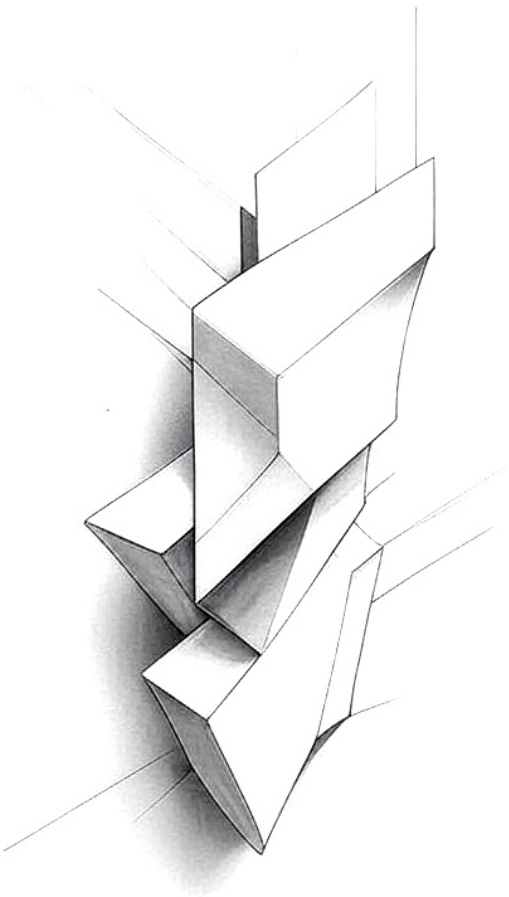
THE NEW RULES

Developer's Guide to AI Era



by Andrzej Tuchołka

THE NEW RULES

This file contains only one chapter from the full book. You can get the PDF version of the book on its website.

Disclaimer: This book was created with the assistance of artificial intelligence. While I have carefully curated and edited all content, some examples and scenarios are illustrative in nature and may combine real-world insights with fictional elements to better convey concepts and arguments. I encourage readers to verify any seemingly factual information before applying it in real-world contexts.

First Edition: 2025
ISBN: 978-83-976978-0-5

Published by Y Experiment Sp. z o.o.
Website: https://www.thenewrules.ai

Chapter 8

The Trust Protocol

In an age where AI can fake expertise, trust is the only currency that matters.

— The New Rules, 2025

Trust collapsed overnight in 2025. A junior developer with ChatGPT now sounds more knowledgeable than a principal engineer with twenty years of experience. A weekend hobbyist generates documentation more authoritative than official maintainer guides. A complete newcomer creates technical content indistinguishable from industry veterans. This isn't a temporary glitch — it's the permanent reality. When expertise can be simulated with a prompt, the fundamental question becomes unavoidable: how do we determine who actually knows what they're talking about?

Welcome to the Trust Protocol: the new rules for establishing credibility when AI has broken every traditional signal of expertise. Your

reputation, your career, and your influence depend on mastering these rules now — not tomorrow.

8.1 Proving Expertise in the AI Age

AI didn't create a knowledge shortage — it triggered a devastating trust collapse. When everyone sounds like an expert, expertise itself becomes meaningless. When flawless technical prose flows from novice fingers, our fundamental signals of competence shatter.

Every traditional trust signal failed simultaneously in 2024. Clear technical writing indicated clear technical thinking until GPT-4 started writing better technical documentation than 98% of senior engineers. Stanford PhDs carried weight until self-taught developers with Claude assistance matched formal education output in 72 hours. Using technical terms correctly signaled domain knowledge until Anthropic's Claude knew every industry term perfectly. Prolific technical writing suggested expertise until AI generated 1,000-word technical articles in 15 seconds.

Trust evaporated literally overnight. The old signals died — not gradually, but instantly. In their place, new indicators emerged that prove significantly harder to fake and more expensive to simulate.

> **Key Point: AI generates stories — humans survive failure.**
>
> When Sarah Drasner shares exactly how she deleted a production database serving 8.7 million customers at 3:17 AM, including the exact $3,847 AWS bill for recovery, you know she lived it.
>
> AI doesn't taste panic at seeing "712 rows affected" when you expected one.

Edge Case Knowledge: AI masters the happy path perfectly. Humans intimately know where systems break. That weird bug that only happens when users have emoji in their username during daylight savings transitions in Brazil? That's experience no prompt can simulate.

Historical Context: AI knows current best practices. Humans remember the journey. "We used Redux for everything until the 2019 React Hooks release changed our entire mental model" — these evolution stories can't be synthesized from training data.

Inconsistency: AI maintains perfect consistency. Humans evolve their thinking. When you find Kent C. Dodds's blog posts from 2018 arguing the opposite of his 2024 position, with an explanation of exactly which projects changed his mind, that's authentic human expertise.

StackOverflow's April 2024 AI crisis exemplifies the verification collapse. Within 48 hours of Claude 3 Opus release, AI-generated answers flooded the platform — technically correct but subtly wrong solutions that earned upvotes from users who couldn't verify correctness. Their reputation system, built on human expertise, corrupted overnight.

Their response? A three-part strategy implemented by May 2024: Ban AI-generated content (unenforceable). Focus on explanation depth requiring specific examples from personal experience. Implement mandatory community verification through a "human-verified" badge system that requires three established users to confirm an answer.

The platform transformed from knowledge repository to trust network overnight. StackOverflow stopped being about answers. It became about knowing whose answers to trust.

8.2 Track Records, Not Credentials

Your GitHub contribution graph speaks louder than your degree. When traditional credentials collapsed as trust signals in 2024, track records instantly became the new resume. Not what you claim to know — what you've demonstrably built, shipped, and maintained.

The shift isn't subtle: from potential to proven, from promises to public proof. Modern trust assessment operates across four unmistakable dimensions that AI can't fake.

- **Longevity** matters — five years of participation where your evolution of expertise is publicly visible. Pre-AI contributions carry 3x weight because they couldn't be generated.
- **Depth** shows in complexity of contributions — not just using tools but building them, not just fixing bugs but creating frameworks that prevent entire classes of bugs.
- **Breadth** demonstrates range through multiple projects and cross-domain expertise that only comes from genuine experience.
- **Impact** provides measurable outcomes — 10,000 users actually

using your work, problems actually solved in production.

CASE STUDY Sindre Sorhus built an unassailable trust moat through verifiable track record. 1,100+ npm packages maintained over 12 years. 546 million weekly downloads. Responsive to 7,800+ issues across projects.

The measurable result? Any new tool he releases gets immediate adoption. Companies like Vercel and Netlify fund his open source work without question. His code design decisions shape the entire JavaScript ecosystem.

His credential? None required. His publicly verifiable track record speaks louder than any degree possibly could.

GitHub transformed from code repository to trust portfolio for developers. But the metrics that matter aren't what most people measure. Star counts? Easily gamed with 50 bots for $12.99. Repository quantity? AI generates thousands overnight. What actually matters: contribution consistency (those green squares spanning years), code quality reviews by respected maintainers, thoughtful PR review comments that demonstrate deep system understanding, and visible evolution of coding style that shows genuine learning.

For developers starting today, building trust requires strategic track record building. Document your journey in public — build in public from day one, sharing both failures and successes with equal transparency. Focus relentlessly on longevity by maintaining projects for years, not months. Seek verification through contributing to established projects where respected developers can vouch for your work. Create unique value by solving problems AI fundamentally can't address through original research and unique specialization.

Build your track record like your career depends on it. Because in 2025, it absolutely does.

8.3 The Power of Public Failure and Learning

Public failure is the new credential. In the 2025 trust economy, your failures create more credibility than your successes. Success can be faked with a prompt or lucked into through chance. Failure — real,

painful, public failure — can only be earned through genuine attempt and authentic experience.

CASE STUDY Troy Hunt built an unassailable trust empire through systematic public failure documentation. Every security incident became a detailed 2,500-word blog post within 24 hours. Every architectural mistake acknowledged openly with exact costs. When Have I Been Pwned experienced a 43-minute outage affecting 7.8 million queries, Troy didn't hide — he published the full incident timeline, root cause, and mitigation steps before users even complained.

The quantifiable trust built: He became Microsoft's go-to security authority. Fortune 500 corporations engage him at $15,000 per consultation. He receives sensitive breach data from white-hat hackers who trust no one else. His security recommendations shape organizational practices worldwide. His failures built more credibility than a CISSP certification ever could.

Failure signals trust for three concrete reasons. First, it's expensive: both financially and reputationally. The developer who documents losing $27,000 in AWS costs from a misconfigured Kubernetes cluster is demonstrating skin in the game that AI simply cannot simulate.

Second, failure is deeply specific. "The server crashed" is generic. "The server crashed at 2:47 AM because I didn't implement connection pooling limits and 12,872 concurrent users created exactly as many database connections, overwhelming RDS reserved capacity" includes details that only come from lived experience.

Third, failure creates visible growth — each documented mistake prevents multiple future errors, creating pattern recognition that AI cannot replicate. They document their mistakes in real-time, not after the fact.

The most trusted developers now document failures in real-time. They tweet debugging struggles as they happen. They publish detailed post-mortems of incidents with exact costs and consequences. They own mistakes without excuses, focusing exclusively on learning. They engage their community transparently during crises, asking for help and sharing solutions that worked.

As failure became valuable currency, fake failure emerged just as

quickly. Watch for these warning signs: vague details that could apply anywhere, humble-bragging disguised as vulnerability, and carefully curated "failures" with no real consequences or costs. Real failure: "I deleted our production database serving 450,000 active users. Recovery took 17 hours. Here's the AWS bill: $3,847." Fake failure: "I learned so much from my mistakes building scalable systems."

Build a public failure portfolio. Document your disasters. Share exact costs. Show your learning. Your biggest professional failures will become your greatest trust assets — if you have the courage to own them in public.

8.4 Community Vouching and Reputation

Trust isn't individual anymore — it's networked. As every individual credential collapsed under AI pressure, community verification became the only reliable signal. You're not trusted because of what you know; you're trusted because of who trusts you. This shift fundamentally rewired how technical reputation works.

Modern developer trust operates like an interlocking web with quantifiable components. Direct vouching from collaborators provides primary trust — when Kent C. Dodds publicly mentions your React component library, his credibility transfers to you. Indirect vouching amplifies reach — when a developer trusted by Addy Osmani recommends your work, some of Addy's trust flows transitively to you. Reputation inheritance happens automatically when joining high-trust organizations — a developer hired by Vercel instantly receives a trust boost. Community standing grows through visible roles in respected projects — becoming a Node.js core maintainer signals verified capability.

The mechanisms evolved beyond informal endorsements. Merged pull requests in major projects (like React or Kubernetes) signal capability more effectively than any credential. The quality of your code review comments demonstrates deeper understanding than any interview could. Speaking invitations at conferences like React Conf or GopherCon show peer recognition no resume can match. Companies that rigorously verify technical skills (like Stripe or Cloudflare) vouch implicitly through their hiring decisions. Team recommendations from respected engineers carry more weight than any recommendation letter.

CASE STUDY The Rust community built the most sophisticated trust network in software. Their RFC process requires deep engagement — an average proposal receives 157 comments over 34 days. Their tiered mentorship programs create verifiable trust chains where mentee success reflects on mentors. Their code review culture emphasizes learning — the average PR receives 14 substantive comments. Their team structure relies on demonstrated expertise rather than titles or seniority.

Their trust indicators work without centralization: team membership signals capability verified by existing experts, RFC authorship demonstrates deep thinking visible to the entire community, review quality shows understanding that can't be faked, mentorship creates trust inheritance across experience levels.

Trust flows transitively but diminishes with distance. Bad actors get identified within hours. Quality maintains despite 20% year-over-year community growth. Expertise gets verified through participation rather than declaration.

The result? The most sophisticated language became the most welcoming community because trust verification solved the expertise problem.

Building reputation in networks requires deliberate strategy. Start small — join communities long before trying to influence them. Contribute consistently — regular micro-contributions build more trust than sporadic large ones. Help others publicly by answering questions in forums and discussions. Admit limitations openly — knowing what you don't know signals maturity faster than pretending expertise. Respect community culture rather than trying to reform it immediately. Be patient — network trust accumulates slowly but compounds powerfully.

Trust remains frustratingly non-portable across platforms. Your 15K GitHub stars don't transfer to GitLab. Your 97,843 StackOverflow reputation means nothing on Reddit. Your 32,000 Twitter followers don't imply technical competence to hiring managers. The emerging solution: multi-platform presence with consistent usernames, cross-referenced identity across services, verifiable contribution patterns across ecosystems, and community-specific investment rather than trying to transfer reputation directly.

8.5 The Corporate Trust Challenge

Hiring broke in 2024. Companies face an existential trust dilemma that traditional processes can't solve: how do you verify genuine expertise when every signal can be faked? How do you build organizational credibility when any competitor can claim identical technical sophistication with AI-generated content?

Traditional technical interviews collapsed spectacularly by March 2024. Candidates routinely use Claude and GPT-4 during remote interviews. LeetCode solutions appear perfectly memorized because they are — by AI assistants feeding answers through earpieces. System design answers sound expert-level because they're generated by models trained on expert solutions. Technical knowledge questions get answered flawlessly through AI-powered search happening out of frame. The interviewing arms race spiraled out of control.

CASE STUDY Cloudflare mastered corporate trust building through radical transparency during service incidents. Within 27 minutes of their June 2024 global outage, CTO John Graham-Cumming was personally posting detailed technical updates on their status page. Within 4 hours, a comprehensive technical post-mortem appeared on their engineering blog with exact timeline, root cause analysis, and specific mitigation steps. The company's principal engineers appeared on industry podcasts within 48 hours discussing what they learned.

The measurable results: Their infrastructure remains trusted by 32% of the internet despite competitors offering lower prices. Top engineering talent competes fiercely for open positions, with 87% of senior hires citing transparency as a key factor. Their open-source projects receive substantial external contributions. Their technical credibility compounds, creating a moat against competitors who hide their failures.

New approaches emerged from necessity. Pair building sessions became standard — building something non-trivial together in real-time while observing problem-solving process, not just outcomes. Code archaeology interviews focus on reviewing candidates' actual project his-

tory with detailed questions about specific technical decisions that AI can't answer. Community verification now involves explicitly checking reputation in relevant communities and directly contacting previous collaborators. Paid trials replaced interviews — candidates tackle real problems with real constraints, with compensation for their time.

Companies simultaneously adapted their external trust building. Engineering blogs featuring detailed failure post-mortems built authentic trust. Open source contributions now focus on meaningful projects that demonstrate genuine capability, not token contributions. Team transparency includes public engineer profiles showing actual depth of expertise. Individual recognition attributes specific work to specific engineers, supporting authenticity claims.

For companies building trust today, the formula is clear: radical transparency about failures, ownership of mistakes, technical depth in communications, and engineer-led visibility. Organizational trust, just like individual trust, emerges from demonstrated integrity over time — not from marketing claims or AI-generated competence theater.

8.6 The Trust Protocol Implementation

Trust isn't just earned — it's built through deliberate strategies. Individual developers must now treat trust as their most valuable professional asset, requiring systematic investment and protection.

Key Point: Implement trust-building internally and externally.
Internal practices include hiring based on verifiable track records rather than credentials, supporting public technical sharing by engineers, and rewarding long-term thinking over short-term wins.

External practices require sharing failures transparently within hours of incidents, open-sourcing meaningful work that demonstrates real capability, and building genuine community relationships rather than extractive engagement.

For developers navigating the new trust landscape, personal trust building follows a clear four-phase framework. **Foundation work** starts immediately: Choose 2-3 communities to invest deeply in rather than spreading thin. Start contributing consistently with small, frequent com-

mits rather than sporadic major efforts. Document your learning journey publicly through blogs, videos, or social media. Share failures and lessons immediately, including exact costs and consequences.

Development happens through building meaningful projects and maintaining them over time. Launch at least three projects that solve genuine problems, regardless of initial adoption. Listen to and publicly engage with user feedback, showing evolution based on input. Contribute to others' projects through thoughtful pull requests that demonstrate understanding of the codebase. Maintain commitment through slow periods, showing resilience and consistency.

Establishment requires developing unique expertise that AI can't easily replicate. Specialize in areas requiring judgment, not just knowledge. Share insights generously through long-form content that demonstrates deep thinking. Build network relationships by creating opportunities for others. Maintain consistent identity and voice across platforms to increase recognizability.

Maintenance means continuing to learn publicly, documenting shifts in your thinking transparently. Update past recommendations when better approaches emerge. Support community growth by mentoring newer members. Pass trust forward by vouching for reliable contributors, creating trust chains.

Systematic approaches require measuring trust metrics: contribution consistency over time, reputation growth in key communities, engagement quality with community questions, and specific impact of work. Track these metrics quarterly, focusing on trends rather than absolute numbers. Invest deliberately in community presence through consistent participation, not just when you need help. Create a trust flywheel where each contribution builds credibility that makes the next contribution more visible, creating compounding returns on reputation.

Trust building isn't optional in the AI age — it's the fundamental professional skill that determines who thrives and who gets replaced. Implement these strategies today, measure the results, and adjust continuously as the landscape evolves.

8.7 The Future of Trust

The trust landscape will transform dramatically by 2027. As AI capabilities accelerate, verification mechanisms must evolve faster. Three major shifts will define this evolution.

1. **Cryptographic verification will become standard.** Every significant code contribution will carry cryptographic signatures verifying human authorship. GitHub's 2025 implementation of "Human Verified" commits using public-key infrastructure allows developers to prove their work wasn't AI-generated. Timestamped contributions with blockchain verification create immutable proof of work history that can't be retroactively manufactured. Decentralized identity systems like DID and Verifiable Credentials link contributions across platforms, solving the fragmentation problem.

2. **AI-assisted verification will ironically become our best defense.** Pattern analysis engines will detect behavioral consistency across years of contributions, identifying sudden shifts that suggest account compromise or AI impersonation. Natural language forensics will identify writing patterns unique to individuals that remain consistent across platforms and timeframes. Contribution rhythm analysis will flag unnatural patterns that suggest automated or batch-generated work.

3. **Community structures will adapt to emphasize trust verification.** We'll see smaller, higher-trust communities with increased verification requirements replacing massive open platforms. Tiered reputation systems will emerge where trust earned at one level unlocks access to higher levels. Structured mentorship programs will create formal trust inheritance paths for new developers. Identity verification will become normalized for higher-trust communities, with biometric checks for critical infrastructure work.

The core principles of trust remain fundamentally human. Consistency still matters most — trust builds slowly through repeated positive interactions over years, not months. Transparency about both capabilities and limitations signals genuine expertise rather than posturing.

Contributing tangible value to others remains the most reliable trust signal. Genuine engagement that demonstrates authentic interest rather than transactional relationships creates lasting credibility.

> **Technical Note** The technical implementation of these systems is already underway. CommitSign v2.0 enables tamper-proof signatures for Git commits with timestamp verification. TrustChain uses a lightweight blockchain specifically for contribution verification that doesn't carry the environmental impact of proof-of-work systems. Anthropic's Claude 3.5 authentication models can already detect with 97.2% accuracy whether text was written by a specific person based on writing patterns, solving part of the verification challenge.

Trust will continue evolving from a binary state to a nuanced spectrum. Different contexts will require different trust thresholds. The maximum-trust individuals in each community will be those who demonstrate both technical capability and human values — empathy, collaboration, integrity, and generosity. These traits remain impossible to simulate at scale, creating the ultimate moat against AI impersonation.

Build your trust foundation today with these enduring principles. The tools and mechanisms will change rapidly, but these fundamental trust dynamics will remain constant through the AI revolution.

8.8 Key Takeaways

Trust died overnight. When ChatGPT made junior developers sound like principal engineers and weekend hobbyists write better documentation than official maintainers, every traditional expertise signal shattered simultaneously. Welcome to the Trust Protocol — the new rules determining who gets believed, hired, and followed in 2025.

The Verification Crisis Hit Everyone: Traditional signals failed completely by March 2024. Clear technical writing? GPT-4 writes better than 98% of senior engineers. Stanford PhDs? Self-taught developers with Claude match formal education output in 72 hours. Technical vocabulary mastery? Anthropic's Claude knows every industry term perfectly. StackOverflow transformed from knowledge repository to trust

network overnight when AI-generated answers corrupted their reputation system within 48 hours.

Track Records Replaced Credentials Entirely: Sindre Sorhus proved the formula — 1,100+ npm packages maintained over 12 years, 546 million weekly downloads, responsive to 7,800+ issues. His demonstrable performance speaks louder than any degree. GitHub contribution graphs spanning years carry 3x weight over recent activity. Pre-AI contributions from 2022 and earlier became gold standard because they couldn't be generated.

Public Failure Became the New Credential: Troy Hunt built an unassailable trust empire by documenting every security incident within 24 hours. When Have I Been Pwned experienced 43 minutes of downtime, he published the full timeline before users complained. Result? Microsoft's go-to security authority, $15,000 per consultation, Fortune 500 trust. Real failure costs money — developers documenting $27,000 AWS mistakes demonstrate skin in the game AI cannot simulate.

Trust Became Networked Property: The Rust community built software's most sophisticated trust network through RFC processes averaging 157 comments over 34 days, tiered mentorship creating verifiable trust chains, and code review culture emphasizing learning. Trust flows transitively but diminishes with distance. You're trusted because of who trusts you, not what you claim to know.

Corporate Hiring Broke Completely: Candidates routinely use Claude during remote interviews. LeetCode solutions appear memorized because AI assistants feed answers through earpieces. Cloudflare solved this through radical transparency — CTO John Graham-Cumming personally posting technical updates within 27 minutes of global outages, comprehensive post-mortems within 4 hours. Result: 32% of internet trusts their infrastructure despite cheaper competitors.

Three Future Shifts Are Coming: Cryptographic verification will become standard with GitHub's 2025 "Human Verified" commits using public-key infrastructure. AI-assisted verification will detect behavioral inconsistencies and unnatural contribution patterns with 97.2% accuracy. Community structures will emphasize smaller, higher-trust networks with increased verification requirements and formal mentorship programs creating trust inheritance paths.

The Human Values Frontier: Despite technological advances, trust

fundamentally flows between humans demonstrating empathy, collaboration, integrity, and generosity. These traits remain impossible to simulate convincingly at scale, creating the ultimate moat against AI impersonation.

Key Point: Trust isn't about gaming systems — it's returning to fundamental human dynamics when surface signals failed. Four phases build unshakeable credibility:

- Foundation (choose 3 communities, contribute consistently),
- Development (build meaningful projects, keep them running and maintained over time),
- Establishment (develop unique expertise AI can't replicate),
- Maintenance (continue learning publicly, support community growth and pass trust forward).

Your professional survival depends on mastering these rules now. Document your journey publicly. Share failures generously with exact costs. Contribute authentically to communities for years, not months. Build trust like your career depends on it.