The Core Idea & Contribution

At its heart, this research solves a fundamental integration problem in modern 5G networks: how to securely connect the vast number of Wi-Fi-only devices that lack 5G credentials and therefore cannot be authenticated or managed by the 5G Core (5GC) using standard procedures.

The proposed solution is a practical, non-disruptive framework centered on an intelligent 5G Residential Gateway (5G-RG). This gateway acts as a mediator, performing a two-step process:

1. It first authenticates devices on the local network using a strong, standard-based security protocol, EAP-TLS.

2. Then, for each successfully authenticated device, the gateway leverages its own 5G credentials to establish a unique PDU Session with the 5GC.

This PDU Session effectively becomes a "proxy identity" for the Wi-Fi-only device. From the 5GC's perspective, it is simply managing another data session for a legitimate UE (the 5G-RG). From the device's perspective, it has undergone a standard network authentication. This elegant abstraction allows for individual device management, traffic isolation, and policy enforcement within the 5G ecosystem, all without requiring any modification to the 5G Core's network functions or the firmware of the end devices.

The Problem & Its Significance in Detail

The Technical Gap

3GPP standards define how 5G-capable devices authenticate. However, a class of devices defined as NAUN3 (Non-Authenticable Non-3GPP) have no mechanism for direct 5GC authentication. They are "invisible" to the core network's identity management framework.

Critique of Current 3GPP Approaches

CGID (Connectivity Group ID)

This 3GPP concept allows for grouping devices behind a gateway (e.g., by SSID) onto a single, shared PDU session. While useful, it's a blunt instrument. It prevents the network from applying specific security rules, QoS profiles, or traffic policies to an individual device within that group. All devices are treated as one.

Release 19 Developments

More recent 3GPP work introduces the "Non-3GPP Device Identifier," which allows for distinguishing traffic from specific devices for QoS purposes. However, this is primarily a mechanism for traffic differentiation, not for authentication. It assumes the device is already "known" to the gateway but doesn't standardize a secure method for how that initial trust is established. My work addresses this prerequisite authentication step.

Why It Matters

The inability to integrate these devices individually is a major bottleneck for true 5G-Wi-Fi convergence. Enterprises need to apply granular security policies to IoT devices. Residential users could benefit from per-device parental controls or prioritized bandwidth for a home-office computer. My solution enables these use cases by giving each device a manageable network identity.

The Proposed Solution: A Deeper Dive

The framework's logic is centralized entirely within the 5G-RG to maximize transparency and minimize disruption.

Pillar 1: Secure Local Authentication (EAP-TLS)

- **Components:** The authentication process involves the Supplicant (the Wi-Fi-only device running wpa_supplicant), the Authenticator (the 5G-RG running hostapd), and the Authentication Server (an operator-managed server running FreeRADIUS).
- **Process Flow:** The 5G-RG doesn't terminate the EAP session; it acts as a relay. It packages EAP messages from the device into RADIUS packets and sends them over a dedicated "backhaul" PDU session to the FreeRADIUS server.
- **Security:** EAP-TLS is used because it provides mutual, certificate-based authentication, which is far stronger than shared secrets or passwords. It establishes a "zero-trust" style of security at the network edge.

Pillar 2: Identity Management (Proxy PDU Sessions)

- **The "Interceptor" Application:** This is the custom Go application I developed, which acts as the brain of the 5G-RG. It connects to hostapd's control interface to listen for events.
- **Trigger and Action:** When the Interceptor sees a CTRL-EVENT-EAP-SUCCESS message for a device, it knows that device is trusted. It then immediately executes a command via the UERANSIM nr-cli tool to request a new PDU session from the 5GC on behalf of that device.
- **The Proxy Identity in Practice:** The 5GC's SMF allocates resources and an IP address for this new PDU session. The UERANSIM stack on the 5G-RG creates a corresponding virtual network interface (e.g., uesimtun1, uesimtun2). This interface is the proxy identity — a tangible network endpoint through which the device's traffic will now flow.

Pillar 3: Traffic Management (Policy-Based Routing)

- **The Challenge:** The 5G-RG now has multiple active PDU sessions (one for backhaul, and one for each authenticated device). It needs a robust way to direct the right traffic to the right session.
- **Technical Implementation:** The Interceptor automates the configuration of iptables and ip-route2 to achieve this:
    - **Marking (iptables -t mangle):** An iptables rule in the PREROUTING chain marks every packet arriving from the device's unique MAC address with a unique identifier (e.g., the PDU session

ID).

  - **Policy Routing (ip rule):** A routing policy rule is created that says, "any packet with this mark must use a specific routing table."
  - **Dedicated Route (ip route):** A custom routing table is created that contains only one rule: a default route pointing to the PDU session's virtual interface (uesimtunX).
  - **NAT (iptables -t nat):** A MASQUERADE rule in the POSTROUTING chain performs NAT on all traffic exiting that virtual interface, replacing the device's local IP with the 5G-Core-assigned IP for that session.

Validation & Key Results

The testbed was built using Vagrant to automate the setup of a multi-VM environment running Open5GS (5GC) and UERANSIM (gNB/UE).

- **Result 1: Successful & Isolated Onboarding:** Tests confirmed that for each of two connecting devices (naun301, naun302), a unique PDU session was created (PDU Session2 with IP 10.46.0.2 and PDU Session3 with IP 10.46.0.3). The ping -R command provided definitive proof of traffic isolation, showing the route for each device was correctly NAT'd through its distinct proxy IP.
- **Result 2: Full Lifecycle Management:** When a device was disconnected, the Interceptor's logs showed it detecting the stale ARP entry. It then correctly executed the cleanup sequence: deauthenticating the client via hostapd, removing its routing rules from iptables, and executing nr-cli ps-release to terminate the PDU session. Subsequent checks confirmed the session was gone from the 5GC and the virtual interface was removed.
- **Result 3: Traffic Segregation:** Packet captures confirmed that RADIUS/EAP control traffic was exclusively transported over the backhaul PDU session, while iperf3 user data from the devices was transported over their respective clients PDU sessions.

Potential Questions from the Jury (and How to Answer Them)

Here are some questions you might face, with suggested answer strategies.

Q1: Your solution relies on creating a full PDU session for every single Wi-Fi device. Isn't this inefficient and a waste of network resources compared to the 3GPP approach of using a shared session?

Answer Strategy: Acknowledge the premise but reframe it around value and capability.

Suggested Answer: "That's an excellent point. While creating a PDU session per device is more resource-intensive upfront than a shared session, it's a trade-off for gaining a level of security, isolation, and manageability that is impossible with a shared model. With a dedicated session, the network operator can apply per-device policies, enforce granular QoS, monitor individual traffic patterns, and completely isolate one device from another—which is critical for a secure IoT environment. My framework provides this capability, which the standard CGID model does not. It turns unmanageable devices into fully managed network endpoints."

Q2: You mentioned the 3GPP Release 19 introduction of a "Non-3GPP Device Identifier." How is your solution different or better? Why not just use that?

Answer Strategy: Position your work as complementary and solving a prerequisite problem.

Suggested Answer: "The Release 19 developments are very important and validate the industry's need for this kind of granularity. However, the 'Non-3GPP Device Identifier' is primarily a mechanism for QoS differentiation—it assumes the device is already connected and trusted by the gateway. The specifications explicitly state that these devices are not authenticated by the 5G Core. My work addresses the crucial, preceding step: how do you securely authenticate that device in the first place? My framework uses EAP-TLS to establish trust, and then provides a robust identity via the proxy PDU session. My solution can be seen as a practical implementation that provides the authenticated foundation upon which these new QoS mechanisms could then be applied."

Q3: The 33-second onboarding delay you measured is quite high for many applications. What is the primary cause of this latency, and how realistically can it be fixed?

Answer Strategy: Be specific about the cause and confident about the solution.

Suggested Answer: "The 33-second delay is a key limitation of the proof-of-concept, not a fundamental flaw of the architecture. The vast majority of that time is consumed by my Interceptor application shelling out to the UERANSIM command-line interface, nr-cli, and then polling its text output to see when the session is active. This is an artifact of the simulation tools. In a production environment, the Interceptor would use a native modem API—like AT commands, QMI, or a vendor-specific SDK—to manage PDU sessions. These interactions are near-instantaneous. By replacing the CLI-based orchestration with direct API calls, I am confident this delay could be reduced to just a few seconds, which would be well within acceptable limits."

Q4: Your solution relies on EAP-TLS, which requires a Public Key Infrastructure (PKI) to manage certificates. How would this scale to millions of IoT devices in a real-world operator deployment?

Answer Strategy: Acknowledge the challenge but point to existing industry solutions.

Suggested Answer: "That's a very practical and important question. Managing a PKI at scale is a significant operational task, but it is a well-understood problem that is already being solved in the industry. Large-scale device management platforms, like those for cable modems using DOCSIS, or IoT platforms from major cloud providers, already incorporate automated certificate enrollment and lifecycle management protocols. An operator would integrate a similar system. For instance, devices could use an initial identity or a bootstrapping process to securely request and install their unique certificate upon first connection. While my work didn't implement this PKI management layer, the framework is designed to use the standard EAP-TLS protocol, making it fully compatible with these industry-standard solutions."

Q5: You used NAT, which prevents inbound connections to the devices. How does this limit your solution, and what is your proposed fix?

Answer Strategy: Clearly state the limitation and the proposed future work.

Suggested Answer: "The use of NAT is indeed a limitation for use cases requiring a device to act as a server, such as a security camera you want to connect to from outside the home. In my current implementation, all connections must be initiated from the device outward. However, this is a solvable problem. My dissertation proposes exploring the use of Framed-Route, a RADIUS attribute. Upon successful authentication, the RADIUS server could send this attribute to the 5G-RG, instructing it to assign a specific, publicly routable IP address or subnet to the device's PDU session. This would eliminate the need for NAT for that device and allow direct inbound connections, enabling peer-to-peer and server-based applications."