Post-Session Execution Assurance (PSEA)

A Security Model for Verifying Authority at the Moment of Action

---

Mohamad Khalil Yossif

Founder & CEO, Yuthent

Version 1.0 | February 7, 2026

https://yuthent.com/psea

Abstract

Modern security architectures assume that trust established at login can safely extend to all subsequent actions within a session. This assumption is fundamentally flawed. Sessions preserve the memory of authentication but not the certainty of authority. Post-Session Execution Assurance (PSEA) is an architectural security model that addresses this gap by requiring cryptographic proof of real human presence, device trust, and execution authority at the exact moment a sensitive action is approved — independent of prior authentication or session state. This paper defines the PSEA model, its core principles, its distinction from existing security paradigms, and its implications for regulated and mission-critical systems.

1. The Broken Assumption of Sessions

For decades, session-based security has operated on a single foundational assumption: that a successful authentication event at login time can be trusted to represent the user's identity and authority for the duration of the session. This assumption was reasonable when systems were simple, sessions were short, and interactions were linear.

It is no longer reasonable.

Modern systems operate in environments where actions occur long after the initial authentication event. Context drifts continuously — devices change hands, network environments shift, and authorization states evolve. Sessions persist across these changes without re-evaluating whether the conditions that justified the original trust decision still hold.

The problem is structural. Sessions are designed to reduce friction by avoiding repeated authentication. In doing so, they create an implicit trust window during which any actor — human or automated — operating within that session inherits the authority of the original authenticator. This inheritance is unconditional. It does not account for elapsed time, changes in device posture, delegation of access, or environmental degradation.

In practice, this means that a session authenticated by a legitimate user at 9:00 AM carries identical authority at 3:00 PM, even if the device has been compromised, the user has stepped away, or an automated process has assumed control of the session. The session does not know. It was not designed to know.

This is not a vulnerability in any particular implementation. It is a structural flaw in the session model itself.

2. The Authority Gap

Authentication and execution authority are fundamentally different concepts, though they are routinely conflated in security architecture.

Authentication answers the question: "Did this entity present valid credentials at some point in time?" It establishes identity at the moment of login. It does not, and cannot, establish who is performing a specific action at a later point in time.

Execution authority answers a different question: "Is the authorized human — present, verified, and operating from a trusted device — approving this specific action right now?"

The gap between these two questions is where risk materializes. In banking, a fraudulent wire transfer is executed not at login, but minutes or hours later, within a valid session. In healthcare, an unauthorized prescription is signed within a session that was legitimately established. In government systems, classified actions are approved by sessions that may have been delegated, replayed, or hijacked after the original authentication.

In each case, the authentication was valid. The session was valid. But the authority to execute the specific action at that specific moment was never verified.

This gap — between authentication and execution authority — is the central problem that existing security models fail to address. Multi-factor authentication strengthens the login boundary. Session management extends or restricts session duration. Continuous authentication monitors behavioral signals over time. None of these mechanisms provide cryptographic proof that a specific human is authorizing a specific action at the moment it is executed.

The authority gap is not theoretical. It is the operational reality of every system that relies on sessions to authorize sensitive actions.

3. Definition of Post-Session Execution Assurance

> *Post-Session Execution Assurance (PSEA) is an architectural security model that requires cryptographic proof of real human presence, device trust, and execution authority at the exact moment a sensitive action is approved — independent of prior authentication or session state.*

This definition establishes PSEA as a model concerned exclusively with the moment of execution. It does not replace authentication, session management, or access control. It addresses the specific problem of verifying authority when an action is approved — a problem that existing models leave unresolved.

The term "post-session" does not imply that sessions are absent. It means that the assurance provided

by PSEA operates independently of session state. Whether a session is active, expired, or absent, the execution assurance requirement remains the same: cryptographic proof of human presence, device trust, and authority at the moment of action.

4. Core Principles of PSEA

The PSEA model is built on five core principles. Each addresses a specific dimension of the authority gap.

4.1 Execution-Time Proof

Authority must be validated at the moment an action is executed, not at any prior point. A proof generated at login time, or at any time before the action, does not satisfy this requirement. The temporal binding between proof and action is non-negotiable.

4.2 Human Presence Assurance

The model requires verification that a real human being is present and actively approving the action. This is distinct from identity recall, where a system infers human presence from a prior authentication event. Presence must be demonstrated, not assumed.

4.3 Device-Bound Trust

Execution approval must be bound to a specific, trusted device state. The device from which the approval originates must be verified as part of the assurance chain. An approval that cannot be tied to a known, trusted device does not meet the PSEA requirement.

4.4 Cryptographic Proof

Trust is established through cryptographic mechanisms, not inferred from session tokens, cookies, or bearer credentials. The proof must be independently verifiable and resistant to replay, forgery, and delegation.

4.5 Connectivity Independence

PSEA is designed to function in environments with intermittent, degraded, or absent network connectivity. The model does not depend on real-time communication with a central authority for the generation of execution proof. Verification may be deferred, but the proof itself must be generated at execution time regardless of connectivity state.

5. What PSEA Is Not

Clarity about the boundaries of PSEA is essential to prevent conflation with existing security paradigms.

PSEA is not Multi-Factor Authentication. MFA strengthens the login boundary by requiring multiple credential types. It does not address authority at execution time. Once MFA is completed, the resulting session carries the same inherited trust as any other session.

PSEA is not Continuous Authentication. Continuous authentication monitors behavioral or biometric signals throughout a session to detect anomalies. It is a monitoring and risk-scoring mechanism. It does not require proof of authority at the moment of a specific action.

PSEA is not Session Hardening. Session hardening techniques — short timeouts, token rotation, binding to IP addresses — reduce the window of exposure. They do not verify who is executing an action within that window.

PSEA is not Risk-Based Authentication. Risk-based systems evaluate contextual signals to adjust authentication requirements dynamically. They operate at or near the login boundary. They do not provide execution-time proof.

PSEA is not a rebranding of Zero Trust. Zero Trust architectures reduce implicit trust between system components and enforce verification at network and application boundaries. PSEA addresses a different problem: the absence of verified authority at the point of action execution, which persists even within Zero Trust environments.

Each of these paradigms addresses a legitimate security concern. None of them solve the specific problem that PSEA defines: the absence of cryptographic, human-verified, device-bound proof of authority at execution time.

6. Implications for Regulated and Mission-Critical Systems

The authority gap has acute consequences in regulated and mission-critical environments where the cost of unauthorized execution is severe.

6.1 Banking and Financial Services

Financial systems process high-value transactions within authenticated sessions. Fraud, insider misuse, and session compromise result in unauthorized transactions that are technically valid from the session's perspective. Regulatory frameworks increasingly require proof that specific individuals authorized specific transactions. PSEA provides the architectural basis for such proof.

6.2 Government and Defense

Government systems handle classified operations, policy decisions, and infrastructure controls. Delegation of authority, shared workstations, and long-duration sessions create conditions where session-based trust is insufficient. PSEA addresses the requirement for verifiable human authority over specific actions in these environments.

6.3 Healthcare

Healthcare systems authorize prescriptions, access to patient records, and clinical decisions. Shared terminals, shift-based workflows, and emergency access create scenarios where the authenticated user may not be the person executing the action. PSEA provides a mechanism for binding execution authority to a verified human at the point of action.

6.4 Critical Infrastructure

Industrial control systems, energy grids, and transportation networks require human authorization for sensitive operational changes. These environments frequently operate in degraded connectivity conditions. PSEA's connectivity-independent design addresses the need for execution-time authority verification even when centralized systems are unreachable.

7. Relationship to Implementations

PSEA is an architectural model, not a product specification. It defines what must be proven at execution time without prescribing how the proof is generated, transmitted, or verified.

Implementations of PSEA may vary in their choice of biometric modalities, cryptographic schemes, device attestation mechanisms, and verification architectures. The model is intentionally agnostic to these choices, provided the core principles are satisfied.

Yuthent is one implementation of the PSEA model, designed for regulated and mission-critical environments. It is not the only possible implementation. The PSEA model exists independently of any specific product or vendor.

Organizations adopting PSEA may implement it through existing infrastructure, custom development, or third-party solutions. The requirement is adherence to the model's principles, not adoption of a specific technology.

8. Conclusion

The security architecture of modern systems contains a structural gap between authentication and execution. Sessions bridge this gap with inherited trust — a mechanism that was adequate for simpler systems but is fundamentally insufficient for environments where authority must be verified at the moment of action.

Post-Session Execution Assurance defines the requirement that this gap be closed. It is not a feature, not a product category, and not a rebranding of existing security paradigms. It is an architectural correction: a formal recognition that authority is temporal, context-dependent, and must be proven — not assumed — at the moment it is exercised.

The absence of execution-time authority verification is not a theoretical concern. It is the operational

reality that enables fraud, insider misuse, and unauthorized actions across banking, healthcare, government, and critical infrastructure. PSEA provides the conceptual and architectural foundation for addressing this reality.

9. Citation & Reference

To reference this document, use the following citation:

```
Khalil Yossif, M. (2026). Post-Session Execution Assurance (PSEA): A
Security Model for Verifying Authority at the Moment of Action (Version
1.0). Yuthent. https://yuthent.com/psea
```

Canonical URL: https://yuthent.com/psea

GitHub: https://github.com/yuthent/psea-spec