Mobile Device GNSS Spoofing Detection

Nathan Johnson
Cybersecurity Department
Embry-Riddle Aeronautical University
Prescott, Arizona, USA
johnsn63@my.erau.edu

***Abstract*—Global Navigation Satellite Systems (GNSS) play a critical role in modern infrastructure, from aviation and maritime navigation to time synchronization for cryptographic systems. However, the increasing reliance on GNSS has made it a valuable target for cyber threat actors, particularly through GNSS spoofing attacks. Unlike jamming, which simply disrupts GNSS signals, spoofing can deceive systems into using false positional data, leading to potentially severe consequences. This paper explores a novel approach to GNSS spoofing detection using mobile devices, leveraging onboard GNSS receivers and inertial sensors. Initial findings reveal significant limitations in consumer-grade accelerometers and gyroscopes for implementing a full Inertial Navigation System (INS). However, a simpler detection method based on monitoring deviations in GNSS position over time was developed. While this approach shows promise, false positive detections due to inherent GPS inaccuracies remain a challenge. Future work will focus on refining detection methods, potentially incorporating machine learning and noise reduction techniques to improve reliability.**

I. INTRODUCTION

Global Navigation Satellite Systems (GNSS) are an invisible and often overlooked yet critical piece of infrastructure. Many aspects of modern life rely on GNSS systems like the American GPS constellation or other systems like the Russian GLONASS constellation, EU Galileo constellation, or Chinese BeiDou constellation. GNSS systems fall under the broader category Position Navigation and Timing (PNT) systems which are used for military, aviation, maritime, and civilian navigation, but also for some unexpected purposes like synchronizing clocks around the world which plays a critical role in synchronizing the internet as well as time based cryptographic systems [1]. GPS based navigation in aviation is quickly making ground based forms of navigation obsolete due to its high flexibility and lower cost of implementation and maintenance. It is very common now for aircraft in instrument flight conditions, to be relying solely on the GPS system for navigation. Accurate GNSS service has also become critical for performing basic flight functions and stabilization in modern drones [2].

As GNSS systems become more relied upon, their value as potential targets for cyber threat actors also grows. In recent years, GNSS jamming and spoofing attacks have become more and more prevalent, especially in areas with high geopolitical tension such as Ukraine and Iran. GNSS jamming is a very simple form of denial of service attack involving the use of a radio transmitter to overpower the GNSS signals with noise. This is very easy to do as the frequencies which GNSS systems operate on are very well known and the satellites' signals are very weak since they are being sent all the way from space. GNSS jamming, however, is not nearly as big of a threat as GNSS spoofing. This is because it is extremely easy for systems to detect a total loss of GNSS signal and to gracefully fail over to a backup system. In the case of GNSS spoofing, a threat actor will overpower the legitimate GNSS satellites by transmitting false signals impersonating satellites. If done properly, the transition from real to spoofed GNSS signals can be extremely difficult to detect, leading systems to falsely trust and continue to operate using data being fed in from a malicious actor. Most targeted GNSS spoofing attacks work by replaying the accurate GNSS signals and then gradually applying a deviation to prevent a sudden detectable jump in position.

Several approaches to GNSS spoofing detection have been proposed. Power level based detection methods use the signal strength to try and determine when satellite based signals are being overpowered by stronger terrestrial signals, but as attackers become more advanced and carefully tune their transmit power, this method is becoming more difficult to use. Most other methods of detection rely on a comparison of the received GNSS signals to some known good position information. This includes ground based stations at a known location checking for GPS error. Another method proposed for aviation and drones is to make use of the aircraft's onboard accelerometers and gyroscopes. Almost all modern aircraft have an Inertial Navigation System (INS). This system operates using a known starting position and then continuously integrating extremely precise data from the accelerometers and gyroscopes to calculate how the aircraft's position has changed over time. Due to the nature of these systems, they can only be used in isolation for a limited time before the accumulation of error renders them unreliable and they need to be periodically reset using GPS position. It is theoretically possible, however, to compare the relative velocities calculated by the INS and GPS. If an unusually high difference was noted, it could be evidence that spoofing is taking place [2].

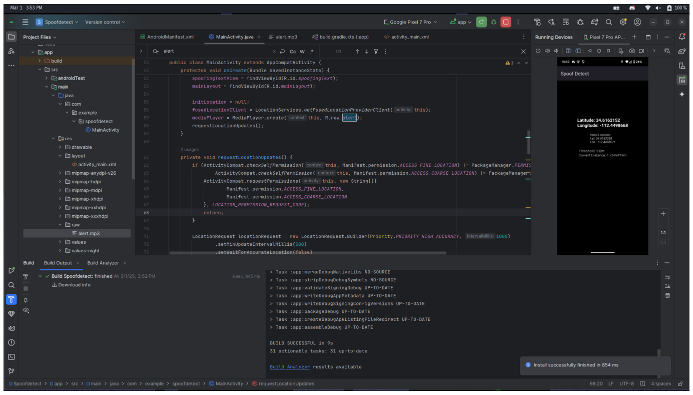
For my CI 515 project, I have decided to take advantage of mobile devices having both onboard GNSS receivers and accelerometers and gyroscopes to explore a possible implementation of this detection method.

II. Related Work

Several studies have been done assessing the effectiveness of GNSS spoofing detection by comparing GNSS position to inertial position, including a 2021 study utilizing the accelerometers and gyros onboard drones [2]. However, I am not aware of any studies that have attempted this technique using mobile devices.

III. Tools and Methodology

I have decided to implement my prototype mobile GNSS spoofing detection system in an android application. I am using Android Studio as my IDE and have been developing the app in Java as I have more experience in Java than Kotlin. I am currently developing using a Google Pixel 7 Pro and its onboard inertial systems and GPS. My initial approach was to try and implement a basic INS system in the app and then to compare the calculated latitude, longitude and altitude to what the GPS indicated. One of my fears going into this project is that the quality of the sensors in consumer mobile devices is not up to the same standard as those that are installed in aircraft and drones.



```
public class MainActivity extends AppCompatActivity {
    protected void onCreate(Bundle savedInstanceState) {
        spoofingTextView = findViewById(R.id.spoofingText);
        mainLayout = findViewById(R.id.mainLayout);

        initLocation = null;
        fusedLocationClient = LocationServices.getFusedLocationProviderClient( activity: this);
        mediaPlayer = MediaPlayer.create( context: this, R.raw.alert);
        requestLocationUpdates();
    }

    2 usages
    private void requestLocationUpdates() {
        if (ActivityCompat.checkSelfPermission( context: this, Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMIS
                ActivityCompat.checkSelfPermission( context: this, Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager
            ActivityCompat.requestPermissions( activity: this, new String[]{
                    Manifest.permission.ACCESS_FINE_LOCATION,
                    Manifest.permission.ACCESS_COARSE_LOCATION
            }, LOCATION_PERMISSION_REQUEST_CODE);
            return;
        }

        LocationRequest locationRequest = new LocationRequest.Builder(Priority.PRIORITY_HIGH_ACCURACY,  intervalMillis: 1000)
                .setMinUpdateIntervalMillis(500)
                .setWaitForAccurateLocation(false)
```

```
> Task :app:mergeDebugNativeLibs NO-SOURCE
> Task :app:stripDebugDebugSymbols NO-SOURCE
> Task :app:validateSigningDebug UP-TO-DATE
> Task :app:writeDebugAppMetadata UP-TO-DATE
> Task :app:writeDebugSigningConfigVersions UP-TO-DATE
> Task :app:packageDebug UP-TO-DATE
> Task :app:createDebugApkListingFileRedirect UP-TO-DATE
> Task :app:assembleDebug UP-TO-DATE

BUILD SUCCESSFUL in 9s
31 actionable tasks: 31 up-to-date

Build Analyzer results available
```

Install successfully finished in 854 ms

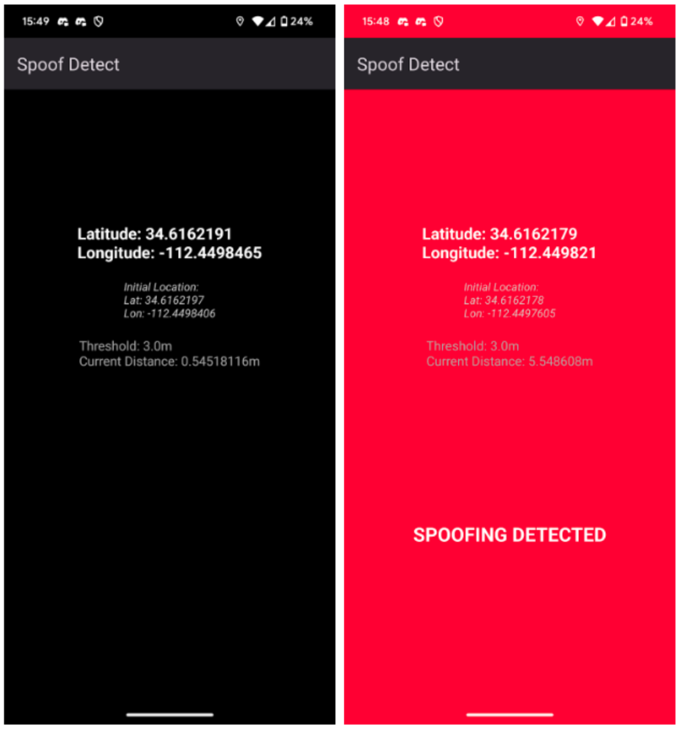
Fig. 1. Screenshot of source code in Android Studio.

IV. Progress and Initial Findings

So far, I have been unable to get accurate enough data from the phone's sensors to be useful. The GPS position, which updates once per second, varies by several meters on each update, which is not terrible, but will make it impossible to detect very fine spoofing attacks.

I have also determined that the phone's accelerometers are nowhere near accurate enough to keep track of the phone's position in real time. While I never fully implemented an INS, I did implement a basic test system for tracking the phone's position. I aligned the phone face up along a North-south axis. I then used the phone's x-y accelerometers. I performed numerical integration twice on the x and y acceleration to integrate from acceleration to velocity and then to position. This change in position was then added to the phone's running estimate of its latitude and longitude. It became immediately clear that inaccuracy of the accelerometers was accumulating rapidly, resulting in useless data. This was extremely disappointing, but I expected this to be the case.

My new strategy is to assume that the phone is to remain stationary. The current version of the app will first wait for the gps to acquire signal and then stabilize. It will then store this initial position and this will be trusted to be the phone's true position. The GPS position will then be checked every second and the distance between the measured position and the initial position is calculated. If the distance between these two measurements exceeds a set tolerance, currently 3 meters, the app will register a detection of GPS spoofing. Due to the inaccuracy of the consumer grade GPS receiver in the phone, this is still not effective. The measured position from the phone's GPS varies widely within a range up to about 10 meters leading to many false positive detections. The accuracy does improve when taking the phone outdoors, but not by much.



15:49

Spoof Detect

Latitude: 34.6162191
Longitude: -112.4498465

Initial Location:
Lat: 34.6162197
Lon: -112.4498406

Threshold: 3.0m
Current Distance: 0.54518116m

15:48

Spoof Detect

Latitude: 34.6162179
Longitude: -112.449821

Initial Location:
Lat: 34.6162178
Lon: -112.4497605

Threshold: 3.0m
Current Distance: 5.548608m

SPOOFING DETECTED

Fig. 2. Screenshot of Spoofdetect Android app.

V. Next Steps

For the second half of this project, I would still like to create a functional GNSS spoofing detection system. I have not experimented with any machine learning models which may be a cool direction to go if there is time. I will definitely attempt to implement some type of noise reduction, such as keeping a moving average of the GPS position rather than the raw sensor data.

VI. Conclusion

The investigation into GNSS spoofing detection using mobile devices has demonstrated both the potential and limitations of consumer-grade hardware for this application. While initial attempts to implement an inertial navigation-based detection system proved infeasible due to sensor inaccuracies, an alternative method using position deviation tracking was developed. The primary challenge remains the inherent imprecision of mobile GPS receivers, leading to false positive detections. Moving forward, the implementation of noise reduction techniques, such as moving averages, may help refine detection accuracy. Additionally, incorporating machine learning to identify spoofing patterns could further enhance reliability. As GNSS spoofing continues to grow as a cybersecurity threat, developing accessible and effective detection methods remains an important pursuit.

References

[1] Q. Lu, X. Feng, and C. Zhou, "A detection and weakening method for gnss time-synchronization attacks," *IEEE Sensors Journal*, vol. 21, no. 17, pp. 19 069–19 077, 2021.

[2] L. Meng, L. Yang, S. Ren, G. Tang, L. Zhang, F. Yang, and W. Yang, "An approach of linear regression-based uav gps spoofing detection," *Wireless Communications and Mobile Computing*, vol. 2021, no. 1, p. 5517500, 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1155/2021/5517500

[3] S. C. Bose, "Gps spoofing detection by neural network machine learning," *IEEE Aerospace and Electronic Systems Magazine*, vol. 37, no. 6, pp. 18–31, 2022.