Security Policy

Thank you for taking the time to investigate the security of Deno. The security of Deno is our topmost priority. We appreciate investigative work into system security by well-intentioned, ethical security researchers. If you discover a vulnerability, however small, we would like to know about it to address it with appropriate measures as quickly as possible. This document outlines the method we use to work with the security research community to address runtime security.

Reporting a vulnerability

Please email findings to engineering@deno.com. We strive to resolve all problems as quickly as possible, and are more than happy to play an active role in publication of writeups after the problem is resolved.

Try to include as much information as possible in the initial email, so we can quickly address the issue.

Please do not open security issues in the public issue tracker.

Please do the following

- Do not take advantage of the vulnerability or problem you have discovered.
- Do not publish or reveal the problem until it has been resolved.
- Do not use attacks on physical security or applications of third parties.
- Do provide sufficient information to reproduce the problem, so we will be able to resolve it as quickly as possible. Usually, a list of steps to follow, and the vulnerable Deno version is enough. More complex vulnerabilities may require further explanation.

Our commitment to you

- If you act in accordance with this policy, we will not take legal action against you in regard to your report.
- We will handle your report with strict confidentiality, and not pass on your personal details to third parties without your permission.

Security Model

The following paragraphs outline the rough security model for Deno. The model may change slightly over time, but in general the model is as follows:

- All JavaScript run in Deno is considered untrusted, so permissions are thus never enforced in JavaScript.
- All JavaScript run in a single Deno process is considered to be part of the same program and is not isolated from itself. This means that Deno does not guarantee that values set by one JS module will be inaccessible

to another, or that a value set in one web worker can not be accessed by another.
- All runtime I/O is considered to be privileged and must always be guarded by a runtime permission. This includes filesystem access, network access, etc.
- Users should not be able to self-escalate their permissions without explicit consent.
- I/O required to build an initial static module graph should always follow the permissions of its parent. If there is no parent, all permissions are granted. As an example, this means that the initial static module graph that is constructed when doing `deno run`, does not have any permission restrictions. However, the module graph constructed as the result of loading a web worker or dynamic import will be restricted to the permissions of the caller (the main worker most likely).
- We consider the V8 VM to be a secure sandbox.