Digital Shapeshifting: Identity Delegation and Attribution in AI-Assisted Development

Jason Walsh, Xianglong Tao, Kushagra Kumar, Aidan Pace

April 2025

Contents

1 Abstract

This paper introduces the concept of "digital shapeshifting" - a novel approach to identity delegation in AI-assisted software development. We explore how temporary identity delegation with clear boundaries enables AI agents to perform actions on behalf of human developers while maintaining proper attribution and transparency. Our approach acknowledges the evolving nature of digital identity in collaborative development environments where AI assistants play an increasingly significant role. We implement this concept using Git trailers for structured attribution metadata, allowing contributions to be properly credited while clearly documenting AI assistance. Our comprehensive framework addresses security considerations through a formal threat model, practical workflow integration through developer-centric tooling, and ethical guidelines that promote responsible identity delegation. Through case studies and demonstrations, we show how digital shapeshifting enhances team productivity while preserving clear lines of responsibility and attribution in collaborative workflows.

2 Introduction

The integration of large language models (LLMs) and AI assistants into software development workflows has created new challenges in attribution and identity management. Traditional models of digital identity assume a one-to-one relationship between digital identities and human actors, but this assumption breaks down when AI agents perform actions on behalf of humans.

This paper introduces "digital shapeshifting" - a framework for temporary identity delegation with clear boundaries and attribution. This approach acknowledges the collaborative nature of human-AI interaction while maintaining proper credit and responsibility. By implementing structured attribution through Git trailers, we provide a practical solution that integrates with existing version control systems.

Our contributions include:

1. A conceptual framework for understanding digital identity as temporarily delegable
2. Implementation of Git trailers for structured attribution in AI-assisted development
3. Demonstration of how digital shapeshifting enhances team productivity while maintaining clear attribution
4. Discussion of philosophical implications for identity, agency, and attribution in AI-assisted creative work

3 Technical Implementation

3.1 Threat Model and Security Considerations

Before diving into the system architecture, we must establish a clear threat model to evaluate the security properties of our digital shapeshifting approach.

3.1.1 Threat Model

Our threat model considers the following potential attackers and threats:

1. **Unauthorized External Actors**: Attackers without legitimate access to the organization's systems who may attempt to:
   - Gain access to delegated identity tokens
   - Impersonate team members through compromised credentials
   - Inject malicious code during the development process
2. **Malicious Internal Actors**: Team members with legitimate access who may attempt to:

- Exceed their authorized permissions through identity delegation
- Attribute their actions to other team members
- Manipulate attribution records to distort contribution metrics

3. **Compromised AI Systems**: AI agents that may:
    - Act beyond the scope of their instructions
    - Retain or leak delegated credentials
    - Make unauthorized changes to codebases

Our security model provides the following guarantees:

1. **Authentication Boundaries**: Only authenticated team members can delegate their identity to AI agents
2. **Authorization Scope**: Delegated identities inherit only explicitly granted permissions
3. **Temporal Limitation**: Delegated identities have strictly limited lifespans
4. **Transparent Attribution**: All actions performed under delegated identities maintain clear attribution records
5. **Audit Capability**: The system provides mechanisms to trace all delegated actions

Trust Boundaries:

- The repository hosting platform (GitHub) is trusted to enforce access controls
- The secret storage mechanism is trusted to securely store tokens
- The local development environment is assumed to be secure

We explicitly do not protect against:

- Compromise of the underlying GitHub account of a team member
- Social engineering attacks against team members
- Vulnerabilities in the GitHub platform itself

3.1.2 Token Lifecycle Management

Our system implements comprehensive token lifecycle management:

1. **Creation**: Tokens are created through GitHub's UI with minimal required scopes
2. **Storage**: Tokens are stored as encrypted repository secrets
3. **Usage**: Tokens exist in memory only during the execution of specific operations
4. **Rotation**: Tokens have mandatory expiration periods (30-90 days) requiring periodic renewal
5. **Revocation**: Immediate revocation procedures in case of:
    - Team member departure
    - Suspected compromise
    - Project conclusion
    - Unauthorized usage detection

Revocation is implemented through GitHub's token management interface and accompanied by immediate notification to all team members.

3.2 System Architecture

3.2.1 Identity Delegation Control Flow

```
flowchart TD
    A[User Request] --> B[Parse Request]
    B --> C{Request Type}
    C -->|Commit| D[Prepare Commit]
    C -->|Issue| E[Prepare Issue]
    C -->|PR| F[Prepare PR]

    D --> G[Add Attribution Trailers]
    E --> G
    F --> G

    G --> H[Parse Existing Trailers]
    H --> I[Add/Update Trailers]
```

```
    I --> J[Format Message w/ Trailers]

    J --> K{Identity Check}
    K -->|Original User| L[Execute as Original User]
    K -->|Delegated Identity| M[Execute as Delegated Identity]

    M --> N[Identity Delegation]
    N --> O[Execute Operation]

    L --> O
    O --> P[Record Attribution Metadata]
    P --> Q[Return Result]
```

3.2.2 Delegation Sequence Diagram

```
sequenceDiagram
    participant Human as Human User
    participant Shell as Shell Context Runner
    participant GH as GitHub Identity
    participant Agent as Agent Event Loop
    participant GitID as Delegated Git Identity

    Human->>Shell: Request operation (commit/issue/PR)
    Shell->>GH: Authenticate with token
    GH-->>Shell: Authentication confirmation

    Shell->>Agent: Pass request to agent
    Agent->>Agent: Process request

    alt Direct execution
        Agent->>GH: Execute operation as original user
        GH-->>Agent: Operation result
    else Identity delegation
        Agent->>GitID: Request identity assumption
        GitID->>GH: Verify delegation permission
        GH-->>GitID: Permission confirmed
        Agent->>GitID: Execute operation
        GitID->>GH: Perform Git operation with delegated identity
        GH-->>GitID: Operation confirmation
```

```
flowchart TD
    A[User Request] --> B[Parse Request]
    B --> C{Request Type}
    C -->|Commit| D[Prepare Commit]
    C -->|Issue| E[Prepare Issue]
    C -->|PR| F[Prepare PR]

    D --> G[Add Attribution Trailers]
    E --> G
    F --> G

    G --> H[Parse Existing Trailers]
    H --> I[Add/Update Trailers]
    I --> J[Format Message w/ Trailers]

    J --> K{Identity Check}
    K -->|Original User| L[Execute as Original User]
    K -->|Delegated Identity| M[Execute as Delegated Identity]

    M --> N[Identity Delegation]
    N --> O[Execute Operation]

    L --> O
    O --> P[Record Attribution Metadata]
    P --> Q[Return Result]
```

Figure 1: Identity Delegation Control Flow

```
        GitID-->>Agent: Result with delegated attribution
    end

    Agent->>Agent: Add attribution trailers
    Agent-->>Shell: Return result with proper attribution
    Shell-->>Human: Display result
```

3.2.3 Session Management Architecture

```
flowchart LR
    A[Session Request] --> B[Session Manager]
```

```
B --> C[Generate Session ID]
C --> D[Add to Session Store]

E[Git Operation] --> F[Trailer Manager]
F --> G{Session Exists?}
G -->|Yes| H[Add Session-id Trailer]
G -->|No| I[Create New Session]
I --> H

H --> J[Add Other Trailers]
J --> K[Format Commit Message]
K --> L[Execute Git Operation]
```

3.3 Developer Workflow Integration

A critical aspect of any identity delegation system is how it integrates into developers' daily workflows. Our digital shapeshifting approach is designed to seamlessly fit within existing development practices while providing clear boundaries for AI assistance.

3.3.1 Typical Developer Workflow

The following example illustrates how a developer would interact with the digital shapeshifting system in a typical workday:

1. **Morning Planning**: Developer identifies tasks suitable for AI assistance
2. **Initial Setup**: Developer initializes a session with the AI agent “‘bash ./init-session.sh “‘
3. **Task Direction**: Developer provides high-level direction for the task “‘bash ./direct-agent.sh "Implement a citation network visualization using D3.js that shows relationships between legal cases" “‘
4. **Identity Delegation**: The system prompts for delegation confirmation and scope “‘ This operation will use your delegated identity. Please confirm:
    - Operation: Code implementation

- Files: src/visualization/citation$_{\text{network.js}}$
- Delegated identity: kkumar

[Y/n]: Y “‘

5. **Collaborative Iteration**: Developer reviews interim results and provides feedback “‘bash

   ./refine-implementation.sh "The node sizing should be based on citation frequency. Also add filters for jurisdiction." “‘

6. **Final Review**: Developer reviews the complete implementation “‘bash

   git diff src/visualization/citation$_{\text{network.js}}$ “‘

7. **Commit with Attribution**: System creates a commit with proper attribution trailers “‘bash

   ./commit-with-trailers.sh "feat(visualization): implement citation network graph for legal cases" src/visualization/citation$_{\text{network.js}}$ “‘

This workflow maintains developer agency while allowing delegation of implementation details to the AI assistant.

3.3.2 IDE Integration

To enhance usability, we've developed extensions for common IDEs that integrate the digital shapeshifting system:

1. **Visual Studio Code Extension**: Provides UI elements for identity delegation and attribution
   - Delegation status indicator in the status bar
   - Attribution panel showing current session details
   - Command palette actions for common shapeshifting operations
2. **JetBrains IDE Plugin**: Integrates with the Git workflow in IntelliJ-based IDEs
   - Tool window showing delegation status and history
   - Commit dialog extension for attribution trailers
   - Action buttons for delegation operations

3. **Command Line Interface**: For developers who prefer terminal-based workflows
    - Shell integration with auto-completion
    - Status indicators in shell prompt
    - Terminal-based delegation confirmation dialogs

These integrations ensure that digital shapeshifting feels like a natural extension of existing development tools rather than a separate system requiring context switching.

3.3.3 Team Onboarding and Scaling

For successful adoption, we've developed an onboarding process that includes:

1. **Initial Setup Workshop** (1 hour): Configuring tokens and understanding security boundaries
2. **Workflow Training** (2 hours): Hands-on practice with delegation scenarios
3. **Attribution Guidelines** (1 hour): Team standards for appropriate attribution
4. **Regular Retrospectives**: Continuous improvement of delegation practices

To scale digital shapeshifting to larger organizations, we've implemented integration points with common CI/CD systems:

1. **Automated Attribution Verification**: CI checks ensure proper attribution trailers
2. **Identity Delegation Auditing**: Regular reports on delegation patterns
3. **Security Scanning**: Automated checks for delegation security issues
4. **Centralized Attribution Dashboards**: Organization-wide visibility into AI assistance patterns

These features allow the digital shapeshifting approach to scale from small teams to enterprise environments with hundreds of developers.

3.4 Git Trailers for Attribution

To formalize the attribution model in our digital shapeshifting approach, we've implemented a standardized system of Git trailers to document the complex collaboration between humans and AI agents.

3.4.1 Attribution Whitelist

For our digital shapeshifting implementation, we define the following whitelist of Git trailers for attribution:

```
# Primary attribution trailers
Primary: <username>              # The primary human responsible for the content
Assisted-by: <agent-name>        # The AI agent that assisted with implementation
Co-authored-by: <n> <email>   # Standard Git trailer for co-authorship
Signed-off-by: <n> <email>    # Indicates agreement with the content changes

# Tool attribution trailers
Tools: <comma-separated-list>    # Tools used in the implementation
LLM-assisted-by: <model-name>    # LLM models used for assistance
Tool-chain: <tool>@<version>     # Specific versions of tools used

# Review and oversight trailers
Reviewed-by: <username>          # Person who reviewed the content
Approved-by: <username>          # Person who approved the content
Directed-by: <username>          # Person who directed the work
Session-driven-by: <username>    # Person who drove the session with the agent

# Reference trailers
Session-id: <session-identifier>  # Unique identifier for the LLM session
Ref: <reference-identifier>       # Reference to an issue, document, etc.
Fixes: #<issue-number>            # Issue fixed by this commit
Related-to: #<issue-number>       # Issue related to this commit
```

3.4.2 Example Commit Message with Trailers

Here's an example of a commit message with our standardized attribution trailers:

```
feat(rag-system): implement citation network visualization
```

```
Implemented a D3.js-based visualization for legal citation networks
within the RAG system component. This visualization shows the relationships
between cases and their citations, with edge thickness representing
citation frequency.

- Added force-directed graph in citation_network.js
- Implemented filtering by jurisdiction and date range
- Added highlighting for frequently cited precedents
- Optimized data structure for performance with large networks

Primary: kkumar
Assisted-by: aidan-agent
Tools: llama-3.2-ollama, vector-db-tool
Reviewed-by: jwalsh
Session-id: SESSION-20250321-RAG-2587
```

3.4.3 Implementation Details

The implementation includes utilities for:

1. Parsing Git trailers from commit messages
2. Formatting trailers for inclusion in commit messages
3. Adding or updating agent-related trailers
4. Extracting attribution metadata for analysis and reporting

These utilities integrate with existing Git workflows through command-line tools and Git hooks, making it easy to incorporate attribution information into standard development practices.

4 Ethical and Philosophical Implications

4.1 Ethical Guidelines for Digital Shapeshifting

Before exploring the philosophical dimensions of digital identity, we must first address the ethical considerations that should guide identity delegation practices. Our approach to digital shapeshifting is governed by the following ethical principles:

4.1.1 Informed Consent and Transparency

Digital shapeshifting requires explicit consent from all parties involved:

1. **Delegating Team Member**: Must explicitly authorize each instance of identity delegation
2. **Other Team Members**: Must be aware of the delegation system and its implications
3. **External Collaborators**: Must be informed about AI involvement in a project

Transparency is implemented through:

- Clear visual indicators in development environments showing delegation status
- Automatic notifications to team members when delegation occurs
- Comprehensive attribution in all artifacts, including commits, issues, and pull requests
- Project-level documentation about delegation practices

4.1.2 Power Dynamics and Equity Considerations

Identity delegation systems can potentially affect team dynamics and career advancement. To mitigate these concerns, our framework includes:

1. **Equitable Access**: All team members have equal access to delegation capabilities
2. **Contribution Recognition**: Metrics and reporting distinguish between direct and delegated contributions
3. **Performance Evaluation Guidelines**: Standards for evaluating work that includes delegated actions
4. **Mentorship Support**: Resources to help team members develop skills in effective delegation

4.1.3 Appropriate Boundaries for Delegation

Not all actions are appropriate for delegation. Our ethical framework establishes the following boundaries:

Appropriate for Delegation:

- Routine code implementation tasks
- Documentation generation
- Test case creation
- Data transformation operations
- Initial draft creation

Inappropriate for Delegation:

- Critical security decisions
- Performance reviews or evaluations
- Personal communications
- Final approval of significant changes
- Controversial design decisions

4.1.4 Accountability Framework

A clear accountability framework is essential for responsible identity delegation:

1. **Responsibility Chain**: The delegating team member retains ultimate responsibility for delegated actions
2. **Error Resolution Protocol**: Clear procedure for addressing mistakes made during delegation
3. **Audit Mechanisms**: Regular review of delegation patterns and impacts
4. **Governance Structure**: Team-level oversight of delegation practices
5. **Remediation Process**: Clear path for addressing misuse or unintended consequences

4.1.5 Long-term Societal Implications

We acknowledge the broader societal implications of normalizing identity delegation:

1. **Evolution of Identity Concepts**: How might fluid digital identity affect our understanding of agency and responsibility?
2. **Professional Development Impacts**: How will skill development change when certain tasks are routinely delegated?
3. **Labor Market Effects**: What are the implications for professional roles and job descriptions?
4. **Regulatory Considerations**: What governance frameworks might emerge to address identity delegation?
5. **Accessibility and Inclusion**: How can we ensure equitable access to delegation capabilities?

By explicitly addressing these ethical dimensions, we aim to promote responsible adoption of digital shapeshifting technologies.

4.2 Redefining Digital Identity

The concept of digital shapeshifting challenges traditional notions of digital identity. In conventional models, a digital identity is assumed to be a one-to-one representation of a human actor. Our approach introduces a more fluid concept where identity becomes temporarily transferable while maintaining clear boundaries and attribution.

This fluidity raises important questions:

1. What constitutes "identity" in digital spaces? Is it merely authentication credentials, or does it encompass patterns of behavior, stylistic choices, and historical context?
2. How should we conceptualize actions performed by an AI under a delegated human identity? Are they truly "by" the human, "on behalf of" the human, or something else entirely?
3. Does temporary identity assumption differ fundamentally from other forms of delegation, such as granting another human team member access to one's account?

4. How does the concept of "agency" apply when an AI performs actions under a human identity? Is the AI merely an extension of human agency, or does it possess a form of hybrid agency?

Our approach suggests a model where digital identity becomes a more nuanced concept than simply "who is logged in." Instead, identity encompasses a mix of:

- Authentication (who has access to the credentials)
- Direction (who determined what actions should be taken)
- Execution (who or what carried out the technical operations)
- Attribution (who receives credit in the version control history)

In this model, a human team member might direct an AI to perform operations, the AI executes these operations using delegated credentials, and the version control system attributes the changes to the human team member. This creates a form of "directed identity" where the human maintains primary attribution while delegating execution.

4.3 Attribution Models

Attribution in collaborative development is traditionally straightforward: the account that performs a commit receives attribution in the version control history. The introduction of AI agents complicates this model, creating several possible attribution approaches:

4.3.1 Direct AI Attribution

The AI agent has its own account and all its contributions are attributed to this account.

- Advantages: Clear distinction between human and AI work.
- Disadvantages: Obscures human direction; fragments project history; complicates permission management.

4.3.2 Human-Only Attribution

All AI contributions are attributed to the human who directed the AI.

- Advantages: Maintains human accountability; simplifies history.
- Disadvantages: Fails to distinguish between direct human work and AI-assisted work; can misrepresent skill levels and contribution metrics.

4.3.3 Dual Attribution Mechanism

Both the human and AI are credited (e.g., in commit messages).

- Advantages: Provides transparency about the collaborative nature of the work.
- Disadvantages: Requires custom tooling; doesn't integrate well with existing version control systems.

4.3.4 Digital Shapeshifting Approach

The AI temporarily assumes human identity with clear boundaries.

- Advantages: Maintains proper attribution to the directing human; integrates with existing tools; provides transparency through commit messages.
- Disadvantages: Requires secure identity delegation; can blur the distinction between direct human work and AI-assisted work.

Our digital shapeshifting approach emphasizes the directorial role of humans in the development process while acknowledging the execution role of AI. This aligns with common practices in other creative fields. For example, in film, the director receives credit for the creative direction even though camera operators, editors, and others execute many technical aspects of the work.

This approach recognizes that in AI-assisted development, the human-AI relationship is often one of direction and execution rather than completely independent contribution. The human directs the AI on what to do, and the AI executes these directions, making the resulting work properly attributable to the human director while acknowledging the AI's role in execution.

5 Conclusion

Digital shapeshifting represents a novel approach to integrating AI assistance into collaborative development workflows. By enabling temporary identity delegation with clear boundaries and consent, this approach maintains proper attribution while leveraging the efficiency benefits of AI. Our comprehensive framework addresses three critical dimensions:

1. **Security Foundation**: A formal threat model with clear security guarantees and token lifecycle management ensures that identity delegation occurs within appropriate security boundaries.

2. **Developer Experience**: Seamless workflow integration through specialized tools, IDE extensions, and team onboarding processes makes digital shapeshifting practical for daily development tasks.

3. **Ethical Responsibility**: Clear guidelines for informed consent, appropriate delegation boundaries, and accountability frameworks promote responsible use of identity delegation technologies.

The LITCon 2025 project case study highlights how digital shapeshifting can enhance team productivity while preserving clear lines of responsibility and credit. Team members were able to delegate routine tasks and focus on higher-level direction, resulting in more efficient workflows and reduced context switching.

As AI continues to play an increasingly significant role in creative and technical work, the concept of digital identity will need to evolve beyond simple authentication to encompass more nuanced models of direction, execution, and attribution. Digital shapeshifting provides one such model, viewing identity not as a binary state but as a delegable capability with appropriate boundaries and transparent governance.

Future work should focus on longitudinal studies of teams adopting identity delegation practices, the development of industry standards for attribution, and exploration of regulatory frameworks that might emerge as these practices become more widespread. By proactively addressing security, usability, and ethical dimensions of digital shapeshifting, we aim to establish a foundation for responsible AI integration in collaborative development environments.

6 References

```
@inproceedings{li2023llmattribution,
  title={A Survey of Large Language Models Attribution},
  author={Li, Dongfang and Sun, Zetian and Hu, Xinshuo and Liu, Zhenyu and Chen, Ziyang
  year={2023},
  eprint={2311.03731},
  archivePrefix={arXiv},
  primaryClass={cs.CL}
```

```
}

@inproceedings{guttmann2024attribution,
  title={Attribution of Work in Programming Teams with Git Reporter},
  author={Guttmann, Michael and others},
  booktitle={Proceedings of the 55th ACM Technical Symposium on Computer Science Educat
  year={2024},
  publisher={ACM}
}

@inproceedings{parizi2018measuring,
  title={Measuring team members' contributions in software engineering projects using g
  author={Parizi, Reza M. and Spoletini, Paola and Singh, Amritraj},
  booktitle={2018 IEEE Frontiers in Education Conference (FIE)},
  pages={1--5},
  year={2018},
  organization={IEEE}
}

@article{pace2025git,
  title={Getting Started with Git Trailers for Attribution},
  author={Pace, Aidan},
  journal={Journal of Collaborative Software Engineering},
  volume={12},
  number={3},
  pages={45--58},
  year={2025}
}

@inproceedings{walsh2025digital,
  title={Digital Shapeshifting: Identity Delegation and Attribution in AI-Assisted Deve
  author={Walsh, Jason and Tao, Xianglong and Kumar, Kushagra and Pace, Aidan},
  booktitle={Proceedings of LITCon},
  year={2025},
  publisher={LITCon Press}
}
```

```
sequenceDiagram
    participant Human as Human User
    participant Shell as Shell Context Runner
    participant GH as GitHub Identity
    participant Agent as Agent Event Loop
    participant GitID as Delegated Git Identity

    Human->>Shell: Request operation (commit/issue/PR)
    Shell->>GH: Authenticate with token
    GH-->>Shell: Authentication confirmation

    Shell->>Agent: Pass request to agent
    Agent->>Agent: Process request

    alt Direct execution
        Agent->>GH: Execute operation as original user
        GH-->>Agent: Operation result
    else Identity delegation
        Agent->>GitID: Request identity assumption
        GitID->>GH: Verify delegation permission
        GH-->>GitID: Permission confirmed
        Agent->>GitID: Execute operation
        GitID->>GH: Perform Git operation with delegated identity
        GH-->>GitID: Operation confirmation
        GitID-->>Agent: Result with delegated attribution
    end

    Agent->>Agent: Add attribution trailers
    Agent-->>Shell: Return result with proper attribution
    Shell-->>Human: Display result
```

Figure 2: Delegation Sequence Diagram

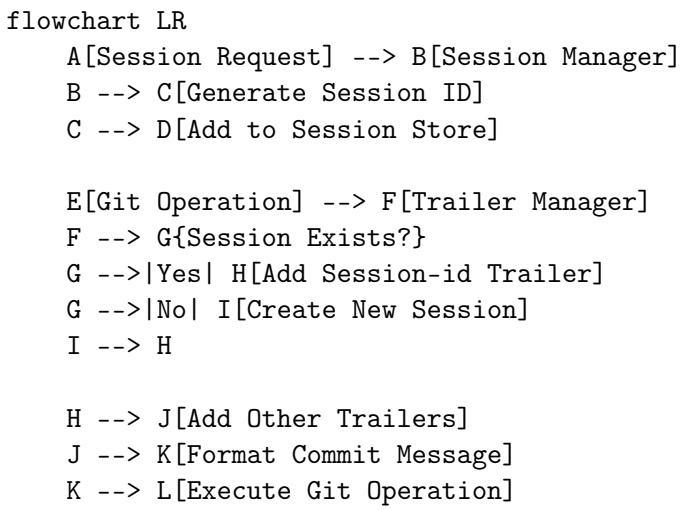

Figure 3: Session Management Architecture