RUSTSEC-2021-0050

swap_index can write out of bounds and return uninitialized memory

Reported	February 24, 2021
Issued	March 31, 2021 (last modified: October 19, 2021)
Package	reorder (crates.io)
Type	Vulnerability
Keywords	#memory-corruption #out-of-bounds
Aliases	CVE-2021-29941 CVE-2021-29942
Details	https://github.com/tiby312/reorder/issues/1
CVSS Score	7.3 HIGH
CVSS Details	

Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchanged
Confidentiality	Low
Integrity	Low
Availability	Low

CVSS Vector	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:L
Patched	>=1.1.0

Description

`swap_index` takes an iterator and swaps the items with their corresponding indexes. It reserves capacity and sets the length of the vector based on the `.len()` method of the iterator.

If the `len()` returned by the iterator is larger than the actual number of elements yielded, then `swap_index` creates a vector containing uninitialized members. If the `len()` returned by the iterator is smaller than the actual number of members yielded, then `swap_index` can write out of bounds past its allocated vector.

As noted by the Rust documentation, `len()` and `size_hint()` are primarily meant for optimization and incorrect values from their implementations should not lead to memory safety violations.

Patch

A new version crate was pushed that marks this function as unsafe.

reorder = "1.1.0"

Previous versions have also been yanked from crates.io.