitHub, or other code source apps.

2.1 Cybersecurity VR Simulations

Several VR simulations have been developed since the release of modern equipment that is available for commercial purchase. These projects aim to offer a hands-on, secure, and realistic training environment for individuals in a specific field such as cybersecurity. For instance, 'VR for industrial cybersecurity training' stands as an example.

The primary goal of this project involves training users in the same digital online VR environment to interact and manipulate objects within the simulation [14]. These digital elements included action cards and gaming fields, intended to overcome in-person trainers or web-based calls [14]. This project's secondary approach was to maintain safety protocols during the COVID-19 pandemic in 2020 [14]. Results indicated that VR-based simulation training proved more effective in the users managing security within a virtual industrial setting compared to them shadowing managerial approaches or reading a company code of conduct [14].

Figure 8: Kaspersky Interactive Protection Simulation in Virtual Reality [14]

This study contrasts itself as a cybersecurity tool rather than instructing and training individuals on the features or protocols within the cybersecurity industry. Unlike the training simulation confined within their fixed digital environment, the AR wireless network mapper operates as an adaptable AR setting that is active in any location, delivering freedom of movement in all directions. While the industry training simulation demands a continuous internet connection for interaction with others, the project study functions independently, requiring no network connection for activation on the installed device. This function allows for movement beyond regular AR settings within a single floor, the possibility could range from multiple buildings to mapping across an entire campus.

2.2 iWifi App and WiFi Solver FDTD

This other older wifi mapper project involved an AR application designed for Android phones, attempting to transform an apartment space into an AR-based map highlighting the network signal waves of a router. The related work process was assisted by two other apps: WiFi Solver FDTD which simulated a map of Wi-Fi waves from the router's origin point, and iWifi to functions similarly to the Netsh Command to display the device signal strength in specific spots for Wi-Fi connectivity testing.

Figure 9: iWifi + Wifi Solver FDTD Signal Mapping Results [16]

The results of the project began with using the WiFi Solver FDTD to map a physical grid with tape, then conducting a long process of using the iWifi app to grab the signal strength within each laid-out square, and inputting the outcome on a printed paper of the layout grid [15]. Finally using AR capabilities, the results were hard-coded into the app and then revealed a grid layout in AR digital environment elements of the Android phone [15].

The AR wireless Mapper research project is to automate these procedures, enabling users to achieve mapping of desired areas from just the device. The device would also incorporate more extensive information beyond signal strength or dBm. It aims to showcase RSSI signal strength alongside SSID, BSSID, Authentication Type, and potential security vulnerabilities. Additionally, it will update the map to override any changed data, ensuring updated outcomes within the area.

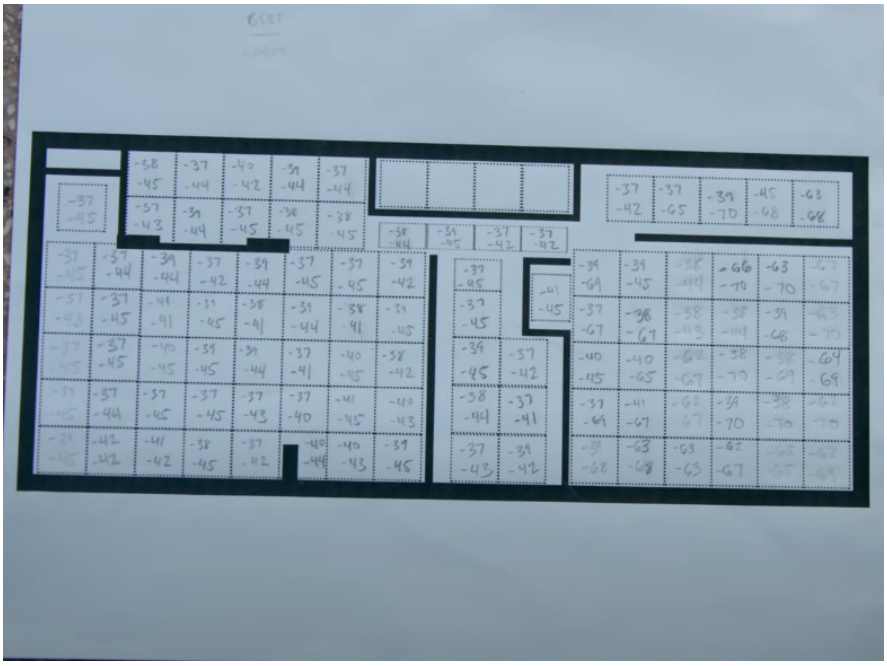

Figure 10: Physical Paper of Wifi Signals Written for AR Display [16]

3 Implementation

The Augmented Reality (AR) Wireless Network Security Mapper project was initiated by acquiring software such as Unity Version 2020.03, Unity Hub, a coding editing platforms like Visual Studios. Applications to assist testing would include the Oculus Home App, Oculus Mobile App, and Side Quest. Ideally, this research is suited for more adaptable AR gear similar to the Google Glass or Windows Hololens, however, testing and implementation were accomplished using the Oculus Quest 2/Meta Quest 2 VR Headset.

Several challenges during the project's launch included enabling wireless access for the Quest 2 headset to ensure unrestricted movement when testing the AR properties. Additionally, acquiring connected network details within an Android environment equivalent to Windows PowerShell Netsh command. The most definite challenge was validating this research as a cybersecurity tool such as functionality that alerts users of vulnerable routers or networks. Further challenges would arise later when crafting objects within an AR environment with Unity's Engines or attempting to pull methods with Unity's Android Engines.

3.1 Augment Reality Compatible Equipment

To have the Oculus Quest 2 VR headset compatible with our study, a series of initial steps were needed. The device would need developer mode active by creating a Meta/Facebook account for access to the Meta/Quest apps and linking the headset to our mobile device that had the mobile app downloaded. Accomplishing this would require logging into the official Oculus website with the account and creating a company organization under it. This approval would be needed for development with certain settings on our device. Enabling developer mode within the mobile app provided access to critical settings connected to our Quest 2, including toggling Boundary mode, permitting access for unknown devices,

activating Passthrough mode, and experimental hand tracking.

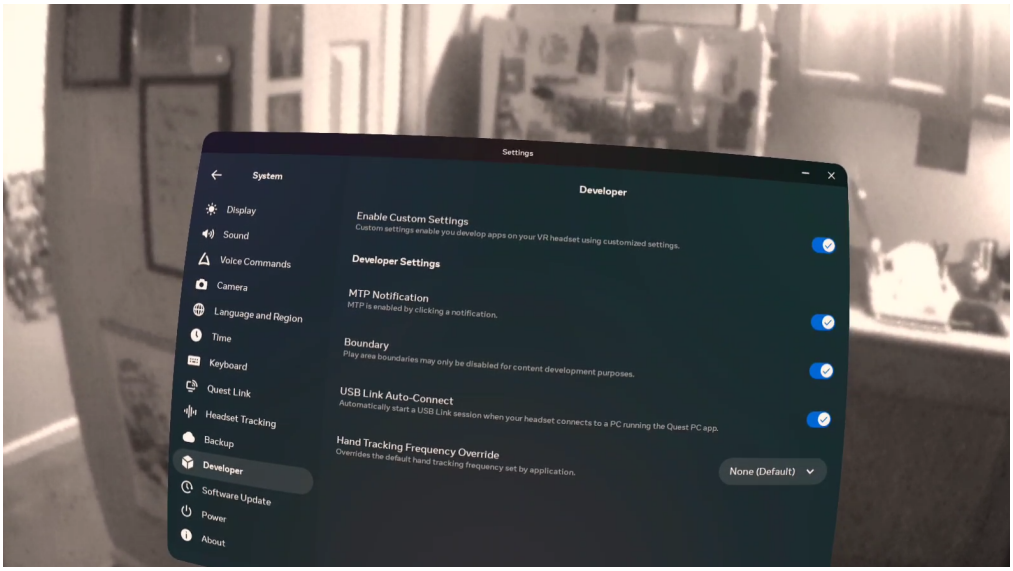

Figure 11: Augmented Reality Quest 2 Developer Settings Menu

Oculus Rift was needed for testing our code within our Unity application which was through the PC Oculus Home app and an Oculus Rift Official C-Type Data Transfer Cable. After configuring the Quest 2 to Rift-connect to our compatible PC we can allow our headset to be verified by our system. This would allow cross-access between our devices for platforms such as SteamVR test labs, Streaming our progress for later presentation, and importantly Unity VR/AR test applications and building.

Table 1: COMPATIBILITY CHART FOR SPECS AND QUEST RIFT [18]

Component	**Recommended Specs**	**Recommended Specs**
Processor	Intel i5-4590 / AMD Ryzen 5 1500X or greater	Intel i3-6100 / AMD Ryzen 3 1200, FX4350 or greater
Graphics Card	NVIDIA GTX 1060 / AMD Radeon RX 480 or greater	NVIDIA GTX 1050 Ti / AMD Radeon RX 470 or greater
Alternative Graphics Card	NVIDIA GTX 970 / AMD Radeon R9 290 or greater	NVIDIA GTX 960 4GB / AMD Radeon R9 290 or greater
Memory	8GB+ RAM	8GB+ RAM
Operating System	Windows 10+	Windows 10+
USB Ports	1x USB 3.0 ports	1x USB 3.0 ports
Video Output	Compatible DisplayPort video output	Compatible mini-DisplayPort video output (mini-DisplayPort to DisplayPort adapter included with Rift S)

3.2 Mixed Reality Capture Tool

With a new blank project in Unity Version 2020.03, the acquirement of packages was needed from the Mixed Reality Feature Tool. This toolkit imports settings and assets into a blank Unity project, assisting with foundations for projects that are applying to AR/VR/XR applications. Fundamental packages from this toolkit that were imported to our project are the Mixed Reality Toolkit Examples 2.8.3, Mixed Reality Toolkit Extensions 2.8.3, Mixed Reality Toolkit Foundation 2.8.3, and Mixed Reality Toolkit Standard Assets 2.8.3.

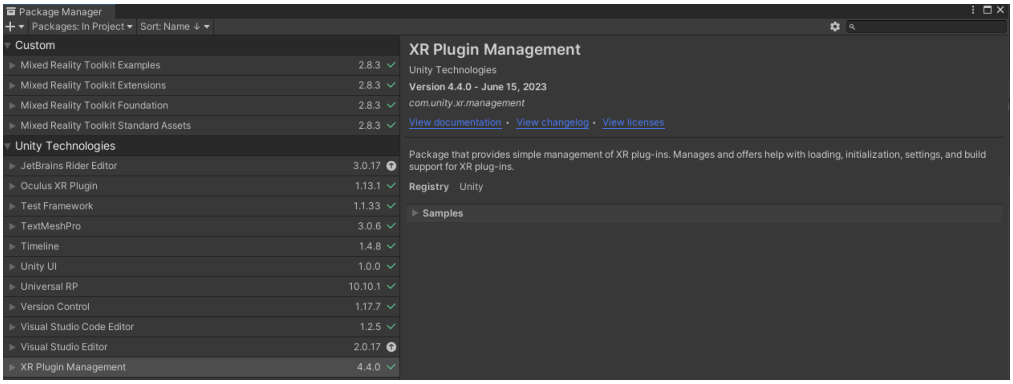

Figure 12: Unity Mixed Reality, Oculus, and XR Package Manager

Additional Unity packages Oculus XR Plugin and XR Plugin Manager were installed for the project's overall functionality. Oculus XR Plugin grants control settings to the Passthrough mode in Quest 2, allowing it to project the user's physical environment within Unity. The XR Plugin Manager enables hand tracking within Unity, eliminating the need for the controllers by having our Quest 2 cameras track the user's hands. Successfully importing these packages provides sample assets and Unity prefabs that we can make use of as presented in Figure 13

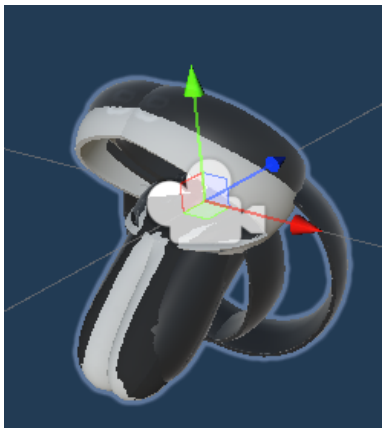

Figure 13: Imported Controller Prefab From Mixed Reality Package

3.3 Unity Android APK Builds w/ Side Quest

From the installation of the imported packages, a Mixed Reality sample scene was loaded into the Unity project to test AR functionalities within Quest 2. While the Oculus Rift connected to our PC through our Rift wire, many problems were encountered with the Passthrough mode and internet connection. These issues were solved by adding specific permissions to the Android manifest file which controls the project-approved functions, as outlined in Table 2.

Table 2: XML PERMISSIONS AND DESCRIPTIONS IN UNITY ANDROID MANIFEST

Permission Added	**Description**
INTERNET	Grants the app permission to access the internet.
RECORD_AUDIO	Permits the app to record audio.
MODIFY_AUDIO_SETTINGS	Allows the app to modify global audio settings.
ACCESS_WIFI_STATE	Allows the app to access information about Wi-Fi networks.
CHANGE_WIFI_STATE	Allows the app to change the Wi-Fi connectivity state.
ACCESS_NETWORK_STATE	Allows access to information about network connectivity.
ACCESS_FINE_LOCATION	Allows precise location information with VR resources.

With the successful outcome of testing with a hard-connected wire and Oculus Rift, the next focus was transferring the program into an APK file for installation on the Quest 2 device. To have proper configuration and settings for our APK build, access to Quest 2's files is granted to the Unity Engine. Further development of APK files and builds will have our Quest 2 device settings saved within the project.

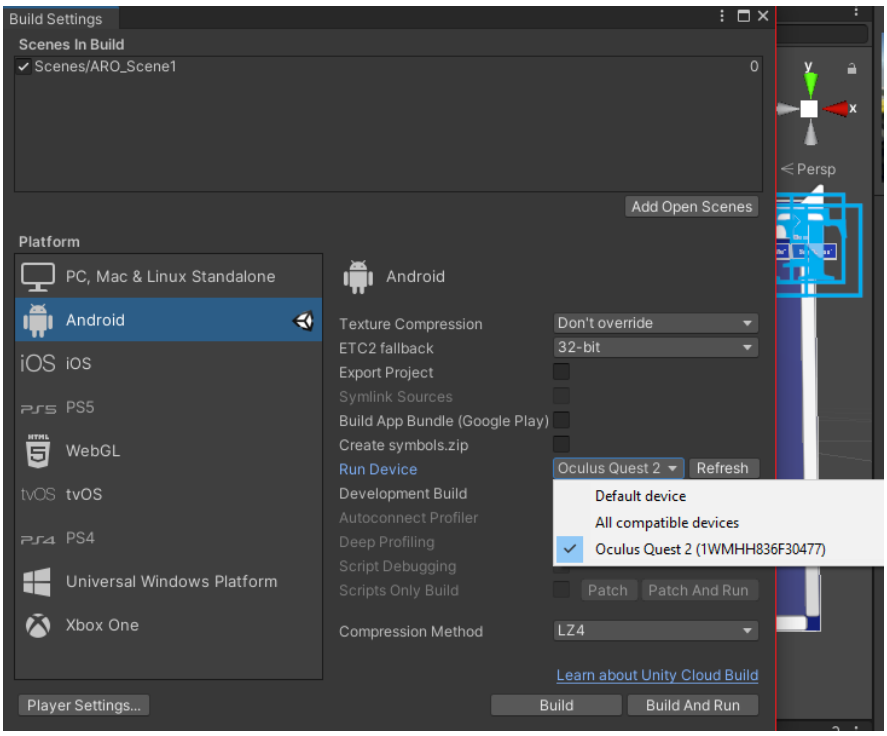

Figure 14: Oculus Device in Build APK settings

For direct installation for our built APK files, installation with account creation of Sidequest is needed for both the PC and Quest 2. Sidequest, a third-party application for our devices, allows files to be straightforwardly installed into Quest 2 after being allowed the same access as Unity. With the APK file dragged and dropped into our file folder on our Quest 2, we can initiate wireless testing with the same Unity project. The Quest 2 could now be tested with Mixed Reality assets in any location, along with the environment of AR.

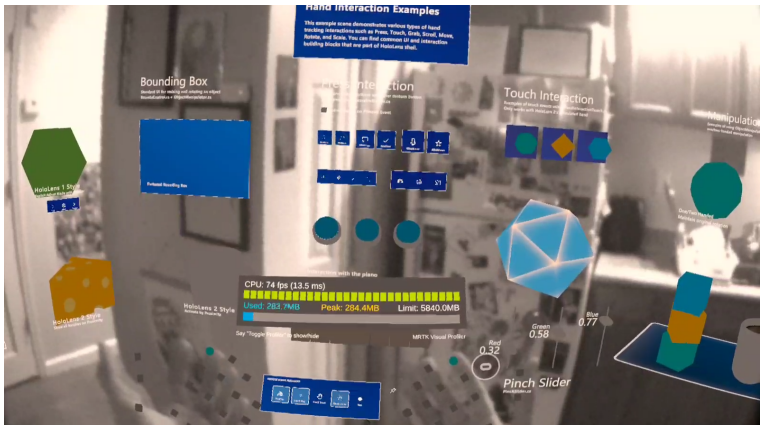

Figure 15: Mixed Reality Imported AR Sample Lab

3.4 Unity Android Engines and Methods

Saving the sample lab as APK 001, the project had the majority of unnecessary elements and assets like the chrome interface or the piano removed. For testing, a single text screen that trails the user, hand-tracking assets, and a CPU and GPU power gauge were kept. The text screen operated as a designated display for code outputs when obtaining network information and variables. Hand-tracking assets will be used in future functionalities such as interaction with prefabs and collision testing. Additionally, the device power gauge will help evaluate the power of the Quest 2 to avoid implementation crashes or lag.

Figure 16: Text Screen, Hand tracking, and Power Gauge for testing

Acquiring connected network information started with the creation of a Windows PowerShell terminal technique employing the NETSH command on our PC. While successful for our PC displaying network information and connectivity, installation of this method on the Quest 2 led to a crash. This outcome is due to the Quest 2's Android operating system not having NETSH commands within itself, unlike the Windows platform. To resolve this, the development of Android Java objects with individual named method calls that are compatible with the Quest 2 began. The desired results can be matched with a NETSH command can be achieved with abstracted objects in Table 3.

Table 3: COMPATIBLE ANDROID JAVA OBJECTS FOR QUEST 2

Method Object Name	**Description**
"currentActivity"	retrieves the reference to the current Android activity, allowing Unity to interact with the Android system.
"getSystemService", "wifi"	requests the Android system service responsible for managing Wi-Fi functionality.
"getConnectionInfo"	retrieve information about the current Wi-Fi connection.
"getSSID"	retrieve the SSID (Service Set Identifier) of the connected Wi-Fi network.
"getBSSID"	retrieve the BSSID (Basic Service Set Identifier) of the connected Wi-Fi network.
"getRssi"	retrieve the received signal strength indicator (RSSI) of the connected Wi-Fi network.

Once variables that were needed were acquired, the implementation shifted to integrating them into Unity's text objects, enabling the information upon network connection to change with new connections. However, location network details required repeated disconnections and re-connections. The process to get the connection was used with the Update() method for retrieval per frame, but the execution led to a crash in Quest 2 from seeing the CPU and GPU gauge skyrocket. To address this, the code utilizing Unity's WaitForSeconds(1.0f) and InvokeRepeating() methods instead of Update(), limiting code execution to

every X seconds. With adjustments, a balance of swift data return, and constant smooth frames without overwhelming the Quest 2 was invoking every 6 seconds.

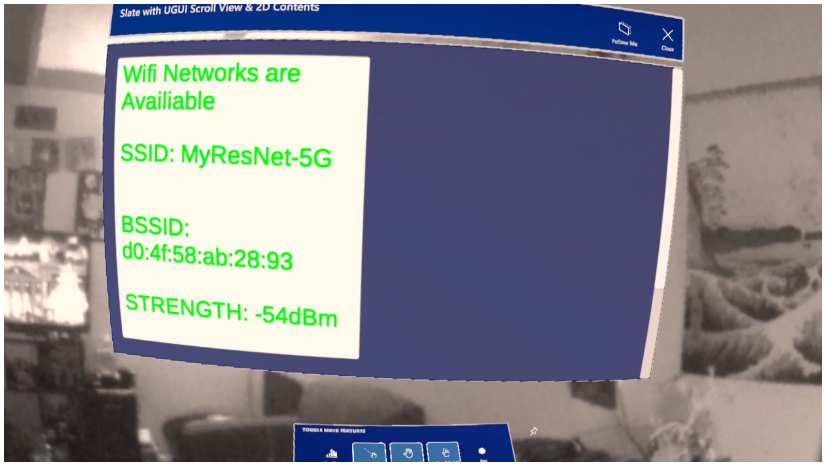

Figure 17: Test Output of Network Details

In parallel, another script was implemented for the mapping segment of the research. This script executed also every 6 seconds, would spawn a modified mixed reality prefab (just a cube for now) in the user's position and stay as the application ran to identify where they've connected to the network. Additionally, as these cubes mapped out the area, their prefab color would be based on the getRssi variable of dBm signal strength.

Table 4: "getRSSI" dBm SIGNAL STRENGTH

Strength of dBm	**Definition of Signal Strength**	**Color for dBm**
-67dBm or greater	Great connection / Full bars	Green
-68dBm to -79dBm	Good or OK connection / half bars	Yellow
-80dBm or less	Bad or No connection / No bars or 1	Red

These cubes also retained the network information displayed at the time allowing users to revisit and access prior data. To maintain an organized digital AR space, zone restrictions were implemented in Unity, allowing cubes to only spawn when they are X distance from each other.

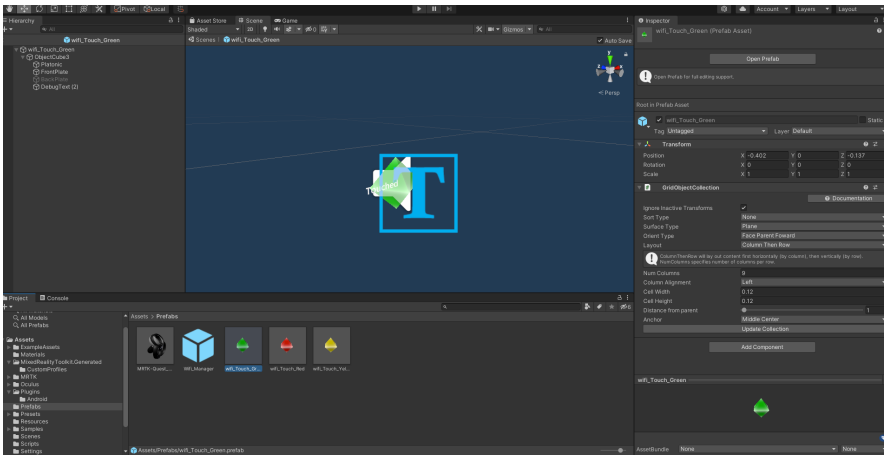

Figure 18: Unity Application w/ Cube Prefab

Advancing the APK development, the Quest 2 still lacked a dedicated cybersecurity function to map the network connections within AR. Within the Android variables, it was challenging to find the correct combination of method names that were Quest 2 compatible to retrieve the authentication or security type of the connected network. Resorting to resources such as ChatGPT and advanced GitHub repository searches, eventually 'Security type:' was the correct method call. With this method, a discovery of the Quest 2 was found that the security type was returned as an integer value. By constructing a switch statement to interpret the value sent by Quest 2, identified security protocols were functioning within the research project. With other assets, a new Unity text object can be adaptable to changing network connections as the following Table 5.

Table 5: QUEST 2 CUSTOM SECURITY RETURN TYPE CHEAT SHEET

Int Value	**Auth Type**	**Unity Color**	**Security Definition**
-1	No Security/Signal	Color.red	No connected network
0	OPEN	Color.red	Dangerous
1	WEP	Color.yellow	Old & Cautious
2	WPA/WPA2	Color.green	Safe Security
3	WPA3	Color.blue	Very Safe Security
3+	Unknown	Color.magenta	Unknown Security

Security type details are implemented into our prefab objects marking our project as a cybersecurity tool. This equips the user with a reliable tool to gauge both the strength and security aspects of connected networks within AR technology. Improving the visualization of the prefabs will remain a future challenge until we have enough functionality secured in the code.

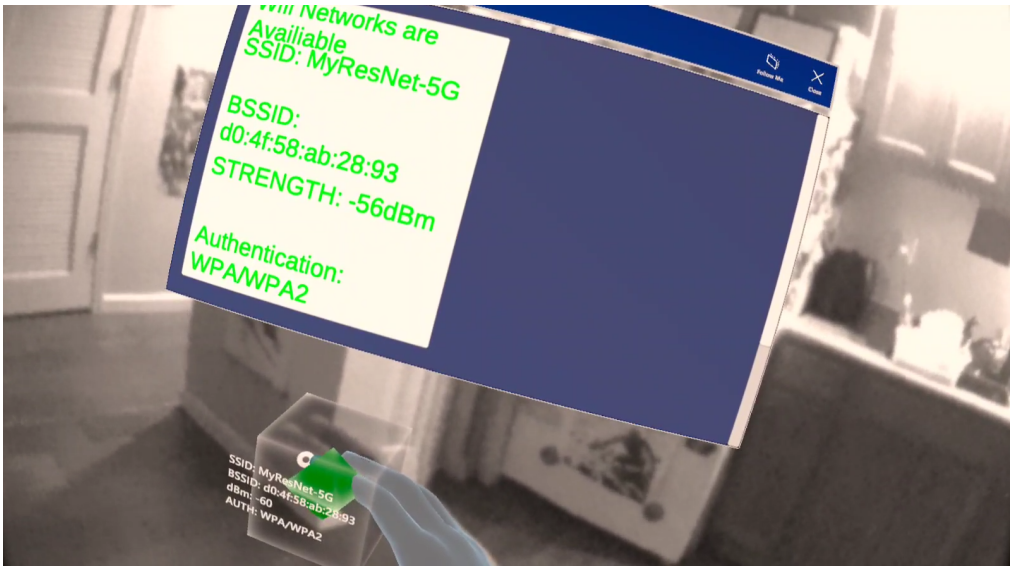

Figure 19: Output Results w/ Authentication & Prefab Details

3.5 Router Authentication Security

To verify the Authentication feature, testing will occur with an intentionally configured old Netgear83 router to operate with vulnerable security types. When using the application to switch between networks in our device settings, moving from a WPA/WPA2 network displayed in the green color text indicating good security to the open Netgear83 network which displayed a security risk with red text as intended.

Upon further testing, an accident led to code that could assess a router's capabilities beyond its current set authentication status. This function allows the project to check not only the existing authentication setting but also the maximum security potential of a router.

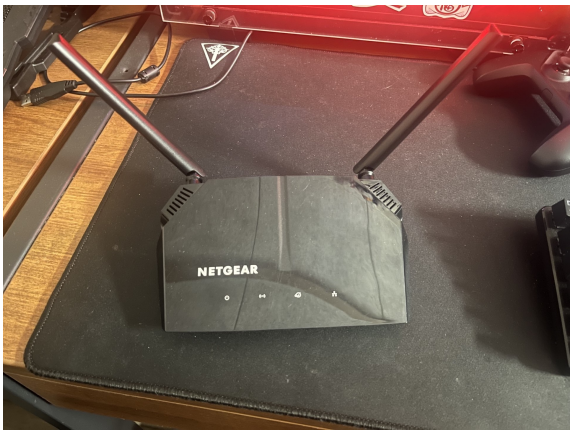

Figure 20: Netgear83 Router w/ Vulnerable Security

As an example, The Netgear 83 router was set to OPEN for testing purposes, however, the function can confirm that the router can be switched to a higher security mode of WPA2 based on a specific numeric bool value result returned by the router, as detailed in Table 6.

Table 6: ANDROID ENGINE "hasCapability" CHEAT SHEET

Network Capabilities Value return	**Security ValueType Definition**
<bool>("hasCapability", 12) = if TRUE	Router Can have internet Access
<bool>("hasCapability", 15) = if TRUE	Route Can have WEP security active
<bool>("hasCapability", 13) = if TRUE	Route Can have WPA/WPA2 security active
<bool>("hasCapability", 6) = if TRUE	Route Can have CCMP security active
<bool>("hasCapability", 26) = if TRUE	Route Can have WPA3 security active

4 Results

This project has surpassed the capabilities of a standard PC or Linux machine to accomplish the outcome within the AR environment. Designed ideally for advanced AR equipment, this study has successfully displayed a mapping of network details based on the user's location every 6 seconds. The application has an area covered with cube prefabs that mark the user's past locations along with the details based on their creation. Additionally, they change color to represent signal strength and authentication types to serve as a visual

indicator for this cybersecurity tool. The position of the cubes is placed at a certain distance from the user using Unity's Android Engine code, avoiding obstruction to their view. Although attempts were successful to place the cubes under the user near hip level, the X and Z axes were determined by the user's initial facing direction at the program's launch so placing the prefab in front of the user by a foot needs more additional implementation.

Alongside, the function that organizes the digital mapping area, Quest 2's Mixed Reality AR function allows cubes to be visible through walls and even across multiple floors. For example, prefabs created on the third floor are noticeable from the first floor of a building. Finally, the program runs in the background while switching between the network in the Quest 2's settings, however, it can function independently without requiring any network connection. This AR function offers the possibility to test network limitations, extending beyond walls, leaving outside, or across different buildings.

5 Conclusion

Essential network details like SSID, BSSID, and RSSI, this research project detects authentication vulnerabilities to notify the user about the security status of connected routers and wireless networks. With more testing with older routers limited to WEP, WPA, and WPA3 protocols, functions can identify the maximum security capacity of the connected network. Additional implementation will focus on spawning cubes one meter ahead of the user based on their initial facing direction. An updated prefab design is overdue for the research project, having the cubes include color-coded hex values based on connection strength in dBm. Additional functionality will include the following for the prefabs: Shaped based on security type, Height based on signal strength, and transparent radius to create better mapping visualization.

Much further development goals include creating a new prefab to spawn when a new access point or router is discovered on the same network. To maintain previous mapping efforts, a saving mechanism will be introduced to the research. This feature is to allow users to reload the application and access the same mapped area they created earlier. The anticipated outcome and results for next semester is for the device to generate a 3D digital map of the scanned area to export to other devices connected within the reach of the Meta/Oculus Quest 2 account.

Figure 21: Current APK Build containing all Implementations & Functionality

References

[1] Fitzpatrick, J. (2016, September 21). The Difference Between WEP, WPA, and WPA2 Wi-Fi Passwords. How-To Geek. https://www.howtogeek.com/167783/htg-explains-the-difference-between-wep-wpa-and-wpa2-wireless-encryption-and-why-it-matters/

[2] Greenwald, W. (2023, June 6). Augmented Reality (AR) vs. Virtual Reality (VR): What's the Difference? PCMAG. https://www.pcmag.com/news/augmented-reality-ar-vs-virtual-reality-vr-whats-the-difference

[3] Barnard, D. (2023, June 14). History of VR - Timeline of Events and Tech Development. VirtualSpeech. https://virtualspeech.com/blog/history-of-vr

[4] Loeffler, J. (2021, September 30). The History and Science of Virtual Reality Headsets. Interesting Engineering. https://interestingengineering.com/innovation/the-history-and-science-of-virtual-reality-headsets

[5] Poetker, B. (2023, August 9). A Brief History of Augmented Reality (+ Future Trends and Impact). G2. https://www.g2.com/articles/history-of-augmented-reality

[6] Bindu, Y. (2021, November 11). Park Assist Technology: A History. Get My Parking Blog. https://blog.getmyparking.com/2021/11/11/park-assist-technology-a-history/

[7] Heather. (2023, August 6). Oculus: A Complete Guide — History, Products, Founding, and More. History Computer. https://history-computer.com/oculus-history/

[8] Moore-Colyer, R., and McMillan, M. (2023 November 12). Meta Quest 3 release date, price, specs, and latest news. Tom's Guide. https://www.tomsguide.com/news/oculus-quest-3-release-date-rumors-specs-news

[9] Bezmalinovic, T. (2023, November 19). Is Meta Quest 3 subsidized? One analysis suggests so. Tom's Guide. https://www.tomsguide.com/news/oculus-quest-3-release-date-rumors-specs-news

[10] XR Terra. (2020, October 9). Developing for VR with Quest 2 and Unity for the First Time – A Step-by-Step Guide. https://www.xrterra.com/developing-for-vr-with-quest-2-unity-for-the-first-time-a-step-by-step-guide

[11] Unity Technologies. (n.d.). AR Overview. https://docs.unity3d.com/Manual/AROverview.html

[12] Biba, J. (n.d.). What Is Mixed Reality? Built In. https://builtin.com/media-gaming/mixed-reality

[13] DavidMVRGo. (2022, November 2). Hauntify Mixed Reality Full Setup and Gameplay Speed Run Video. YouTube. https://www.youtube.com/watch?v=OU_wi40JFDk

[14] Pankov, N. (2021, June 10). Virtual Reality (VR) for industrial cybersecurity training. https://www.kaspersky.com/blog/vr-interactive-simulation/40188/

[15] Farsace, B. (2020, July 21). I mapped my entire apartment's Wi-Fi signal. The Verge. https://www.theverge.com/science/2020/7/21/21328522/wifi-mapping-signal-physics-science-isp-router-tips-tricks-verge-science

[16] Verge Science. (2020, July 21). I mapped my entire apartment's Wi-Fi signal [Video]. YouTube. https://www.youtube.com/watch?v=6ADqAX-heFY

[17] Wright, T. (2015). Game Development for an Augmented Reality System. Stetson University Archives. https://archives.stetson.edu/digital/collection/Research/id/26589/rec/8

[18] Meta. (2023). Oculus Rift S and Rift minimum requirements and system specifications. https://www.meta.com/help/quest/articles/headsets-and-accessories/oculus-rift-s/rift-s-minimum-requirements/

[19] Orgzales / Orion Gonzales. (2023). AUGMENTED REALITY WIRELESS NETWORK SECURITY MAPPER. https://github.com/Orgzales/Unity-Ar-Project