Search Results

There are **423** CVE Records that match your search.

Name	Description
CVE-2023-5217	Heap buffer overflow in vp8 encoding in libvpx in Google Chrome prior to 117.0.5938.132 and libvpx 1.13.1 allowed a remote attacker to potentially exploit heap corruption via a crafted HTML page. (Chromium security severity: High)
CVE-2023-46135	rs-stellar-strkey is a Rust lib for encode/decode of Stellar Strkeys. A panic vulnerability occurs when a specially crafted payload is used.`inner_payload_len` should not above 64. This vulnerability has been patched in version 0.0.8.
CVE-2023-45812	The Apollo Router is a configurable, high-performance graph router written in Rust to run a federated supergraph that uses Apollo Federation. Affected versions are subject to a Denial-of-Service (DoS) type vulnerability which causes the Router to panic and terminate when a multi-part response is sent. When users send queries to the router that uses the `@defer` or Subscriptions, the Router will panic. To be vulnerable, users of Router must have a coprocessor with `coprocessor.supergraph.response` configured in their `router.yaml` and also to support either `@defer` or Subscriptions. Apollo Router version 1.33.0 has a fix for this vulnerability which was introduced in PR #4014. Users are advised to upgrade. Users unable to upgrade should avoid using the coprocessor supergraph response or disable defer and subscriptions support and continue to use the coprocessor supergraph response.
CVE-2023-43669	The Tungstenite crate before 0.20.1 for Rust allows remote attackers to cause a denial of service (minutes of CPU consumption) via an excessive length of an HTTP header in a client handshake. The length affects both how many times a parse is attempted (e.g., thousands of times) and the average amount of data for each parse attempt (e.g., millions of bytes).
CVE-2023-42811	aes-gcm is a pure Rust implementation of the AES-GCM. Starting in version 0.10.0 and prior to version 0.10.3, in the AES GCM implementation of decrypt_in_place_detached, the decrypted ciphertext (i.e. the correct plaintext) is exposed even if tag verification fails. If a program using the `aes-gcm` crate's `decrypt_in_place*` APIs accesses the buffer after decryption failure, it will contain a decryption of an unauthenticated input. Depending on the specific nature of the program this may enable Chosen Ciphertext Attacks (CCAs) which can cause a catastrophic breakage of the cipher including full plaintext recovery. Version 0.10.3 contains a fix for this issue.
CVE-2023-42447	blurhash-rs is a pure Rust implementation of Blurhash, software for encoding images into ASCII strings that can be turned into a gradient of colors representing the original image. In version 0.1.1, the blurhash parsing code may panic due to multiple panic-guarded out-of-bounds accesses on untrusted input. In a typical deployment, this may get triggered by feeding a maliciously crafted blurhashes over the network. These may include UTF-8 compliant strings containing multi-byte UTF-8 characters. A patch is available in version 0.2.0, which requires user intervention because of slight API churn. No known workarounds are available.
CVE-2023-42444	phonenumber is a library for parsing, formatting and validating international phone numbers. Prior to versions `0.3.3+8.13.9` and `0.2.5+8.11.3`, the phonenumber parsing code may panic due to a panic-guarded out-of-bounds access on the phonenumber string. In a typical deployment of `rust-phonenumber`, this may get triggered by feeding a maliciously crafted phonenumber over the network, specifically the string `.;phone-context=`. Versions `0.3.3+8.13.9` and `0.2.5+8.11.3` contain a patch for this issue. There are no known workarounds.
CVE-2023-41317	The Apollo Router is a configurable, high-performance graph router written in Rust to run a federated supergraph that uses Apollo Federation 2. Affected versions are subject to a Denial-of-Service (DoS) type vulnerability which causes the Router to panic and terminate when GraphQL Subscriptions are enabled. It can be triggered when **all of the following conditions are met**: 1. Running Apollo Router v1.28.0, v1.28.1 or v1.29.0 ("impacted versions"); **and** 2. The Supergraph schema provided to the Router (either via Apollo Uplink or explicitly via other configuration) **has a `subscription` type** with root-fields defined; **and** 3. The YAML configuration provided to the Router **has subscriptions enabled** (they are _disabled_ by default), either by setting `enabled: true` _or_ by setting a valid `mode` within the `subscriptions` object (as seen in [subscriptions' documentation](https://www.apollographql.com/docs/router/executing-operations/subscription-support/#router-setup)); **and** 4. An [anonymous](https://spec.graphql.org/draft/#sec-Anonymous-Operation-Definitions) (i.e., un-named) `subscription` operation (e.g., `subscription { ... }`) is received by the Router If **all four** of these criteria are met, the impacted versions will panic and terminate. There is no data-privacy risk or sensitive-information exposure aspect to this vulnerability. This is fixed in Apollo Router v1.29.1. Users are advised to upgrade. Updating to v1.29.1 should be a clear and simple upgrade path for those running impacted versions. However, if Subscriptions are **not** necessary for your Graph – but are enabled via configuration — then disabling subscriptions is another option to mitigate the risk.
CVE-2023-41051	In a typical Virtual Machine Monitor (VMM) there are several components, such as boot loader, virtual device drivers, virtio backend drivers and vhost drivers, that need to access the VM physical memory. The vm-memory rust crate provides a set of traits to decouple VM memory consumers from VM memory providers. An issue was discovered in the default implementations of the `VolatileMemory::{get_atomic_ref, aligned_as_ref, aligned_as_mut, get_ref, get_array_ref}` trait functions, which allows out-of-bounds memory access if the `VolatileMemory::get_slice` function returns a `VolatileSlice` whose length is less than the function’s `count` argument. No implementations of `get_slice` provided in `vm_memory` are affected. Users of custom `VolatileMemory` implementations may be impacted if the custom implementation does not adhere to `get_slice`'s documentation. The issue started in version 0.1.0 but was fixed in version 0.12.2 by inserting a check that verifies that the `VolatileSlice` returned by `get_slice` is of the correct length. Users are advised to upgrade. There are no known workarounds for this issue.
CVE-2023-40030	Cargo downloads a Rust project’s dependencies and compiles the project. Starting in Rust 1.60.0 and prior to 1.72, Cargo did not escape Cargo feature names when including them in the report generated by `cargo build --timings`. A malicious package included as a dependency may inject nearly arbitrary HTML here, potentially leading to cross-site scripting if the report is subsequently uploaded somewhere. The vulnerability affects users relying on dependencies from git, local paths, or alternative registries. Users who solely depend on crates.io are unaffected. Rust 1.60.0 introduced `cargo build --timings`, which produces a report of how long the different steps of the build process took. It includes lists of Cargo features for each crate. Prior to Rust 1.72, Cargo feature names were allowed to contain almost any characters (with some exceptions as used by the feature syntax), but it would produce a future incompatibility warning about them since Rust 1.49. crates.io is far more stringent about what it considers a valid feature name and has not allowed such feature names. As the feature names were included unescaped in the timings report, they could be used to inject Javascript into the page, for example with a feature name like `features = ["<img src='' onerror=alert(0)"]`. If this report were subsequently uploaded to a domain that uses credentials, the injected Javascript could access resources from the website visitor. This issue was fixed in Rust 1.72 by turning the future incompatibility warning into an error. Users should still exercise care in which package they download, by only including trusted dependencies in their projects. Please note that even with these vulnerabilities fixed, by design Cargo allows arbitrary code execution at build time thanks to build scripts and procedural macros: a malicious

Name	Description
	dependency will be able to cause damage regardless of these vulnerabilities. crates.io has server-side checks preventing this attack, and there are no packages on crates.io exploiting these vulnerabilities. crates.io users still need to excercise care in choosing their dependencies though, as remote code execution is allowed by design there as well.
CVE-2023-38497	Cargo downloads the Rust project’s dependencies and compiles the project. Cargo prior to version 0.72.2, bundled with Rust prior to version 1.71.1, did not respect the umask when extracting crate archives on UNIX-like systems. If the user downloaded a crate containing files writeable by any local user, another local user could exploit this to change the source code compiled and executed by the current user. To prevent existing cached extractions from being exploitable, the Cargo binary version 0.72.2 included in Rust 1.71.1 or later will purge caches generated by older Cargo versions automatically. As a workaround, configure one's system to prevent other local users from accessing the Cargo directory, usually located in `~/.cargo`.
CVE-2023-3766	A vulnerability was discovered in the odoh-rs rust crate that stems from faulty logic during the parsing of encrypted queries. This issue specifically occurs when processing encrypted query data received from remote clients and enables an attacker with knowledge of this vulnerability to craft and send specially designed encrypted queries to targeted ODOH servers running with odoh-rs. Upon successful exploitation, the server will crash abruptly, disrupting its normal operation and rendering the service temporarily unavailable.
CVE-2023-34449	ink! is an embedded domain specific language to write smart contracts in Rust for blockchains built on the Substrate framework. Starting in version 4.0.0 and prior to version 4.2.1, the return value when using delegate call mechanics, either through `CallBuilder::delegate` or `ink_env::invoke_contract_delegate`, is decoded incorrectly. This bug was related to the mechanics around decoding a call's return buffer, which was changed as part of pull request 1450. Since this feature was only released in ink! 4.0.0, no previous versions are affected. Users who have an ink! 4.x series contract should upgrade to 4.2.1 to receive a patch.
CVE-2023-34411	The xml-rs crate before 0.8.14 for Rust and Crab allows a denial of service (panic) via an invalid <! token (such as <!DOCTYPEs/%<!A nesting) in an XML document. The earliest affected version is 0.8.9.
CVE-2023-33290	The git-url-parse crate through 0.4.4 for Rust allows Regular Expression Denial of Service (ReDos) via a crafted URL to normalize_url in lib.rs, a similar issue to CVE-2023-32758 (Python).
CVE-2023-33289	The urlnorm crate through 0.1.4 for Rust allows Regular Expression Denial of Service (ReDos) via a crafted URL to lib.rs.
CVE-2023-33192	ntpd-rs is an NTP implementation written in Rust. ntpd-rs does not validate the length of NTS cookies in received NTP packets to the server. An attacker can crash the server by sending a specially crafted NTP packet containing a cookie shorter than what the server expects. The server also crashes when it is not configured to handle NTS packets. The issue was caused by improper slice indexing. The indexing operations were replaced by safer alternatives that do not crash the ntpd-rs server process but instead properly handle the error condition. A patch was released in version 0.3.3.
CVE-2023-30624	Wasmtime is a standalone runtime for WebAssembly. Prior to versions 6.0.2, 7.0.1, and 8.0.1, Wasmtime's implementation of managing per-instance state, such as tables and memories, contains LLVM-level undefined behavior. This undefined behavior was found to cause runtime-level issues when compiled with LLVM 16 which causes some writes, which are critical for correctness, to be optimized away. Vulnerable versions of Wasmtime compiled with Rust 1.70, which is currently in beta, or later are known to have incorrectly compiled functions. Versions of Wasmtime compiled with the current Rust stable release, 1.69, and prior are not known at this time to have any issues, but can theoretically exhibit potential issues. The underlying problem is that Wasmtime's runtime state for an instance involves a Rust-defined structure called `Instance` which has a trailing `VMContext` structure after it. This `VMContext` structure has a runtime-defined layout that is unique per-module. This representation cannot be expressed with safe code in Rust so `unsafe` code is required to maintain this state. The code doing this, however, has methods which take `&self` as an argument but modify data in the `VMContext` part of the allocation. This means that pointers derived from `&self` are mutated. This is typically not allowed, except in the presence of `UnsafeCell`, in Rust. When compiled to LLVM these functions have `noalias readonly` parameters which means it's UB to write through the pointers. Wasmtime's internal representation and management of `VMContext` has been updated to use `&mut self` methods where appropriate. Additionally verification tools for `unsafe` code in Rust, such as `cargo miri`, are planned to be executed on the `main` branch soon to fix any Rust-level issues that may be exploited in future compiler versions. Precomplied binaries available for Wasmtime from GitHub releases have been compiled with at most LLVM 15 so are not known to be vulnerable. As mentioned above, however, it's still recommended to update. Wasmtime version 6.0.2, 7.0.1, and 8.0.1 have been issued which contain the patch necessary to work correctly on LLVM 16 and have no known UB on LLVM 15 and earlier. If Wasmtime is compiled with Rust 1.69 and prior, which use LLVM 15, then there are no known issues. There is a theoretical possibility for undefined behavior to exploited, however, so it's recommended that users upgrade to a patched version of Wasmtime. Users using beta Rust (1.70 at this time) or nightly Rust (1.71 at this time) must update to a patched version to work correctly.
CVE-2023-30610	aws-sigv4 is a rust library for low level request signing in the aws cloud platform. The `aws_sigv4::SigningParams` struct had a derived `Debug` implementation. When debug-formatted, it would include a user's AWS access key, AWS secret key, and security token in plaintext. When TRACE-level logging is enabled for an SDK, `SigningParams` is printed, thereby revealing those credentials to anyone with access to logs. All users of the AWS SDK for Rust who enabled TRACE-level logging, either globally (e.g. `RUST_LOG=trace`), or for the `aws-sigv4` crate specifically are affected. This issue has been addressed in a set of new releases. Users are advised to upgrade. Users unable to upgrade should disable TRACE-level logging for AWS Rust SDK crates.
CVE-2023-28631	comrak is a CommonMark + GFM compatible Markdown parser and renderer written in rust. A Comrak AST can be constructed manually by a program instead of parsing a Markdown document with `parse_document`. This AST can then be converted to HTML via `html::format_document_with_plugins`. However, the HTML formatting code assumes that the AST is well-formed. For example, many AST notes contain `[u8]` fields which the formatting code assumes is valid UTF-8 data. Several bugs can be triggered if this is not the case. Version 0.17.0 contains adjustments to the AST, storing strings instead of unvalidated byte arrays. Users are advised to upgrade. Users unable to upgrade may manually validate UTF-8 correctness of all data when assigning to `&[u8]` and `Vec<u8>` fields in the AST. This issue is also tracked as `GHSL-2023-049`.
CVE-2023-28626	comrak is a CommonMark + GFM compatible Markdown parser and renderer written in rust. A range of quadratic parsing issues are present in Comrak. These can be used to craft denial-of-service attacks on services that use Comrak to parse Markdown. This issue has been addressed in version 0.17.0. Users are advised to upgrade. There are no known workarounds for this vulnerability. This issue is also tracked as `GHSL-2023-047`
CVE-2023-28448	Versionize is a framework for version tolerant serializion/deserialization of Rust data structures, designed for usecases that need fast deserialization times and minimal size overhead. An issue was discovered in the ‘Versionize::deserialize’ implementation provided by the ‘versionize’ crate for ‘vmm_sys_utils::fam::FamStructWrapper', which can lead to out of bounds memory accesses. The impact started with version 0.1.1. The issue was corrected in version 0.1.10 by inserting a check that verifies, for any deserialized header, the lengths of compared flexible arrays are equal and aborting deserialization otherwise.
CVE-2023-28446	Deno is a simple, modern and secure runtime for JavaScript and TypeScript that uses V8 and is built in Rust. Arbitrary program names without any ANSI filtering allows any malicious program to clear the first 2 lines of a `op_spawn_child` or `op_kill` prompt and replace it with any desired text. This works with any command on the respective platform, giving the program the full ability to choose what program they wanted to run. This problem can not be exploited on systems that do not attach an interactive prompt (for example headless servers). This issue has been patched in version 1.31.2.

Name	Description
CVE-2023-28445	Deno is a runtime for JavaScript and TypeScript that uses V8 and is built in Rust. Resizable ArrayBuffers passed to asynchronous functions that are shrunk during the asynchronous operation could result in an out-of-bound read/write. It is unlikely that this has been exploited in the wild, as the only version affected is Deno 1.32.0. Deno Deploy users are not affected. The problem has been resolved by disabling resizable ArrayBuffers temporarily in Deno 1.32.1. Deno 1.32.2 will re-enable resizable ArrayBuffers with a proper fix. As a workaround, run with `--v8-flags=--no-harmony-rab-gsab` to disable resizable ArrayBuffers.
CVE-2023-28113	russh is a Rust SSH client and server library. Starting in version 0.34.0 and prior to versions 0.36.2 and 0.37.1, Diffie-Hellman key validation is insufficient, which can lead to insecure shared secrets and therefore breaks confidentiality. Connections between a russh client and server or those of a russh peer with some other misbehaving peer are most likely to be problematic. These may vulnerable to eavesdropping. Most other implementations reject such keys, so this is mainly an interoperability issue in such a case. This issue is fixed in versions 0.36.2 and 0.37.1
CVE-2023-22895	The bzip2 crate before 0.4.4 for Rust allow attackers to cause a denial of service via a large file that triggers an integer overflow in mem.rs. NOTE: this is unrelated to the https://crates.io/crates/bzip2-rs product.
CVE-2023-22499	Deno is a runtime for JavaScript and TypeScript that uses V8 and is built in Rust. Multi-threaded programs were able to spoof interactive permission prompt by rewriting the prompt to suggest that program is waiting on user confirmation to unrelated action. A malicious program could clear the terminal screen after permission prompt was shown and write a generic message. This situation impacts users who use Web Worker API and relied on interactive permission prompt. The reproduction is very timing sensitive and can’t be reliably reproduced on every try. This problem can not be exploited on systems that do not attach an interactive prompt (for example headless servers). The problem has been fixed in Deno v1.29.3; it is recommended all users update to this version. Users are advised to upgrade. Users unable to upgrade may run with --no-prompt flag to disable interactive permission prompts.
CVE-2023-22466	Tokio is a runtime for writing applications with Rust. Starting with version 1.7.0 and prior to versions 1.18.4, 1.20.3, and 1.23.1, when configuring a Windows named pipe server, setting `pipe_mode` will reset `reject_remote_clients` to `false`. If the application has previously configured `reject_remote_clients` to `true`, this effectively undoes the configuration. Remote clients may only access the named pipe if the named pipe's associated path is accessible via a publicly shared folder (SMB). Versions 1.23.1, 1.20.3, and 1.18.4 have been patched. The fix will also be present in all releases starting from version 1.24.0. Named pipes were introduced to Tokio in version 1.7.0, so releases older than 1.7.0 are not affected. As a workaround, ensure that `pipe_mode` is set first after initializing a `ServerOptions`.
CVE-2022-46176	Cargo is a Rust package manager. The Rust Security Response WG was notified that Cargo did not perform SSH host key verification when cloning indexes and dependencies via SSH. An attacker could exploit this to perform man-in-the-middle (MITM) attacks. This vulnerability has been assigned CVE-2022-46176. All Rust versions containing Cargo before 1.66.1 are vulnerable. Note that even if you don't explicitly use SSH for alternate registry indexes or crate dependencies, you might be affected by this vulnerability if you have configured git to replace HTTPS connections to GitHub with SSH (through git's [`url.<base>.insteadOf`][1] setting), as that'd cause you to clone the crates.io index through SSH. Rust 1.66.1 will ensure Cargo checks the SSH host key and abort the connection if the server's public key is not already trusted. We recommend everyone to upgrade as soon as possible.
CVE-2022-46149	Cap'n Proto is a data interchange format and remote procedure call (RPC) system. Cap'n Proro prior to versions 0.7.1, 0.8.1, 0.9.2, and 0.10.3, as well as versions of Cap'n Proto's Rust implementation prior to 0.13.7, 0.14.11, and 0.15.2 are vulnerable to out-of-bounds read due to logic error handling list-of-list. This issue may lead someone to remotely segfault a peer by sending it a malicious message, if the victim performs certain actions on a list-of-pointer type. Exfiltration of memory is possible if the victim performs additional certain actions on a list-of-pointer type. To be vulnerable, an application must perform a specific sequence of actions, described in the GitHub Security Advisory. The bug is present in inlined code, therefore the fix will require rebuilding dependent applications. Cap'n Proto has C++ fixes available in versions 0.7.1, 0.8.1, 0.9.2, and 0.10.3. The `capnp` Rust crate has fixes available in versions 0.13.7, 0.14.11, and 0.15.2.
CVE-2022-45299	An issue in the IpFile argument of rust-lang webbrowser-rs v0.8.2 allows attackers to access arbitrary files via supplying a crafted URL.
CVE-2022-39397	aliyun-oss-client is a rust client for Alibaba Cloud OSS. Users of this library will be affected, the incoming secret will be disclosed unintentionally. This issue has been patched in version 0.8.1.
CVE-2022-39354	SputnikVM, also called evm, is a Rust implementation of Ethereum Virtual Machine. A custom stateful precompile can use the `is_static` parameter to determine if the call is executed in a static context (via `STATICCALL`), and thus decide if stateful operations should be done. Prior to version 0.36.0, the passed `is_static` parameter was incorrect -- it was only set to `true` if the call came from a direct `STATICCALL` opcode. However, once a static call context is entered, it should stay static. The issue only impacts custom precompiles that actually uses `is_static`. For those affected, the issue can lead to possible incorrect state transitions. Version 0.36.0 contains a patch. There are no known workarounds.
CVE-2022-39294	conduit-hyper integrates a conduit application with the hyper server. Prior to version 0.4.2, `conduit-hyper` did not check any limit on a request's length before calling [`hyper::body::to_bytes`](https://docs.rs/hyper/latest/hyper/body/fn.to_bytes.html). An attacker could send a malicious request with an abnormally large `Content-Length`, which could lead to a panic if memory allocation failed for that request. In version 0.4.2, `conduit-hyper` sets an internal limit of 128 MiB per request, otherwise returning status 400 ("Bad Request"). This crate is part of the implementation of Rust's crates.io, but that service is not affected due to its existing cloud infrastructure, which already drops such malicious requests. Even with the new limit in place, `conduit-hyper` is not recommended for production use, nor to directly serve the public Internet.
CVE-2022-39252	matrix-rust-sdk is an implementation of a Matrix client-server library in Rust, and matrix-sdk-crypto is the Matrix encryption library. Prior to version 0.6, when a user requests a room key from their devices, the software correctly remembers the request. When the user receives a forwarded room key, the software accepts it without checking who the room key came from. This allows homeservers to try to insert room keys of questionable validity, potentially mounting an impersonation attack. Version 0.6 fixes this issue.
CVE-2022-36125	It is possible to crash (panic) an application by providing a corrupted data to be read. This issue affects Rust applications using Apache Avro Rust SDK prior to 0.14.0 (previously known as avro-rs). Users should update to apache-avro version 0.14.0 which addresses this issue.
CVE-2022-36124	It is possible for a Reader to consume memory beyond the allowed constraints and thus lead to out of memory on the system. This issue affects Rust applications using Apache Avro Rust SDK prior to 0.14.0 (previously known as avro-rs). Users should update to apache-avro version 0.14.0 which addresses this issue.
CVE-2022-36114	Cargo is a package manager for the rust programming language. It was discovered that Cargo did not limit the amount of data extracted from compressed archives. An attacker could upload to an alternate registry a specially crafted package that extracts way more data than its size (also known as a "zip bomb"), exhausting the disk space on the machine using Cargo to download the package. Note that by design Cargo allows code execution at build time, due to build scripts and procedural macros. The vulnerabilities in this advisory allow performing a subset of the possible damage in a harder to track down way. Your dependencies must still be trusted if you want to be protected from attacks, as it's possible to perform the same attacks with build scripts and procedural macros. The vulnerability is present in all versions of Cargo. Rust 1.64, to be released on September 22nd, will include a fix for it. Since the vulnerability is just a more limited way to accomplish what a malicious build scripts or procedural macros can do, we decided not to publish Rust point releases backporting the security fix. Patch files are available for Rust 1.63.0 are available in the wg-security-response repository for people building their own toolchain. We recommend users of alternate registries to excercise care in which

Name	Description
	package they download, by only including trusted dependencies in their projects. Please note that even with these vulnerabilities fixed, by design Cargo allows arbitrary code execution at build time thanks to build scripts and procedural macros: a malicious dependency will be able to cause damage regardless of these vulnerabilities. crates.io implemented server-side checks to reject these kinds of packages years ago, and there are no packages on crates.io exploiting these vulnerabilities. crates.io users still need to excercise care in choosing their dependencies though, as the same concerns about build scripts and procedural macros apply here.
CVE-2022-36113	Cargo is a package manager for the rust programming language. After a package is downloaded, Cargo extracts its source code in the ~/.cargo folder on disk, making it available to the Rust projects it builds. To record when an extraction is successful, Cargo writes "ok" to the .cargo-ok file at the root of the extracted source code once it extracted all the files. It was discovered that Cargo allowed packages to contain a .cargo-ok symbolic link, which Cargo would extract. Then, when Cargo attempted to write "ok" into .cargo-ok, it would actually replace the first two bytes of the file the symlink pointed to with ok. This would allow an attacker to corrupt one file on the machine using Cargo to extract the package. Note that by design Cargo allows code execution at build time, due to build scripts and procedural macros. The vulnerabilities in this advisory allow performing a subset of the possible damage in a harder to track down way. Your dependencies must still be trusted if you want to be protected from attacks, as it's possible to perform the same attacks with build scripts and procedural macros. The vulnerability is present in all versions of Cargo. Rust 1.64, to be released on September 22nd, will include a fix for it. Since the vulnerability is just a more limited way to accomplish what a malicious build scripts or procedural macros can do, we decided not to publish Rust point releases backporting the security fix. Patch files are available for Rust 1.63.0 are available in the wg-security-response repository for people building their own toolchain. Mitigations We recommend users of alternate registries to exercise care in which package they download, by only including trusted dependencies in their projects. Please note that even with these vulnerabilities fixed, by design Cargo allows arbitrary code execution at build time thanks to build scripts and procedural macros: a malicious dependency will be able to cause damage regardless of these vulnerabilities. crates.io implemented server-side checks to reject these kinds of packages years ago, and there are no packages on crates.io exploiting these vulnerabilities. crates.io users still need to exercise care in choosing their dependencies though, as remote code execution is allowed by design there as well.
CVE-2022-35922	Rust-WebSocket is a WebSocket (RFC6455) library written in Rust. In versions prior to 0.26.5 untrusted websocket connections can cause an out-of-memory (OOM) process abort in a client or a server. The root cause of the issue is during dataframe parsing. Affected versions would allocate a buffer based on the declared dataframe size, which may come from an untrusted source. When `Vec::with_capacity` fails to allocate, the default Rust allocator will abort the current process, killing all threads. This affects only sync (non-Tokio) implementation. Async version also does not limit memory, but does not use `with_capacity`, so DoS can happen only when bytes for oversized dataframe or message actually got delivered by the attacker. The crashes are fixed in version 0.26.5 by imposing default dataframe size limits. Affected users are advised to update to this version. Users unable to upgrade are advised to filter websocket traffic externally or to only accept trusted traffic.
CVE-2022-35724	It is possible to provide data to be read that leads the reader to loop in cycles endlessly, consuming CPU. This issue affects Rust applications using Apache Avro Rust SDK prior to 0.14.0 (previously known as avro-rs). Users should update to apache-avro version 0.14.0 which addresses this issue.
CVE-2022-31173	Juniper is a GraphQL server library for Rust. Affected versions of Juniper are vulnerable to uncontrolled recursion resulting in a program crash. This issue has been addressed in version 0.15.10. Users are advised to upgrade. Users unable to upgrade should limit the recursion depth manually.
CVE-2022-31162	Slack Morphism is an async client library for Rust. Prior to 0.41.0, it was possible for Slack OAuth client information to leak in application debug logs. Stricter and more secure debug formatting was introduced in v0.41.0 for OAuth secret types to reduce the possibility of printing sensitive information in application logs. As a workaround, do not print/output requests and responses for OAuth and client configurations in logs.
CVE-2022-31053	Biscuit is an authentication and authorization token for microservices architectures. The Biscuit specification version 1 contains a vulnerable algorithm that allows malicious actors to forge valid Γ-signatures. Such an attack would allow an attacker to create a token with any access level. The version 2 of the specification mandates a different algorithm than gamma signatures and as such is not affected by this vulnerability. The Biscuit implementations in Rust, Haskell, Go, Java and Javascript all have published versions following the v2 specification. There are no known workarounds for this issue.
CVE-2022-29185	totp-rs is a Rust library that permits the creation of 2FA authentification tokens per time-based one-time password (TOTP). Prior to version 1.1.0, token comparison was not constant time, and could theorically be used to guess value of an TOTP token, and thus reuse it in the same time window. The attacker would have to know the password beforehand nonetheless. Starting with patched version 1.1.0, the library uses constant-time comparison. There are currently no known workarounds.
CVE-2022-27943	libiberty/rust-demangle.c in GNU GCC 11.2 allows stack consumption in demangle_const, as demonstrated by nm-new.
CVE-2022-24713	regex is an implementation of regular expressions for the Rust language. The regex crate features built-in mitigations to prevent denial of service attacks caused by untrusted regexes, or untrusted input matched by trusted regexes. Those (tunable) mitigations already provide sane defaults to prevent attacks. This guarantee is documented and it's considered part of the crate's API. Unfortunately a bug was discovered in the mitigations designed to prevent untrusted regexes to take an arbitrary amount of time during parsing, and it's possible to craft regexes that bypass such mitigations. This makes it possible to perform denial of service attacks by sending specially crafted regexes to services accepting user-controlled, untrusted regexes. All versions of the regex crate before or equal to 1.5.4 are affected by this issue. The fix is include starting from regex 1.5.5. All users accepting user-controlled regexes are recommended to upgrade immediately to the latest version of the regex crate. Unfortunately there is no fixed set of problematic regexes, as there are practically infinite regexes that could be crafted to exploit this vulnerability. Because of this, it us not recommend to deny known problematic regexes.
CVE-2022-23639	crossbeam-utils provides atomics, synchronization primitives, scoped threads, and other utilities for concurrent programming in Rust. crossbeam-utils prior to version 0.8.7 incorrectly assumed that the alignment of `{i,u}64` was always the same as `Atomic{I,U}64`. However, the alignment of `{i,u}64` on a 32-bit target can be smaller than `Atomic{I,U}64`. This can cause unaligned memory accesses and data race. Crates using `fetch_*` methods with `AtomicCell<{i,u}64>` are affected by this issue. 32-bit targets without `Atomic{I,U}64` and 64-bit targets are not affected by this issue. This has been fixed in crossbeam-utils 0.8.7. There are currently no known workarounds.
CVE-2022-23486	libp2p-rust is the official rust language Implementation of the libp2p networking stack. In versions prior to 0.45.1 an attacker node can cause a victim node to allocate a large number of small memory chunks, which can ultimately lead to the victim’s process running out of memory and thus getting killed by its operating system. When executed continuously, this can lead to a denial of service attack, especially relevant on a larger scale when run against more than one node of a libp2p based network. Users are advised to upgrade to `libp2p` `v0.45.1` or above. Users unable to upgrade should reference the DoS Mitigation page for more information on how to incorporate mitigation strategies, monitor their application, and respond to attacks: https://docs.libp2p.io/reference/dos-mitigation/.
CVE-2022-21658	Rust is a multi-paradigm, general-purpose programming language designed for performance and safety, especially safe concurrency. The Rust Security Response WG was notified that the `std::fs::remove_dir_all` standard library function is vulnerable a race condition enabling symlink following (CWE-363). An attacker could use this security issue to trick a privileged program into deleting files and directories the attacker couldn't otherwise access or delete. Rust 1.0.0 through Rust 1.58.0 is affected by this vulnerability with 1.58.1 containing a patch. Note that the following build targets don't have usable APIs to properly mitigate the attack, and are thus still vulnerable even with a patched toolchain: macOS before version 10.10 (Yosemite) and REDOX. We recommend everyone to update to Rust 1.58.1 as soon as possible, especially people developing programs expected to run in privileged contexts (including

Name	Description
	system daemons and setuid binaries), as those have the highest risk of being affected by this. Note that adding checks in your codebase before calling remove_dir_all will not mitigate the vulnerability, as they would also be vulnerable to race conditions like remove_dir_all itself. The existing mitigation is working as intended outside of race conditions.
CVE-2021-46195	GCC v12.0 was discovered to contain an uncontrolled recursion via the component libiberty/rust-demangle.c. This vulnerability allows attackers to cause a Denial of Service (DoS) by consuming excessive CPU and memory resources.
CVE-2021-45720	An issue was discovered in the lru crate before 0.7.1 for Rust. The iterators have a use-after-free, as demonstrated by an access after a pop operation.
CVE-2021-45719	An issue was discovered in the rusqlite crate 0.25.x before 0.25.4 and 0.26.x before 0.26.2 for Rust. update_hook has a use-after-free.
CVE-2021-45718	An issue was discovered in the rusqlite crate 0.25.x before 0.25.4 and 0.26.x before 0.26.2 for Rust. rollback_hook has a use-after-free.
CVE-2021-45717	An issue was discovered in the rusqlite crate 0.25.x before 0.25.4 and 0.26.x before 0.26.2 for Rust. commit_hook has a use-after-free.
CVE-2021-45716	An issue was discovered in the rusqlite crate 0.25.x before 0.25.4 and 0.26.x before 0.26.2 for Rust. create_collation has a use-after-free.
CVE-2021-45715	An issue was discovered in the rusqlite crate 0.25.x before 0.25.4 and 0.26.x before 0.26.2 for Rust. create_window_function has a use-after-free.
CVE-2021-45714	An issue was discovered in the rusqlite crate 0.25.x before 0.25.4 and 0.26.x before 0.26.2 for Rust. create_aggregate_function has a use-after-free.
CVE-2021-45713	An issue was discovered in the rusqlite crate 0.25.x before 0.25.4 and 0.26.x before 0.26.2 for Rust. create_scalar_function has a use-after-free.
CVE-2021-45712	An issue was discovered in the rust-embed crate before 6.3.0 for Rust. A ../ directory traversal can sometimes occur in debug mode.
CVE-2021-45711	An issue was discovered in the simple_asn1 crate 0.6.0 before 0.6.1 for Rust. There is a panic if UTCTime data, supplied by a remote attacker, has a second character greater than 0x7f.
CVE-2021-45710	An issue was discovered in the tokio crate before 1.8.4, and 1.9.x through 1.13.x before 1.13.1, for Rust. In certain circumstances involving a closed oneshot channel, there is a data race and memory corruption.
CVE-2021-45709	An issue was discovered in the crypto2 crate through 2021-10-08 for Rust. During Chacha20 encryption and decryption, an unaligned read of a u32 may occur.
CVE-2021-45708	An issue was discovered in the abomonation crate through 2021-10-17 for Rust. Because transmute operations are insufficiently constrained, there can be an information leak or ASLR bypass.
CVE-2021-45707	An issue was discovered in the nix crate 0.16.0 and later before 0.20.2, 0.21.x before 0.21.2, and 0.22.x before 0.22.2 for Rust. unistd::getgrouplist has an out-of-bounds write if a user is in more than 16 /etc/groups groups.
CVE-2021-45706	An issue was discovered in the zeroize_derive crate before 1.1.1 for Rust. Dropped memory is not zeroed out for an enum.
CVE-2021-45705	An issue was discovered in the nanorand crate before 0.6.1 for Rust. There can be multiple mutable references to the same object because the TlsWyRand Deref implementation dereferences a raw pointer.
CVE-2021-45704	An issue was discovered in the metrics-util crate before 0.7.0 for Rust. There is a data race and memory corruption because AtomicBucket<T> unconditionally implements the Send and Sync traits.
CVE-2021-45703	An issue was discovered in the tectonic_xdv crate before 0.1.12 for Rust. XdvParser::<T>::process may read from uninitialized memory locations.
CVE-2021-45702	An issue was discovered in the tremor-script crate before 0.11.6 for Rust. A merge operation may result in a use-after-free.
CVE-2021-45701	An issue was discovered in the tremor-script crate before 0.11.6 for Rust. A patch operation may result in a use-after-free.
CVE-2021-45700	An issue was discovered in the ckb crate before 0.40.0 for Rust. Attackers can cause a denial of service (Nervos CKB blockchain node crash) via a dead call that is used as a DepGroup.
CVE-2021-45699	An issue was discovered in the ckb crate before 0.40.0 for Rust. Remote attackers may be able to conduct a 51% attack against the Nervos CKB blockchain by triggering an inability to allocate memory for the misbehavior HashMap.
CVE-2021-45698	An issue was discovered in the ckb crate before 0.40.0 for Rust. A get_block_template RPC call may fail in situations where it is supposed to select a Nervos CKB blockchain transaction with a higher fee rate than another transaction.
CVE-2021-45697	An issue was discovered in the molecule crate before 0.7.2 for Rust. A FixVec partial read has an incorrect result.
CVE-2021-45696	An issue was discovered in the sha2 crate 0.9.7 before 0.9.8 for Rust. Hashes of long messages may be incorrect when the AVX2-accelerated backend is used.
CVE-2021-45695	An issue was discovered in the mopa crate through 2021-06-01 for Rust. It incorrectly relies on Trait memory layout, possibly leading to future occurrences of arbitrary code execution or ASLR bypass.
CVE-2021-45694	An issue was discovered in the rdiff crate through 2021-02-03 for Rust. Window may read from uninitialized memory locations.
CVE-2021-45693	An issue was discovered in the messagepack-rs crate through 2021-01-26 for Rust. deserialize_string_primitive may read from uninitialized memory locations.
CVE-2021-45692	An issue was discovered in the messagepack-rs crate through 2021-01-26 for Rust. deserialize_extension_others may read from uninitialized memory locations.
CVE-2021-45691	An issue was discovered in the messagepack-rs crate through 2021-01-26 for Rust. deserialize_string may read from uninitialized memory locations.
CVE-2021-45690	An issue was discovered in the messagepack-rs crate through 2021-01-26 for Rust. deserialize_binary may read from uninitialized memory locations.
CVE-2021-45689	An issue was discovered in the gfx-auxil crate through 2021-01-07 for Rust. gfx_auxil::read_spirv may read from uninitialized memory locations.
CVE-2021-45688	An issue was discovered in the ash crate before 0.33.1 for Rust. util::read_spv may read from uninitialized memory locations.
CVE-2021-45687	An issue was discovered in the raw-cpuid crate before 9.1.1 for Rust. If the serialize feature is used (which is not the the default), a Deserialize operation may lack sufficient validation, leading to memory corruption or a panic.
CVE-2021-45686	An issue was discovered in the csv-sniffer crate through 2021-01-05 for Rust. preamble_skipcount may read from uninitialized memory locations.
CVE-2021-45685	An issue was discovered in the columnar crate through 2021-01-07 for Rust. ColumnarReadExt::read_typed_vec may read from uninitialized memory locations.

Name	Description
CVE-2021-45684	An issue was discovered in the flumedb crate through 2021-01-07 for Rust. read_entry may read from uninitialized memory locations.
CVE-2021-45683	An issue was discovered in the binjs_io crate through 2021-01-03 for Rust. The Read method may read from uninitialized memory locations.
CVE-2021-45682	An issue was discovered in the bronzedb-protocol crate through 2021-01-03 for Rust. ReadKVExt may read from uninitialized memory locations.
CVE-2021-45681	An issue was discovered in the derive-com-impl crate before 0.1.2 for Rust. An invalid reference (and memory corruption) can occur because AddRef might not be called before returning a pointer.
CVE-2021-45680	An issue was discovered in the vec-const crate before 2.0.0 for Rust. It tries to construct a Vec from a pointer to a const slice, leading to memory corruption.
CVE-2021-43620	An issue was discovered in the fruity crate through 0.2.0 for Rust. Security-relevant validation of filename extensions is plausibly affected. Methods of NSString for conversion to a string may return a partial result. Because they call CStr::from_ptr on a pointer to the string buffer, the string is terminated at the first '\0' byte, which might not be the end of the string.
CVE-2021-41153	The evm crate is a pure Rust implementation of Ethereum Virtual Machine. In `evm` crate `< 0.31.0`, `JUMPI` opcode's condition is checked after the destination validity check. However, according to Geth and OpenEthereum, the condition check should happen before the destination validity check. This is a **high** severity security advisory if you use `evm` crate for Ethereum mainnet. In this case, you should update your library dependency immediately to on or after `0.31.0`. This is a **low** severity security advisory if you use `evm` crate in Frontier or in a standalone blockchain, because there's no security exploit possible with this advisory. It is **not** recommended to update to on or after `0.31.0` until all the normal chain upgrade preparations have been done. If you use Frontier or other `pallet-evm` based Substrate blockchain, please ensure to update your `spec_version` before updating this. For other blockchains, please make sure to follow a hard-fork process before you update this.
CVE-2021-41150	Tough provides a set of Rust libraries and tools for using and generating the update framework (TUF) repositories. The tough library, prior to 0.12.0, does not properly sanitize delegated role names when caching a repository, or when loading a repository from the filesystem. When the repository is cached or loaded, files ending with the .json extension could be overwritten with role metadata anywhere on the system. A fix is available in version 0.12.0. No workarounds to this issue are known.
CVE-2021-41149	Tough provides a set of Rust libraries and tools for using and generating the update framework (TUF) repositories. The tough library, prior to 0.12.0, does not properly sanitize target names when caching a repository, or when saving specific targets to an output directory. When targets are cached or saved, files could be overwritten with arbitrary content anywhere on the system. A fix is available in version 0.12.0. No workarounds to this issue are known.
CVE-2021-39219	Wasmtime is an open source runtime for WebAssembly & WASI. Wasmtime before version 0.30.0 is affected by a type confusion vulnerability. As a Rust library the `wasmtime` crate clearly marks which functions are safe and which are `unsafe`, guaranteeing that if consumers never use `unsafe` then it should not be possible to have memory unsafety issues in their embeddings of Wasmtime. An issue was discovered in the safe API of `Linker::func_*` APIs. These APIs were previously not sound when one `Engine` was used to create the `Linker` and then a different `Engine` was used to create a `Store` and then the `Linker` was used to instantiate a module into that `Store`. Cross-`Engine` usage of functions is not supported in Wasmtime and this can result in type confusion of function pointers, resulting in being able to safely call a function with the wrong type. Triggering this bug requires using at least two `Engine` values in an embedding and then additionally using two different values with a `Linker` (one at the creation time of the `Linker` and another when instantiating a module with the `Linker`). It's expected that usage of more-than-one `Engine` in an embedding is relatively rare since an `Engine` is intended to be a globally shared resource, so the expectation is that the impact of this issue is relatively small. The fix implemented is to change this behavior to `panic!()` in Rust instead of silently allowing it. Using different `Engine` instances with a `Linker` is a programmer bug that `wasmtime` catches at runtime. This bug has been patched and users should upgrade to Wasmtime version 0.30.0. If you cannot upgrade Wasmtime and are using more than one `Engine` in your embedding it's recommended to instead use only one `Engine` for the entire program if possible. An `Engine` is designed to be a globally shared resource that is suitable to have only one for the lifetime of an entire process. If using multiple `Engine`s is required then code should be audited to ensure that `Linker` is only used with one `Engine`.
CVE-2021-38512	An issue was discovered in the actix-http crate before 3.0.0-beta.9 for Rust. HTTP/1 request smuggling (aka HRS) can occur, potentially leading to credential disclosure.
CVE-2021-38511	An issue was discovered in the tar crate before 0.4.36 for Rust. When symlinks are present in a TAR archive, extraction can create arbitrary directories via .. traversal.
CVE-2021-38196	An issue was discovered in the better-macro crate through 2021-07-22 for Rust. It intentionally demonstrates that remote attackers can execute arbitrary code via proc-macros, and otherwise has no legitimate purpose.
CVE-2021-38195	An issue was discovered in the libsecp256k1 crate before 0.5.0 for Rust. It can verify an invalid signature because it allows the R or S parameter to be larger than the curve order, aka an overflow.
CVE-2021-38194	An issue was discovered in the ark-r1cs-std crate before 0.3.1 for Rust. It does not enforce any constraints in the FieldVar::mul_by_inverse method. Thus, a prover can produce a proof that is unsound but is nonetheless verified.
CVE-2021-38193	An issue was discovered in the ammonia crate before 3.1.0 for Rust. XSS can occur because the parsing differences for HTML, SVG, and MathML are mishandled, a similar issue to CVE-2020-26870.
CVE-2021-38192	An issue was discovered in the prost-types crate before 0.8.0 for Rust. An overflow can occur during conversion from Timestamp to SystemTime.
CVE-2021-38191	An issue was discovered in the tokio crate before 1.8.1 for Rust. Upon a JoinHandle::abort, a Task may be dropped in the wrong thread.
CVE-2021-38190	An issue was discovered in the nalgebra crate before 0.27.1 for Rust. It allows out-of-bounds memory access because it does not ensure that the number of elements is equal to the product of the row count and column count.
CVE-2021-38189	An issue was discovered in the lettre crate before 0.9.6 for Rust. In an e-mail message body, an attacker can place a . character after two <CR><LF> sequences and then inject arbitrary SMTP commands.
CVE-2021-38188	An issue was discovered in the iced-x86 crate through 1.10.3 for Rust. In Decoder::new(), slice.get_unchecked(slice.length()) is used unsafely.
CVE-2021-38187	An issue was discovered in the anymap crate through 0.12.1 for Rust. It violates soundness via conversion of a *u8 to a *u64.
CVE-2021-38186	An issue was discovered in the comrak crate before 0.10.1 for Rust. It mishandles & characters, leading to XSS via &# HTML entities.
CVE-2021-3530	A flaw was discovered in GNU libiberty within demangle_path() in rust-demangle.c, as distributed in GNU Binutils version 2.36. A crafted symbol can cause stack memory to be exhausted leading to a crash.

Name	Description
CVE-2021-32810	crossbeam-deque is a package of work-stealing deques for building task schedulers when programming in Rust. In versions prior to 0.7.4 and 0.8.0, the result of the race condition is that one or more tasks in the worker queue can be popped twice instead of other tasks that are forgotten and never popped. If tasks are allocated on the heap, this can cause double free and a memory leak. If not, this still can cause a logical bug. Crates using `Stealer::steal`, `Stealer::steal_batch`, or `Stealer::steal_batch_and_pop` are affected by this issue. This has been fixed in crossbeam-deque 0.8.1 and 0.7.4.
CVE-2021-32715	hyper is an HTTP library for rust. hyper's HTTP/1 server code had a flaw that incorrectly parses and accepts requests with a `Content-Length` header with a prefixed plus sign, when it should have been rejected as illegal. This combined with an upstream HTTP proxy that doesn't parse such `Content-Length` headers, but forwards them, can result in "request smuggling" or "desync attacks". The flaw exists in all prior versions of hyper prior to 0.14.10, if built with `rustc` v1.5.0 or newer. The vulnerability is patched in hyper version 0.14.10. Two workarounds exist: One may reject requests manually that contain a plus sign prefix in the `Content-Length` header or ensure any upstream proxy handles `Content-Length` headers with a plus sign prefix.
CVE-2021-32714	hyper is an HTTP library for Rust. In versions prior to 0.14.10, hyper's HTTP server and client code had a flaw that could trigger an integer overflow when decoding chunk sizes that are too big. This allows possible data loss, or if combined with an upstream HTTP proxy that allows chunk sizes larger than hyper does, can result in "request smuggling" or "desync attacks." The vulnerability is patched in version 0.14.10. Two possible workarounds exist. One may reject requests manually that contain a `Transfer-Encoding` header or ensure any upstream proxy rejects `Transfer-Encoding` chunk sizes greater than what fits in 64-bit unsigned integers.
CVE-2021-32619	Deno is a runtime for JavaScript and TypeScript that uses V8 and is built in Rust. In Deno versions 1.5.0 to 1.10.1, modules that are dynamically imported through `import()` or `new Worker` might have been able to bypass network and file system permission checks when statically importing other modules. The vulnerability has been patched in Deno release 1.10.2.
CVE-2021-32256	An issue was discovered in GNU libiberty, as distributed in GNU Binutils 2.36. It is a stack-overflow issue in demangle_type in rust-demangle.c.
CVE-2021-31996	An issue was discovered in the algorithmica crate through 2021-03-07 for Rust. There is a double free in merge_sort::merge().
CVE-2021-31919	An issue was discovered in the rkyv crate before 0.6.0 for Rust. When an archive is created via serialization, the archive content may contain uninitialized values of certain parts of a struct.
CVE-2021-31162	In the standard library in Rust before 1.52.0, a double free can occur in the Vec::from_iter function if freeing the element panics.
CVE-2021-30457	An issue was discovered in the id-map crate through 2021-02-26 for Rust. A double free can occur in remove_set upon a panic in a Drop impl.
CVE-2021-30456	An issue was discovered in the id-map crate through 2021-02-26 for Rust. A double free can occur in get_or_insert upon a panic of a user-provided f function.
CVE-2021-30455	An issue was discovered in the id-map crate through 2021-02-26 for Rust. A double free can occur in IdMap::clone_from upon a .clone panic.
CVE-2021-30454	An issue was discovered in the outer_cgi crate before 0.2.1 for Rust. A user-provided Read instance receives an uninitialized memory buffer from KeyValueReader.
CVE-2021-29942	An issue was discovered in the reorder crate through 2021-02-24 for Rust. swap_index can return uninitialized values if an iterator returns a len() that is too large.
CVE-2021-29941	An issue was discovered in the reorder crate through 2021-02-24 for Rust. swap_index has an out-of-bounds write if an iterator returns a len() that is too small.
CVE-2021-29940	An issue was discovered in the through crate through 2021-02-18 for Rust. There is a double free (in through and through_and) upon a panic of the map function.
CVE-2021-29939	An issue was discovered in the stackvector crate through 2021-02-19 for Rust. There is an out-of-bounds write in StackVec::extend if size_hint provides certain anomalous data.
CVE-2021-29938	An issue was discovered in the slice-deque crate through 2021-02-19 for Rust. A double drop can occur in SliceDeque::drain_filter upon a panic in a predicate function.
CVE-2021-29937	An issue was discovered in the telemetry crate through 2021-02-17 for Rust. There is a drop of uninitialized memory if a value.clone() call panics within misc::vec_with_size().
CVE-2021-29936	An issue was discovered in the adtensor crate through 2021-01-11 for Rust. There is a drop of uninitialized memory via the FromIterator implementation for Vector and Matrix.
CVE-2021-29935	An issue was discovered in the rocket crate before 0.4.7 for Rust. uri::Formatter can have a use-after-free if a user-provided function panics.
CVE-2021-29934	An issue was discovered in PartialReader in the uu_od crate before 0.0.4 for Rust. Attackers can read the contents of uninitialized memory locations via a user-provided Read operation.
CVE-2021-29933	An issue was discovered in the insert_many crate through 2021-01-26 for Rust. Elements may be dropped twice if a .next() method panics.
CVE-2021-29932	An issue was discovered in the parse_duration crate through 2021-03-18 for Rust. It allows attackers to cause a denial of service (CPU and memory consumption) via a duration string with a large exponent.
CVE-2021-29931	An issue was discovered in the arenavec crate through 2021-01-12 for Rust. A double drop can sometimes occur upon a panic in T::drop().
CVE-2021-29930	An issue was discovered in the arenavec crate through 2021-01-12 for Rust. A drop of uninitialized memory can sometimes occur upon a panic in T::default().
CVE-2021-29929	An issue was discovered in the endian_trait crate through 2021-01-04 for Rust. A double drop can occur when a user-provided Endian impl panics.
CVE-2021-29922	library/std/src/net/parser.rs in Rust before 1.53.0 does not properly consider extraneous zero characters at the beginning of an IP address string, which (in some situations) allows attackers to bypass access control that is based on IP addresses, because of unexpected octal interpretation.
CVE-2021-29511	evm is a pure Rust implementation of Ethereum Virtual Machine. Prior to the patch, when executing specific EVM opcodes related to memory operations that use `evm_core::Memory::copy_large`, the `evm` crate can over-allocate memory when it is not needed, making it possible for an attacker to perform denial-of-service attack. The flaw was corrected in commit `19ade85`. Users should upgrade to `==0.21.1, ==0.23.1, ==0.24.1, ==0.25.1, >=0.26.1`. There are no workarounds. Please upgrade your `evm` crate version.
CVE-2021-28879	In the standard library in Rust before 1.52.0, the Zip implementation can report an incorrect size due to an integer overflow. This bug can lead to a buffer overflow when a consumed Zip iterator is used again.
CVE-2021-28878	In the standard library in Rust before 1.52.0, the Zip implementation calls __iterator_get_unchecked() more than once for the same index (under certain conditions) when next_back() and next() are used together. This bug could lead to a memory safety violation due to an unmet safety requirement for the TrustedRandomAccess trait.
CVE-2021-28877	In the standard library in Rust before 1.51.0, the Zip implementation calls __iterator_get_unchecked() for the same index more than once when nested. This bug can lead to a memory safety violation due to an unmet safety requirement for the TrustedRandomAccess trait.
CVE-2021-28876	In the standard library in Rust before 1.52.0, the Zip implementation has a panic safety issue. It calls __iterator_get_unchecked() more than once for the same index when the underlying iterator panics (in certain conditions). This bug could lead to a memory safety violation due to an unmet safety requirement for the TrustedRandomAccess trait.

Name	Description
CVE-2021-28875	In the standard library in Rust before 1.50.0, read_to_end() does not validate the return value from Read in an unsafe context. This bug could lead to a buffer overflow.
CVE-2021-28308	An issue was discovered in the fltk crate before 0.15.3 for Rust. There is an out-of bounds read because the pixmap constructor lacks pixmap input validation.
CVE-2021-28307	An issue was discovered in the fltk crate before 0.15.3 for Rust. There is a NULL pointer dereference during attempted use of a non-raster image for a window icon.
CVE-2021-28306	An issue was discovered in the fltk crate before 0.15.3 for Rust. There is a NULL pointer dereference during attempted use of a multi label type if the image is nonexistent.
CVE-2021-28305	An issue was discovered in the diesel crate before 1.4.6 for Rust. There is a use-after-free in the SQLite backend because the semantics of sqlite3_column_name are not followed.
CVE-2021-28037	An issue was discovered in the internment crate before 0.4.2 for Rust. There is a data race that can cause memory corruption because of the unconditional implementation of Sync for Intern<T>.
CVE-2021-28036	An issue was discovered in the quinn crate before 0.7.0 for Rust. It may have invalid memory access for certain versions of the standard library because it relies on a direct cast of std::net::SocketAddrV4 and std::net::SocketAddrV6 data structures.
CVE-2021-28035	An issue was discovered in the stack_dst crate before 0.6.1 for Rust. Because of the push_inner behavior, a drop of uninitialized memory can occur upon a val.clone() panic.
CVE-2021-28034	An issue was discovered in the stack_dst crate before 0.6.1 for Rust. Because of the push_inner behavior, a double free can occur upon a val.clone() panic.
CVE-2021-28033	An issue was discovered in the byte_struct crate before 0.6.1 for Rust. There can be a drop of uninitialized memory if a certain deserialization method panics.
CVE-2021-28032	An issue was discovered in the nano_arena crate before 0.5.2 for Rust. There is an aliasing violation in split_at because two mutable references can exist for the same element, if Borrow<Idx> behaves in certain ways. This can have a resultant out-of-bounds write or use-after-free.
CVE-2021-28031	An issue was discovered in the scratchpad crate before 1.3.1 for Rust. The move_elements function can have a double-free upon a panic in a user-provided f function.
CVE-2021-28030	An issue was discovered in the truetype crate before 0.30.1 for Rust. Attackers can read the contents of uninitialized memory locations via a user-provided Read operation within Tape::take_bytes.
CVE-2021-28029	An issue was discovered in the toodee crate before 0.3.0 for Rust. The row-insertion feature allows attackers to read the contents of uninitialized memory locations.
CVE-2021-28028	An issue was discovered in the toodee crate before 0.3.0 for Rust. Row insertion can cause a double free upon an iterator panic.
CVE-2021-28027	An issue was discovered in the bam crate before 0.1.3 for Rust. There is an integer underflow and out-of-bounds write during the loading of a bgzip block.
CVE-2021-27671	An issue was discovered in the comrak crate before 0.9.1 for Rust. XSS can occur because the protection mechanism for data: and javascript: URIs is case-sensitive, allowing (for example) Data: to be used in an attack.
CVE-2021-27378	An issue was discovered in the rand_core crate before 0.6.2 for Rust. Because read_u32_into and read_u64_into mishandle certain buffer-length checks, a random number generator may be seeded with too little data.
CVE-2021-27377	An issue was discovered in the yottadb crate before 1.2.0 for Rust. For some memory-allocation patterns, ydb_subscript_next_st and ydb_subscript_prev_st have a use-after-free.
CVE-2021-27376	An issue was discovered in the nb-connect crate before 1.0.3 for Rust. It may have invalid memory access for certain versions of the standard library because it relies on a direct cast of std::net::SocketAddrV4 and std::net::SocketAddrV6 data structures.
CVE-2021-26958	An issue was discovered in the xcb crate through 2021-02-04 for Rust. It has a soundness violation because transmutation to the wrong type can happen after xcb::base::cast_event uses std::mem::transmute to return a reference to an arbitrary type.
CVE-2021-26957	An issue was discovered in the xcb crate through 2021-02-04 for Rust. It has a soundness violation because there is an out-of-bounds read in xcb::xproto::change_property(), as demonstrated by a format=32 T=u8 situation where out-of-bounds bytes are sent to an X server.
CVE-2021-26956	An issue was discovered in the xcb crate through 2021-02-04 for Rust. It has a soundness violation because bytes from an X server can be interpreted as any data type returned by xcb::xproto::GetPropertyReply::value.
CVE-2021-26955	An issue was discovered in the xcb crate through 2021-02-04 for Rust. It has a soundness violation because xcb::xproto::GetAtomNameReply::name() calls std::str::from_utf8_unchecked() on unvalidated bytes from an X server.
CVE-2021-26954	An issue was discovered in the qwutils crate before 0.3.1 for Rust. When a Clone panic occurs, insert_slice_clone can perform a double drop.
CVE-2021-26953	An issue was discovered in the postscript crate before 0.14.0 for Rust. It might allow attackers to obtain sensitive information from uninitialized memory locations via a user-provided Read implementation.
CVE-2021-26952	An issue was discovered in the ms3d crate before 0.1.3 for Rust. It might allow attackers to obtain sensitive information from uninitialized memory locations via IoReader::read.
CVE-2021-26951	An issue was discovered in the calamine crate before 0.17.0 for Rust. It allows attackers to overwrite heap-memory locations because Vec::set_len is used without proper memory claiming, and this uninitialized memory is used for a user-provided Read operation, as demonstrated by Sectors::get.
CVE-2021-26308	An issue was discovered in the marc crate before 2.0.0 for Rust. A user-provided Read implementation can gain access to the old contents of newly allocated memory, violating soundness.
CVE-2021-26307	An issue was discovered in the raw-cpuid crate before 9.0.0 for Rust. It allows __cpuid_count() calls even if the processor does not support the CPUID instruction, which is unsound and causes a deterministic crash.
CVE-2021-26306	An issue was discovered in the raw-cpuid crate before 9.0.0 for Rust. It has unsound transmute calls within as_string() methods.
CVE-2021-26305	An issue was discovered in Deserializer::read_vec in the cdr crate before 0.2.4 for Rust. A user-provided Read implementation can gain access to the old contents of newly allocated heap memory, violating soundness.
CVE-2021-25908	An issue was discovered in the fil-ocl crate through 2021-01-04 for Rust. From<EventList> can lead to a double free.
CVE-2021-25907	An issue was discovered in the containers crate before 0.9.11 for Rust. When a panic occurs, a util::{mutate,mutate2} double drop can be performed.
CVE-2021-25906	An issue was discovered in the basic_dsp_matrix crate before 0.9.2 for Rust. When a TransformContent panic occurs, a double drop can be performed.

Name	Description
CVE-2021-25905	An issue was discovered in the bra crate before 0.1.1 for Rust. It lacks soundness because it can read uninitialized memory.
CVE-2021-25904	An issue was discovered in the av-data crate before 0.3.0 for Rust. A raw pointer is dereferenced, leading to a read of an arbitrary memory address, sometimes causing a segfault.
CVE-2021-25903	An issue was discovered in the cache crate through 2021-01-01 for Rust. A raw pointer is dereferenced.
CVE-2021-25902	An issue was discovered in the glsl-layout crate before 0.4.0 for Rust. When a panic occurs, map_array can perform a double drop.
CVE-2021-25901	An issue was discovered in the lazy-init crate through 2021-01-17 for Rust. Lazy lacks a Send bound, leading to a data race.
CVE-2021-25900	An issue was discovered in the smallvec crate before 0.6.14 and 1.x before 1.6.1 for Rust. There is a heap-based buffer overflow in SmallVec::insert_many.
CVE-2021-24117	In Apache Teaclave Rust SGX SDK 1.1.3, a side-channel vulnerability in base64 PEM file decoding allows system-level (administrator) attackers to obtain information about secret RSA keys via a controlled-channel and side-channel attack on software running in isolated environments that can be single stepped, especially Intel SGX.
CVE-2021-21299	hyper is an open-source HTTP library for Rust (crates.io). In hyper from version 0.12.0 and before versions 0.13.10 and 0.14.3 there is a vulnerability that can enable a request smuggling attack. The HTTP server code had a flaw that incorrectly understands some requests with multiple transfer-encoding headers to have a chunked payload, when it should have been rejected as illegal. This combined with an upstream HTTP proxy that understands the request payload boundary differently can result in "request smuggling" or "desync attacks". To determine if vulnerable, all these things must be true: 1) Using hyper as an HTTP server (the client is not affected), 2) Using HTTP/1.1 (HTTP/2 does not use transfer-encoding), 3) Using a vulnerable HTTP proxy upstream to hyper. If an upstream proxy correctly rejects the illegal transfer-encoding headers, the desync attack cannot succeed. If there is no proxy upstream of hyper, hyper cannot start the desync attack, as the client will repair the headers before forwarding. This is fixed in versions 0.14.3 and 0.13.10. As a workaround one can take the following options: 1) Reject requests that contain a `transfer-encoding` header, 2) Ensure any upstream proxy handles `transfer-encoding` correctly.
CVE-2021-21269	Keymaker is a Mastodon Community Finder based Matrix Community serverlist page Server. In Keymaker before version 0.2.0, the assets endpoint did not check for the extension. The rust `join` method without checking user input might have made it abe to do a Path Traversal attack causing to read more files than allowed. This is fixed in version 0.2.0.
CVE-2021-21235	kamadak-exif is an exif parsing library written in pure Rust. In kamadak-exif version 0.5.2, there is an infinite loop in parsing crafted PNG files. Specifically, reader::read_from_container can cause an infinite loop when a crafted PNG file is given. This is fixed in version 0.5.3. No workaround is available. Applications that do not pass files with the PNG signature to Reader::read_from_container are not affected.
CVE-2021-20332	Specific MongoDB Rust Driver versions can include credentials used by the connection pool to authenticate connections in the monitoring event that is emitted when the pool is created. The user's logging infrastructure could then potentially ingest these events and unexpectedly leak the credentials. Note that such monitoring is not enabled by default.
CVE-2020-5499	Baidu Rust SGX SDK through 1.0.8 has an enclave ID race. There are non-deterministic results in which, sometimes, two global IDs are the same.
CVE-2020-36514	An issue was discovered in the acc_reader crate through 2020-12-27 for Rust. fill_buf may read from uninitialized memory locations.
CVE-2020-36513	An issue was discovered in the acc_reader crate through 2020-12-27 for Rust. read_up_to may read from uninitialized memory locations.
CVE-2020-36512	An issue was discovered in the buffoon crate through 2020-12-31 for Rust. InputStream::read_exact may read from uninitialized memory locations.
CVE-2020-36511	An issue was discovered in the bite crate through 2020-12-31 for Rust. read::BiteReadExpandedExt::read_framed_max may read from uninitialized memory locations.
CVE-2020-36472	An issue was discovered in the max7301 crate before 0.2.0 for Rust. The ImmediateIO and TransactionalIO types implement Sync for all Expander<EI> types that they contain.
CVE-2020-36471	An issue was discovered in the generator crate before 0.7.0 for Rust. It does not ensure that a function (for yielding values) has Send bounds.
CVE-2020-36470	An issue was discovered in the disrustor crate through 2020-12-17 for Rust. RingBuffer doe not properly limit the number of mutable references.
CVE-2020-36469	An issue was discovered in the appendix crate through 2020-11-15 for Rust. For the generic K and V type parameters, Send and Sync are implemented unconditionally.
CVE-2020-36468	An issue was discovered in the cgc crate through 2020-12-10 for Rust. Ptr::write performs non-atomic write operations on an underlying pointer.
CVE-2020-36467	An issue was discovered in the cgc crate through 2020-12-10 for Rust. Ptr::get returns more than one mutable reference to the same object.
CVE-2020-36466	An issue was discovered in the cgc crate through 2020-12-10 for Rust. Ptr implements Send and Sync for all types.
CVE-2020-36465	An issue was discovered in the generic-array crate before 0.13.3 for Rust. It violates soundness by using the arr! macro to extend lifetimes.
CVE-2020-36464	An issue was discovered in the heapless crate before 0.6.1 for Rust. The IntoIter Clone implementation clones an entire underlying Vec without considering whether it has already been partially consumed.
CVE-2020-36463	An issue was discovered in the multiqueue crate through 2020-12-25 for Rust. There are unconditional implementations of Send for InnerSend<RW, T>, InnerRecv<RW, T>, FutInnerSend<RW, T>, and FutInnerRecv<RW, T>.
CVE-2020-36462	An issue was discovered in the syncpool crate before 0.1.6 for Rust. There is an unconditional implementation of Send for Bucket2.
CVE-2020-36461	An issue was discovered in the noise_search crate through 2020-12-10 for Rust. There are unconditional implementations of Send and Sync for MvccRwLock.
CVE-2020-36460	An issue was discovered in the model crate through 2020-11-10 for Rust. The Shared data structure has an implementation of the Send and Sync traits without regard for the inner type.
CVE-2020-36459	An issue was discovered in the dces crate through 2020-12-09 for Rust. The World type is marked as Send but lacks bounds on its EntityStore and ComponentStore.
CVE-2020-36458	An issue was discovered in the lexer crate through 2020-11-10 for Rust. For ReaderResult<T, E>, there is an implementation of Sync with a trait bound of T: Send, E: Send.
CVE-2020-36457	An issue was discovered in the lever crate before 0.1.1 for Rust. AtomicBox<T> implements the Send and Sync traits for all types T.
CVE-2020-36456	An issue was discovered in the toolshed crate through 2020-11-15 for Rust. In CopyCell<T>, the Send trait lacks bounds on the contained type.
CVE-2020-36455	An issue was discovered in the slock crate through 2020-11-17 for Rust. Slock<T> unconditionally implements Send and Sync.
CVE-2020-36454	An issue was discovered in the parc crate through 2020-11-14 for Rust. LockWeak<T> has an unconditional implementation of Send without trait bounds on T.
CVE-2020-36453	An issue was discovered in the scottqueue crate through 2020-11-15 for Rust. There are unconditional implementations of Send and Sync for Queue<T>.

Name	Description
CVE-2020-36452	An issue was discovered in the array-tools crate before 0.3.2 for Rust. FixedCapacityDequeLike::clone() has a drop of uninitialized memory.
CVE-2020-36451	An issue was discovered in the rcu_cell crate through 2020-11-14 for Rust. There are unconditional implementations of Send and Sync for RcuCell<T>.
CVE-2020-36450	An issue was discovered in the bunch crate through 2020-11-12 for Rust. There are unconditional implementations of Send and Sync for Bunch<T>.
CVE-2020-36449	An issue was discovered in the kekbit crate before 0.3.4 for Rust. For ShmWriter<H>, Send is implemented without requiring H: Send.
CVE-2020-36448	An issue was discovered in the cache crate through 2020-11-24 for Rust. There are unconditional implementations of Send and Sync for Cache<K>.
CVE-2020-36447	An issue was discovered in the v9 crate through 2020-12-18 for Rust. There is an unconditional implementation of Sync for SyncRef<T>.
CVE-2020-36446	An issue was discovered in the signal-simple crate through 2020-11-15 for Rust. There are unconditional implementations of Send and Sync for SyncChannel<T>.
CVE-2020-36445	An issue was discovered in the convec crate through 2020-11-24 for Rust. There are unconditional implementations of Send and Sync for ConVec<T>.
CVE-2020-36444	An issue was discovered in the async-coap crate through 2020-12-08 for Rust. Send and Sync are implemented for ArcGuard<RC, T> without trait bounds on RC.
CVE-2020-36443	An issue was discovered in the libp2p-deflate crate before 0.27.1 for Rust. An uninitialized buffer is passed to AsyncRead::poll_read(), which is a user-provided trait function.
CVE-2020-36442	An issue was discovered in the beef crate before 0.5.0 for Rust. beef::Cow has no Sync bound on its Send trait.
CVE-2020-36441	An issue was discovered in the abox crate before 0.4.1 for Rust. It implements Send and Sync for AtomicBox<T> with no requirement for T: Send and T: Sync.
CVE-2020-36440	An issue was discovered in the libsbc crate before 0.1.5 for Rust. For Decoder<R>, it implements Send for any R: Read.
CVE-2020-36439	An issue was discovered in the ticketed_lock crate before 0.3.0 for Rust. There are unconditional implementations of Send for ReadTicket<T> and WriteTicket<T>.
CVE-2020-36438	An issue was discovered in the tiny_future crate before 0.4.0 for Rust. Future<T> does not have bounds on its Send and Sync traits.
CVE-2020-36437	An issue was discovered in the conqueue crate before 0.4.0 for Rust. There are unconditional implementations of Send and Sync for QueueSender<T>.
CVE-2020-36436	An issue was discovered in the unicycle crate before 0.7.1 for Rust. PinSlab<T> and Unordered<T, S> do not have bounds on their Send and Sync traits.
CVE-2020-36435	An issue was discovered in the ruspiro-singleton crate before 0.4.1 for Rust. In Singleton, Send and Sync do not have bounds checks.
CVE-2020-36434	An issue was discovered in the sys-info crate before 0.8.0 for Rust. sys_info::disk_info calls can trigger a double free.
CVE-2020-36433	An issue was discovered in the chunky crate through 2020-08-25 for Rust. The Chunk API does not honor an alignment requirement.
CVE-2020-36432	An issue was discovered in the alg_ds crate through 2020-08-25 for Rust. There is a drop of uninitialized memory in Matrix::new().
CVE-2020-36323	In the standard library in Rust before 1.52.0, there is an optimization for joining strings that can cause uninitialized bytes to be exposed (or the program to crash) if the borrowed string changes after its length is checked.
CVE-2020-36318	In the standard library in Rust before 1.49.0, VecDeque::make_contiguous has a bug that pops the same element more than once under certain condition. This bug could result in a use-after-free or double free.
CVE-2020-36317	In the standard library in Rust before 1.49.0, String::retain() function has a panic safety problem. It allows creation of a non-UTF-8 Rust string when the provided closure panics. This bug could result in a memory safety violation when other string APIs assume that UTF-8 encoding is used on the same string.
CVE-2020-36220	An issue was discovered in the va-ts crate before 0.0.4 for Rust. Because Demuxer<T> omits a required T: Send bound, a data race and memory corruption can occur.
CVE-2020-36219	An issue was discovered in the atomic-option crate through 2020-10-31 for Rust. Because AtomicOption<T> implements Sync unconditionally, a data race can occur.
CVE-2020-36218	An issue was discovered in the buttplug crate before 1.0.4 for Rust. ButtplugFutureStateShared does not properly consider (!Send|!Sync) objects, leading to a data race.
CVE-2020-36217	An issue was discovered in the may_queue crate through 2020-11-10 for Rust. Because Queue does not have bounds on its Send trait or Sync trait, memory corruption can occur.
CVE-2020-36216	An issue was discovered in Input<R> in the eventio crate before 0.5.1 for Rust. Because a non-Send type can be sent to a different thread, a data race and memory corruption can occur.
CVE-2020-36215	An issue was discovered in the hashconsing crate before 1.1.0 for Rust. Because HConsed does not have bounds on its Send trait or Sync trait, memory corruption can occur.
CVE-2020-36214	An issue was discovered in the multiqueue2 crate before 0.1.7 for Rust. Because a non-Send type can be sent to a different thread, a data race can occur.
CVE-2020-36213	An issue was discovered in the abi_stable crate before 0.9.1 for Rust. A retain call can create an invalid UTF-8 string, violating soundness.
CVE-2020-36212	An issue was discovered in the abi_stable crate before 0.9.1 for Rust. DrainFilter lacks soundness because of a double drop.
CVE-2020-36211	An issue was discovered in the gfwx crate before 0.3.0 for Rust. Because ImageChunkMut does not have bounds on its Send trait or Sync trait, a data race and memory corruption can occur.
CVE-2020-36210	An issue was discovered in the autorand crate before 0.2.3 for Rust. Because of impl Random on arrays, uninitialized memory can be dropped when a panic occurs, leading to memory corruption.
CVE-2020-36209	An issue was discovered in the late-static crate before 0.4.0 for Rust. Because Sync is implemented for LateStatic with T: Send, a data race can occur.
CVE-2020-36208	An issue was discovered in the conquer-once crate before 0.3.2 for Rust. Thread crossing can occur for a non-Send but Sync type, leading to memory corruption.
CVE-2020-36207	An issue was discovered in the aovec crate through 2020-12-10 for Rust. Because Aovec<T> does not have bounds on its Send trait or Sync trait, a data race and memory corruption can occur.
CVE-2020-36206	An issue was discovered in the rusb crate before 0.7.0 for Rust. Because of a lack of Send and Sync bounds, a data race and memory corruption can occur.
CVE-2020-36205	An issue was discovered in the xcb crate through 2020-12-10 for Rust. base::Error does not have soundness. Because of the public ptr field, a use-after-free or double-free can occur.
CVE-2020-36204	An issue was discovered in the im crate through 2020-11-09 for Rust. Because TreeFocus does not have bounds on its Send trait or Sync trait, a data race can occur.

Name	Description
CVE-2020-36203	An issue was discovered in the reffers crate through 2020-12-01 for Rust. ARefss can contain a !Send,!Sync object, leading to a data race and memory corruption.
CVE-2020-36202	An issue was discovered in the async-h1 crate before 2.3.0 for Rust. Request smuggling can occur when used behind a reverse proxy.
CVE-2020-35928	An issue was discovered in the concread crate before 0.2.6 for Rust. Attackers can cause an ARCache<K,V> data race by sending types that do not implement Send/Sync.
CVE-2020-35927	An issue was discovered in the thex crate through 2020-12-08 for Rust. Thex<T> allows cross-thread data races of non-Send types.
CVE-2020-35926	An issue was discovered in the nanorand crate before 0.5.1 for Rust. It caused any random number generator (even ChaCha) to return all zeroes because integer truncation was mishandled.
CVE-2020-35925	An issue was discovered in the magnetic crate before 2.0.1 for Rust. MPMCConsumer and MPMCProducer allow cross-thread sending of a non-Send type.
CVE-2020-35924	An issue was discovered in the try-mutex crate before 0.3.0 for Rust. TryMutex<T> allows cross-thread sending of a non-Send type.
CVE-2020-35923	An issue was discovered in the ordered-float crate before 1.1.1 and 2.x before 2.0.1 for Rust. A NotNan value can contain a NaN.
CVE-2020-35922	An issue was discovered in the mio crate before 0.7.6 for Rust. It has false expectations about the std::net::SocketAddr memory representation.
CVE-2020-35921	An issue was discovered in the miow crate before 0.3.6 for Rust. It has false expectations about the std::net::SocketAddr memory representation.
CVE-2020-35920	An issue was discovered in the socket2 crate before 0.3.16 for Rust. It has false expectations about the std::net::SocketAddr memory representation.
CVE-2020-35919	An issue was discovered in the net2 crate before 0.2.36 for Rust. It has false expectations about the std::net::SocketAddr memory representation.
CVE-2020-35918	An issue was discovered in the branca crate before 0.10.0 for Rust. Decoding tokens (with invalid base62 data) can panic.
CVE-2020-35917	An issue was discovered in the pyo3 crate before 0.12.4 for Rust. There is a reference-counting error and use-after-free in From<Py<T>>.
CVE-2020-35916	An issue was discovered in the image crate before 0.23.12 for Rust. A Mutable reference has immutable provenance. (In the case of LLVM, the IR may be always correct.)
CVE-2020-35915	An issue was discovered in the futures-intrusive crate before 0.4.0 for Rust. GenericMutexGuard allows cross-thread data races of non-Sync types.
CVE-2020-35914	An issue was discovered in the lock_api crate before 0.4.2 for Rust. A data race can occur because of RwLockWriteGuard unsoundness.
CVE-2020-35913	An issue was discovered in the lock_api crate before 0.4.2 for Rust. A data race can occur because of RwLockReadGuard unsoundness.
CVE-2020-35912	An issue was discovered in the lock_api crate before 0.4.2 for Rust. A data race can occur because of MappedRwLockWriteGuard unsoundness.
CVE-2020-35911	An issue was discovered in the lock_api crate before 0.4.2 for Rust. A data race can occur because of MappedRwLockReadGuard unsoundness.
CVE-2020-35910	An issue was discovered in the lock_api crate before 0.4.2 for Rust. A data race can occur because of MappedMutexGuard unsoundness.
CVE-2020-35909	An issue was discovered in the multihash crate before 0.11.3 for Rust. The from_slice parsing code can panic via unsanitized data from a network server.
CVE-2020-35908	An issue was discovered in the futures-util crate before 0.3.2 for Rust. FuturesUnordered can lead to data corruption because Sync is mishandled.
CVE-2020-35907	An issue was discovered in the futures-task crate before 0.3.5 for Rust. futures_task::noop_waker_ref allows a NULL pointer dereference.
CVE-2020-35906	An issue was discovered in the futures-task crate before 0.3.6 for Rust. futures_task::waker may cause a use-after-free in a non-static type situation.
CVE-2020-35905	An issue was discovered in the futures-util crate before 0.3.7 for Rust. MutexGuard::map can cause a data race for certain closure situations (in safe code).
CVE-2020-35904	An issue was discovered in the crossbeam-channel crate before 0.4.4 for Rust. It has incorrect expectations about the relationship between the memory allocation and how many iterator elements there are.
CVE-2020-35903	An issue was discovered in the dync crate before 0.5.0 for Rust. VecCopy allows misaligned element access because u8 is not always the type in question.
CVE-2020-35902	An issue was discovered in the actix-codec crate before 0.3.0-beta.1 for Rust. There is a use-after-free in Framed.
CVE-2020-35901	An issue was discovered in the actix-http crate before 2.0.0-alpha.1 for Rust. There is a use-after-free in BodyStream.
CVE-2020-35900	An issue was discovered in the array-queue crate through 2020-09-26 for Rust. A pop_back() call may lead to a use-after-free.
CVE-2020-35899	An issue was discovered in the actix-service crate before 1.0.6 for Rust. The Cell implementation allows obtaining more than one mutable reference to the same data.
CVE-2020-35898	An issue was discovered in the actix-utils crate before 2.0.0 for Rust. The Cell implementation allows obtaining more than one mutable reference to the same data.
CVE-2020-35897	An issue was discovered in the atom crate before 0.3.6 for Rust. An unsafe Send implementation allows a cross-thread data race.
CVE-2020-35896	An issue was discovered in the ws crate through 2020-09-25 for Rust. The outgoing buffer is not properly limited, leading to a remote memory-consumption attack.
CVE-2020-35895	An issue was discovered in the stack crate before 0.3.1 for Rust. ArrayVec has an out-of-bounds write via element insertion.
CVE-2020-35894	An issue was discovered in the obstack crate before 0.1.4 for Rust. Unaligned references can occur.
CVE-2020-35893	An issue was discovered in the simple-slab crate before 0.3.3 for Rust. remove() has an off-by-one error, causing memory leakage and a drop of uninitialized memory.
CVE-2020-35892	An issue was discovered in the simple-slab crate before 0.3.3 for Rust. index() allows an out-of-bounds read.
CVE-2020-35891	An issue was discovered in the ordnung crate through 2020-09-03 for Rust. compact::Vec violates memory safety via a remove() double free.
CVE-2020-35890	An issue was discovered in the ordnung crate through 2020-09-03 for Rust. compact::Vec violates memory safety via out-of-bounds access for large capacity.
CVE-2020-35889	An issue was discovered in the crayon crate through 2020-08-31 for Rust. A TOCTOU issue has a resultant memory safety violation via HandleLike.
CVE-2020-35888	An issue was discovered in the arr crate through 2020-08-25 for Rust. Uninitialized memory is dropped by Array::new_from_template.
CVE-2020-35887	An issue was discovered in the arr crate through 2020-08-25 for Rust. There is a buffer overflow in Index and IndexMut.

Name	Description
CVE-2020-35886	An issue was discovered in the arr crate through 2020-08-25 for Rust. An attacker can smuggle non-Sync/Send types across a thread boundary to cause a data race.
CVE-2020-35885	An issue was discovered in the alpm-rs crate through 2020-08-20 for Rust. StrcCtx performs improper memory deallocation.
CVE-2020-35884	An issue was discovered in the tiny_http crate through 2020-06-16 for Rust. HTTP Request smuggling can occur via a malformed Transfer-Encoding header.
CVE-2020-35883	An issue was discovered in the mozwire crate through 2020-08-18 for Rust. A ../ directory-traversal situation allows overwriting local files that have .conf at the end of the filename.
CVE-2020-35882	An issue was discovered in the rocket crate before 0.4.5 for Rust. LocalRequest::clone creates more than one mutable references to the same object, possibly causing a data race.
CVE-2020-35881	An issue was discovered in the traitobject crate through 2020-06-01 for Rust. It has false expectations about fat pointers, possibly causing memory corruption in, for example, Rust 2.x.
CVE-2020-35880	An issue was discovered in the bigint crate through 2020-05-07 for Rust. It allows a soundness violation.
CVE-2020-35879	An issue was discovered in the rulinalg crate through 2020-02-11 for Rust. There are incorrect lifetime-boundary definitions for RowMut::raw_slice and RowMut::raw_slice_mut.
CVE-2020-35878	An issue was discovered in the ozone crate through 2020-07-04 for Rust. Memory safety is violated because of the dropping of uninitialized memory.
CVE-2020-35877	An issue was discovered in the ozone crate through 2020-07-04 for Rust. Memory safety is violated because of out-of-bounds access.
CVE-2020-35876	An issue was discovered in the rio crate through 2020-05-11 for Rust. A struct can be leaked, allowing attackers to obtain sensitive information, cause a use-after-free, or cause a data race.
CVE-2020-35875	An issue was discovered in the tokio-rustls crate before 0.13.1 for Rust. Excessive memory usage may occur when data arrives quickly.
CVE-2020-35874	An issue was discovered in the internment crate through 2020-05-28 for Rust. ArcIntern::drop has a race condition and resultant use-after-free.
CVE-2020-35873	An issue was discovered in the rusqlite crate before 0.23.0 for Rust. Memory safety can be violated because sessions.rs has a use-after-free.
CVE-2020-35872	An issue was discovered in the rusqlite crate before 0.23.0 for Rust. Memory safety can be violated via the repr(Rust) type.
CVE-2020-35871	An issue was discovered in the rusqlite crate before 0.23.0 for Rust. Memory safety can be violated via an Auxdata API data race.
CVE-2020-35870	An issue was discovered in the rusqlite crate before 0.23.0 for Rust. Memory safety can be violated via an Auxdata API use-after-free.
CVE-2020-35869	An issue was discovered in the rusqlite crate before 0.23.0 for Rust. Memory safety can be violated because rusqlite::trace::log mishandles format strings.
CVE-2020-35868	An issue was discovered in the rusqlite crate before 0.23.0 for Rust. Memory safety can be violated via UnlockNotification.
CVE-2020-35867	An issue was discovered in the rusqlite crate before 0.23.0 for Rust. Memory safety can be violated via create_module.
CVE-2020-35866	An issue was discovered in the rusqlite crate before 0.23.0 for Rust. Memory safety can be violated via VTab / VTabCursor.
CVE-2020-35865	An issue was discovered in the os_str_bytes crate before 2.0.0 for Rust. It has false expectations about char::from_u32_unchecked behavior.
CVE-2020-35864	An issue was discovered in the flatbuffers crate through 2020-04-11 for Rust. read_scalar (and read_scalar_at) can transmute values without unsafe blocks.
CVE-2020-35863	An issue was discovered in the hyper crate before 0.12.34 for Rust. HTTP request smuggling can occur. Remote code execution can occur in certain situations with an HTTP server on the loopback interface.
CVE-2020-35862	An issue was discovered in the bitvec crate before 0.17.4 for Rust. BitVec to BitBox conversion leads to a use-after-free or double free.
CVE-2020-35861	An issue was discovered in the bumpalo crate before 3.2.1 for Rust. The realloc feature allows the reading of unknown memory. Attackers can potentially read cryptographic keys.
CVE-2020-35860	An issue was discovered in the cbox crate through 2020-03-19 for Rust. The CBox API allows dereferencing raw pointers without a requirement for unsafe code.
CVE-2020-35859	An issue was discovered in the lucet-runtime-internals crate before 0.5.1 for Rust. It mishandles sigstack allocation. Guest programs may be able to obtain sensitive information, or guest programs can experience memory corruption.
CVE-2020-35858	An issue was discovered in the prost crate before 0.6.1 for Rust. There is stack consumption via a crafted message, causing a denial of service (e.g., x86) or possibly remote code execution (e.g., ARM).
CVE-2020-35857	An issue was discovered in the trust-dns-server crate before 0.18.1 for Rust. DNS MX and SRV null targets are mishandled, causing stack consumption.
CVE-2020-35711	An issue has been discovered in the arc-swap crate before 0.4.8 (and 1.x before 1.1.0) for Rust. Use of arc_swap::access::Map with the Constant test helper (or with a user-supplied implementation of the Access trait) could sometimes lead to dangling references being returned by the map.
CVE-2020-28247	The lettre library through 0.10.0-alpha for Rust allows arbitrary sendmail option injection via transport/sendmail/mod.rs.
CVE-2020-26297	mdBook is a utility to create modern online books from Markdown files and is written in Rust. In mdBook before version 0.4.5, there is a vulnerability affecting the search feature of mdBook, which could allow an attacker to execute arbitrary JavaScript code on the page. The search feature of mdBook (introduced in version 0.1.4) was affected by a cross site scripting vulnerability that allowed an attacker to execute arbitrary JavaScript code on an user's browser by tricking the user into typing a malicious search query, or tricking the user into clicking a link to the search page with the malicious search query prefilled. mdBook 0.4.5 fixes the vulnerability by properly escaping the search query. Owners of websites built with mdBook have to upgrade to mdBook 0.4.5 or greater and rebuild their website contents with it.
CVE-2020-26281	async-h1 is an asynchronous HTTP/1.1 parser for Rust (crates.io). There is a request smuggling vulnerability in async-h1 before version 2.3.0. This vulnerability affects any webserver that uses async-h1 behind a reverse proxy, including all such Tide applications. If the server does not read the body of a request which is longer than some buffer length, async-h1 will attempt to read a subsequent request from the body content starting at that offset into the body. One way to exploit this vulnerability would be for an adversary to craft a request such that the body contains a request that would not be noticed by a reverse proxy, allowing it to forge forwarded/x-forwarded headers. If an application trusted the authenticity of these headers, it could be misled by the smuggled request. Another potential concern with this vulnerability is that if a reverse proxy is sending multiple http clients' requests along the same keep-alive connection, it would be possible for the smuggled request to specify a long content and capture another user's request in its body. This content could be captured in a post request to an endpoint that allows the content to be subsequently retrieved by the adversary. This has been addressed in async-h1 2.3.0 and previous versions have been yanked.
CVE-2020-26235	In Rust time crate from version 0.2.7 and before version 0.2.23, unix-like operating systems may segfault due to dereferencing a dangling pointer in specific circumstances. This requires the user to set any environment variable in a different thread than the affected functions. The affected functions are time::UtcOffset::local_offset_at, time::UtcOffset::try_local_offset_at,

Name	Description
	time::UtcOffset::current_local_offset, time::UtcOffset::try_current_local_offset, time::OffsetDateTime::now_local and time::OffsetDateTime::try_now_local. Non-Unix targets are unaffected. This includes Windows and wasm. The issue was introduced in version 0.2.7 and fixed in version 0.2.23.
CVE-2020-26222	Dependabot is a set of packages for automated dependency management for Ruby, JavaScript, Python, PHP, Elixir, Rust, Java, .NET, Elm and Go. In Dependabot-Core from version 0.119.0.beta1 before version 0.125.1, there is a remote code execution vulnerability in dependabot-common and dependabot-go_modules when a source branch name contains malicious injectable bash code. For example, if Dependabot is configured to use the following source branch name: "/$({curl,127.0.0.1})", Dependabot will make a HTTP request to the following URL: 127.0.0.1 when cloning the source repository. The fix was applied to version 0.125.1. As a workaround, one can escape the branch name prior to passing it to the Dependabot::Source class.
CVE-2020-25796	An issue was discovered in the sized-chunks crate through 0.6.2 for Rust. In the InlineArray implementation, an unaligned reference may be generated for a type that has a large alignment requirement.
CVE-2020-25795	An issue was discovered in the sized-chunks crate through 0.6.2 for Rust. In the Chunk implementation, insert_from can have a memory-safety issue upon a panic.
CVE-2020-25794	An issue was discovered in the sized-chunks crate through 0.6.2 for Rust. In the Chunk implementation, clone can have a memory-safety issue upon a panic.
CVE-2020-25793	An issue was discovered in the sized-chunks crate through 0.6.2 for Rust. In the Chunk implementation, the array size is not checked when constructed with From<InlineArray<A, T>>.
CVE-2020-25792	An issue was discovered in the sized-chunks crate through 0.6.2 for Rust. In the Chunk implementation, the array size is not checked when constructed with pair().
CVE-2020-25791	An issue was discovered in the sized-chunks crate through 0.6.2 for Rust. In the Chunk implementation, the array size is not checked when constructed with unit().
CVE-2020-25576	An issue was discovered in the rand_core crate before 0.4.2 for Rust. Casting of byte slices to integer slices mishandles alignment constraints.
CVE-2020-25575	** UNSUPPORTED WHEN ASSIGNED ** An issue was discovered in the failure crate through 0.1.5 for Rust. It may introduce "compatibility hazards" in some applications, and has a type confusion flaw when downcasting. NOTE: This vulnerability only affects products that are no longer supported by the maintainer. NOTE: This may overlap CVE-2019-25010.
CVE-2020-25574	An issue was discovered in the http crate before 0.1.20 for Rust. An integer overflow in HeaderMap::reserve() could result in denial of service (e.g., an infinite loop).
CVE-2020-25573	An issue was discovered in the linked-hash-map crate before 0.5.3 for Rust. It creates an uninitialized NonNull pointer, which violates a non-null constraint.
CVE-2020-25016	A safety violation was discovered in the rgb crate before 0.8.20 for Rust, leading to (for example) dereferencing of arbitrary pointers or disclosure of uninitialized memory. This occurs because structs can be treated as bytes for read and write operations.
CVE-2020-15093	The tough library (Rust/crates.io) prior to version 0.7.1 does not properly verify the threshold of cryptographic signatures. It allows an attacker to duplicate a valid signature in order to circumvent TUF requiring a minimum threshold of unique signatures before the metadata is considered valid. A fix is available in version 0.7.1. CVE-2020-6174 is assigned to the same vulnerability in the TUF reference implementation.
CVE-2020-13759	rust-vmm vm-memory before 0.1.1 and 0.2.x before 0.2.1 allows attackers to cause a denial of service (loss of IP networking) because read_obj and write_obj do not properly access memory. This affects aarch64 (with musl or glibc) and x86_64 (with musl).
CVE-2019-25055	An issue was discovered in the libpulse-binding crate before 2.6.0 for Rust. It mishandles a panic that crosses a Foreign Function Interface (FFI) boundary.
CVE-2019-25054	An issue was discovered in the pnet crate before 0.27.2 for Rust. There is a segmentation fault (upon attempted dereference of an uninitialized descriptor) because of an erroneous IcmpTransportChannelIterator compiler optimization.
CVE-2019-25010	An issue was discovered in the failure crate through 2019-11-13 for Rust. Type confusion can occur when __private_get_type_id__ is overridden.
CVE-2019-25009	An issue was discovered in the http crate before 0.1.20 for Rust. The HeaderMap::Drain API can use a raw pointer, defeating soundness.
CVE-2019-25007	An issue was discovered in the streebog crate before 0.8.0 for Rust. The Streebog hash function can cause a panic.
CVE-2019-25006	An issue was discovered in the streebog crate before 0.8.0 for Rust. The Streebog hash function can produce the wrong answer.
CVE-2019-25005	An issue was discovered in the chacha20 crate before 0.2.3 for Rust. A ChaCha20 counter overflow makes it easier for attackers to determine plaintext.
CVE-2019-25004	An issue was discovered in the flatbuffers crate before 0.6.1 for Rust. Arbitrary bytes can be reinterpreted as a bool, defeating soundness.
CVE-2019-25003	An issue was discovered in the libsecp256k1 crate before 0.3.1 for Rust. Scalar::check_overflow allows a timing side-channel attack; consequently, attackers can obtain sensitive information.
CVE-2019-25002	An issue was discovered in the sodiumoxide crate before 0.2.5 for Rust. generichash::Digest::eq compares itself to itself and thus has degenerate security properties.
CVE-2019-25001	An issue was discovered in the serde_cbor crate before 0.10.2 for Rust. The CBOR deserializer can cause stack consumption via nested semantic tags.
CVE-2019-16882	An issue was discovered in the string-interner crate before 0.7.1 for Rust. It allows attackers to read from memory locations associated with dangling pointers, because of a cloning flaw.
CVE-2019-16881	An issue was discovered in the portaudio-rs crate through 0.3.1 for Rust. There is a use-after-free with resultant arbitrary code execution because of a lack of unwind safety in stream_callback and stream_finished_callback.
CVE-2019-16880	An issue was discovered in the linea crate through 0.9.4 for Rust. There is double free in the Matrix::zip_elements method.
CVE-2019-16760	Cargo prior to Rust 1.26.0 may download the wrong dependency if your package.toml file uses the `package` configuration key. Usage of the `package` key to rename dependencies in `Cargo.toml` is ignored in Rust 1.25.0 and prior. When Rust 1.25.0 and prior is used Cargo may download the wrong dependency, which could be squatted on crates.io to be a malicious package. This not only affects manifests that you write locally yourself, but also manifests published to crates.io. Rust 1.0.0 through Rust 1.25.0 is affected by this advisory because Cargo will ignore the `package` key in manifests. Rust 1.26.0 through Rust 1.30.0 are not affected and typically will emit an error because the `package` key is unstable. Rust 1.31.0 and after are not affected because Cargo understands the `package` key. Users of the affected versions are strongly encouraged to update their compiler to the latest available one. Preventing this issue from happening requires updating your compiler to be either Rust 1.26.0 or newer. There will be no point release for Rust versions prior to 1.26.0. Users of Rust 1.19.0 to Rust 1.25.0 can instead apply linked patches to mitigate the issue.
CVE-2019-16144	An issue was discovered in the generator crate before 0.6.18 for Rust. Uninitialized memory is used by Scope, done, and yield_ during API calls.

Name	Description
CVE-2019-16143	An issue was discovered in the blake2 crate before 0.8.1 for Rust. The BLAKE2b and BLAKE2s algorithms, when used with HMAC, produce incorrect results because the block sizes are half of the required sizes.
CVE-2019-16142	An issue was discovered in the renderdoc crate before 0.5.0 for Rust. Multiple exposed methods take self by immutable reference, which is incompatible with a multi-threaded application.
CVE-2019-16141	An issue was discovered in the once_cell crate before 1.0.1 for Rust. There is a panic during initialization of Lazy.
CVE-2019-16140	An issue was discovered in the chttp crate before 0.1.3 for Rust. There is a use-after-free during buffer conversion.
CVE-2019-16139	An issue was discovered in the compact_arena crate before 0.4.0 for Rust. Generativity is mishandled, leading to an out-of-bounds write or read.
CVE-2019-16138	An issue was discovered in the image crate before 0.21.3 for Rust, affecting the HDR image format decoder. Vec::set_len is called on an uninitialized vector, leading to a use-after-free and arbitrary code execution.
CVE-2019-16137	An issue was discovered in the spin crate before 0.5.2 for Rust, when RwLock is used. Because memory ordering is mishandled, two writers can acquire the lock at the same time, violating mutual exclusion.
CVE-2019-15554	An issue was discovered in the smallvec crate before 0.6.10 for Rust. There is memory corruption for certain grow attempts with less than the current capacity.
CVE-2019-15553	An issue was discovered in the memoffset crate before 0.5.0 for Rust. offset_of and span_of can cause exposure of uninitialized memory.
CVE-2019-15552	An issue was discovered in the libflate crate before 0.1.25 for Rust. MultiDecoder::read has a use-after-free, leading to arbitrary code execution.
CVE-2019-15551	An issue was discovered in the smallvec crate before 0.6.10 for Rust. There is a double free for certain grow attempts with the current capacity.
CVE-2019-15550	An issue was discovered in the simd-json crate before 0.1.15 for Rust. There is an out-of-bounds read and an incorrect crossing of a page boundary.
CVE-2019-15549	An issue was discovered in the asn1_der crate before 0.6.2 for Rust. Attackers can trigger memory exhaustion by supplying a large value in a length field.
CVE-2019-15548	An issue was discovered in the ncurses crate through 5.99.0 for Rust. There are instr and mvwinstr buffer overflows because interaction with C functions is mishandled.
CVE-2019-15547	An issue was discovered in the ncurses crate through 5.99.0 for Rust. There are format string issues in printw functions because C format arguments are mishandled.
CVE-2019-15546	An issue was discovered in the pancurses crate through 0.16.1 for Rust. printw and mvprintw have format string vulnerabilities.
CVE-2019-15545	An issue was discovered in the libp2p-core crate before 0.8.1 for Rust. Attackers can spoof ed25519 signatures.
CVE-2019-15544	An issue was discovered in the protobuf crate before 2.6.0 for Rust. Attackers can exhaust all memory via Vec::reserve calls.
CVE-2019-15543	An issue was discovered in the slice-deque crate before 0.2.0 for Rust. There is memory corruption in certain allocation cases.
CVE-2019-15542	An issue was discovered in the ammonia crate before 2.1.0 for Rust. There is uncontrolled recursion during HTML DOM tree serialization.
CVE-2019-15541	rustls-mio/examples/tlsserver.rs in the rustls crate before 0.16.0 for Rust allows attackers to cause a denial of service (loop of conn_event and ready) by arranging for a client to never be writable.
CVE-2019-13225	A NULL Pointer Dereference in match_at() in regexec.c in Oniguruma 6.9.2 allows attackers to potentially cause denial of service by providing a crafted regular expression. Oniguruma issues often affect Ruby, as well as common optional libraries for PHP and Rust.
CVE-2019-13224	A use-after-free in onig_new_deluxe() in regext.c in Oniguruma 6.9.2 allows attackers to potentially cause information disclosure, denial of service, or possibly code execution by providing a crafted regular expression. The attacker provides a pair of a regex pattern and a string, with a multi-byte encoding that gets handled by onig_new_deluxe(). Oniguruma issues often affect Ruby, as well as common optional libraries for PHP and Rust.
CVE-2019-12083	The Rust Programming Language Standard Library 1.34.x before 1.34.2 contains a stabilized method which, if overridden, can violate Rust's safety guarantees and cause memory unsafety. If the `Error::type_id` method is overridden then any type can be safely cast to any other type, causing memory safety vulnerabilities in safe code (e.g., out-of-bounds write or read). Code that does not manually implement Error::type_id is unaffected.
CVE-2019-1010299	The Rust Programming Language Standard Library 1.18.0 and later is affected by: CWE-200: Information Exposure. The impact is: Contents of uninitialized memory could be printed to string or to log file. The component is: Debug trait implementation for std::collections::vec_deque::Iter. The attack vector is: The program needs to invoke debug printing for iterator over an empty VecDeque. The fixed version is: 1.30.0, nightly versions after commit b85e4cc8fadaabd41da5b9645c08c68b8f89908d.
CVE-2019-1010182	yaml-rust 0.4.0 and earlier is affected by: Uncontrolled Recursion. The impact is: Denial of service by impossible to catch abort. The component is: YamlLoader::load_from_str function. The attack vector is: Parsing of a malicious YAML document. The fixed version is: 0.4.1 and later.
CVE-2019-10052	An issue was discovered in Suricata 4.1.3. If the network packet does not have the right length, the parser tries to access a part of a DHCP packet. At this point, the Rust environment runs into a panic in parse_clientid_option in the dhcp/parser.rs file.
CVE-2018-25028	An issue was discovered in the libpulse-binding crate before 1.2.1 for Rust. get_context can cause a use-after-free.
CVE-2018-25027	An issue was discovered in the libpulse-binding crate before 1.2.1 for Rust. get_format_info can cause a use-after-free.
CVE-2018-25026	An issue was discovered in the actix-web crate before 0.7.15 for Rust. It can add the Send marker trait to an object that cannot be sent between threads safely, leading to memory corruption.
CVE-2018-25025	An issue was discovered in the actix-web crate before 0.7.15 for Rust. It can unsoundly extend the lifetime of a string, leading to memory corruption.
CVE-2018-25024	An issue was discovered in the actix-web crate before 0.7.15 for Rust. It can unsoundly coerce an immutable reference into a mutable reference, leading to memory corruption.
CVE-2018-25023	An issue was discovered in the smallvec crate before 0.6.13 for Rust. It can create an uninitialized value of any type, including a reference type.
CVE-2018-25008	In the standard library in Rust before 1.29.0, there is weak synchronization in the Arc::get_mut method. This synchronization issue can be lead to memory safety issues through race conditions.
CVE-2018-25001	An issue was discovered in the libpulse-binding crate before 2.5.0 for Rust. proplist::Iterator can cause a use-after-free.

Name	Description
CVE-2018-21000	An issue was discovered in the safe-transmute crate before 0.10.1 for Rust. A constructor's arguments are in the wrong order, causing heap memory corruption.
CVE-2018-20999	An issue was discovered in the orion crate before 0.11.2 for Rust. reset() calls cause incorrect results.
CVE-2018-20998	An issue was discovered in the arrayfire crate before 3.6.0 for Rust. Addition of the repr() attribute to an enum is mishandled, leading to memory corruption.
CVE-2018-20997	An issue was discovered in the openssl crate before 0.10.9 for Rust. A use-after-free occurs in CMS Signing.
CVE-2018-20996	An issue was discovered in the crossbeam crate before 0.4.1 for Rust. There is a double free because of destructor mishandling.
CVE-2018-20995	An issue was discovered in the slice-deque crate before 0.1.16 for Rust. move_head_unchecked allows memory corruption because deque updates are mishandled.
CVE-2018-20994	An issue was discovered in the trust-dns-proto crate before 0.5.0-alpha.3 for Rust. There is infinite recursion because DNS message compression is mishandled.
CVE-2018-20993	An issue was discovered in the yaml-rust crate before 0.4.1 for Rust. There is uncontrolled recursion during deserialization.
CVE-2018-20992	An issue was discovered in the claxon crate before 0.4.1 for Rust. Uninitialized memory can be exposed because certain decode buffer sizes are mishandled.
CVE-2018-20991	An issue was discovered in the smallvec crate before 0.6.3 for Rust. The Iterator implementation mishandles destructors, leading to a double free.
CVE-2018-20990	An issue was discovered in the tar crate before 0.4.16 for Rust. Arbitrary file overwrite can occur via a symlink or hardlink in a TAR archive.
CVE-2018-20989	An issue was discovered in the untrusted crate before 0.6.2 for Rust. Error handling can trigger an integer underflow and panic.
CVE-2018-1000810	The Rust Programming Language Standard Library version 1.29.0, 1.28.0, 1.27.2, 1.27.1, 127.0, 126.2, 126.1, 126.0 contains a CWE-680: Integer Overflow to Buffer Overflow vulnerability in standard library that can result in buffer overflow. This attack appear to be exploitable via str::repeat, passed a large number, can overflow an internal buffer. This vulnerability appears to have been fixed in 1.29.1.
CVE-2018-1000657	Rust Programming Language Rust standard library version Commit bfa0e1f58acf1c28d500c34ed258f09ae021893e and later; stable release 1.3.0 and later contains a Buffer Overflow vulnerability in std::collections::vec_deque::VecDeque::reserve() function that can result in Arbitrary code execution, but no proof-of-concept exploit is currently published.. This vulnerability appears to have been fixed in after commit fdfafb510b1a38f727e920dccbeeb638d39a8e60; stable release 1.22.0 and later.
CVE-2018-1000622	The Rust Programming Language rustdoc version Between 0.8 and 1.27.0 contains a CWE-427: Uncontrolled Search Path Element vulnerability in rustdoc plugins that can result in local code execution as a different user. This attack appear to be exploitable via using the --plugin flag without the --plugin-path flag. This vulnerability appears to have been fixed in 1.27.1.
CVE-2017-20004	In the standard library in Rust before 1.19.0, there is a synchronization problem in the MutexGuard object. MutexGuards can be used across threads with any types, allowing for memory safety issues through race conditions.
CVE-2017-18589	An issue was discovered in the cookie crate before 0.7.6 for Rust. Large integers in the Max-Age of a cookie cause a panic.
CVE-2017-18588	An issue was discovered in the security-framework crate before 0.1.12 for Rust. Hostname verification for certificates does not occur if ClientBuilder uses custom root certificates.
CVE-2017-18587	An issue was discovered in the hyper crate before 0.9.18 for Rust. It mishandles newlines in headers.
CVE-2017-1000430	rust-base64 version <= 0.5.1 is vulnerable to a buffer overflow when calculating the size of a buffer to use when encoding base64 using the 'encode_config_buf' and 'encode_config' functions
CVE-2016-10933	An issue was discovered in the portaudio crate through 0.7.0 for Rust. There is a man-in-the-middle issue because the source code is downloaded over cleartext HTTP.
CVE-2016-10932	An issue was discovered in the hyper crate before 0.9.4 for Rust on Windows. There is an HTTPS man-in-the-middle vulnerability because hostname verification was omitted.
CVE-2016-10931	An issue was discovered in the openssl crate before 0.9.0 for Rust. There is an SSL/TLS man-in-the-middle vulnerability because certificate verification is off by default and there is no API for hostname verification.
CVE-2015-20001	In the standard library in Rust before 1.2.0, BinaryHeap is not panic-safe. The binary heap is left in an inconsistent state when the comparison of generic elements inside sift_up or sift_down_range panics. This bug leads to a drop of zeroed memory as an arbitrary type, which can result in a memory safety violation.