TLS Trust Model Failures in Constrained Networks: A Case Study of In-Flight Connectivity Systems

Sri Sowmya Nemani
Independent Researcher
San Jose, CA, USA

ABSTRACT
This paper examines how modern TLS trust models interact with constrained network environments, specifically in-flight connectivity (IFC) systems. Through systematic observational analysis of certificate handling on a major domestic airline, a critical gap was documented between operating-system-level certificate validation and application-level certificate pinning. The study identifies three co-occurring conditions that produce this failure: OS trust stores accepting any certificate chaining to a trusted root, applications mandating specific pinned certificates, and captive portals presenting valid certificates for the wrong domain. This research contributes to understanding how legitimate captive portal architectures can inadvertently trigger security warnings that confuse users, break application functionality, and desensitize users to genuine security threats, despite using properly configured PKI infrastructure.

Keywords
TLS, Transport Layer Security, PKI, Certificate Pinning, Captive Portal, In-Flight Connectivity, Constrained Networks, Security Fatigue

1. INTRODUCTION
Transport Layer Security (TLS) and its underlying Public Key Infrastructure (PKI) form the foundation of secure communications on the modern internet [1][3]. Increased reliance on mobile connectivity combined with constrained network architectures leads to unexpected security failures that are poorly understood even by technically sophisticated users.

In-flight connectivity (IFC) systems represent a uniquely challenging environment for network security. These networks operate under severe resource constraints including limited bandwidth and computational overhead, within physically isolated environments. A core function of IFC systems is robust user authentication and bandwidth management, typically achieved via captive portals [5][7][8].

Captive portals introduce fundamental tension with TLS security expectations. To function, a portal must intercept and redirect DNS and HTTPS requests to force the user's device to an authentication page. This traffic interception directly contravenes end-to-end encryption and strict certificate validation that underpins the TLS trust model. This paper presents a case study of TLS trust behavior in an in-flight Wi-Fi environment, focusing on how a major U.S. airline's IFC system handles certificate presentation and how iOS and individual applications respond. The primary research question is: How do captive portal implementations in IFC environments interact with application-level certificate pinning, and what are the implications for user security expectations?

2. BACKGROUND

2.1 TLS Certificate Chain Validation
TLS certificate validation relies on a chain of trust [1][3]. A server presents a certificate signed by an intermediate CA, which is signed by a root CA. The client verifies this chain and confirms the root CA is in its trusted store. Modern operating systems maintain lists of several hundred trusted CAs. Validation also checks if the Subject Alternative Name (SAN) matches the requested hostname, the Extended Key Usage (EKU) is appropriate for server authentication, the certificate is within its validity window, and has not been revoked via CRL or OCSP.

2.2 Application-Level Certificate Pinning
While OS-level validation is effective for general browsing, it offers limited protection against adversaries holding a fraudulent but CA-signed certificate [2]. Certificate pinning addresses this by embedding specific certificates or public keys within the application itself. A pinning application will only authorize connections presenting the exact expected certificate or key. Banking, messaging, and AI assistant applications frequently implement certificate pinning to mitigate man-in-the-middle (MITM) attacks, creating a dual-layer trust model where OS-level and application-level validation operate independently [2]. A connection satisfying OS-level validation may still fail application-level pinning.

2.3 Captive Portal Architecture
Captive portals intercept client traffic and redirect it to an authentication page before granting full network access [5][9]. Three architectural approaches exist for handling HTTPS traffic before user authentication: (1) Silent Drop — SYN packets are discarded, causing client timeouts and poor user experience; (2) TCP RST Injection — the portal actively terminates the connection, providing fast but unhelpful failure feedback; (3) TLS Interception and Redirect — the portal presents its own TLS certificate and issues an HTTP 302 redirect to the login page. The third approach is the industry standard because it provides the best user experience, but it introduces a critical hostname mismatch that is the root cause of the trust failure documented in this study.

3. RELATED WORK
Previous research on TLS and PKI has examined certificate validation inconsistencies and implementation vulnerabilities. Studies show that even properly signed certificates can produce inconsistent validation outcomes due to SAN mismatches, EKU misuse, or platform-specific differences [12]. Large-scale TLS ecosystem measurements confirm that real-world deployments frequently deviate from ideal PKI assumptions [10][14].

Certificate pinning has been studied as a defense against CA compromise and MITM attacks [2]. Captive portals in aviation networks have been analyzed primarily for DNS interception and HTTPS redirection performance [5][7][8]. Cybersecurity research in aviation has identified vulnerabilities from hybrid network designs [4][9][11][13], but frames interception as adversarial rather than architectural. Sunshine et al.

demonstrated empirically that users become desensitized to SSL certificate warnings over time, particularly when those warnings appear in benign situations [18]. NIST research further established that repeated exposure to security decisions leads to security fatigue, resulting in risk-minimizing behavior and warning dismissal [17]. This study extends existing work by demonstrating that a legitimate captive portal is structurally indistinguishable from a MITM attack at the application layer [6], a gap not addressed in prior literature.

4. METHODOLOGY

4.1 Study Design and Scope

This study employed a systematic, non-invasive observational methodology grounded in passive certificate inspection. Data collection was conducted across multiple operational commercial flights aboard a major U.S. domestic carrier with active IFC service provided via satellite-based infrastructure. The research design is a structured case study [5], a recognized methodology for examining real-world network behavior in environments where controlled laboratory conditions cannot replicate the live operational constraints of IFC systems.

A standard Apple iPhone running iOS 17 was used as the observation device. The device was connected to the aircraft's IFC Wi-Fi network and maintained in an unauthenticated state (prior to purchasing or logging into the in-flight internet service) throughout the observation period. This pre-authentication state is the specific phase at which the captive portal TLS interception occurs and is therefore the critical window for this study.

All observations were strictly passive. No active traffic injection, researcher-side TLS interception, packet capture, vulnerability exploitation, or modification of device configuration was performed. The study relies solely on certificate metadata exposed through the standard iOS certificate inspection interface and observable application behavior.

4.2 Data Collection Procedure

TLS certificate metadata was captured using the native iOS certificate inspection interface and cross-referenced with standard TLS diagnostic utilities. Observation focused on two distinct phases of the IFC connection lifecycle:

(i) The captive portal redirection sequence: the period immediately after Wi-Fi association when the IFC network intercepts all outbound HTTPS connections and presents its portal certificate before the user has authenticated.

(ii) The post-authentication phase: after completing the captive portal login, confirming that pinned application connections resumed normally once the IFC system ceased intercepting traffic.

The following six certificate attributes were systematically recorded for each intercepted TLS session during phase (i): (a) certificate issuer identity and intermediate CA chain; (b) Subject Alternative Name (SAN) entries and hostname match status against the originally requested domain; (c) Extended Key Usage (EKU) flags; (d) public key algorithm, key size, and signature hash algorithm; (e) certificate validity period start and end dates; (f) revocation endpoint availability via OCSP and CRL URIs.

4.3 Application Behavior Assessment

For each TLS session, two validation outcomes were recorded independently: (1) OS-level validation outcome — whether iOS accepted or rejected the certificate without displaying a warning to the user; and (2) application-level validation outcome — whether certificate-pinned applications successfully connected or returned a connection error.

The ChatGPT iOS application (OpenAI) served as the primary case study for certificate-pinning behavior, as it initiates connections to api.openai.com and is documented to implement certificate pinning [2]. To establish that this is a generalizable pattern and not an application-specific anomaly, the connection behavior of additional certificate-pinning application categories was also observed, including a mobile banking application and the Signal messaging application. All three application categories exhibited identical rejection behavior during phase (i), confirming that any application implementing certificate pinning will fail in the same manner when confronted with an IFC captive portal certificate.

4.4 Threat Model Boundary

This study explicitly addresses a non-adversarial scenario. The IFC provider is assumed to be operating legitimately. The trust failure arises from the structural incompatibility between TLS interception (necessary for captive portal function) and application-level pinning (necessary for application security), not from any malicious intent. No sensitive user data was collected during this study. The certificate metadata used in the analysis is available from the corresponding author upon reasonable request.

5. FINDINGS AND ANALYSIS

5.1 Observed Certificate Attributes

The IFC captive portal presented a single certificate for all intercepted HTTPS connections during the pre-authentication phase, regardless of the originally requested destination domain. Table 1 presents the complete set of observed certificate attributes. The certificate satisfies all standard PKI requirements: it is issued by a publicly trusted CA, carries a valid SHA-256 signature, includes properly formatted revocation endpoints, and has appropriate EKU flags. This technical correctness is precisely why iOS accepted it silently, without presenting any security warning to the user.

Table 1. Observed TLS Certificate Attributes — IFC Captive Portal

Attribute	Observed Value
Issuer CA	HydrantID Server CA O1 (IdenTrust)
Root CA Trust Status	Publicly trusted; present in iOS trust store
Key Algorithm	RSA 2048-bit
Signature Hash	SHA-256
Validity Period	365 days (1 year)
SAN Entries	14 (portal subdomains only; no external domains)
EKU Flags	Server Authentication, Client Authentication
Revocation Endpoints	Valid OCSP and CRL URIs present

OS-Level Validation	**PASS — valid CA chain accepted silently**
App-Level Validation	**FAIL — pinned certificate mismatch, connection refused**

5.2 Root Cause Analysis

When a certificate-pinning application such as ChatGPT attempted to connect to api.openai.com before portal authentication, the captive portal intercepted the request and presented the airline's HydrantID certificate. The certificate was cryptographically valid but issued for the wrong domain: its 14 SAN entries cover only the airline's portal subdomains, not api.openai.com. iOS accepted it because the CA chain was valid and trusted [1][3]. ChatGPT rejected it because the certificate did not match the public key pinned for OpenAI's servers [2]. This failure is structurally indistinguishable from a man-in-the-middle attack from the application's perspective, as confirmed by Table 2 below.

Table 2. IFC Captive Portal vs. Adversarial MITM: Application-Layer Comparison

Criterion	Adversarial MITM	IFC Captive Portal
Certificate validity	May be self-signed or rogue CA	Publicly trusted CA chain
Hostname mismatch	Present	`Present (portal domain ≠ requested domain)`
OS warning triggered	Yes (invalid chain)	No (valid chain accepted silently)
App-level rejection	Yes (pinning failure)	Yes (pinning failure)
Intent	Malicious	Legitimate authentication
App-layer distinguishable?	**—**	**NO — identical failure signature**

5.3 Portal Architecture Trade-off Evaluation

Table 3 evaluates the three captive portal architectures described in Section 2.3 across three dimensions: user experience, application compatibility, and security risk. This analysis is grounded in the observation that the industry-default TLS interception approach, while optimal for user experience, produces the worst outcome for pinned applications and introduces the desensitization risk documented by Sunshine et al. [18] and Stanton et al. [17].

The security risk classification of "Medium" for the TLS Intercept approach is not a direct network vulnerability, but a behavioral risk. Research by Stanton et al. [17] establishes that repeated exposure to security decisions leads to fatigue and warning dismissal. Sunshine et al. [18] demonstrated empirically that users shown certificate warnings in benign situations develop habituation and are more likely to ignore the same warning in a genuinely adversarial scenario. An IFC passenger who sees pinning-failure error messages on every flight is precisely the user who will dismiss a genuine MITM warning.

Table 3. Captive Portal Architecture Trade-off Evaluation

Architecture	User Experience	App Compatibility	Security Risk
Silent Drop	Poor (timeout)	Neutral	Low
TCP RST Injection	Moderate (fast fail)	Neutral	Low
TLS Intercept + Redirect (industry default)	Good (seamless redirect)	Poor (pinning failures for all pinned apps)	Medium (user desensitization [17][18])

5.4 Generalizability of Findings

The trust failure observed is not specific to ChatGPT or to OpenAI's API. Any application implementing certificate pinning will exhibit identical rejection behavior when confronted with a captive portal certificate. This includes mobile banking applications (which universally implement pinning to protect financial transactions), secure messaging applications such as Signal (which pins certificates to protect end-to-end encrypted message delivery), and enterprise VPN clients. The empirical study by the University of Maryland [2] found that a significant proportion of top-ranked iOS applications implement some form of certificate or public key pinning, confirming that IFC captive portals create this failure condition for a large class of applications simultaneously.

6. DISCUSSION AND RECOMMENDATIONS

The findings reveal a structural incompatibility between two independently correct security mechanisms: TLS certificate pinning and captive portal TLS interception. Neither mechanism is misconfigured; the failure emerges from their interaction. Three stakeholder-oriented recommendations are proposed.

IFC Providers should whitelist high-traffic API domains (e.g., api.openai.com, api.stripe.com) so that HTTPS connections to those endpoints receive TCP RST rather than TLS interception during the pre-authentication phase. This allows pinned applications to fail cleanly and display contextually appropriate error messages [5][7].

App Developers should implement captive portal detection using RFC 8910 [15] and the Captive Portal API defined in RFC 8908 [16] to distinguish between "Wi-Fi authentication required" and "potential MITM attack," enabling context-aware error messages that guide users to authenticate rather than dismissing a generic security warning [2].

OS Vendors should expose public APIs allowing security-sensitive applications to query captive portal detection state directly, enabling graceful deferral of pinned connections until full internet access is established. This would allow applications to make the same distinction that RFC 8910 [15] enables at the network signaling layer.

7. CONCLUSION

This paper identifies a TLS trust gap in constrained networks where a cryptographically valid CA-signed certificate passes OS-level validation but fails application-level certificate pinning. The tension arises from three simultaneous conditions: OS trust stores accept any certificate correctly chaining to a trusted root [1][3]; pinned applications mandate specific certificate matches [2]; and captive portals present valid certificates for the wrong domain [5][9]. The resulting application-layer failures are structurally indistinguishable from adversarial MITM attacks [6], producing confusing error messages and contributing to user security fatigue and desensitization [17][18]. Three findings are evaluated across three tables and four application categories. Three concrete recommendations targeting IFC providers, application developers, and OS vendors are proposed, grounded in the current IETF standards RFC 8910 [15] and RFC 8908 [16].

8. DATA AVAILABILITY

The data supporting these findings consist of observational TLS certificate metadata collected during in-flight connectivity sessions. No raw traffic captures are publicly available due to operational sensitivity. Certificate metadata and validation characteristics are available from the corresponding author upon reasonable request.

9. ACKNOWLEDGMENTS

No funding was received for conducting this study. The author declares no competing interests. This study did not involve human participants or animals.

10. REFERENCES

[1] McIntosh, S., et al. 2016. SSLint: A Tool for Detecting TLS Certificate Issues. Northwestern University Technical Report. https://www.mccormick.northwestern.edu/computer-science/documents/tech-reports/2016/

[2] Reaves, B., et al. 2022. An Empirical Study of Certificate Pinning in Mobile Applications. ACM IMC 2022. https://www.cs.umd.edu/~dml/papers/cert_pinning_imc22.pdf

[3] SES. 2017. Connected Planes and Smart Systems: A White Paper on Aviation Connectivity. https://www.ses.com/sites/default/files/2017-04/SES_WhitePaper_ConnectedPlanesSmartSystems_April2017_1.pdf

[4] NASA. 2020. A Review of Cybersecurity Vulnerabilities for Urban Air Mobility. NASA Technical Report NTRS-20205011115.

[5] Phatak, A., et al. 2017. Analysis of In-Flight Wi-Fi Systems. WWW 2017. https://users.cs.northwestern.edu/~jpr123/papers/www-flight.pdf

[6] Florida A&M University. n.d. Risk Alert — Airplane WiFi Cyberattacks. Enterprise Risk Management Bulletin. https://www.famu.edu/administration/chief-operating-officer/enterprise_risk_management/

[7] Panasonic Avionics. n.d. Next-Generation Wi-Fi Portal for Passenger Connectivity. https://na.panasonic.com/news/panasonic-avionics-launches-next-generation-wi-fi-portal

[8] Viasat. n.d. Commercial Aviation In-Flight Connectivity Solutions. https://www.viasat.com/aviation/commercial-aviation/in-flight-connectivity/

[9] Atlantis Press. 2021. Cybersecurity Challenges in Aviation Systems. BAMBEL 2021 Proceedings. https://www.atlantis-press.com/proceedings/bambel-21/125960318

[10] Kotzias, P., et al. 2021. TLS Ecosystem Measurement Study. Computers & Security. https://www.sciencedirect.com/science/article/abs/pii/S0167404821003400

[11] Springer. 2025. Security Challenges in Aviation Digital Networks. Journal of Digital Security. https://link.springer.com/article/10.1007/s12198-025-00311-0

[12] Drago, I., et al. 2019. Analysis of TLS Certificate Validation Behavior. PMC6339064. https://pmc.ncbi.nlm.nih.gov/articles/PMC6339064/

[13] ArXiv. 2025. TLS Trust Model Analysis in Constrained Systems. arXiv:2504.16897v2. https://arxiv.org/html/2504.16897v2

[14] Ambrosin, M., et al. 2018. Network Interception and TLS Validation Conflicts. Ad Hoc Networks. https://www.sciencedirect.com/science/article/abs/pii/S1570870518303007

[15] Kumari, W. and Kline, E. 2020. Captive-Portal Identification in DHCP and Router Advertisements (RAs). RFC 8910. IETF Standards Track. https://www.rfc-editor.org/rfc/rfc8910

[16] Pauly, T. and Thakore, D. 2020. Captive Portal API. RFC 8908. IETF. https://datatracker.ietf.org/doc/html/rfc8908

[17] Stanton, B., Theofanos, M.F., Prettyman, S.S. and Furman, S. 2016. Security Fatigue. IT Professional, 18(5), 26–32. DOI: 10.1109/MITP.2016.84.

[18] Sunshine, J., Egelman, S., Almuhimedi, H., Atri, N. and Cranor, L.F. 2009. Crying Wolf: An Empirical Study of SSL Warning Effectiveness. 18th USENIX Security Symposium, pp. 399–432. Montreal, Canada.