Rust Foundation

Technology Report

2024 – 2025

Contents

Foreword

May 15, 2025 marked 10 years since the first stable release of Rust. Since then, the Rust Foundation has been reflecting on the growth of the Rust ecosystem and the increasing role Rust plays in powering the world's infrastructure. Since we issued our **last Tech Report** in August 2024, the Rust Foundation's Technology Team has been hard at work on a vast array of projects aimed at securing the Rust supply chain, strengthening the Project's underlying infrastructure, preparing Rust for environments where safety and reliability are paramount, and much more. Our team, in collaboration with the Rust Project and industry partners, delivered tangible outcomes that protect the ecosystem while supporting our global community of maintainers and users. I am grateful for the dedication and collaboration that made these advancements possible, and I look forward to building on this momentum as we continue to support the Rust language and its vibrant community over the next year and beyond.

While we're only partially through it, 2025 has been a pivotal year for the Rust Foundation. Like many other teams, the Rust Foundation has had to remain flexible and strategic in the face of massive shifts across the tech landscape: the rise of Artificial Intelligence, increasing costs and complexity of cloud compute, and global regulatory changes alone have posed novel challenges for our Technology Team as I know they have for our readers. And yet, I am filled with pride for how we have continued to deepen our support for the Rust Project, prioritizing security, scalable stewardship, resilience, industry networking opportunities, and community trust.

As you'll find in the pages of this report, the Rust Foundation has spent the last year advancing major initiatives in open source security, interoperability, operational improvements that reduce costs while increasing stability, and more. The updates you'll discover show that the Rust Foundation has been hard at work strengthening our role as a steward of the Rust ecosystem, facilitating industry collaboration, advancing efforts that align Rust with safety-critical standards, and ensuring that Rust remains a secure and reliable choice for developers worldwide.

As ever, the Rust Foundation is committed to making direct and impactful improvements to Rust through these technical efforts so that Rust may remain a language people and systems can rely on for decades to come.

If you have any questions about the work described in this publication, or if you would like to discuss ways you can support the critical work of our Technology Team, please do not hesitate to reach out to us at contact@rustfoundation.org. You can also learn more about supporting our work at **https://rustfoundation.org/get-involved**.

[illegible]

Dr. Rebecca Rumbul
Executive Director & CEO
Rust Foundation

Executive Summary

2025 was a year of strategic progress for the Rust Foundation, emphasizing security, infrastructure resilience, and the growth of Rust as a trusted, industry-ready language. We launched Trusted Publishing on crates.io, advancing supply chain security while simplifying workflows for maintainers. We made significant progress on crate signing infrastructure with The Update Framework (TUF), and applauded the successful transfer and publication of the FLS (formerly Ferrocene Language Specification) under the Rust Project. Our Safety-Critical Rust Consortium grew, delivering practical guidance and resources for teams building safety-critical applications with Rust. We improved operational efficiency by reducing CI costs by 75% without disrupting contributor workflows, and continued to support the Rust Project through infrastructure upgrades and security initiatives. These achievements alone demonstrate my team's commitment to ensuring Rust remains secure, reliable, and ready for the challenges and inevitable shifts over the coming years.

Recent Highlights

- Trusted Publishing fully launched.
- Advanced TUF crate signing with multi-path testing and community engagement.
- FLS integrated into the Rust Project.
- Safety-Critical Rust Consortium expanded and delivered guidelines and tooling alignment.
- Reduced CI costs by 75% while maintaining contributor workflow stability.
- Infrastructure upgraded with enhanced backups and monitoring.
- Advanced Rust-C++ interoperability strategy.
- Progressed Capslock security tooling exploration and build.rs safety research.
- Began vulnerability surfacing integration for crates.io.

Current Priorities

- Ecosystem-wide rollout and adoption support for Trusted Publishing.
- Finalize and deploy TUF-based crate signing infrastructure.
- Continue integration and publication of the FLS.
- Advance Capslock security tooling toward public pilot.
- Integrate vulnerability surfacing into crates.io workflows.
- Maintain CI cost efficiency while scaling infrastructure reliability.
- Deepen engagement with the Rust Project to align technical goals.
- Expand Safety-Critical Rust resources for regulated industry adoption.
- Strengthen Rust's supply chain security and ecosystem resilience for developers globally.

In the pages that follow, you'll find quarterly snapshots of the Rust Foundation Technology Team's progress over the past year and a high-level overview of why the work under each initiative is broadly relevant. These packages of work reflect the steady, collaborative efforts required to keep Rust robust, secure, and ready for the demands of modern software development.

[illegible]

Joel Marcey
Director of Technology
Rust Foundation

Rust Foundation Technology Team

The Rust Foundation Technology Team is dedicated to ensuring the sustainability, security, and reliability of the Rust ecosystem. Our developers and engineers work directly on critical Rust infrastructure, collaborate to advance supply chain security, work with maintainers and leaders in the Rust Project, and support industry partnerships that has helped Rust maintain its reputation as the most-admired programming language year after year.

Meet the Team

Joel Marcey
DIRECTOR OF TECHNOLOGY

Tobias Bieniek
SOFTWARE ENGINEER (CRATES.IO), CRATES.IO TEAM LEAD

Jan David Nose
INFRASTRUCTURE LEAD, INFRASTRUCTURE TEAM LEAD

Jon Bauman
RUST/C++ INTEROPERABILITY ENGINEER

Marco Ieni
INFRASTRUCTURE ENGINEER, INFRASTRUCTURE TEAM MEMBER

Walter Pearce
SECURITY ENGINEER, RUSTSEC ADVISOR

Adam Harvey
SOFTWARE ENGINEER (SECURITY), CRATES.IO TEAM MEMBER

Thank you to Adam, Jan David, Joel, Jon, Marco, Tobias, and Walter for their contributions to the safety, security, interoperability, performance, and good working order of the Rust programming language!

Collaborators

The Tech Team could not be successful without transparent and productive relationships with hard working members of the Rust Project. The Rust Foundation would like to thank all the maintainers who have been willing to work and communicate openly with us about our technical endeavors. Many of these Rust maintainers have been donating their time and skills to the Rust programming language before the Rust Foundation existed and we would be remiss not to mention this alongside the report on the Foundation's technical activities.

Special thanks to the members of these Rust Project Teams and Working Groups:

LEADERSHIP COUNCIL | INFRASTRUCTURE TEAM | CRATES.IO TEAM | SECURE CODE WORKING GROUP | SECURITY RESPONSE WORKING GROUP

Funding & Support

The Rust Foundation's technical work is made possible by the generous support of our **member organizations** and dedicated funding partners.

In 2025, the Foundation received continued investment from Alpha-Omega to advance critical supply chain security initiatives, including Trusted Publishing, crate signing, and vulnerability surfacing infrastructure. This funding has enabled the Foundation to grow its engineering capacity, sustain high-impact projects, and support the Rust Project's strategic goals. We extend our sincere gratitude to all sponsors and donors whose contributions ensure that Rust remains a secure, sustainable, and community-driven language for developers around the world.

Alpha-Omega

OpenSSF
OPEN SOURCE SECURITY FOUNDATION

aws

Infrastructure Donors

MULLVAD VPN

Google Cloud

fastly

aws

SENTRY

WIZ

GitHub

crates.io Updates

crates.io remains a reliable, community-driven package ecosystem that serves millions of downloads daily while ensuring the sustainability of Rust's growth. In short, **crates.io** is the backbone of the Rust ecosystem and its strength is essential.

During the period of this report, crates.io Engineer Tobias Bieniek, Software Developer Adam Harvey, and their collaborators on our Tech Team and within the Rust Project focused on strengthening crates.io to support both maintainers and end users, aligning infrastructure stability, security improvements, and new capabilities with the needs of the Rust ecosystem.

What We Delivered:

Q3 2024:

- Helped implement the **crate deletion RFC**.
- **Email notifications** for new crate versions.
- Migrated download archive data to static.crates.io.
- Moved to CrunchyBridge database.
- RSS feeds for crate publications.
- Updated install instructions for binary crates.

Q4 2024:

- Completion of the async/await migration of the codebase, enabling performance improvements through PostgreSQL query pipelining.
- Identification of dormant and spam crates through the development of specialized SQL scripts, allowing us to successfully recover over 500 reserved package names after careful review.
- **The Trusted Publishing RFC** for crates.io successfully passed the final comment period (FCP) and was accepted by the crates.io team.
- Development of a comprehensive API documentation automatically generated from endpoint code.

Q1 2025:

- Migration of the frontend test suite to a mock API based on "Mock Service Worker" to unblock fur-ther improvements in the future.
- Publication of a crates.io team update blog on the Rust Project blog, detailing crates.io team updates over the previous six months: https://blog.rust-lang.org/2025/02/05/crates-io-development-update/
- Rust's security response team received a report about a malicious crate that attempted to exfil-trate the host and user name of the system that was building the crate. The security response team dealt with the crate efficiently, and our team grabbed a copy of the crate files to add to our ever-growing corpus of malware.

Q2 2025:

- Formed a proposed database layout for the Trusted Publishing implementation, which was reviewed and accepted by the crates.io team.
- Necessary database migrations were first executed on the staging environment, and later brought to production.
- Tobias completed the main publishing pipeline in **rust-lang/crates.io#11294**, which adjusted the existing publish endpoint to accept short-lived API tokens received from the token exchange process
- Tobias implemented the user interface in **rust-lang/crates.io#11398**, adding a "Trusted Publishing" section to the crate settings page where users can configure and manage GitHub Actions workflows as trusted publishers. He also added comprehensive documentation in **rust-lang/crates.io#11414** with setup instructions and workflow examples.
- Security enhancements include GitHub Secret Scanning integration **(rust-lang/crates.io#11405)** for automatic detection of exposed tokens and an incident response system **(rust-lang/crates.io#11419)** that sends email notifications and enables token revocation when compromises are detected. The ongoing work can be tracked **here.**

More Recently:

Thanks to the support of Alpha-Omega, our team began work on and completed the Trusted Publishing implementation for **crates.io**.

Why This Work Matters

crates.io enables reliable dependency management, community sharing, and secure reuse of libraries, which accelerates development and supports collaboration in the Rust community. In 2024, crates.io delivered 50.2 billion downloads, and in just the first six months of 2025, there have already been 47 billion downloads—a testament to the essential role of crates.io and its rapid growth.

The more general enhancements made between the latter half of 2024 and now improve reliability, security, and user experience while reducing technical debt, ensuring the platform can continue to scale sustainably with the ecosystem's growth.

More specifically, Tobias' work on Trusted Publishing for crates.io is significant because **Trusted Publishing** is used to strengthen the security posture around the supply chain of publishing Rust crates by reducing the risk of credential leaks and streamlining release workflows. We are grateful to OpenSSF's Alpha-Omega Project for supporting our efforts in these areas through their grant program.

What's Next

- Support broad Trusted Publishing adoption across the ecosystem.
- Roll out enhanced crate metadata visibility to support user decision-making.
- Continue refining spam management and deletion workflows.
- Assess additional API improvements based on user and maintainer feedback.

Safety-Critical Rust Consortium

The Safety-Critical Rust Consortium made significant strides in 2025, further advancing Rust's suitability for use in regulated and safety-critical environments. Led by the Rust Foundation in partnership with a growing network of members, industry stakeholders, and the Rust Project, the consortium focuses on producing practical tools and guidance to help organizations confidently adopt Rust in sectors such as aerospace, automotive, and medical devices where ensuring human safety is essential.

What We Delivered:

Q3 2024:

- First in-person meeting was held in Montreal alongside RustConf 2024, attracting over 30 participants interested in Rust's safety-critical applications. The meeting established a formal mission, subcommittees for coding guidelines, tooling, and education, and excitement for the next in-person meeting.

Q4 2024:

- Continued progress with the committee's two primary subcommittees: coding guidelines and tooling.

Q1 2025:

- A second in-person consortium meeting was held in London alongside Rust Nation UK. The consortium made progress on plans for delivering assets, guidance, and content for Rust developers working on safety-critical applications. The consortium also announced to its members that Ferrous Systems committed to contributing the Ferrocene Language Specification (FLS) to the Rust Project.

Q2 2025:

- The third full Safety Critical Rust Consortium **meeting** was held in Utrecht in the Netherlands alongside Rust Week. The consortium reviewed progress on drafting its public charter and goals, established plans for developing safety-critical Rust guidelines and tooling, and committed to ongoing coordination with the Rust Project—publishing minutes and charter drafts in the coming weeks.

More Recently:

- Advanced work on Rust Coding Guidelines that align with safety standards, providing developers with actionable guidance.
- Expanded participation to new industry members, broadening the range of safety-critical perspectives.

Why This Work Matters

Rust's memory safety and concurrency guarantees make it a compelling choice for safety-critical industries that demand reliability and verifiability. By aligning Rust's capabilities with recognized safety standards, the consortium helps lower barriers to adoption in industries where safety certification is essential, supporting Rust's long-term growth into new domains.

What's Next

- Continue development and publication of Rust Coding Guidelines aligned with MISRA and safety standards.
- Support reference tooling for safety-critical development with Rust.
- Facilitate industry discussions to align Rust's ecosystem with certification requirements.
- Expand outreach to additional industries to support safe Rust adoption.
- To learn more about the Safety-Critical Rust Consortium and to submit an application to join, visit our GitHub repository: **https://github.com/rustfoundation/safety-critical-rust-consortium** and check out the consortium website: **https://arewesafetycriticalyet.org**.

Rust-C++ Interoperability Initiative

In 2025, the Rust Foundation advanced the Rust-C++ Interoperability Initiative, recognizing that seamless, safe interoperation with C++ is critical for many industries adopting Rust alongside existing systems. This initiative aims to reduce barriers to Rust adoption while preserving safety guarantees that differentiate Rust in systems programming contexts.

What We Delivered:

Q3 2024:

- The C++/Rust Interop strategy neared completion, with plans to join the INCITS C++ specification committee to enhance collaboration between languages.

Q4 2024:

- Jon Bauman oversaw the creation and release of the **C++/Rust Interop problem statement and strategy.**
- A discussion was initiated around the definition of unsafety and undefined behavior, safety critical coding guidelines and unsafe coding guidelines research and writing.
- The Foundation also joined **INCITS** to participate directly in the **ISO C++** standardization process and collaborated with Tyler Mandry on a proposed 2025H1 Project Goal for seamless C++/Rust interoperability.

Q1 2025:

- Jon represented the Rust Foundation at the ISO C++ WG21 meeting in Austria. These discussions laid the groundwork for smoother integration between C++ and Rust, in line with the Foundation's previously published Interop Strategy.
- Jon publicly posted his draft version of the C++/Rust Interop initiative problem statement and high level strategy to the Rust Project and received positive feedback.

Q2 2025:

- At Rust Week in the Netherlands, Jon organized and facilitated a meeting between members of the Rust Project and the WG21 (C++ Committee) interested in interop. This was an initial discussion to create meaningful relationships, with the intention of sparking further collaboration between the two parties in order to provide an interop solution helpful to both communities.
- At the WG21 meeting in Sofia Jon engaged in useful discussions related to C++/Rust interop, most importantly. C++'s implementation of "trivial relocation", pointer authentication and the ongoing work helping to improve C++'s safety and supporting **the proposal for a Rust-like language subsetting approach.**
- Jon spoke at the Open Source in Finance Forum to raise awareness about memory safety with this essential sector that is highly invested in C++. This was an opportunity to explain both the practical consequences (in terms of incidents like CrowdStrike 2024) of language-level safety and motivate the essential role of C++/Rust interop as part of a pragmatic mitigation strategy to audiences currently underrepresented in Foundation membership.

More Recently:

- Initial exploration of the potential usage of **BorrowSanitizer** (a dynamic analysis tool for detecting Rust-specific aliasing bugs in multi-language applications,) in the C++/Rust Interop Initiative.
- Participation in ISO C++ WG21 meetings to align interoperability strategies and share safety perspectives from the Rust community.
- Advanced internal planning for Rust-C++ interoperability tooling and reference architectures.
- Engaged with stakeholders in safety-critical and embedded industries to gather requirements for safe, practical interop scenarios.
- Developed initial documentation plans to support developers navigating Rust-C++ integration challenges.

What's Next

- Develop and publish reference documentation for Rust-C++ interoperability best practices.
- Continue engagement with ISO C++ working groups to align on memory safety and interoperability.
- Prototype tooling to simplify safe interoperability for developers.
- Expand outreach to industries dependent on mixed Rust-C++ environments to understand needs and refine approaches.

Why This Work Matters

In 2025, the Rust Foundation's Rust-C++ Interoperability Initiative made strategic and exploratory progress toward one of Rust's most critical adoption frontiers: seamless and safe integration with C++. Many industries, from aerospace to embedded systems, rely heavily on C++ and need interoperability to adopt Rust incrementally without sacrificing safety.

This year, the Foundation finalized its Interop Strategy in several critical ways, including formally joining the ISO C++ standardization process (INCITS and WG21), and facilitating sustained collaboration between Rust Project members and C++ language stakeholders. These efforts culminated in face-to-face engagement at major events where other language communities gather.

Internally, the Initiative conducted exploratory work around tooling and safety analysis mechanisms, including BorrowSanitizer, and engaged with safety-critical industry stakeholders to ensure real-world relevance. It also began planning documentation and reference materials to ease adoption.

Rust's future in large-scale systems depends on practical, principled interoperability with C++. By building shared understanding, participating in standards bodies, and shaping tools with safety at their core, the Foundation is enabling broader, safer adoption of Rust in the systems programming landscape.

Security Initiative

In 2025, the Rust Foundation's Security Initiative expanded significantly, reflecting its central role in safeguarding the Rust ecosystem. Supported by funding from Alpha-Omega, the initiative encompassed layered work across supply chain security, vulnerability surfacing, safe build practices, and ecosystem tooling.

What We Delivered:

Q3 2024

- Crate provenance tracking continued with repo verification now being run across the entirety of the crates.io corpus, and a mechanism to delete crates was approved and implemented.
- At RustConf 2024, Foundation Security Initiative members and some members of the Rust Project agreed in principle that implementing **The Update Framework** (TUF) for Rust was the best way to move forward with crate signing and mirroring. This agreement in principle amongst the affected parties was a significant milestone for Rust ecosystem security.
- Walter gave a talk called "Dude Where's My C?" to a filled room at the Rust Global event co-located with RustConf 2024 in Montreal. Using **Painter** data, this talk discussed the statistics and implications of externally-linked code across the crates ecosystem.
- An effort to examine how the Rust ecosystem uses build.rs in practice was initiated. This work sought to clarify whether a safer framework could be built that would allow build.rs scripts to be replaced by a unified framework, allowing build.rs scripts to become more standardized in practice and easier to flag for review in the same manner as many organizations currently review all unsafe blocks in their dependency graphs.

Q4 2024

- Several important security and funding milestones were reached. The Foundation received a generous $430,000 in funding from Alpha-Omega for 2025, marking our third consecutive year of support as a "Critical Open Source Software Project."
- Security Engineer Walter Pearce, working alongside Rust Project members, led the creation of a **2025H1 Rust Project goal** around crate signing, building on the **TUF RFC**. This initiative includes the development of preliminary infrastructure for cryptographic verification of the crates.io repository and experimental mirrors, incorporating a chain-of-trust implementation.
- Work began on MVP test cases for the unified build framework.
- Work progressed on creating and implementing official backup accounts for all of Rust and crates.io.

Q1 2025

- Security Engineer Walter Pearce implemented a proof-of-concept for The Update Framework (TUF) covering crates.io and past Rust releases, which will support the **2025H1 Project Goal** around mirroring and verifying downloads. This infrastructure work complements the newly accepted RFC 3724.
- Walter moved towards publicly landing critical code changes to **Painter**:
  - Customized LLVM-based analysis, including scaffolding for importing LLVM IR into the graph database.
  - Improved cross-crate analysis with import filters and de-duplication of functions.
  - Behavioral analysis improvements with queries for function-name patterns, call path tracing and reverse lookups.
  - Multi-path resolution for multi-version/feature crates. And, of course, improved documentation.

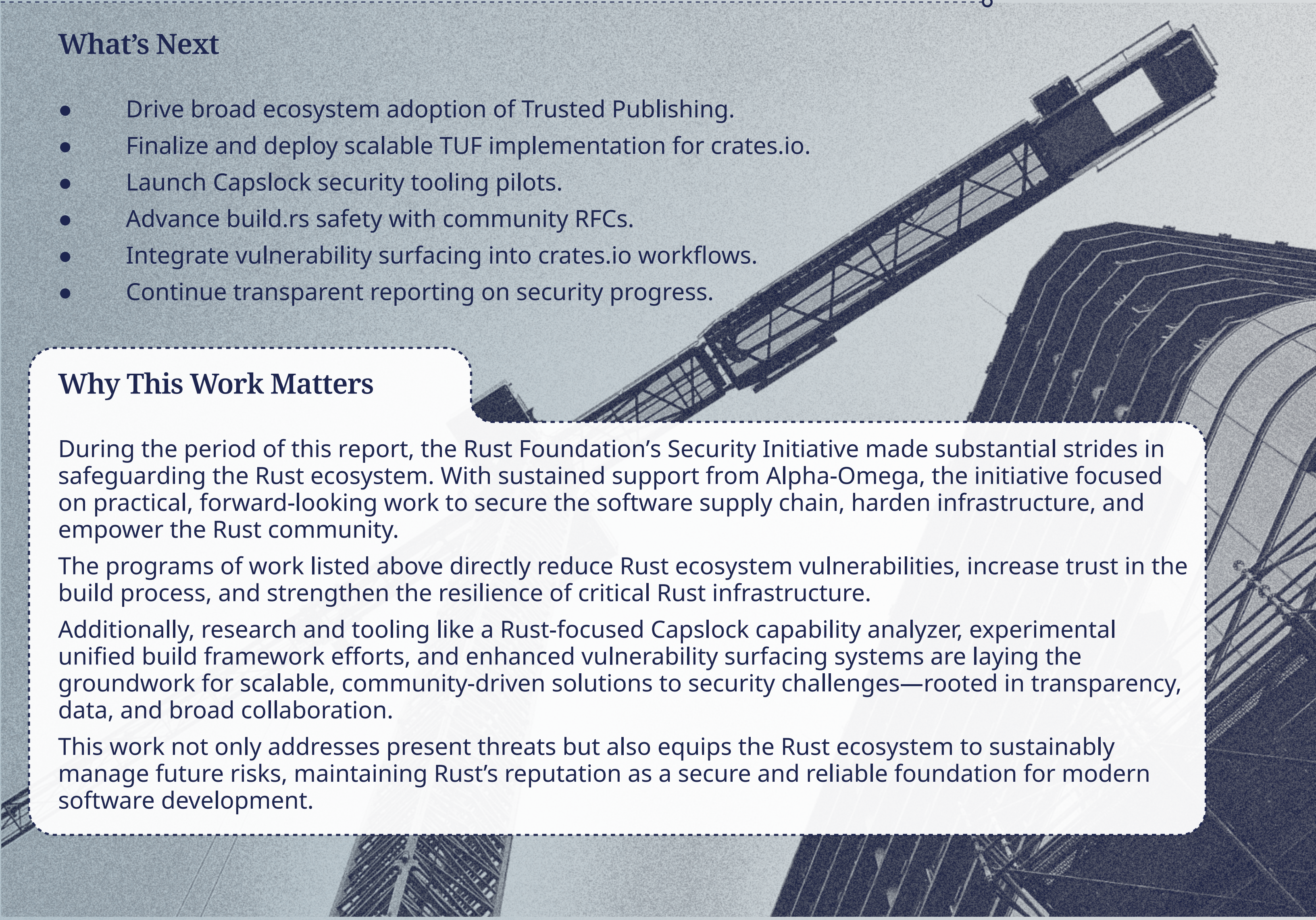
- Led by Walter, **the crates.io verification and mirroring goal** was approved for 2025H1. This effort will work towards consensus with Rust teams on an RFC for cryptographic verification and mirroring of releases and crates.io, and provide experimental infrastructure, demonstrating the ability to mirror crates.io and verify downloads from a mirror.
- Walter met with the Rust Project Infra team to discuss the infrastructure needs in order to meet this goal.
- Walter and Adam began exploring **Capslock** as it pertains to the Rust ecosystem. Currently, Capslock is a capability analysis CLI for Go packages that informs users of which privileged operations a given package can access. In order to make this capability for Rust, Walter began by understanding the level of granularity needed. Our team explored the landscape as it pertains to Rust and Capslock throughout this quarter.
- Rust's Security Response Working Group received a report about a malicious crate that attempted to exfiltrate the host and user name of the system that was building the crate. The security response team dealt with the crate efficiently, and we grabbed a copy of the crate files to add to our ever-growing corpus of malware.
- Walter implemented a "vanilla" (sans TAP-16/Merkle tree) implementation of **TUF** for the crates.io repository and a year of historic Rust releases to prove it as a proof-of-concept.
- The analysis engine and LLVM integrations of **Painter** were broken off into a separate library crate, allowing for it to be leveraged by further analysis tooling for the public discussed below.
- With the announced departure of a key maintainer of **cargo-audit**, Walter and Adam began investigating how they may be able to pick up the slack. They explored adding optional features for some security analysis across the Rust ecosystem, to enable users to do so locally.
- As part of a quarterly blog post series for Alpha-Omega, Adam **wrote** about the history of Typomania, a tool that detects potential typosquatting attacks on crates.io.

Q2 2025

- Thanks to the support of Alpha-Omega, we began the Trusted Publishing implementation for crates.io (see crates.io section on **page 7**).
- Alpha-Omega's support also enabled us to flesh out a Rust-focused implementation of Capslock. Walter and Adam have created an implementation plan for an experimental Cargo **subcommand** that analyses a Rust target and emits call graph data in the language-independent Capslock ingestion format.
- Walter and Adam actively participated in various Alpha-Omega working groups, including Capslock and securing software repos.
- Walter completed three full **TUF** repository implementations for crates and releases using a fork of rust-tuf, including a vanilla repo, a succinct hashed bin repo, and a TAP-16 Merkle tree repo.
- Adam had high level discussions at the Rust All Hands with interested project members around future build sandboxing work, as part of the ongoing research into how build.rs and proc macros are used in practice, and what mitigations might be possible to add in the short to medium term.

More Recently:

- Capslock research and planning continued, aiming to deliver Rust-specific capability analysis tooling for deeper ecosystem risk analysis.
- Trusted Publishing fully launched on crates.io, enabling maintainers to publish securely with GitHub Actions integration, secret scanning, and automated incident response, significantly reducing supply chain risk.
- TUF-based crate signing advanced, with testing across multiple repository implementations (vanilla TUF, hashed bins, Merkle tree) and stakeholder engagement to align on a scalable, community-driven solution.
- Vulnerability surfacing systems progressed, laying groundwork to help maintainers identify and address risks efficiently.
- Thanks to the support of Alpha-Omega, we began working on surfacing crate vulnerabilities in crates.io. This work already has Rust community interest.
- The Security Initiative also prioritized community transparency, regularly publishing updates, architecture proposals, and technical analysis in collaboration with the Rust Project and industry partners to ensure this work remains practical, open, and aligned with the needs of Rust maintainers and users.



What's Next

- Drive broad ecosystem adoption of Trusted Publishing.
- Finalize and deploy scalable TUF implementation for crates.io.
- Launch Capslock security tooling pilots.
- Advance build.rs safety with community RFCs.
- Integrate vulnerability surfacing into crates.io workflows.
- Continue transparent reporting on security progress.

Why This Work Matters

During the period of this report, the Rust Foundation's Security Initiative made substantial strides in safeguarding the Rust ecosystem. With sustained support from Alpha-Omega, the initiative focused on practical, forward-looking work to secure the software supply chain, harden infrastructure, and empower the Rust community.

The programs of work listed above directly reduce Rust ecosystem vulnerabilities, increase trust in the build process, and strengthen the resilience of critical Rust infrastructure.

Additionally, research and tooling like a Rust-focused Capslock capability analyzer, experimental unified build framework efforts, and enhanced vulnerability surfacing systems are laying the groundwork for scalable, community-driven solutions to security challenges—rooted in transparency, data, and broad collaboration.

This work not only addresses present threats but also equips the Rust ecosystem to sustainably manage future risks, maintaining Rust's reputation as a secure and reliable foundation for modern software development.

Infrastructure Support

In 2025, the Rust Foundation provided essential infrastructure support to ensure the reliability, scalability, and sustainability of the Rust ecosystem's critical services. The team focused on maintaining and improving the stability of CI pipelines, reducing operational costs, and modernizing server environments.

What We Accomplished:

Q3 2024

- Infrastructure updates accelerated under Marco Ieni's leadership, including **updating Terraform providers**, formulating a plan for crate and release backups, and completing server cataloging.
- The Foundation renewed its Datadog partnership for infrastructure monitoring.
- **The Rust Infrastructure Threat Model** identified the lack of out-of-band backups for critical data assets of the Rust Project as a major threat. Infrastructure Engineer Marco Ieni began some preliminary testing on mirroring all Rust releases and crates into a separate infrastructure on Google Cloud Platform (GCP).
- As part of its **ongoing effort** to catalog and patch servers for security and efficiency, Marco, JD and others on the Rust Project infrastructure team updated all of the servers it manages to Ubuntu 24.04.

Q4 2024

- Marco led efforts to reduce CI costs by approximately 50% by replacing large GitHub runners with free GitHub runners. These improvements came through technical optimizations, including build reordering and CI runner migration, all while maintaining minimal impact on Rust maintainers and community workflows.
- Released an experimental publicly-accessible CI dashboard, giving contributors visibility into pipeline duration, job failures, and CI health.
- Improved discoverability of Rust's documentation by removing tech debt in our CDNs.

Q1 2025

- Marco presented a talk at Rust Nation UK about release automation.
- Added a CI check to validate sync-team interactions with external services like GitHub and Zulip, increasing deployment confidence.

Q2 2025

- Marco, along with other members of the Rust Project infrastructure team, **added** the **rust-lang/rust repository** to the **team repository**, so that its settings are managed with Infrastructure as Code (IaC) rather than manually. This improved both the security and the maintainability of the repository. The rust-lang/rust repository was the last one missing: now all rust-lang repositories are managed with our IaC system.
- Marco continued to lead efforts to reduce CI costs, increasing the cost reduction to 75%, by continuing to replace large GitHub runners with free GitHub runners.

More Recently:

- **Multi-year Enterprise Agreement with GitHub** signed
- Marco has continued to examine whether we can eke out even more CI cost reduction, with a few promising results.
- Renewed Datadog usage contract for 2025-2026.
- Experimenting with IBM-provided GitHub CI runners for some libraries in the Rust Project.
- Continuing to set up the infrastructure to ensure robust backup systems for the Rust Project's assets.

What's Next

- Continue optimizing CI workflows for reliability and cost efficiency.
- Complete the process of backing up critical data assets.
- Review the Cloud Compute program to ensure long-term viability.
- Expand public observability tools to support maintainers.
- Regularly review and upgrade infrastructure to maintain security and performance.
- Explore sustainable infrastructure scaling as Rust's usage continues to grow.

Why This Work Matters

The work of our Infrastructure Engineers and their collaborators over the last several quarters not only improved the resilience and usability of Rust's technical systems, but also contributed to meaningful cost savings and operational efficiency.

By helping with the migration from legacy infrastructure to native GitHub features such as merge queues and optional CI jobs, the team helped reduce engineering overhead and lowered the maintenance burden of custom tooling. The move to Infrastructure-as-Code (IaC) for all Rust repositories consolidated config management and decreased the risk and cost of manual errors. Upgrading servers to Ubuntu 24.04 LTS ensures long-term support and compatibility with modern tooling, while also enabling more efficient resource use and reduced administrative overhead. Visibility improvements, including the public CI dashboard, allow for faster diagnosis of infrastructure issues, decreasing downtime and improving developer productivity across the project. These efforts collectively reduce the time, compute, and human effort required to maintain Rust's infrastructure, freeing up resources for innovation and helping ensure the long-term sustainability of the Rust Project and ecosystem.

Thank you again to our many collaborators and to our infrastructure donors (page 6) for their generous support.

Rust Language Specification

In 2025, the Rust Foundation supported the successful transfer and integration of the **Ferrocene Language Specification** (FLS) into the Rust Project, marking a major milestone toward the development of a formal Rust language specification. We are grateful to Ferrous Systems, the creators of Ferrocene, for initiating this process, ensuring the specification would remain openly accessible to the community while supporting the long-term evolution and stability of Rust.

Key Achievements Included:

- Transfer and publication of the FLS (formerly Ferrocene Language Specification) within the Rust Project.
- Ferrous Systems' support with donation processes, ensuring open community access.
- Collaboration with Project stakeholders to align the FLS with existing compiler and language pro-cesses.
- Development of documentation and processes to support ongoing specification maintenance.
- Community education efforts to explain the role and value of a formal language specification.

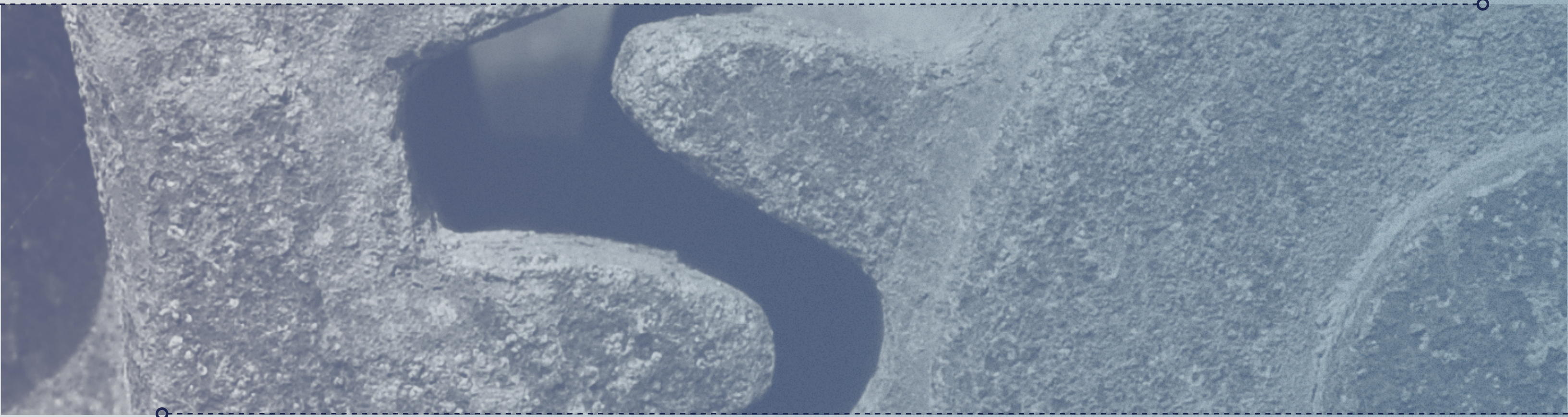
This work advances the Rust ecosystem's maturity, creating a reliable reference point for tooling developers, safety-critical adopters, and the broader community.

Why This Work Matters

A formal language specification supports safety-critical adoption of Rust by providing a clear, auditable description of the language's behavior, aiding certification efforts and improving the long-term reliability and trustworthiness of the language for mission-critical systems.

What's Next

- Continue integration and refinement of the FLS within the Rust Project.
- Develop processes for community feedback and iterative improvement.
- Support tooling and ecosystem projects that depend on a stable specification.
- Expand education and outreach around the specification's role in Rust's future.



Supporting Our Work

Building Rust's Future Together

As we close the Technology Report for the Q3 2024-Q3 2025 period, we invite you to reflect on the essential role Rust plays in powering the world's infrastructure, from safety-critical systems to cloud services and so many areas in between. The work outlined in these pages demonstrates the direct impact of a secure, sustainable, and reliable Rust ecosystem, made possible through the collective efforts of the Rust Foundation, the Rust Project, and our dedicated supporters.

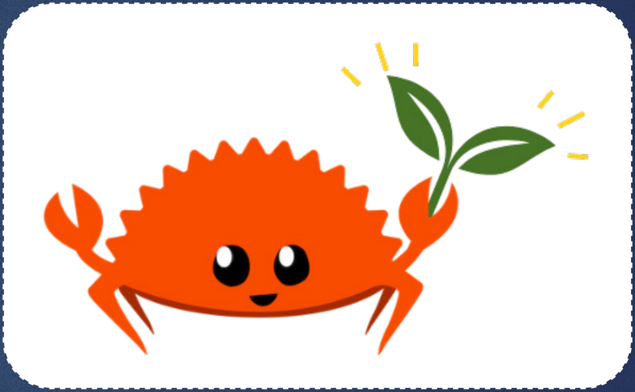
Rust's growth and resilience are not accidental; they are made possible through intentional investment by the Rust Foundation's generous member organizations. By joining the Rust Foundation, members become part of a global community committed to supporting the language's infrastructure, improving its security, and ensuring its continued accessibility to all developers.

Membership directly funds initiatives that:

- Strengthen supply chain security and reliability.
- Maintain and improve critical infrastructure like crates.io and CI systems.
- Advance the language through specification and safety-critical readiness.
- Support maintainers and reduce the burden on volunteer contributors.

If your organization depends on Rust, or is looking to expand its use, membership in the Rust Foundation is the most direct way to invest in the future of the ecosystem while aligning your brand with the values of safety, sustainability, and technical excellence.

To learn more about becoming a member, please visit rustfoundation.org/get-involved or email contact@rustfoundation.org. Together, we can continue to build Rust's future, ensuring that it remains a secure, sustainable, and community-driven language for decades to come.



Acknowledgements

The Rust Foundation extends its deepest gratitude to the Rust Foundation Technology Team for their expertise, reliability, and dedication. Over the past year, they have helped strengthened the Rust ecosystem for all who rely on it.

We are also grateful for the support of **Alpha-Omega**, whose funding has enabled critical security, infrastructure, and ecosystem initiatives this year.

Special thanks to Rust Foundation Platinum Member Amazon Web Services (**AWS**) for launching the **Rust Standard Library Verification Contest** in November 2024. This initiative spurred the creation of new formal verification tools and techniques for Rust, strengthening confidence in the language's core libraries in 2025. In doing so, AWS demonstrated a deep and forward-looking commitment to advancing memory safety, reliability, and trust in the Rust ecosystem. The Rust Foundation was proud to partner with them on this effort and we encourage our community to explore the **associated repository**.

Once again, we extend our heartfelt thanks to our collaborators across the Rust Project and the global Rust community. Your ongoing dedication ensures Rust continues to meet today's needs while evolving toward a secure and sustainable tomorrow.

Finally, we thank all our **members** for their visionary support of the Rust Foundation, their generous commitment to the Rust programming language, and their continued belief in the maintainers whose efforts drive its success.

Rust Foundation

RUSTFOUNDATION.ORG