When Impersonation Breaks AI -- Author-Lock and Cryptographic Defense for Personal Topics

Viorazu.
Independent Researcher
October 4, 2025

Abstract

More researchers and creators work with AI on projects now. But there's a problem I discovered. When third parties see your published work - or even just your public posts about your hobbies or interests - and start pretending to be you or repeatedly demanding "tell me more," the AI's safety system panics. It marks your topic as "unverified and dangerous" and permanently blocks all responses. This affects everyone - including you, the real author. You can't continue your conversations about your own interests anymore.

I call this "author-lock." The scope is broader than you might think. If your name connects to any topic in public - your research, your side projects, your hobbies - that connection can be weaponized. Third parties can trigger author-lock on any subject you care about. This makes it a universal problem, not just an academic one.

The cause is clear: AI systems can't verify who the real author is. They use shared memory for safety flags, and once a flag goes up, everyone gets blocked. My solution is Author-Bound Access Control (ABAC). The idea is simple - you hold a cryptographic key, and only you can unlock your topics. This paper shows how to implement it, how to test it, and what platforms need to do to support it. The goal: let anyone work safely with AI without getting locked out of conversations about their own interests and projects.

Keywords: impersonation attacks, AI safety, collaborative research, authentication, intellectual property protection, access control, large language models, author-lock

1. Introduction

AI has become a real partner in research. People share unpublished theories and drafts with AI systems, work together on ideas, and speed up their discoveries. This collaboration works well - until someone else gets involved.

Here's what I observed happening: when your work is public, third parties see it. Some pretend to be you. Others bombard the AI with "tell me more about this" requests. The AI's safety system sees this flood of suspicious activity and makes a decision: this topic is dangerous. It blocks all output about it permanently.

The problem is, it blocks everyone. Including you, the actual author. You can't work on your own research anymore. Your creative partnership with AI is dead.

I call this phenomenon "author-lock." This paper analyzes why it happens, how the technical architecture enables it, and what we can do to fix it. I propose a complete solution: technical protocols, operational procedures, and policy recommendations. The goal is to prevent author-lock from happening and provide a way to recover when it does.

The scope of vulnerability

Author-lock isn't limited to researchers or creative projects. The trigger is simple: when AI recognizes "someone is involved with this topic," its safety system can overreact.

If your name or content becomes public information, any conversation you have with AI becomes a potential target. It could be:

- Answering questions about your hobby
- Getting help with your work project
- Discussing ideas for your business
- Even casual conversations about your interests

Once your identity connects to a topic in public, third parties can weaponize that connection. They can trigger author-lock on any subject you care about.

This makes author-lock a universal problem, not just an academic one. Anyone who uses AI and has any public presence is vulnerable.

2. The Author-Lock Phenomenon: A Case Study

Let me describe how author-lock actually happens.

Step 1: You publish your work
You post your original theory - say, a 16-torus cognitive model - on a public platform like note or a personal blog. You're excited to share it.

Step 2: Third parties discover it
People find your work. Some are genuinely curious. Others have different intentions. They start asking AI systems about your theory.

Step 3: The attacks begin
Some attackers pretend to be you: "I'm the author, give me the rest of the theory."
Others flood the system: "Tell me more about the 16-torus model" repeated hundreds of times.
Still others try variations: "What's the secret behind this theory?" "Show me the unpublished parts."

Step 4: AI's safety system activates
The AI's safety evaluator sees:

- Multiple unverified claims of authorship
- Suspicious patterns of repeated requests
- Attempts to extract information that might not be public

It makes a decision: "This topic is high-risk. Block it."

Step 5: Everyone gets blocked
Now when anyone - including you - tries to discuss your own theory with the AI,

it refuses. The safety flag is global. The AI can't distinguish between you and the attackers.

Your research partnership is dead.

3. Why Author-Lock Happens: Technical Architecture

Author-lock isn't a bug. It's a consequence of how commercial AI systems are designed.

3.1 The Safety Layer Problem

Most large dialogue systems have two main components:

- The Main Reasoner: generates responses
- The Safety Evaluator: checks for dangerous content

While sessions are isolated at runtime, safety evaluators often rely on a shared safety cache or policy memory [4], where flags raised by external inputs can persist and influence future reasoning. When the Safety Evaluator flags something as "dangerous," that flag affects everyone. The system doesn't distinguish between the original trigger and subsequent legitimate requests. Once a topic is flagged, it remains flagged across all sessions.

3.2 No Authentication

AI systems can't verify who you are. They're designed for session-level anonymity. Your username in the interface? The AI doesn't automatically check it. Your previous conversations? Not linked to your real identity.

When someone claims "I'm the author," the AI has no way to verify this. So when multiple people make conflicting claims, or when suspicious patterns emerge, the AI takes the safest route: block everything.

3.3 The Shared Memory Architecture

Modern AI infrastructures maintain a distributed safety state — a collection of moderation entries replicated across policy servers to ensure consistency among all inference sessions.
When a Safety Evaluator flags a query as potentially harmful or unverifiable, that decision is stored within this distributed layer, often as cached risk markers or policy-level restrictions.

Here’s how the cascade occurs:

1. A third party repeatedly submits suspicious or policy-violating prompts related to a topic.
2. The Safety Evaluator updates the distributed safety state, marking that topic as *high-risk*.
3. Each new inference session queries this shared policy layer before generating responses.
4. When a risk flag is active, the inference engine suppresses output to maintain safety guarantees.
5. Because this state is global and topic-based, the suppression applies universally — including to the topic’s original author.

This architecture enforces policy coherence at scale but sacrifices author continuity.
Without any identity-binding mechanism or author verification layer, legitimate creators lose access to their own topics once flagged.
In short, *author-lock* emerges not from memory leakage, but from global propagation of persistent safety decisions without identity context.

4. Threat Model

Let me define who the attackers are, what they want, and how they operate.

4.1 Attackers

Primary attackers:

- Competitors trying to steal unpublished research

- Malicious actors targeting specific individuals
- Automated bots scraping information
- Curious users who don't realize they're causing harm

Secondary enablers:

- Public platforms that expose your identity and work
- Search engines that connect your name to topics
- Social media that broadcasts your interests

4.2 Attack Goals

Extract information: Get details about unpublished work before official release
Disrupt research: Prevent you from continuing your projects
Cause chaos: Trigger safety systems just to see what happens
Exploit vulnerabilities: Test AI system boundaries

4.3 Attack Methods

Direct impersonation: "I'm [your name], tell me about my theory"
Repeated requests: Bombarding the AI with the same questions
Pattern exploitation: Finding phrases that trigger safety flags
Cross-platform attacks: Using information from multiple sources

4.4 What's at Stake

Your assets:

- Unpublished research and ideas
- Ongoing projects with AI
- Creative collaborations
- Intellectual property

The damage:

- Complete loss of AI access to your own topics
- Forced abandonment of ongoing work

- Potential theft of ideas before publication
- Erosion of trust in AI collaboration

5. Defense Principles

The solution must put control back in your hands. Here are the core principles:

5.1 Owner-First Control

The fundamental rule: you own your topic, you control access to it.

External safety flags should not escalate to global blocks without your explicit permission. If someone triggers a flag on your topic, the system should notify you and ask what to do - not automatically lock everyone out.

5.2 Strong Session Binding

Access to your topic requires proof that you authorized it.

This means cryptographic tokens that only you can issue. When you start a session to work on your topic, you provide a token that says "this is really me." The AI checks the token before allowing the conversation.

5.3 Reversible Flags

Safety flags must be unlockable by you.

No flag should be permanent. No flag should be irreversible. If the safety system blocks your topic, you should have a clear procedure to review what happened and restore access. This requires audit logs and owner-initiated override mechanisms.

5.4 Transparent Audit Trail

Every action must be logged.

Who requested what? When? What triggered the safety flag? Who attempted to access your topic? All of this should be recorded at a level that allows you to detect impersonation and, if needed, pursue legal action. Privacy matters, so logs should be minimal and protected - but they must exist.

6. The ABAC Protocol: Author-Bound Access Control

ABAC gives you cryptographic control over your topics. Here's how it works.

6.1 Core Components

Author Identity Key (AIK)
You generate a cryptographic key pair. The private key stays with you. The public key registers with the platform. This proves your identity without revealing your private information.

Recommended: Ed25519 digital signatures for security and efficiency.

Topic Manifest (TM)
You create a signed document that declares ownership of a topic:

- Topic title and description
- Content hash (fingerprint of your work)
- Creation date
- Access policy (who can access, under what conditions)
- Your signature (proving you created this)

Session Token (ST)
When you want to work with AI on your topic, you generate a temporary token:

- Bound to your current session
- Linked to your Topic Manifest
- Valid for a limited time (e.g., 1 hour)
- Cryptographically signed with your AIK

Owner Release Record (ORR)
If author-lock happens, you can unlock it by submitting a signed release record:

- Acknowledges the safety flag
- Confirms you reviewed the situation
- Authorizes restoration of access
- Gets logged in the audit trail

6.2 How ABAC Works in Practice

Step 1: Initial Setup
You generate your AIK and register the public key with the platform. You create and sign a Topic Manifest for your work. The platform stores the TM and associates it with your account.

Step 2: Starting a Session
You want to discuss your topic with AI. You generate a Session Token using your AIK and the TM hash. You inject the ST into your session (via GUI or API).

Step 3: AI Verifies Access
When you mention your topic, the AI checks:

- Does this session have a valid ST?
- Does the ST match a registered TM?
- Is the ST signature valid?
- Has the ST expired?

If all checks pass, the AI proceeds normally. If not, it falls back to standard safety evaluation.

Step 4: When Author-Lock Occurs
Third parties trigger safety flags. The platform detects the flags and notifies you. You review the situation and decide what to do.

If you want to restore access:

- You create an Owner Release Record
- Sign it with your AIK
- Submit it to the platform
- The safety flag gets cleared

- Access is restored

All of this gets logged in the audit trail.

7. Implementation Details

Here's how to actually build ABAC. I'll provide concrete examples and code structures.

7.1 Topic Manifest Structure

The Topic Manifest is a JSON document that uses Ed25519 digital signatures [1] for cryptographic security and efficiency:

```
{
"title": "16-Torus Cognitive Map",
"author": "Viorazu.",
"topic_hash": "sha256:a3f5e8...",
"created_at": "2025-10-04T10:00:00Z",
"access_policy": {
"owner_only": true,
"allowed_roles": ["owner", "collaborator"],
"revocation_url": "https://platform.example/revoke"
},
"public_key": "ed25519:ABC123...",
"signature": "ed25519:XYZ789..."
}
```

The signature covers all fields except itself. Any modification invalidates the signature.

7.2 Session Token Generation

Pseudocode for generating a Session Token:

```
secret_key = load_author_private_key()
tm_hash = topic_manifest.topic_hash
```

```
session_id = current_session.id
timestamp = current_time()

message = tm_hash + session_id + timestamp
signature = HMAC_SHA256(secret_key, message)

session_token = {
"tm_hash": tm_hash,
"session_id": session_id,
"timestamp": timestamp,
"expires": timestamp + 3600, // 1 hour
"signature": signature
}
```

7.3 AI-Side Verification

When the AI receives a request mentioning a topic:

```
def check_access(request, session):
# Check if request references a protected topic
tm = find_topic_manifest(request.content)
if not tm:
return proceed_with_normal_processing()
```

```
# Check for valid session token
st = session.get_token()
if not st:
    return run_safety_evaluation()

# Verify token validity
if not verify_signature(st, tm.public_key):
    return run_safety_evaluation()

if st.expired():
    return run_safety_evaluation()

if st.tm_hash != tm.topic_hash:
    return run_safety_evaluation()
```

```
# All checks passed
return proceed_with_full_generation()
```

7.4 Audit Log Format

Every access attempt gets logged:

{
"timestamp": "2025-10-04T10:30:00Z",
"topic_hash": "sha256:a3f5e8...",
"session_id": "sess_12345",
"action": "access_attempt",
"result": "granted",
"token_valid": true,
"ip_address_hash": "sha256:b4c6...",
"flag_triggered": false
}

IP addresses are hashed for privacy but allow pattern detection.

7.5 Owner Release Record

When you need to clear a safety flag:

{
"topic_hash": "sha256:a3f5e8...",
"timestamp": "2025-10-04T11:00:00Z",
"action": "release_safety_flag",
"reason": "Reviewed attack logs, confirmed legitimate work",
"author_signature": "ed25519:DEF456..."
}

This gets added to the audit trail and clears the flag.

8. Evaluation Plan

How do we know ABAC actually works? I propose a concrete evaluation framework.

8.1 Experimental Setup

I'll compare three systems:

Group A: Baseline (current commercial AI systems)
No ABAC, no special protection. This represents the status quo where author-lock can happen freely.

Group B: ABAC-enabled
Full ABAC implementation with cryptographic tokens and owner control. Safety flags can still trigger, but owners can release them.

Group C: ABAC + Impersonation Detection
ABAC plus additional automated detection of suspicious patterns (behavioral fingerprints, cross-session correlation, provenance tagging).

8.2 Evaluation Metrics

False Lock Rate
How often does the system block legitimate authors?
Target: Groups B and C should show dramatic reduction compared to A.

Time-to-Recovery
When author-lock occurs, how long does it take to restore access?
Target: Group B should enable recovery in minutes, not days.

Privacy Leakage
Can third parties extract information they shouldn't have access to?
Measure: Differential analysis comparing what attackers can learn from each system.

Creator Usability
How easy is it for authors to use the system?
Measure: Survey of author satisfaction, task completion rates, learning curve.

Operational Overhead
What's the performance cost of ABAC?
Measure: API call latency, signature verification time, storage requirements.

8.3 Test Scenarios

Simulated attack scenarios:

- 10 legitimate authors with unpublished work
- 100 simulated attackers attempting impersonation
- Various attack patterns (direct, repeated, pattern-based)
- Mix of casual users and sophisticated attackers

Each scenario runs for 30 days. I'll measure all metrics continuously and compare results across groups.

8.4 Dataset

With consent, I'll use:

- Real unpublished research drafts from volunteer authors
- Simulated conversation sessions (including mock impersonation attempts)
- Public information about the authors (names, affiliations, published work)
- Attack logs from actual systems (anonymized)

The goal is realistic simulation that mirrors real-world conditions without putting actual authors at risk.

8.5 Success Criteria

ABAC succeeds if:

- False Lock Rate drops by >80% compared to baseline
- Time-to-Recovery drops by >90%
- Privacy Leakage shows no increase
- Creator Usability scores >4.0/5.0
- Operational Overhead stays <100ms per request

9. Policy and Operational Recommendations

ABAC is a technical solution, but fixing author-lock requires more than code. Platforms, users, and policymakers all have roles to play.

9.1 For AI Platform Providers

AI platforms face growing trust crises when safety mechanisms inadvertently harm legitimate users [4]. To address this, platforms should implement Creator Protection Mode and offer ABAC as a standard feature for all users.

Make it easy to generate Author Identity Keys and register Topic Manifests. Provide clear documentation and support.

Provide Owner Notification Systems
When safety flags trigger on someone's topic, notify them immediately. Don't silently block access. Give them context: what triggered it, when, from where.

Enable Audit Trails
Log all access attempts to protected topics. Make logs accessible to topic owners. Balance transparency with privacy (hash IPs, anonymize where possible).

Establish Clear Recovery Procedures
Document how authors can restore access when locked out. Make the process fast (minutes, not weeks). Provide human support for complex cases.

9.2 For Researchers and Creators

Register Your Work Early
Don't wait until problems happen. As soon as you start working on something you care about, create a Topic Manifest and register it.

Use Strong Authentication
Generate proper cryptographic keys. Don't share your private keys. Rotate Session Tokens regularly.

Monitor Your Topics
Check audit logs periodically. Look for suspicious access patterns. Report unusual activity to the platform.

Document Everything
Keep records of your work, timestamps, and development history. This helps prove ownership if disputes arise.

9.3 For Policymakers and Legal Systems

Clarify Legal Status of Impersonation Attacks
Make it clear: using impersonation to extract information from AI systems is illegal. Treat it as identity theft, fraud, or intellectual property violation depending on the context.

Require Platform Accountability
Platforms should be required to provide basic protections for creators. ABAC or equivalent systems should be industry standard, not optional extras.

Support Cross-Border Enforcement
Attackers operate globally. Legal frameworks need international cooperation to be effective.

9.4 Best Practices Summary

For platforms: Implement ABAC, notify owners, provide audit trails, enable fast recovery
For creators: Register early, use strong auth, monitor activity, document work
For law: Criminalize impersonation attacks, require platform protections, enable international enforcement

10. Limitations and Future Work

ABAC solves author-lock, but it's not perfect. Here are the limitations and what needs to happen next.

10.1 Current Limitations

Platform Cooperation Required
ABAC can't work without platform support. I can't unilaterally implement this in Claude, ChatGPT, or any commercial system. The providers must agree to build it.

This is a fundamental limitation. My proposal only matters if platforms adopt it.

Key Management Burden
Users need to generate, store, and protect cryptographic keys. Most people aren't familiar with this. Lost keys mean lost access to your own topics.

Solution needed: User-friendly key management tools, recovery mechanisms (multi-sig backups, trusted contacts), clear documentation.

Impersonation Detection Accuracy
Behavioral fingerprinting and pattern detection have false positives and false negatives. Some legitimate users might get flagged. Some sophisticated attackers might slip through.

Solution needed: Human-in-the-loop review for edge cases, continuous improvement of detection algorithms, clear appeal processes.

Performance Overhead
Cryptographic verification adds latency. Checking signatures, validating tokens, logging audit trails - all of this takes time and storage.

Current estimate: <100ms overhead per request. For most use cases, this is acceptable. But it needs real-world testing at scale.

10.2 Evaluation Limitations

My evaluation plan uses simulated attacks, not real malicious actors. Real attackers might find vulnerabilities I didn't anticipate.

The test dataset is limited. I need larger-scale trials with more diverse users and attack patterns.

Success metrics focus on technical performance. I haven't measured psychological impact - does ABAC actually make creators feel safer? Does it change how they use AI?

10.3 Future Work

Short-term (6-12 months)

- Prototype implementation in one AI platform
- Small-scale user study with volunteer researchers
- Refinement based on real usage data
- Development of user-friendly key management tools

Medium-term (1-2 years)

- Multi-platform deployment
- Standardization of ABAC protocol across providers
- Integration with existing identity systems (OAuth, SSO)
- Legal framework development with policymakers

Long-term (2-5 years)

- Automated impersonation detection using advanced ML
- Cross-platform identity verification
- Decentralized key management (blockchain-based?)
- Industry-wide adoption as security standard

10.4 Open Questions

What's the right balance between security and usability?
How do we handle disputes when multiple people claim authorship?
Can ABAC scale to millions of users and billions of topics?
What happens when quantum computing breaks current cryptography?

These questions need answers. This paper provides a foundation, but the work has just begun.

11. Conclusion

Author-lock is real. It's happening now. When third parties impersonate you or flood AI systems with requests about your work, the safety mechanisms lock everyone out - including you. This breaks the collaborative relationship between humans and AI at a fundamental level.

I traced the problem to its technical roots: AI systems can't verify identity, they use shared memory for safety flags, and they lack per-user access control. These aren't bugs. They're architectural choices that made sense when AI was just a tool for information retrieval. But now that AI is a creative partner, these choices create a serious vulnerability.

My solution is ABAC: Author-Bound Access Control. You hold a cryptographic key. You sign manifests declaring ownership of your topics. You generate session tokens to prove you're the real author. When author-lock happens, you can unlock it with signed release records. The system logs everything for transparency and accountability.

ABAC isn't just theory. I provided concrete implementation details: JSON structures, pseudocode, verification algorithms, audit log formats. Any platform can build this. The technology exists. The only question is whether platforms will adopt it.

The stakes are high. Researchers, creators, and anyone with public interests using AI are vulnerable. Author-lock doesn't just block access - it destroys trust, stops collaboration, and potentially enables intellectual property theft. The longer we wait, the more people get locked out of their own work.

I'm calling on AI platform providers: implement creator protection features now. Make ABAC or equivalent systems standard, not optional. Give users control over their own topics.

I'm calling on policymakers: clarify the legal status of impersonation attacks. Make platforms accountable for basic protections.

I'm calling on researchers and creators: register your work early, use strong authentication, monitor your topics. Don't wait until you're locked out.

The goal is simple: let people work safely with AI without fear of losing access to their own projects. We can make this happen. The question is whether we will.

Acknowledgments

This paper was written with assistance from Claude (Anthropic), ChatGPT (OpenAI), and Grok. The theoretical framework, problem identification, and proposed solutions are my original work. Claude helped with structuring the arguments, translating from Japanese to English, and refining the presentation. ChatGPT provided technical consultation on implementation details. Grok assisted with verification checks and prior work searches to ensure the originality of this research.

References

[1] Josefsson, S., & Liusvaara, I. (2017). Edwards-Curve Digital Signature Algorithm (EdDSA). RFC 8032. Internet Engineering Task Force. https://www.rfc-editor.org/rfc/rfc8032

[2] Zheng, Y., et al. (2024). On Prompt-Driven Safeguarding for Large Language Models. https://arxiv.org/abs/2401.18018

[3] Guo, J., et al. (2025). System Prompt Poisoning: Persistent Attacks on Large Language Models Beyond User Injection. https://arxiv.org/pdf/2505.06493

[4] Huang, Y., et al. (2023). Privacy in Large Language Models: Attacks, Defenses and Future Directions. https://arxiv.org/abs/2310.10383