Halborn CTF Secutiy Report

Halborn Offensive Security Engineer applying for a full time position

Meek Msaki

Version v0.1, 03.26.2024: Work in progress

Table of Contents

Introduction

Summary

This security assessment was conducted on the commit e0e91e5…ca1ee of master CTFs repo starting on March 25th, 2024 and ended on April 2nd, 2024.

Document Revisions

0.1	Draft report	03.26.2024
1.0	Final report	04.02.2024
1.1	Fixes review	-

Part I: Critical

1. C-1 btcd mishandles witness size checking

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-2chg-86hq-7w38

File HalbornCTF_Golang_Cosmos/go.mod#L59-L63

```
	github.com/breml/errchkjson v0.2.3 // indirect
	github.com/btcsuite/btcd v0.22.0-beta // indirect
	github.com/butuzov/ireturn v0.1.1 // indirect
	github.com/cenkalti/backoff/v4 v4.1.2 // indirect
	github.com/cespare/xxhash v1.1.0 // indirect
```

btcd before 0.23.2, as used in Lightning Labs lnd before 0.15.2-beta and other Bitcoin-related products, mishandles witness size checking.

1.1. Specific Go Packages Affected

github.com/btcsuite/btcd/wire

1.2. Impact: Critical 9.8 / 10

Table 1. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	High
Integrity	High
Availability	High

1.3. References

- https://nvd.nist.gov/vuln/detail/CVE-2022-44797
- https://github.com/lightningnetwork/lnd/issues/7002
- https://github.com/btcsuite/btcd/pull/1896
- https://github.com/btcsuite/btcd/releases/tag/v0.23.2
- https://github.com/lightningnetwork/lnd/releases/tag/v0.15.2-beta
- https://github.com/btcsuite/btcd/pull/1896/commits/f523d4ccaa5f34a2f761f16a05f5d6e6665b1168
- https://github.com/advisories/GHSA-2chg-86hq-7w38
- https://pkg.go.dev/vuln/GO-2022-1098

1.4. Recommendation

Consider alternatives of this dependency.

2. C-2 crossbeam-deque Data Race before v0.7.4 and v0.8.1

Tags: `runtime`, Weaknesses: CWE-362, CVE ID: CVE-2021-32810, GHSA ID: GHSA-pqqp-xmhj-wgcw

File HalbornCTF_Rust_Substrate/Cargo.lock#L1004-L1008

```
[[package]]
name = "crossbeam-deque"
version = "0.7.3"
source = "registry+https://github.com/rust-lang/crates.io-index"
checksum = "9f02af974daeee82218205558e51ec8768b48cf524bd01d550abe5573a608285"
```

In the affected version of this crate, 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.

2.1. Impact: Critical 9.8 / 10

Table 2. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	High
Integrity	High
Availability	High

2.2. Patches

This has been fixed in crossbeam-deque 0.8.1 and 0.7.4.

<RECOMMENDATION>

2.3. References

3. C-3 crossbeam-deque Data Race before v0.7.4 and v0.8.1

Tags: `runtime`, Weaknesses: CWE-362, CVE ID: CVE-2021-32810, GHSA ID: GHSA-pqqp-xmhj-wgcw

File HalbornCTF_Rust_Substrate/Cargo.lock#L1004-L1008

```
[[package]]
name = "crossbeam-deque"
version = "0.7.3"
source = "registry+https://github.com/rust-lang/crates.io-index"
checksum = "9f02af974daeee82218205558e51ec8768b48cf524bd01d550abe5573a608285"
```

In the affected version of this crate, 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.

3.1. Impact: Critical 9.8 / 10

Table 3. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	High
Integrity	High
Availability	High

3.2. Patches

This has been fixed in crossbeam-deque 0.8.1 and 0.7.4.

<RECOMMENDATION>

3.3. References

4. C-4 Overflow in libsecp256k1

Tags: runtime, Weaknesses: GHSA ID: GHSA-g4vj-x7v9-h82m CWE-190, CWE-347

File HalbornCTF_Rust_Substrate/Cargo.lock#L2798-L2802

```
"lazy_static",
"libsecp256k1",
"log",
"multihash",
"multistream-select",
```

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.

4.1. Impact: Critical 9.8 / 10

Table 4. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	High
Integrity	High
Availability	High

4.2. References

- https://nvd.nist.gov/vuln/detail/CVE-2021-38195
- https://github.com/paritytech/libsecp256k1/pull/67
- https://rustsec.org/advisories/RUSTSEC-2021-0076.html
- https://github.com/advisories/GHSA-g4vj-x7v9-h82m

4.3. Recommendation

Consider alternatives of this dependency.

5. C-5 Out of bounds write in nalgebra

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-3w8g-xr3f-2mp8 CWE-119, CWE-787

File HalbornCTF_Rust_Substrate/Cargo.lock#L3213-L3217

```
dependencies = [
 "nalgebra 0.25.4",
 "statrs",
]
```

The `Deserialize` implementation for `VecStorage` did not maintain the invariant that the number of elements must equal `nrows * ncols`. Deserialization of specially crafted inputs could allow memory access beyond allocation of the vector.

This flaw was introduced in v0.11.0 (`086e6e`) due to the addition of an automatically derived implementation of `Deserialize` for `MatrixVec`. `MatrixVec` was later renamed to `VecStorage` in v0.16.13 (`0f66403`) and continued to use the automatically derived implementation of `Deserialize`.

5.1. Impact: Critical 9.8 / 10

Table 5. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	High
Integrity	High
Availability	High

5.2. References

- https://nvd.nist.gov/vuln/detail/CVE-2021-38190
- https://github.com/dimforge/nalgebra/issues/883
- https://rustsec.org/advisories/RUSTSEC-2021-0070.html
- https://github.com/dimforge/nalgebra/pull/889
- https://github.com/dimforge/nalgebra/commit/a803271fcce75b7c151e92aa099dfa546db4adc5
- https://github.com/dimforge/nalgebra/blob/dev/CHANGELOG.md#0270
- https://github.com/advisories/GHSA-3w8g-xr3f-2mp8

5.3. Recommendation

Consider alternatives of this dependency.

6. C-6 Rust Failure Crate Vulnerable to Type confusion

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-r98r-j25q-rmpr CWE-843

File HalbornCTF_Rust_Substrate/Cargo.lock#L1408-L1412

```
[[package]]
name = "failure"
version = "0.1.8"
source = "registry+https://github.com/rust-lang/crates.io-index"
checksum = "d32e9bd16cc02eae7db7ef620b392808b89f6a5e16bb3497d159c6b92a0f4f86"
```

Safe Rust code can implement malfunctioning *`private_get_type_id`* and cause type confusion when downcasting, which is an undefined behavior.

Users who derive Fail trait are not affected.

6.1. Impact: Critical 9.8 / 10

Table 6. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	High
Integrity	High
Availability	High

6.2. References

- https://nvd.nist.gov/vuln/detail/CVE-2019-25010
- https://github.com/rust-lang-nursery/failure/issues/336
- https://rustsec.org/advisories/RUSTSEC-2019-0036.html
- https://github.com/advisories/GHSA-r98r-j25q-rmpr

6.3. Recommendation

Consider alternatives of this dependency.

7. C-7 Memory flaw in zeroize_derive

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-c5hx-w945-j4pq CWE-459

File HalbornCTF_Rust_Substrate/Cargo.lock#L8931-L8935

```
dependencies = [
 "zeroize_derive",
]

[[package]]
```

An issue was discovered in the zeroize_derive crate before 1.1.1 for Rust. Dropped memory is not zeroed out for an enum.

7.1. Impact: Critical 9.8 / 10

Table 7. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	High
Integrity	High
Availability	High

7.2. References

- https://nvd.nist.gov/vuln/detail/CVE-2021-45706
- https://raw.githubusercontent.com/rustsec/advisory-db/main/crates/zeroize_derive/RUSTSEC-2021-0115.md
- https://rustsec.org/advisories/RUSTSEC-2021-0115.html
- https://github.com/iqlusioninc/crates/issues/876
- https://github.com/advisories/GHSA-c5hx-w945-j4pq

7.3. Recommendation

Consider alternatives of this dependency.

8. C-8 Type confusion if *private_get_type_id* is overriden

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-jq66-xh47-j9f3 CWE-843

File HalbornCTF_Rust_Substrate/Cargo.lock#L1408-L1412

```
[[package]]
name = "failure"
version = "0.1.8"
source = "registry+https://github.com/rust-lang/crates.io-index"
checksum = "d32e9bd16cc02eae7db7ef620b392808b89f6a5e16bb3497d159c6b92a0f4f86"
```

An issue was discovered in the failure crate through 0.1.5 for Rust. It has a type confusion flaw when downcasting. NOTE: This vulnerability only affects products that are no longer supported by the maintainer.

8.1. Impact: Critical 9.8 / 10

Table 8. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	High
Integrity	High
Availability	High

8.2. References

- https://nvd.nist.gov/vuln/detail/CVE-2020-25575
- https://github.com/rust-lang-nursery/failure/issues/336
- https://github.com/RustCrypto/hashes/pull/91
- https://boats.gitlab.io/blog/post/failure-to-fehler/
- https://github.com/RustSec/advisory-db/blob/main/crates/failure/RUSTSEC-2019-0036.md
- https://rustsec.org/advisories/RUSTSEC-2019-0036.html
- https://rustsec.org/advisories/RUSTSEC-2020-0036.html
- https://github.com/advisories/GHSA-jq66-xh47-j9f3

8.3. Recommendation

Consider alternatives of this dependency.

9. C-9 Deserialization of Untrusted Data in rust-cpuid

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-w428-f65r-h4q2 CWE-502

File HalbornCTF_Rust_Substrate/Cargo.lock#L963-L967

```
  "cranelift-codegen",
  "raw-cpuid",
  "target-lexicon",
]
```

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.

9.1. Impact: Critical 9.8 / 10

Table 9. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	High
Integrity	High
Availability	High

9.2. References

- https://nvd.nist.gov/vuln/detail/CVE-2021-45687
- https://raw.githubusercontent.com/rustsec/advisory-db/main/crates/raw-cpuid/RUSTSEC-2021-0089.md
- https://rustsec.org/advisories/RUSTSEC-2021-0089.html
- https://github.com/gz/rust-cpuid/issues/43
- https://github.com/advisories/GHSA-w428-f65r-h4q2

9.3. Recommendation

Consider alternatives of this dependency.

Part II: High

10. H-1 Uncontrolled Resource Consumption

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-vvpx-j8f3-3w6h CWE-400

File HalbornCTF_Golang_Cosmos/go.mod#L246-L250

```
    golang.org/x/mod v0.6.0-dev.0.20220106191415-9b9b3d81d5e3 // indirect
    golang.org/x/net v0.0.0-20211208012354-db4efeb81f4b // indirect
    golang.org/x/sync v0.0.0-20210220032951-036812b2e83c // indirect
    golang.org/x/sys v0.0.0-20220209214540-3681064d5158 // indirect
    golang.org/x/term v0.0.0-20201126162022-7de9c90e9dd1 // indirect
```

A maliciously crafted HTTP/2 stream could cause excessive CPU consumption in the HPACK decoder, sufficient to cause a denial of service from a small number of small requests.

10.1. Impact: High 7.5 / 10

Table 10. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	High

10.2. References

- https://nvd.nist.gov/vuln/detail/CVE-2022-41723
- https://go.dev/cl/468135
- https://go.dev/issue/57855
- https://groups.google.com/g/golang-announce/c/V0aBFqaFs_E
- https://vuln.go.dev/ID/GO-2023-1571.json
- https://go.dev/cl/468295
- https://pkg.go.dev/vuln/GO-2023-1571
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/4MA5XS5DAOJ5PKKNG5TUXKPQOFHT5VBC/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/RLBQ3A7ROLEQXQLXFDLNJ7MYPKG5GULE/

- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/RGW7GE2Z32ZT47UFAQFDRQE33B7Q7LMT/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/XX3IMUTZKRQ73PBZM4E2JP4BKYH4C6XE/
- https://www.couchbase.com/alerts/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/4BUK2ZIAGCULOOYDNH25JPU6JBES5NF2/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/REMHVVIBDNKSRKNOTV7EQSB7CYQWOUOU/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/T7N5GV4CHH6WAGX3GFMDD3COEOVCZ4RI/
- https://security.gentoo.org/glsa/202311-09
- https://github.com/advisories/GHSA-vvpx-j8f3-3w6h

10.3. Recommendation

Consider alternatives of this dependency.

11. H-2 golang.org/x/crypto/ssh Denial of service via crafted Signer

Tags: runtime, Weaknesses: GHSA ID: GHSA-8c26-wmh5-6g9v CWE-327

File HalbornCTF_Golang_Cosmos/go.mod#L244-L248

```
    go.etcd.io/bbolt v1.3.6 // indirect
    golang.org/x/crypto v0.0.0-20220214200702-86341886e292 // indirect
    golang.org/x/mod v0.6.0-dev.0.20220106191415-9b9b3d81d5e3 // indirect
    golang.org/x/net v0.0.0-20211208012354-db4efeb81f4b // indirect
    golang.org/x/sync v0.0.0-20210220032951-036812b2e83c // indirect
```

The golang.org/x/crypto/ssh package before 0.0.0-20220314234659-1baeb1ce4c0b for Go allows an attacker to crash a server in certain circumstances involving AddHostKey.

11.1. Impact: High 7.5 / 10

Table 11. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	High

11.2. References

- https://nvd.nist.gov/vuln/detail/CVE-2022-27191
- https://groups.google.com/g/golang-announce
- https://groups.google.com/g/golang-announce/c/-cp44ypCT5s
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/HHGBEGJ54DZZGTXFUQNS7ZIG3E624YAF/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/QTFOIDHQRGNI4P6LYN6ILH5G443RYYKB/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/YHYRQB7TRMHDB3NEHW5XBRG7PPMUTPGV/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/ZQNPPQWSTP2IX7SHE6TS4SP4EVMI5EZK/

- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/J5WPM42UR6XIBQNQPNQHM32X7S4LJTRX/
- https://security.netapp.com/advisory/ntap-20220429-0002/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/EZ3S7LB65N54HXXBCB67P4TTOHTNPP5O/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/ZFUNHFHQVJSADNH7EZ3B53CYDZVEEPBP/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/DLUJZV3HBP56ADXU6QH2V7RNYUPMVBXQ/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/ZY2SLWOQR4ZURQ7UBRZ7JIX6H6F5JHJR/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/Z55VUVGO7E5PJFXIOVAY373NZRHBNCI5/
- https://raw.githubusercontent.com/golang/vulndb/df2d3d326300e2ae768f00351ffa96cc2c56cf54/reports/GO-2021-0356.yaml
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/RQXU752ALW53OJAF5MG3WMR5CCZVLWW6/
- https://go.dev/cl/392355
- https://go.googlesource.com/crypto/+/1baeb1ce4c0b006eff0f294c47cb7617598dfb3d
- https://pkg.go.dev/vuln/GO-2021-0356
- https://github.com/advisories/GHSA-8c26-wmh5-6g9v

11.3. Recommendation

Consider alternatives of this dependency.

12. H-3 golang.org/x/net/http2 Denial of Service vulnerability

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-69cg-p879-7622

File HalbornCTF_Golang_Cosmos/go.mod#L246-L250

```
    golang.org/x/mod v0.6.0-dev.0.20220106191415-9b9b3d81d5e3 // indirect
    golang.org/x/net v0.0.0-20211208012354-db4efeb81f4b // indirect
    golang.org/x/sync v0.0.0-20210220032951-036812b2e83c // indirect
    golang.org/x/sys v0.0.0-20220209214540-3681064d5158 // indirect
    golang.org/x/term v0.0.0-20201126162022-7de9c90e9dd1 // indirect
```

In net/http in Go before 1.18.6 and 1.19.x before 1.19.1, attackers can cause a denial of service because an HTTP/2 connection can hang during closing if shutdown were preempted by a fatal error.

12.1. Impact: High 7.5 / 10

Table 12. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	High

12.2. References

- https://nvd.nist.gov/vuln/detail/CVE-2022-27664
- https://groups.google.com/g/golang-announce
- https://groups.google.com/g/golang-announce/c/x49AQzIVX-s
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/TXS2OQ57KZC5XZKK5UW4SYKPVQAHIOJX/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/JXKTHIGE5F576MAPFYCIJXNRGBSPISUF/
- https://security.gentoo.org/glsa/202209-26
- https://security.netapp.com/advisory/ntap-20220923-0004/
- https://pkg.go.dev/vuln/GO-2022-0969

- https://go.dev/cl/428735
- https://go.dev/issue/54658
- https://github.com/advisories/GHSA-69cg-p879-7622

12.3. Recommendation

Consider alternatives of this dependency.

13. H-4 golang.org/x/text/language Denial of service via crafted Accept-Language header

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-69ch-w2m2-3vjp CWE-772

File HalbornCTF_Golang_Cosmos/go.mod#L250-L254

```
	golang.org/x/term v0.0.0-20201126162022-7de9c90e9dd1 // indirect
	golang.org/x/text v0.3.7 // indirect
	golang.org/x/tools v0.1.10 // indirect
	golang.org/x/xerrors v0.0.0-20200804184101-5ec99f83aff1 // indirect
	gopkg.in/ini.v1 v1.66.2 // indirect
```

The BCP 47 tag parser has quadratic time complexity due to inherent aspects of its design. Since the parser is, by design, exposed to untrusted user input, this can be leveraged to force a program to consume significant time parsing Accept-Language headers. The parser cannot be easily rewritten to fix this behavior for various reasons. Instead the solution implemented in this CL is to limit the total complexity of tags passed into ParseAcceptLanguage by limiting the number of dashes in the string to 1000. This should be more than enough for the majority of real world use cases, where the number of tags being sent is likely to be in the single digits.

13.1. Specific Go Packages Affected

golang.org/x/text/language

13.2. Impact: High 7.5 / 10

Table 13. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	High

13.3. References

- https://nvd.nist.gov/vuln/detail/CVE-2022-32149
- https://go.dev/cl/442235
- https://go.dev/issue/56152

- https://groups.google.com/g/golang-announce/c/-hjNw559_tE/m/KlGTfid5CAAJ
- https://pkg.go.dev/vuln/GO-2022-1059
- https://github.com/golang/go/issues/56152
- https://github.com/golang/text/commit/434eadcdbc3b0256971992e8c70027278364c72c
- https://github.com/advisories/GHSA-69ch-w2m2-3vjp

13.4. Recommendation

Consider alternatives of this dependency.

14. H-5 golang.org/x/net/http2/h2c vulnerable to request smuggling attack

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-fxg5-wq6x-vr4w CWE-444

File HalbornCTF_Golang_Cosmos/go.mod#L246-L250

```
    golang.org/x/mod v0.6.0-dev.0.20220106191415-9b9b3d81d5e3 // indirect
    golang.org/x/net v0.0.0-20211208012354-db4efeb81f4b // indirect
    golang.org/x/sync v0.0.0-20210220032951-036812b2e83c // indirect
    golang.org/x/sys v0.0.0-20220209214540-3681064d5158 // indirect
    golang.org/x/term v0.0.0-20201126162022-7de9c90e9dd1 // indirect
```

A request smuggling attack is possible when using MaxBytesHandler. When using MaxBytesHandler, the body of an HTTP request is not fully consumed. When the server attempts to read HTTP2 frames from the connection, it will instead be reading the body of the HTTP request, which could be attacker-manipulated to represent arbitrary HTTP2 requests.

14.1. Specific Go Packages Affected

golang.org/x/net/http2/h2c

14.2. Impact: High 7.5 / 10

Table 14. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	High

14.3. References

- https://nvd.nist.gov/vuln/detail/CVE-2022-41721
- https://go.dev/cl/447396
- https://go.dev/issue/56352
- https://pkg.go.dev/vuln/GO-2023-1495
- https://lists.fedoraproject.org/archives/list/package-

announce@lists.fedoraproject.org/message/X3H3EWQXM2XL5AGBX6UL443JEJ3GQXJN/

- https://github.com/advisories/GHSA-fxg5-wq6x-vr4w

14.4. Recommendation

Consider alternatives of this dependency.

15. H-6 Opencontainers runc Incorrect Authorization vulnerability

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-vpvm-3wq2-2wvm CWE-706

File HalbornCTF_Golang_Cosmos/go.mod#L188-L192

```
    github.com/opencontainers/image-spec v1.0.2 // indirect
    github.com/opencontainers/runc v1.0.3 // indirect
    github.com/pelletier/go-toml v1.9.4 // indirect
    github.com/petermattis/goid v0.0.0-20180202154549-b0b1615b78e5 // indirect
    github.com/phayes/checkstyle v0.0.0-20170904204023-bfd46e6a821d // indirect
```

runc 1.0.0-rc95 through 1.1.4 has Incorrect Access Control leading to Escalation of Privileges, related to `libcontainer/rootfs_linux.go`. To exploit this, an attacker must be able to spawn two containers with custom volume-mount configurations, and be able to run custom images. NOTE: this issue exists because of a CVE-2019-19921 regression.

15.1. Impact: High 7.0 / 10

Table 15. CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:H/I:H/A:H

CVSS base metrics	
Attack vector	Local
Attack complexity	High
Privileges required	Low
User interaction	None
Scope	Unchange
Confidentiality	High
Integrity	High
Availability	High

15.2. References

- https://nvd.nist.gov/vuln/detail/CVE-2023-27561
- https://github.com/opencontainers/runc/issues/2197#issuecomment-1437617334
- https://github.com/opencontainers/runc/issues/3751
- https://gist.github.com/LiveOverflow/c937820b688922eb127fb760ce06dab9
- https://lists.debian.org/debian-lts-announce/2023/03/msg00023.html
- https://github.com/opencontainers/runc/pull/3785
- https://github.com/opencontainers/runc/releases/tag/v1.1.5

- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/DHGVGGMKGZSJ7YO67TGGPFEHBYMS63VF/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/FYVE3GB4OG3BNT5DLQHYO4M5SXX33AQ5/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/I6BF24VCZRFTYBTT3T7HDZUOTKOTNPLZ/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/ANUGDBJ7NBUMSUFZUSKU3ZMQYZ2Z3STN/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/FNB2UEDIIJCRQW4WJLZOPQJZXCVSXMLD/
- https://github.com/advisories/GHSA-vpvm-3wq2-2wvm

15.3. Recommendation

Consider alternatives of this dependency.

16. H-7 Docker Swarm encrypted overlay network may be unauthenticated

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-232p-vwff-86mp CWE-420, CWE-636

File HalbornCTF_Golang_Cosmos/go.mod#L83-L87

```
	github.com/docker/cli v20.10.11+incompatible // indirect
	github.com/docker/docker v20.10.7+incompatible // indirect
	github.com/docker/go-connections v0.4.0 // indirect
	github.com/docker/go-units v0.4.0 // indirect
	github.com/dustin/go-humanize v1.0.0 // indirect
```

Moby is an open source container framework developed by Docker Inc. that is distributed as Docker, Mirantis Container Runtime, and various other downstream projects/products. The Moby daemon component (`dockerd`), which is developed as moby/moby is commonly referred to as **Docker**.

Swarm Mode, which is compiled in and delivered by default in `dockerd` and is thus present in most major Moby downstreams, is a simple, built-in container orchestrator that is implemented through a combination of SwarmKit and supporting network code.

The `overlay` network driver is a core feature of Swarm Mode, providing isolated virtual LANs that allow communication between containers and services across the cluster. This driver is an implementation/user of VXLAN, which encapsulates link-layer (Ethernet) frames in UDP datagrams that tag the frame with a VXLAN Network ID (VNI) that identifies the originating overlay network. In addition, the overlay network driver supports an optional, off-by-default encrypted mode, which is especially useful when VXLAN packets traverses an untrusted network between nodes.

Encrypted overlay networks function by encapsulating the VXLAN datagrams through the use of the IPsec Encapsulating Security Payload protocol in Transport mode. By deploying IPSec encapsulation, encrypted overlay networks gain the additional properties of source authentication through cryptographic proof, data integrity through check-summing, and confidentiality through encryption.

When setting an endpoint up on an encrypted overlay network, Moby installs three iptables (Linux kernel firewall) rules that enforce both incoming and outgoing IPSec. These rules rely on the `u32` iptables extension provided by the `xt_u32` kernel module to directly filter on a VXLAN packet's VNI field, so that IPSec guarantees can be enforced on encrypted overlay networks without interfering with other overlay networks or other users of VXLAN.

Two iptables rules serve to filter incoming VXLAN datagrams with a VNI that corresponds to an encrypted network and discards unencrypted datagrams. The rules are appended to the end of the `INPUT` filter chain, following any rules that have been previously set by the system administrator. Administrator-set rules take precedence over the rules Moby sets to discard unencrypted VXLAN datagrams, which can potentially admit unencrypted datagrams that should have been discarded.

On Red Hat Enterprise Linux and derivatives such as CentOS and Rocky, the `xt_u32` module has been: * moved to the kernel-modules-extra package and no longer installed by default in RHEL 8.3 * officially deprecated in RHEL 8.6 * removed completely in RHEL 9

These rules are not created when `xt_u32` is unavailable, even though the container is still attached to the network.

16.1. Impact

Encrypted overlay networks on affected configurations silently accept cleartext VXLAN datagrams that are tagged with the VNI of an encrypted overlay network. As a result, it is possible to inject arbitrary Ethernet frames into the encrypted overlay network by encapsulating them in VXLAN datagrams.

The injection of arbitrary Ethernet frames can enable a Denial of Service attack. A sophisticated attacker may be able to establish a UDP or TCP connection by way of the container's outbound gateway that would otherwise be blocked by a stateful firewall, or carry out other escalations beyond simple injection by smuggling packets into the overlay network.

16.2. Patches

Patches are available in Moby releases 23.0.3, and 20.10.24. As Mirantis Container Runtime's 20.10 releases are numbered differently, users of that platform should update to 20.10.16.

16.3. Workarounds

- Close the VXLAN port (by default, UDP port 4789) to incoming traffic at the Internet boundary (see GHSA-vwm3-crmr-xfxw) to prevent all VXLAN packet injection.
- Ensure that the `xt_u32` kernel module is available on all nodes of the Swarm cluster.

16.4. Background

- #43382 partially discussed this concern, but did not consider the security implications.
- Mirantis FIELD-5788 essentially duplicates #43382, and was created six months earlier; it similarly overlooked the security implications.
- #45118 is the ancestor of the final patches, and was where the security implications were discovered.

16.5. Related

- CVE-2023-28841: Encrypted overlay network traffic may be unencrypted
- CVE-2023-28842: Encrypted overlay network with a single endpoint is unauthenticated
- GHSA-vwm3-crmr-xfxw: The Swarm VXLAN port may be exposed to attack due to ambiguous documentation
- GHSA-gvm4-2qqg-m333: Security issues in encrypted overlay networks (libnetwork)

16.6. Impact: High 7.5 / 10

Table 16. CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:C/C:H/I:N/A:L

CVSS base metrics	
Attack vector	Network
Attack complexity	High
Privileges required	None
User interaction	None

CVSS base metrics	
Scope	Changed
Confidentiality	High
Integrity	None
Availability	Low

16.7. References

- https://github.com/moby/libnetwork/security/advisories/GHSA-gvm4-2qqg-m333
- https://github.com/moby/moby/security/advisories/GHSA-232p-vwff-86mp
- https://github.com/moby/moby/security/advisories/GHSA-33pg-m6jh-5237
- https://github.com/moby/moby/security/advisories/GHSA-6wrf-mxfj-pf5p
- https://github.com/moby/moby/security/advisories/GHSA-vwm3-crmr-xfxw
- https://github.com/moby/moby/issues/43382
- https://github.com/moby/moby/pull/45118
- https://nvd.nist.gov/vuln/detail/CVE-2023-28840
- https://github.com/advisories/GHSA-232p-vwff-86mp

16.8. Recommendation

Consider alternatives of this dependency.

17. H-8 gopkg.in/yaml.v3 Denial of Service

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-hp87-p4gw-j4gq CWE-502

File HalbornCTF_Golang_Cosmos/go.mod#L254-L258

```
	gopkg.in/ini.v1 v1.66.2 // indirect
	gopkg.in/yaml.v3 v3.0.0-20210107192922-496545a6307b // indirect
	honnef.co/go/tools v0.2.2 // indirect
	mvdan.cc/gofumpt v0.3.0 // indirect
	mvdan.cc/interfacer v0.0.0-20180901003855-c20040233aed // indirect
```

An issue in the Unmarshal function in Go-Yaml v3 can cause a program to panic when attempting to deserialize invalid input.

17.1. Impact: High 7.5 / 10

Table 17. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	High

17.2. References

- https://nvd.nist.gov/vuln/detail/CVE-2022-28948
- https://github.com/go-yaml/yaml/issues/666
- https://github.com/go-yaml/yaml/commit/8f96da9f5d5eff988554c1aae1784627c4bf6754
- https://security.netapp.com/advisory/ntap-20220923-0006/
- https://github.com/advisories/GHSA-hp87-p4gw-j4gq

17.3. Recommendation

Consider alternatives of this dependency.

18. H-9 HTTP/2 rapid reset can cause excessive work in net/http

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-4374-p667-p6c8 CWE-400, CWE-770

File HalbornCTF_Golang_Cosmos/go.mod#L246-L250

```
    golang.org/x/mod v0.6.0-dev.0.20220106191415-9b9b3d81d5e3 // indirect
    golang.org/x/net v0.0.0-20211208012354-db4efeb81f4b // indirect
    golang.org/x/sync v0.0.0-20210220032951-036812b2e83c // indirect
    golang.org/x/sys v0.0.0-20220209214540-3681064d5158 // indirect
    golang.org/x/term v0.0.0-20201126162022-7de9c90e9dd1 // indirect
```

A malicious HTTP/2 client which rapidly creates requests and immediately resets them can cause excessive server resource consumption. While the total number of requests is bounded by the http2.Server.MaxConcurrentStreams setting, resetting an in-progress request allows the attacker to create a new request while the existing one is still executing.

With the fix applied, HTTP/2 servers now bound the number of simultaneously executing handler goroutines to the stream concurrency limit (MaxConcurrentStreams). New requests arriving when at the limit (which can only happen after the client has reset an existing, in-flight request) will be queued until a handler exits. If the request queue grows too large, the server will terminate the connection.

This issue is also fixed in golang.org/x/net/http2 for users manually configuring HTTP/2.

The default stream concurrency limit is 250 streams (requests) per HTTP/2 connection. This value may be adjusted using the golang.org/x/net/http2 package; see the Server.MaxConcurrentStreams setting and the ConfigureServer function.

18.1. Impact: High 7.5 / 10

Table 18. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	High

18.2. References

- https://github.com/golang/go/issues/63417
- https://go.dev/cl/534215
- https://go.dev/cl/534235
- https://go.dev/issue/63417
- https://groups.google.com/g/golang-announce/c/iNNxDTCjZvo/m/UDd7VKQuAAAJ
- https://nvd.nist.gov/vuln/detail/CVE-2023-39325
- https://pkg.go.dev/vuln/GO-2023-2102
- https://security.gentoo.org/glsa/202311-09
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/3OVW5V2DM5K5IC3H7O42YDUGNJ74J35O
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/3SZN67IL7HMGMNAVLOTIXLIHUDXZK4LH
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/4BUK2ZIAGCULOOYDNH25JPU6JBES5NF2
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/5RSKA2II6QTD4YUKUNDVJQSRYSFC4VFR
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/AVZDNSMVDAQJ64LJC5I5U5LDM5753647
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/CHHITS4PUOZAKFIUBQAQZC7JWXMOYE4B
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/CLB4TW7KALB3EEQWNWCN7OUIWWVWWCG2
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/D2BBIDR2ZMB3X5BC7SR4SLQMHRMVPY6L
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/ECRC75BQJP6FJN2L7KCKYZW4DSBD7QSD
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/FTMJ3NJIDAZFWJQQSP3L22MUFJ3UP2PT
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/GSY7SXFFTPZFWDM6XELSDSHZLVW3AHK7
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/HZQIELEIRSZUYTFFH5KTH2YJ4IIQG2KE
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/IPWCNYB5PQ5PCVZ4NJT6G56ZYFZ5QBU6
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/KEOTKBUPZXHE3F352JBYNTSNRXYLWD6P
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/KSEGD2IWKNUO3DWY4KQGUQM5BISRWHQE
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/MZQYOOKHQDQ57LV2IAG6NRFOVXKHJJ3Z
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/NG7IMPL55MVWU3LCI4JQJT3K2U5CHDV7

- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/OXGWPQOJ3JNDW2XIYKIVJ7N7QUIFNM2Q
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/PJCUNGIQDUMZ4Z6HWVYIMR66A35F5S74
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/QF5QSYAOPDOWLY6DUHID56Q4HQFYB45I
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/R3UETKPUB3V5JS5TLZOF3SMTGT5K5APS
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/REMHVVIBDNKSRKNOTV7EQSB7CYQWOUOU
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/T7N5GV4CHH6WAGX3GFMDD3COEOVCZ4RI
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/ULQQONMSCQSH5Z5OWFFQHCGEZ3NL4DRJ
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/UTT7DG3QOF5ZNJLUGHDNLRUIN6OWZARP
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/W2LZSWTV4NV4SNQARNXG5T6LRHP26EW2
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/WCNCBYKZXLDFGAJUB7ZP5VLC3YTHJNVH
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/XFOIBB4YFICHDM7IBOP7PWXW3FX4HLL2
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/XTNLSL44Y5FB6JWADSZH6DCV4JJAAEQY
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/YRKEXKANQ7BKJW2YTAMP625LJUJZLJ4P
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/ZSVEMQV5ROY5YW5QE3I57HT3ITWG5GCV
- https://security.netapp.com/advisory/ntap-20231110-0008
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/L5E5JSJBZLYXOTZWXHJKRVCIXIHVWKJ6
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/YJWHBLVZDM5KQSDFRBFRKU5KSSOLIRQ4
- https://github.com/advisories/GHSA-4374-p667-p6c8

18.3. Recommendation

Consider alternatives of this dependency.

19. H-10 gRPC-Go HTTP/2 Rapid Reset vulnerability

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-m425-mq94-257g

File HalbornCTF_Golang_Cosmos/go.mod#L27-L31

```
    google.golang.org/genproto v0.0.0-20211223182754-3ac035c7e7cb
    google.golang.org/grpc v1.45.0
    google.golang.org/protobuf v1.27.1
    gopkg.in/yaml.v2 v2.4.0
)
```

19.1. Impact

In affected releases of gRPC-Go, it is possible for an attacker to send HTTP/2 requests, cancel them, and send subsequent requests, which is valid by the HTTP/2 protocol, but would cause the gRPC-Go server to launch more concurrent method handlers than the configured maximum stream limit.

19.2. Patches

This vulnerability was addressed by #6703 and has been included in patch releases: 1.56.3, 1.57.1, 1.58.3. It is also included in the latest release, 1.59.0.

Along with applying the patch, users should also ensure they are using the `grpc.MaxConcurrentStreams` server option to apply a limit to the server's resources used for any single connection.

19.3. Workarounds

None.

19.4. References

#6703

19.5. Impact: High 7.5 / 10

Table 19. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange

CVSS base metrics	
Confidentiality	None
Integrity	None
Availability	High

19.6. References

- https://github.com/grpc/grpc-go/security/advisories/GHSA-m425-mq94-257g
- https://github.com/grpc/grpc-go/pull/6703
- https://github.com/grpc/grpc-go/commit/f2180b4d5403d2210b30b93098eb7da31c05c721
- https://github.com/advisories/GHSA-m425-mq94-257g

19.7. Recommendation

Consider alternatives of this dependency.

20. H-11 runc vulnerable to container breakout through process.cwd trickery and leaked fds

Tags: `runtime`, Weaknesses: CWE-403, CWE-668, CVE ID: CVE-2024-21626, GHSA ID: GHSA-xr7r-f8xq-vfvv

File HalbornCTF_Golang_Cosmos/go.mod#L188-L192

```
    github.com/opencontainers/image-spec v1.0.2 // indirect
    github.com/opencontainers/runc v1.0.3 // indirect
    github.com/pelletier/go-toml v1.9.4 // indirect
    github.com/petermattis/goid v0.0.0-20180202154549-b0b1615b78e5 // indirect
    github.com/phayes/checkstyle v0.0.0-20170904204023-bfd46e6a821d // indirect
```

In runc 1.1.11 and earlier, due to an internal file descriptor leak, an attacker could cause a newly-spawned container process (from `runc exec`) to have a working directory in the host filesystem namespace, allowing for a container escape by giving access to the host filesystem ("attack 2"). The same attack could be used by a malicious image to allow a container process to gain access to the host filesystem through `runc run` ("attack 1"). Variants of attacks 1 and 2 could be also be used to overwrite semi-arbitrary host binaries, allowing for complete container escapes ("attack 3a" and "attack 3b").

Strictly speaking, while attack 3a is the most severe from a CVSS perspective, attacks 2 and 3b are arguably more dangerous in practice because they allow for a breakout from inside a container as opposed to requiring a user execute a malicious image. The reason attacks 1 and 3a are scored higher is because being able to socially engineer users is treated as a given for UI:R vectors, despite attacks 2 and 3b requiring far more minimal user interaction (just reasonable `runc exec` operations on a container the attacker has access to). In any case, all four attacks can lead to full control of the host system.

20.1. Impact: High 8.6 / 10

Table 20. CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H

CVSS base metrics	
Attack vector	Local
Attack complexity	Low
Privileges required	None
User interaction	Required
Scope	Changed
Confidentiality	High
Integrity	High
Availability	High

20.2. Patches

<RECOMMENDATION>

20.3. References

21. H-12 Memory access due to code generation flaw in Cranelift module

Tags: runtime, Weaknesses: GHSA ID: GHSA-hpqh-2wqx-7qp5 CWE-125, CWE-788

File HalbornCTF_Rust_Substrate/Cargo.lock#L900-L904

```
[[package]]
name = "cranelift-codegen"
version = "0.69.0"
source = "registry+https://github.com/rust-lang/crates.io-index"
checksum =
"1a54e4beb833a3c873a18a8fe735d73d732044004c7539a072c8faa35ccb0c60"
```

There is a bug in 0.73.0 of the Cranelift x64 backend that can create a scenario that could result in a potential sandbox escape in a WebAssembly module. Users of versions 0.73.0 of Cranelift should upgrade to either 0.73.1 or 0.74 to remediate this vulnerability. Users of Cranelift prior to 0.73.0 should update to 0.73.1 or 0.74 if they were not using the old default backend.

21.1. Description

This bug was introduced in the new backend on 2020-09-08 and first included in a release on 2020-09-30, but the new backend was not the default prior to 0.73.0. The recently-released version 0.73.0 with default settings, and prior versions with an explicit build flag to select the new backend, are vulnerable. The bug in question performs a sign-extend instead of a zero-extend on a value loaded from the stack, under a specific set of circumstances. If those circumstances occur, the bug could allow access to memory addresses up to 2GiB before the start of the heap allocated for the WebAssembly module.

If the heap bound is larger than 2GiB, then it would be possible to read memory from a computable range dependent on the size of the heap's bound.

The impact of this bug is highly dependent on heap implementation; specifically: * if the heap has bounds checks, and * does not rely exclusively on guard pages, and * the heap bound is 2GiB or smaller

then this bug cannot be used to reach memory from another WebAssembly module heap.

The impact of the vulnerability is mitigated if there is no memory mapped in the range accessible using this bug, for example, if there is a 2 GiB guard region before the WebAssembly module heap.

The bug in question performs a sign-extend instead of a zero-extend on a value loaded from the stack when the register allocator reloads a spilled integer value narrower than 64 bits. This interacts poorly with another optimization: the instruction selector elides a 32-to-64-bit zero-extend operator when we know that an instruction producing a 32-bit value actually zeros the upper 32 bits of its destination register. Hence, we rely on these zeroed bits, but the type of the value is still i32, and the spill/reload reconstitutes those bits as the sign extension of the i32's MSB.

The issue would thus occur when: * An i32 value is greater than or equal to 0x8000_0000; * The value is spilled and reloaded by the register allocator due to high register pressure in the program between the value's definition and its use; * The value is produced by an instruction that we know to be "special" in that it zeroes the upper 32 bits of its destination: add, sub, mul, and, or; * The value is then zero-extended to 64 bits; * The resulting 64-bit value is used.

Under these circumstances there is a potential sandbox escape when the i32 value is a pointer. The usual code emitted for heap accesses zero-extends the WebAssembly heap address, adds it to a 64-bit heap base, and accesses the resulting address. If the zero-extend becomes a sign-extend, the module could reach backward and access memory up to 2GiB before the start of its heap.

This bug was identified by developers at Fastly following a report from Javier Cabrera Arteaga, KTH Royal Institute of Technology, with support from project Trustful of Stiftelsen för Strategisk Forskning. In addition to supporting the analysis and remediation of this vulnerability, Fastly will publish a related Fastly Security Advisory at https://www.fastly.com/security-advisories.

In addition to assessing the nature of the code generation bug in Cranelift, we have also determined that under specific circumstances, both Lucet and Wasmtime using this version of Cranelift may be exploitable.

21.2. General Impact to Lucet

Lucet inherits the heap address computation and bounds-checks of Cranelift, which it uses as its backend code generator. Of particular importance specifically is the address-space layout used by Lucet. In the default configuration for Lucet, only a single module is running, and therefore it is not possible to access memory from another module.

By default, the open source implementation of Lucet uses a maximum heap size of 4 GiB, and an instance slot size of 8 GiB, when invoking an instance from the lucet-wasi command-line tool. These settings are within the range of vulnerability described above, but only a single instance is running, so there is no other instance to read. When embedding the runtime (for example, in a long-running daemon), the default for the heap size as described in the source is 1MB; with this setting, the runtime is not vulnerable.

Lucet allocates its WebAssembly module instances into "instance slots", which are contiguous zones of virtual address space that contain the VM context at the bottom, the WebAssembly heap in the next page after that, a guard region in the middle, and other data at the top: the stack and the globals.

If the instance slot size is less than (max heap) + 2GiB, then the lowest accessible address using the bug will overlap with the prior instance’s heap. If the size of VM context + stack + globals is greater than (4GiB - heap limit), then the highest accessible address using the bug will overlap with this critical data. If neither of these conditions are true, the bug should only result in an access to the prior instance’s guard region.

Generally, if the limit is between 2GiB and 4GiB - ~1MB (depending on stack/global size) and the instance slot size is less than 6GiB, the configuration is vulnerable. If the limit is greater than 4GiB - ~1MB, the configuration is vulnerable regardless of instance slot size. Otherwise, the configuration is not vulnerable.

21.3. General Impact on Wasmtime

In Wasmtime, the same Cranelift heap address computations and heap types are used as above. The memory layout, however, is slightly different, with different outcomes: * With the mmap implementation impact is mitigated probabilistically if ASLR is enabled. * With the pooling allocator, the vulnerability only exists if a memory reservation size lower than the default of 6GB is used.

With the default mmap-based instance memory implementation, Wasmtime uses mmap() to allocate a block of memory large enough for the heap and guard region, as specified in its configuration. If the underlying OS implements ASLR (modern Linux, macOS and Windows do) then this address will be randomized, and the region below it will (probabilistically) be free. Hence, the bug is mitigated probabilistically in the default configuration if ASLR is enabled.

If using the pooling allocator, the vulnerability exists if instance memory size (`memory_reservation_size` in InstanceLimit) is strictly less than 6GiB (4 GiB + 2 GiB of guard pages). The default is 6GiB, so the vulnerability is masked in the default pooling allocator configuration.

21.4. Impact: High 7.2 / 10

Table 21. CVSS:3.1/AV:L/AC:H/PR:L/UI:R/S:C/C:H/I:H/A:N

CVSS base metrics	
Attack vector	Local
Attack complexity	High
Privileges required	Low
User interaction	Required
Scope	Changed
Confidentiality	High
Integrity	High
Availability	None

21.5. References

- https://github.com/bytecodealliance/wasmtime/security/advisories/GHSA-hpqh-2wqx-7qp5
- https://nvd.nist.gov/vuln/detail/CVE-2021-32629
- https://github.com/bytecodealliance/wasmtime/commit/95559c01aaa7c061088a433040f31e8291fb09d0
- https://crates.io/crates/cranelift-codegen
- https://www.fastly.com/security-advisories/memory-access-due-to-code-generation-flaw-in-cranelift-module
- https://github.com/RustSec/advisory-db/blob/main/crates/cranelift-codegen/RUSTSEC-2021-0067.md
- https://rustsec.org/advisories/RUSTSEC-2021-0067.html
- https://github.com/advisories/GHSA-hpqh-2wqx-7qp5

21.6. Recommendation

Consider alternatives of this dependency.

22. H-13 Overflow in prost-types

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-x4qm-mcjq-v2gf CWE-120, CWE-190

File HalbornCTF_Rust_Substrate/Cargo.lock#L4917-L4921

```
 "prost",
 "prost-types",
 "tempfile",
 "which",
]
```

Affected versions of this crate contained a bug in which untrusted input could cause an overflow and panic when converting a Timestamp to SystemTime. It is recommended to upgrade to prost-types v0.8 and switch the usage of From<Timestamp> for SystemTime to TryFrom<Timestamp> for SystemTime.

22.1. Impact: High 7.5 / 10

Table 22. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	High

22.2. References

- https://nvd.nist.gov/vuln/detail/CVE-2021-38192
- https://github.com/tokio-rs/prost/issues/438
- https://rustsec.org/advisories/RUSTSEC-2021-0073.html
- https://github.com/tokio-rs/prost/pull/439
- https://github.com/tokio-rs/prost/commit/59f2a7311dd6540696bfd0145f5281ce495f4385
- https://github.com/advisories/GHSA-x4qm-mcjq-v2gf

22.3. Recommendation

Consider alternatives of this dependency.

23. H-14 Soundness issue in raw-cpuid

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-hvqc-pc78-x9wh CWE-198, CWE-400

File HalbornCTF_Rust_Substrate/Cargo.lock#L963-L967

```
 "cranelift-codegen",
 "raw-cpuid",
 "target-lexicon",
]
```

VendorInfo::as_string(), SoCVendorBrand::as_string(), and ExtendedFunctionInfo::processor_brand_string() construct byte slices using std::slice::from_raw_parts(), with data coming from #[repr(Rust)] structs. This is always undefined behavior. This flaw has been fixed in v9.0.0, by making the relevant structs #[repr©].

23.1. Impact: High 7.5 / 10

Table 23. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	High

23.2. References

- https://nvd.nist.gov/vuln/detail/CVE-2021-26306
- https://github.com/gz/rust-cpuid/issues/40
- https://github.com/RustSec/advisory-db/pull/614
- https://rustsec.org/advisories/RUSTSEC-2021-0013.html
- https://github.com/advisories/GHSA-hvqc-pc78-x9wh

23.3. Recommendation

Consider alternatives of this dependency.

24. H-15 Use After Free in lru

Tags: runtime, Weaknesses: GHSA ID: GHSA-v362-2895-h9r2 CWE-416

File HalbornCTF_Rust_Substrate/Cargo.lock#L3055-L3059

```
"log",
"lru",
"minicbor",
"rand 0.7.3",
"smallvec 1.6.1",
```

Lru crate has two functions for getting an iterator. Both iterators give references to key and value. Calling specific functions, like pop(), will remove and free the value, and but it’s still possible to access the reference of value which is already dropped causing use after free.

24.1. Impact: High 7.5 / 10

Table 24. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	High

24.2. References

- https://nvd.nist.gov/vuln/detail/CVE-2021-45720
- https://raw.githubusercontent.com/rustsec/advisory-db/main/crates/lru/RUSTSEC-2021-0130.md
- https://rustsec.org/advisories/RUSTSEC-2021-0130.html
- https://github.com/jeromefroe/lru-rs/issues/120
- https://github.com/advisories/GHSA-v362-2895-h9r2

24.3. Recommendation

Consider alternatives of this dependency.

25. H-16 crossbeam-utils Race Condition vulnerability

Tags: `runtime`, Weaknesses: CWE-362, CVE ID: CVE-2022-23639, GHSA ID: GHSA-qc84-gqf4-9926

File HalbornCTF_Rust_Substrate/Cargo.lock#L313-L317

```
"async-process",
"crossbeam-utils 0.8.5",
"futures-channel",
"futures-core",
"futures-io",
```

The affected version of this crate 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 the following problems:

- Unaligned memory accesses
- 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. 32-bit targets with `Atomic{I,U}64` and `{i,u}64` have the same alignment are also not affected by this issue.

The following is a complete list of the builtin targets that may be affected. (last update: nightly-2022-02-11)

- armv7-apple-ios (tier 3)
- armv7s-apple-ios (tier 3)
- i386-apple-ios (tier 3)
- i586-unknown-linux-gnu
- i586-unknown-linux-musl
- i686-apple-darwin (tier 3)
- i686-linux-android
- i686-unknown-freebsd
- i686-unknown-haiku (tier 3)
- i686-unknown-linux-gnu
- i686-unknown-linux-musl
- i686-unknown-netbsd (tier 3)
- i686-unknown-openbsd (tier 3)
- i686-wrs-vxworks (tier 3)

(script to get list)

25.1. Impact: High 8.1 / 10

Table 25. CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	High
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	High
Integrity	High
Availability	High

25.2. Patches

This has been fixed in crossbeam-utils 0.8.7.

Affected 0.8.x releases have been yanked.

<RECOMMENDATION>

25.3. References

https://github.com/crossbeam-rs/crossbeam/pull/781

26. H-17 Rust’s regex crate vulnerable to regular expression denial of service

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-m5pq-gvj9-9vr8 CWE-400, CWE-1333

File HalbornCTF_Rust_Substrate/Cargo.lock#L11-L15

```
 "lazy_static",
 "regex",
]

[[package]]
```

> This is a cross-post of [the official security advisory][advisory]. The official advisory contains a signed version with our PGP key, as well.

[advisory]: https://groups.google.com/g/rustlang-security-announcements/c/NcNNL1Jq7Yw

The Rust Security Response WG was notified that the `regex` crate did not properly limit the complexity of the regular expressions (regex) it parses. An attacker could use this security issue to perform a denial of service, by sending a specially crafted regex to a service accepting untrusted regexes. No known vulnerability is present when parsing untrusted input with trusted regexes.

This issue has been assigned CVE-2022-24713. The severity of this vulnerability is "high" when the `regex` crate is used to parse untrusted regexes. Other uses of the `regex` crate are not affected by this vulnerability.

26.1. Overview

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.

26.2. Affected versions

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.

26.3. Mitigations

We recommend everyone accepting user-controlled regexes 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, we do not recommend denying known problematic regexes.

26.4. Acknowledgements

We want to thank Addison Crump for responsibly disclosing this to us according to the Rust security policy, and for helping review the fix.

We also want to thank Andrew Gallant for developing the fix, and Pietro Albini for coordinating the disclosure and writing this advisory.

26.5. Impact: High 7.5 / 10

Table 26. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	High

26.6. References

- https://github.com/rust-lang/regex/security/advisories/GHSA-m5pq-gvj9-9vr8
- https://github.com/rust-lang/regex/commit/ae70b41d4f46641dbc45c7a4f87954aea356283e
- https://groups.google.com/g/rustlang-security-announcements/c/NcNNL1Jq7Yw
- https://nvd.nist.gov/vuln/detail/CVE-2022-24713
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/JANLZ3JXWJR7FSHE57K66UIZUIJZI67T/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/PDOWTHNVGBOP2HN27PUFIGRYNSNDTYRJ/
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/O3YB7CURSG64CIPCDPNMGPE4UU24AB6H/
- https://lists.debian.org/debian-lts-announce/2022/04/msg00003.html
- https://www.debian.org/security/2022/dsa-5113
- https://lists.debian.org/debian-lts-announce/2022/04/msg00009.html
- https://www.debian.org/security/2022/dsa-5118
- https://rustsec.org/advisories/RUSTSEC-2022-0013.html
- https://security.gentoo.org/glsa/202208-08
- https://security.gentoo.org/glsa/202208-14
- https://github.com/advisories/GHSA-m5pq-gvj9-9vr8

26.7. Recommendation

Consider alternatives of this dependency.

27. H-18 Use after free in Wasmtime

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-gwc9-348x-qwv2 CWE-416

File HalbornCTF_Rust_Substrate/Cargo.lock#L5822-L5826

```
"sc-executor-wasmi",
"sc-executor-wasmtime",
"sp-api",
"sp-core",
"sp-externalities",
```

There is a use after free vulnerability in Wasmtime when both running Wasm that uses `externref`s and enabling epoch interruption in Wasmtime. If you are not explicitly enabling epoch interruption (it is disabled by default) then you are not affected. If you are explicitly disabling the Wasm reference types proposal (it is enabled by default) then you are also not affected.

The use after free is caused by Cranelift failing to emit stack maps when there are safepoints inside cold blocks. Cold blocks occur when epoch interruption is enabled. Cold blocks are emitted at the end of compiled functions, and change the order blocks are emitted versus defined. This reordering accidentally caused Cranelift to skip emitting some stack maps because it expected to emit the stack maps in block definition order, rather than block emission order. When Wasmtime would eventually collect garbage, it would fail to find live references on the stack because of the missing stack maps, think that they were unreferenced garbage, and therefore reclaim them. Then after the collection ended, the Wasm code could use the reclaimed-too-early references, which is a use after free.

This bug was discovered while extending our fuzz targets for `externref`s and GC in Wasmtime. The updated fuzz target thoroughly exercises these code paths and feature combinations now. We have also added a regression test for this bug. Released versions 0.34.2 and 0.35.2, which fix the vulnerability. We recommend all Wasmtime users upgrade to these patched versions. If upgrading is not an option for you at this time, you can avoid the vulnerability by either disabling the Wasm reference types proposal or by disabling epoch interruption if you were previously enabling it.

27.1. Impact: High 8.1 / 10

Table 27. CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	High
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	High
Integrity	High
Availability	High

27.2. References

- https://github.com/bytecodealliance/wasmtime/security/advisories/GHSA-gwc9-348x-qwv2
- https://nvd.nist.gov/vuln/detail/CVE-2022-24791
- https://github.com/bytecodealliance/wasmtime/commit/666c2554ea0e1728c35aa41178cf235920db888a
- https://docs.rs/wasmtime/latest/wasmtime/struct.Config.html#method.epoch_interruption
- https://github.com/WebAssembly/reference-types
- https://rustsec.org/advisories/RUSTSEC-2022-0016.html
- https://github.com/advisories/GHSA-gwc9-348x-qwv2

27.3. Recommendation

Consider alternatives of this dependency.

28. H-19 Parser creates invalid uninitialized value

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-f67m-9j94-qv9j

File HalbornCTF_Rust_Substrate/Cargo.lock#L2258-L2262

```
[[package]]
name = "hyper"
version = "0.12.36"
source = "registry+https://github.com/rust-lang/crates.io-index"
checksum = "5c843caf6296fc1f93444735205af9ed4e109a539005abb2564ae1d6fad34c52"
```

Affected versions of this crate called `mem::uninitialized()` in the HTTP1 parser to create values of type `httparse::Header` (from the `httparse` crate). This is unsound, since `Header` contains references and thus must be non-null.

The flaw was corrected by avoiding the use of `mem::uninitialized()`, using `MaybeUninit` instead.

28.1. References

- https://github.com/hyperium/hyper/pull/2545
- https://rustsec.org/advisories/RUSTSEC-2022-0022.html
- https://github.com/advisories/GHSA-f67m-9j94-qv9j

28.2. Recommendation

Consider alternatives of this dependency.

29. H-20 Use after free in lru crate

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-qqmc-hwqp-8g2w

File HalbornCTF_Rust_Substrate/Cargo.lock#L3055-L3059

```
"log",
"lru",
"minicbor",
"rand 0.7.3",
"smallvec 1.6.1",
```

Lru crate has use after free vulnerability.

Lru crate has two functions for getting an iterator. Both iterators give references to key and value. Calling specific functions, like pop(), will remove and free the value, and but it’s still possible to access the reference of value which is already dropped causing use after free.

29.1. References

- https://github.com/jeromefroe/lru-rs/issues/120
- https://rustsec.org/advisories/RUSTSEC-2021-0130.html
- https://github.com/advisories/GHSA-qqmc-hwqp-8g2w

29.2. Recommendation

Consider alternatives of this dependency.

30. H-21 Data race in `Iter` and `IterMut`

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-9hpw-r23r-xgm5 CWE-362

File HalbornCTF_Rust_Substrate/Cargo.lock#L7765-L7769

```
[[package]]
name = "thread_local"
version = "1.1.3"
source = "registry+https://github.com/rust-lang/crates.io-index"
checksum = "8018d24e04c95ac8790716a5987d0fec4f8b27249ffa0f7d33f1369bdfb88cbd"
```

In the affected version of this crate, `{Iter, IterMut}::next` used a weaker memory ordering when loading values than what was required, exposing a potential data race when iterating over a `ThreadLocal's values.

Crates using `Iter::next`, or `IterMut::next` are affected by this issue.

30.1. References

- https://github.com/Amanieu/thread_local-rs/issues/33
- https://rustsec.org/advisories/RUSTSEC-2022-0006.html
- https://github.com/advisories/GHSA-9hpw-r23r-xgm5

30.2. Recommendation

Consider alternatives of this dependency.

31. H-22 Wasmtime may have data leakage between instances in the pooling allocator

Tags: `runtime`, Weaknesses: CWE-212, CWE-226, CVE ID: CVE-2022-39393, GHSA ID: GHSA-wh6w-3828-g9qf

File HalbornCTF_Rust_Substrate/Cargo.lock#L5822-L5826

```
"sc-executor-wasmi",
"sc-executor-wasmtime",
"sp-api",
"sp-core",
"sp-externalities",
```

There is a bug in Wasmtime's implementation of it's pooling instance allocator where when a linear memory is reused for another instance the initial heap snapshot of the prior instance can be visible, erroneously to the next instance. The pooling instance allocator in Wasmtime works by preallocating virtual memory for a fixed number of instances to reside in and then new instantiations pick a slot to use. Most conventional modules additionally have an initial copy-on-write "heap image" which is mapped in Wasmtime into the linear memory slot. When a heap slot is deallocated Wasmtime resets all of its contents back to the initial state but it does not unmap the image in case the next instance is an instantiation of the same module.

The bug in Wasmtime occurs when a slot in the pooling allocator previously was used for a module with a heap image, meaning that its current state of memory contains the initial heap contents of that module. If the next instantiation within that slot does not itself contain a heap image then Wasmtime would leave the old heap image in place erroneously and continue with instantiation. This means that instantiations of modules without a heap image can see the initial heap image of the prior instantiation within that slot.

Heap images in Wasmtime are created by precomputing WebAssembly `data` segments into one large mapping to be placed into linear memory at a particular offset. Most modules produced by toolchains today will have a heap image and an initialization snapshot. Creating a module without a heap image would require a hand-crafted `*.wat` file or a specially crafted source program. This consequence means that this bug is highly unlikely to be accidentally triggered and would otherwise require an intentional trigger with a hand-crafted module.

One important part of this vulnerability is Wasmtime is highly likely to segfault when the slot is reused again with a module that itself has an initialization image. For example if module A has a heap initialization image and module B does not have a heap initialization image, then the following sequence of events could happen if they all are instantiated into the same instance slot:

- Module A is instantiated, and then deallocated. This leaves A's heap image in place, reset to its initial contents.
- Module B is instantiated and erroneously can see the initial heap contents of A. Module B is then deallocated and the entire heap is unmapped and reset back to zero.
- Module A is instantiated again, but the state tracking the slot did not account for module B so it thinks the module image is still mapped and proceeds with instantiation. Any action on A's part to access linear memory will then trap and if the host accesses A's memory it will segfault because the data that's supposed to be mapped is all unmapped.

Adding this all together this means that in practice modules must be deliberately crafted to not have an initial heap image to view the contents of a prior image. If this module is instantiated though then when the slot is reused the next, likely image-using, module will believe its memory is mapped when it isn't, causing the host to segfault on unmapped memory it believed was mapped.

31.1. Impact: High 8.6 / 10

Table 28. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:N/A:N

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Changed
Confidentiality	High
Integrity	None
Availability	None

31.2. Patches

This bug has been patched and users should upgrade to Wasmtime 2.0.2.

<RECOMMENDATION>

31.3. References

- `Config::allocation_strategy` - configuration required to enable the pooling allocator.
- `Config::memory_init_cow` - configuration required to enable or disable copy-on-write (this is enabled by default).
- Mailing list announcement
- Patch for `release-2.0.0` branch
- Patch for `main`

32. H-23 libp2p DoS vulnerability from lack of resource management

Tags: `runtime`, Weaknesses: CWE-400, CWE-770, CVE ID: CVE-2022-23486, GHSA ID: GHSA-jvgw-gccv-q5p8

File HalbornCTF_Rust_Substrate/Cargo.lock#L2747-L2751

```
[[package]]
name = "libp2p"
version = "0.34.0"
source = "registry+https://github.com/rust-lang/crates.io-index"
checksum = "d5133112ce42be9482f6a87be92a605dd6bbc9e93c297aee77d172ff06908f3a"
```

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.

32.1. Impact: High 7.5 / 10

Table 29. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	High

32.2. Patches

Users are advised to upgrade to `libp2p v0.45.1` or above.

<RECOMMENDATION>

32.3. References

33. H-24 Race Condition in tokio

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-fg7r-2g4j-5cgr CWE-362

File HalbornCTF_Rust_Substrate/Cargo.lock#L2064-L2068

```
  "string",
  "tokio-io",
]

[[package]]
```

If a tokio::sync::oneshot channel is closed (via the oneshot::Receiver::close method), a data race may occur if the oneshot::Sender::send method is called while the corresponding oneshot::Receiver is awaited or calling try_recv.

When these methods are called concurrently on a closed channel, the two halves of the channel can concurrently access a shared memory location, resulting in a data race. This has been observed to cause memory corruption.

Note that the race only occurs when both halves of the channel are used after the Receiver half has called close. Code where close is not used, or where the Receiver is not awaited and try_recv is not called after calling close, is not affected.

33.1. Impact: High 8.1 / 10

Table 30. CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	High
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	High
Integrity	High
Availability	High

33.2. References

- https://nvd.nist.gov/vuln/detail/CVE-2021-45710
- https://raw.githubusercontent.com/rustsec/advisory-db/main/crates/tokio/RUSTSEC-2021-0124.md
- https://rustsec.org/advisories/RUSTSEC-2021-0124.html
- https://github.com/tokio-rs/tokio/issues/4225
- https://github.com/advisories/GHSA-fg7r-2g4j-5cgr

33.3. Recommendation

Consider alternatives of this dependency.

34. H-25 webpki: CPU denial of service in certificate path building

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-8qv2-5vq6-g2g7 CWE-400

File HalbornCTF_Rust_Substrate/Cargo.lock#L1885-L1889

```
 "rustls 0.19.1",
 "webpki",
]

[[package]]
```

When this crate is given a pathological certificate chain to validate, it will spend CPU time exponential with the number of candidate certificates at each step of path building.

Both TLS clients and TLS servers that accept client certificate are affected.

This was previously reported in https://github.com/briansmith/webpki/issues/69.

`rustls-webpki` is a fork of this crate which contains a fix for this issue and is actively maintained.

34.1. Impact: High 7.5 / 10

Table 31. CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	High

34.2. References

- https://github.com/crypto-com/sgx-vendor
- https://rustsec.org/advisories/RUSTSEC-2023-0052.html
- https://github.com/briansmith/webpki/issues/69
- https://github.com/briansmith/webpki/issues/69#issuecomment-1699894848
- https://github.com/briansmith/webpki/commit/30a108e0802fd09585e0d071013f24b8272d139b
- https://github.com/advisories/GHSA-8qv2-5vq6-g2g7

34.3. Recommendation

Consider alternatives of this dependency.

35. H-26 Multiple issues involving quote API in shlex

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-r7qv-8r2h-pg27

File HalbornCTF_Rust_Substrate/Cargo.lock#L469-L473

```
 "rustc-hash",
 "shlex",
]

[[package]]
```

35.1. Issue 1: Failure to quote characters

Affected versions of this crate allowed the bytes `{` and `\xa0` to appear unquoted and unescaped in command arguments.

If the output of `quote` or `join` is passed to a shell, then what should be a single command argument could be interpreted as multiple arguments.

This does not **directly** allow arbitrary command execution (you can't inject a command substitution or similar). But depending on the command you're running, being able to inject multiple arguments where only one is expected could lead to undesired consequences, potentially including arbitrary command execution.

The flaw was corrected in version 1.2.1 by escaping additional characters. Updating to 1.3.0 is recommended, but 1.2.1 offers a more minimal fix if desired.

Workaround: Check for the bytes `{` and `\xa0` in `quote`/`join` input or output.

(Note: `{` is problematic because it is used for glob expansion. `\xa0` is problematic because it's treated as a word separator in [specific environments][solved-xa0].)

35.2. Issue 2: Dangerous API w.r.t. nul bytes

Version 1.3.0 deprecates the `quote` and `join` APIs in favor of `try_quote` and `try_join`, which behave the same except that they have `Result` return type, returning `Err` if the input contains nul bytes.

Strings containing nul bytes generally cannot be used in Unix command arguments or environment variables, and most shells cannot handle nul bytes even internally. If you try to pass one anyway, then the results might be security-sensitive in uncommon scenarios. [More details here.][nul-bytes]

Due to the low severity, the behavior of the original `quote` and `join` APIs has not changed; they continue to allow nuls.

Workaround: Manually check for nul bytes in `quote`/`join` input or output.

35.3. Issue 3: Lack of documentation for interactive shell risks

The `quote` family of functions does not and cannot escape control characters. With non-interactive shells this is perfectly safe, as control characters have no special effect. But if you writing directly to the standard input of an interactive shell (or through a pty), then control characters [can cause misbehavior including arbitrary command injection.][control-characters]

This is essentially unfixable, and has not been patched. But as of version 1.3.0, documentation has been added.

Future versions of `shlex` may add API variants that avoid the issue at the cost of reduced portability.

[solved-xa0]: https://docs.rs/shlex/latest/shlex/quoting_warning/index.html#solved-xa0 [nul-bytes]: https://docs.rs/shlex/latest/shlex/quoting_warning/index.html#nul-bytes [control-characters]: https://docs.rs/shlex/latest/shlex/quoting_warning/index.html#control-characters-interactive-contexts-only

35.4. References

- https://github.com/comex/rust-shlex/security/advisories/GHSA-r7qv-8r2h-pg27
- https://rustsec.org/advisories/RUSTSEC-2024-0006.html
- https://github.com/advisories/GHSA-r7qv-8r2h-pg27

35.5. Recommendation

Consider alternatives of this dependency.

Part III: Medium

36. M-1 Default inheritable capabilities for linux container should be empty

Tags: runtime, Weaknesses: CWE-276, CVE ID: CVE-2022-29162, GHSA ID: GHSA-f3fp-gc8g-vw66

File HalbornCTF_Golang_Cosmos/go.mod#L188-L192

```
    github.com/opencontainers/image-spec v1.0.2 // indirect
    github.com/opencontainers/runc v1.0.3 // indirect
    github.com/pelletier/go-toml v1.9.4 // indirect
    github.com/petermattis/goid v0.0.0-20180202154549-b0b1615b78e5 // indirect
    github.com/phayes/checkstyle v0.0.0-20170904204023-bfd46e6a821d // indirect
```

A bug was found in runc where `runc exec --cap` executed processes with non-empty inheritable Linux process capabilities, creating an atypical Linux environment and enabling programs with inheritable file capabilities to elevate those capabilities to the permitted set during execve(2).

This bug did not affect the container security sandbox as the inheritable set never contained more capabilities than were included in the container's bounding set.

36.1. Impact: Medium 5.9 / 10

Table 32. CVSS:3.1/AV:L/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:L

CVSS base metrics	
Attack vector	Local
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	Low
Integrity	Low
Availability	Low

36.2. Patches

<RECOMMENDATION>

36.3. References

37. M-2 golang.org/x/sys/unix has Incorrect privilege reporting in syscall

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-p782-xgp4-8hr8 CWE-269

File HalbornCTF_Golang_Cosmos/go.mod#L248-L252

```
    golang.org/x/sync v0.0.0-20210220032951-036812b2e83c // indirect
    golang.org/x/sys v0.0.0-20220209214540-3681064d5158 // indirect
    golang.org/x/term v0.0.0-20201126162022-7de9c90e9dd1 // indirect
    golang.org/x/text v0.3.7 // indirect
    golang.org/x/tools v0.1.10 // indirect
```

Go before 1.17.10 and 1.18.x before 1.18.2 has Incorrect Privilege Reporting in syscall. When called with a non-zero flags parameter, the Faccessat function could incorrectly report that a file is accessible.

37.1. Specific Go Packages Affected

golang.org/x/sys/unix

37.2. Impact: Medium 5.3 / 10

Table 33. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	Low
Integrity	None
Availability	None

37.3. References

- https://nvd.nist.gov/vuln/detail/CVE-2022-29526
- https://github.com/golang/go/issues/52313
- https://groups.google.com/g/golang-announce
- https://groups.google.com/g/golang-announce/c/Y5qrqw_lWdU
- https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/ZY2SLWOQR4ZURQ7UBRZ7JIX6H6F5JHJR/

37.4. Recommendation

Consider alternatives of this dependency.

38. M-3 runc AppArmor bypass with symlinked /proc

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-g2j6-57v7-gm8c CWE-59, CWE-281

File HalbornCTF_Golang_Cosmos/go.mod#L188-L192

```
	github.com/opencontainers/image-spec v1.0.2 // indirect
	github.com/opencontainers/runc v1.0.3 // indirect
	github.com/pelletier/go-toml v1.9.4 // indirect
	github.com/petermattis/goid v0.0.0-20180202154549-b0b1615b78e5 // indirect
	github.com/phayes/checkstyle v0.0.0-20170904204023-bfd46e6a821d // indirect
```

38.1. Impact

It was found that AppArmor, and potentially SELinux, can be bypassed when `/proc` inside the container is symlinked with a specific mount configuration.

38.2. Patches

Fixed in runc v1.1.5, by prohibiting symlinked `/proc`: https://github.com/opencontainers/runc/pull/3785

This PR fixes CVE-2023-27561 as well.

38.3. Workarounds

Avoid using an untrusted container image.

38.4. Impact: Medium 6.1 / 10

Table 34. CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:C/C:L/I:L/A:L

CVSS base metrics	
Attack vector	Local
Attack complexity	Low
Privileges required	None
User interaction	Required
Scope	Changed
Confidentiality	Low
Integrity	Low
Availability	Low

38.5. References

- https://github.com/opencontainers/runc/security/advisories/GHSA-g2j6-57v7-gm8c
- https://nvd.nist.gov/vuln/detail/CVE-2023-28642
- https://github.com/opencontainers/runc/pull/3785
- https://github.com/advisories/GHSA-g2j6-57v7-gm8c

38.6. Recommendation

Consider alternatives of this dependency.

39. M-4 Docker Swarm encrypted overlay network with a single endpoint is unauthenticated

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-6wrf-mxfj-pf5p CWE-420, CWE-636

File HalbornCTF_Golang_Cosmos/go.mod#L83-L87

```
	github.com/docker/cli v20.10.11+incompatible // indirect
	github.com/docker/docker v20.10.7+incompatible // indirect
	github.com/docker/go-connections v0.4.0 // indirect
	github.com/docker/go-units v0.4.0 // indirect
	github.com/dustin/go-humanize v1.0.0 // indirect
```

Moby is an open source container framework developed by Docker Inc. that is distributed as Docker, Mirantis Container Runtime, and various other downstream projects/products. The Moby daemon component (`dockerd`), which is developed as moby/moby is commonly referred to as **Docker**.

Swarm Mode, which is compiled in and delivered by default in `dockerd` and is thus present in most major Moby downstreams, is a simple, built-in container orchestrator that is implemented through a combination of SwarmKit and supporting network code.

The `overlay` network driver is a core feature of Swarm Mode, providing isolated virtual LANs that allow communication between containers and services across the cluster. This driver is an implementation/user of VXLAN, which encapsulates link-layer (Ethernet) frames in UDP datagrams that tag the frame with a VXLAN Network ID (VNI) that identifies the originating overlay network. In addition, the overlay network driver supports an optional, off-by-default encrypted mode, which is especially useful when VXLAN packets traverses an untrusted network between nodes.

Encrypted overlay networks function by encapsulating the VXLAN datagrams through the use of the IPsec Encapsulating Security Payload protocol in Transport mode. By deploying IPSec encapsulation, encrypted overlay networks gain the additional properties of source authentication through cryptographic proof, data integrity through check-summing, and confidentiality through encryption.

When setting an endpoint up on an encrypted overlay network, Moby installs three iptables (Linux kernel firewall) rules that enforce both incoming and outgoing IPSec. These rules rely on the `u32` iptables extension provided by the `xt_u32` kernel module to directly filter on a VXLAN packet's VNI field, so that IPSec guarantees can be enforced on encrypted overlay networks without interfering with other overlay networks or other users of VXLAN.

The `overlay` driver dynamically and lazily defines the kernel configuration for the VXLAN network on each node as containers are attached and detached. Routes and encryption parameters are only defined for destination nodes that participate in the network. The iptables rules that prevent encrypted overlay networks from accepting unencrypted packets are not created until a peer is available with which to communicate.

39.1. Impact

Encrypted overlay networks silently accept cleartext VXLAN datagrams that are tagged with the VNI of an encrypted overlay network. As a result, it is possible to inject arbitrary Ethernet frames into the encrypted overlay network by encapsulating them in VXLAN datagrams. The implications of this can be quite dire, and GHSA-vwm3-crmr-xfxw should be referenced for a deeper exploration.

39.2. Patches

Patches are available in Moby releases 23.0.3, and 20.10.24. As Mirantis Container Runtime's 20.10 releases are numbered differently, users of that platform should update to 20.10.16.

39.3. Workarounds

- In multi-node clusters, deploy a global 'pause' container for each encrypted overlay network, on every node. For example, use the `registry.k8s.io/pause` image and a `--mode global` service.
- For a single-node cluster, do not use overlay networks of any sort. Bridge networks provide the same connectivity on a single node and have no multi-node features. The Swarm ingress feature is implemented using an overlay network, but can be disabled by publishing ports in `host` mode instead of `ingress` mode (allowing the use of an external load balancer), and removing the `ingress` network.
- If encrypted overlay networks are in exclusive use, block UDP port 4789 from traffic that has not been validated by IPSec. For example, `iptables -A INPUT -m udp —-dport 4789 -m policy --dir in --pol none -j DROP`.

39.4. Background

- This issue was discovered while characterizing and mitigating CVE-2023-28840 and CVE-2023-28841.

39.5. Related

- CVE-2023-28841: Encrypted overlay network traffic may be unencrypted
- CVE-2023-28840: Encrypted overlay network may be unauthenticated
- GHSA-vwm3-crmr-xfxw: The Swarm VXLAN port may be exposed to attack due to ambiguous documentation
- GHSA-gvm4-2qqg-m333: Security issues in encrypted overlay networks (libnetwork)

39.6. Impact: Medium 6.8 / 10

Table 35. CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:C/C:N/I:H/A:N

CVSS base metrics	
Attack vector	Network
Attack complexity	High
Privileges required	None
User interaction	None
Scope	Changed
Confidentiality	None
Integrity	High
Availability	None

39.7. References

- https://github.com/moby/libnetwork/security/advisories/GHSA-gvm4-2qqg-m333
- https://github.com/moby/moby/security/advisories/GHSA-232p-vwff-86mp
- https://github.com/moby/moby/security/advisories/GHSA-33pg-m6jh-5237
- https://github.com/moby/moby/security/advisories/GHSA-6wrf-mxfj-pf5p
- https://github.com/moby/moby/security/advisories/GHSA-vwm3-crmr-xfxw

39.8. Recommendation

Consider alternatives of this dependency.

40. M-5 Docker Swarm encrypted overlay network traffic may be unencrypted

Tags: runtime, Weaknesses: GHSA ID: GHSA-33pg-m6jh-5237 CWE-311, CWE-636

File HalbornCTF_Golang_Cosmos/go.mod#L83-L87

```
    github.com/docker/cli v20.10.11+incompatible // indirect
    github.com/docker/docker v20.10.7+incompatible // indirect
    github.com/docker/go-connections v0.4.0 // indirect
    github.com/docker/go-units v0.4.0 // indirect
    github.com/dustin/go-humanize v1.0.0 // indirect
```

Moby is an open source container framework developed by Docker Inc. that is distributed as Docker, Mirantis Container Runtime, and various other downstream projects/products. The Moby daemon component (dockerd), which is developed as moby/moby is commonly referred to as **Docker**.

Swarm Mode, which is compiled in and delivered by default in dockerd and is thus present in most major Moby downstreams, is a simple, built-in container orchestrator that is implemented through a combination of SwarmKit and supporting network code.

The overlay network driver is a core feature of Swarm Mode, providing isolated virtual LANs that allow communication between containers and services across the cluster. This driver is an implementation/user of VXLAN, which encapsulates link-layer (Ethernet) frames in UDP datagrams that tag the frame with a VXLAN Network ID (VNI) that identifies the originating overlay network. In addition, the overlay network driver supports an optional, off-by-default encrypted mode, which is especially useful when VXLAN packets traverses an untrusted network between nodes.

Encrypted overlay networks function by encapsulating the VXLAN datagrams through the use of the IPsec Encapsulating Security Payload protocol in Transport mode. By deploying IPSec encapsulation, encrypted overlay networks gain the additional properties of source authentication through cryptographic proof, data integrity through check-summing, and confidentiality through encryption.

When setting an endpoint up on an encrypted overlay network, Moby installs three iptables (Linux kernel firewall) rules that enforce both incoming and outgoing IPSec. These rules rely on the u32 iptables extension provided by the xt_u32 kernel module to directly filter on a VXLAN packet's VNI field, so that IPSec guarantees can be enforced on encrypted overlay networks without interfering with other overlay networks or other users of VXLAN.

An iptables rule designates outgoing VXLAN datagrams with a VNI that corresponds to an encrypted overlay network for IPsec encapsulation.

On Red Hat Enterprise Linux and derivatives such as CentOS and Rocky, the xt_u32 module has been: * moved to the kernel-modules-extra package and no longer installed by default in RHEL 8.3 * officially deprecated in RHEL 8.6 * removed completely in RHEL 9

This rule is not created when xt_u32 is unavailable, even though the container is still attached to the network.

40.1. Impact

Encrypted overlay networks on affected platforms silently transmit unencrypted data. As a result, overlay networks may appear to be functional, passing traffic as expected, but without any of the expected confidentiality or data integrity guarantees.

It is possible for an attacker sitting in a trusted position on the network to read all of the application traffic that is moving across the overlay network, resulting in unexpected secrets or user data disclosure. Thus, because many database protocols, internal APIs, etc. are not protected by a second layer of encryption, a user may rely on Swarm encrypted overlay networks to provide confidentiality, which due to this vulnerability is no longer guaranteed.

40.2. Patches

Patches are available in Moby releases 23.0.3, and 20.10.24. As Mirantis Container Runtime's 20.10 releases are numbered differently, users of that platform should update to 20.10.16.

40.3. Workarounds

- Close the VXLAN port (by default, UDP port 4789) to outgoing traffic at the Internet boundary (see GHSA-vwm3-crmr-xfxw) in order to prevent unintentionally leaking unencrypted traffic over the Internet.
- Ensure that the `xt_u32` kernel module is available on all nodes of the Swarm cluster.

40.4. Background

- #43382 partially discussed this concern, but did not consider the security implications.
- Mirantis FIELD-5788 essentially duplicates #43382, and was created six months earlier; it similarly overlooked the security implications.
- #45118 is the ancestor of the final patches, and was where the security implications were discovered.

40.5. Related

- CVE-2023-28840: Encrypted overlay network may be unauthenticated
- CVE-2023-28842: Encrypted overlay network with a single endpoint is unauthenticated
- GHSA-vwm3-crmr-xfxw: The Swarm VXLAN port may be exposed to attack due to ambiguous documentation
- GHSA-gvm4-2qqg-m333: Security issues in encrypted overlay networks (libnetwork)

40.6. Impact: Medium 6.8 / 10

Table 36. CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:C/C:H/I:N/A:N

CVSS base metrics	
Attack vector	Network
Attack complexity	High
Privileges required	None
User interaction	None
Scope	Changed
Confidentiality	High

CVSS base metrics	
Integrity	None
Availability	None

40.7. References

- https://github.com/moby/libnetwork/security/advisories/GHSA-gvm4-2qqg-m333
- https://github.com/moby/moby/security/advisories/GHSA-232p-vwff-86mp
- https://github.com/moby/moby/security/advisories/GHSA-33pg-m6jh-5237
- https://github.com/moby/moby/security/advisories/GHSA-6wrf-mxfj-pf5p
- https://github.com/moby/moby/security/advisories/GHSA-vwm3-crmr-xfxw

40.8. Recommendation

Consider alternatives of this dependency.

41. M-6 Cosmos-SDK Cosmovisor component may be vulnerable to denial of service

Tags: `runtime`, Weaknesses: GHSA ID: GHSA-23px-mw2p-46qm

File HalbornCTF_Golang_Cosmos/go.mod#L5-L9

```
require (
    github.com/cosmos/cosmos-sdk v0.45.4
    github.com/cosmos/go-bip39 v1.0.0
    github.com/cosmos/ibc-go/v3 v3.0.0
    github.com/golangci/golangci-lint v1.45.2
```

Component: Cosmovisor **Criticality**: Medium **Affected Versions**: Cosmovisor < v1.0.0 (distributed with Cosmos-SDK < 0.46) **Affected Users**: Validators and Node operators utilizing unsupported versions of Cosmovisor **Impact**: DOS, potential RCE on node depending on configuration

An issue has been identified on unsupported versions of Cosmovisor which may result in a Denial of Service or Remote Code Execution path depending on configuration for a node or validator using the vulnerable version to manage their node.

If a validator is utilizing an affected version of Cosmovisor with `DAEMON_ALLOW_DOWNLOAD_BINARIES` set to true, a non-default configuration, it may be possible for an attacker to trigger a Remote Code Execution path as well on the host. In this configuration it is recommended to immediately stop use of the `DAEMON_ALLOW_DOWNLOAD_BINARIES` feature, and then proceed with an upgrade of Cosmovisor.

It is recommended that all validators utilizing unsupported versions of Cosmovisor to upgrade to the latest supported versions immediately. If you are utilizing a forked version of Cosmos-SDK, it is recommended to stop use of Cosmovisor until it is possible to update to a supported version of Cosmovisor, whether through your project's fork, or directly compiled from the Cosmos-SDK. At the time of this advisory, the latest version of Cosmovisor is v1.5.0.

Additionally, the Amulet team recommends that developers building chains powered by Cosmos-SDK share this advisory with validators and node operators to ensure this information is available to all impacted parties within their ecosystems.

For more information about Cosmovisor, see https://docs.cosmos.network/main/tooling/cosmovisor

This issue was discovered by Maxwell Dulin and Nathan Kirkland, who reported it to the Cosmos Bug Bounty Program. If you believe you have found a bug in the Interchain Stack or would like to contribute to the program by reporting a bug, please see https://hackerone.com/cosmos.

41.1. How to tell if I am affected?

Running the following command will output whether your cosmovisor version is vulnerable to this issue or not.

Vulnerable to this issue:

```
strings ./cosmovisor | grep -q "NEEDED at" && echo "vulnerable" || echo
"NOT vulnerable"
```

```
vulnerable
```

NOT vulnerable to this issue:

```
strings ./cosmovisor_new | grep -q "NEEDED at" && echo "vulnerable" ||
echo "NOT vulnerable"

NOT vulnerable
```

A Note from Amulet on the Security Advisory Process

In the interest of timely resolution of this issue for validators and node operators, the Amulet team has chosen to use existing processes and resources for distributing security advisories within the Cosmos and Interchain Ecosystems. Stay tuned as we implement an improved, more robust security advisory distribution system that will provide equitable access to information about security issues in the Interchain Stack.

=== References

- https://github.com/cosmos/cosmos-sdk/security/advisories/GHSA-23px-mw2p-46qm
- https://github.com/cosmos/cosmos-sdk/blob/tools/cosmovisor/v1.5.0/tools/cosmovisor/CHANGELOG.md
- https://github.com/cosmos/cosmos-sdk/releases/tag/v0.46.0
- https://github.com/advisories/GHSA-23px-mw2p-46qm

=== Recommendation

Consider alternatives of this dependency.

== M-7 Improper rendering of text nodes in golang.org/x/net/html Tags: `runtime`, Weaknesses: GHSA ID: GHSA-2wrh-6pvc-2jm9 CWE-79

File HalbornCTF_Golang_Cosmos/go.mod#L246-L250

```
	golang.org/x/mod v0.6.0-dev.0.20220106191415-9b9b3d81d5e3 // indirect
	golang.org/x/net v0.0.0-20211208012354-db4efeb81f4b // indirect
	golang.org/x/sync v0.0.0-20210220032951-036812b2e83c // indirect
	golang.org/x/sys v0.0.0-20220209214540-3681064d5158 // indirect
	golang.org/x/term v0.0.0-20201126162022-7de9c90e9dd1 // indirect
```

Text nodes not in the HTML namespace are incorrectly literally rendered, causing text which should be escaped to not be. This could lead to an XSS attack.

=== Impact: Medium 6.1 / 10

Table 37. CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:L/I:L/A:N

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	Required
Scope	Changed
Confidentiality	Low
Integrity	Low
Availability	None

=== References

- https://nvd.nist.gov/vuln/detail/CVE-2023-3978
- https://go.dev/cl/514896
- https://go.dev/issue/61615
- https://pkg.go.dev/vuln/GO-2023-1988
- https://github.com/advisories/GHSA-2wrh-6pvc-2jm9

=== Recommendation

Consider alternatives of this dependency.

== M-8 HTTP/2 Stream Cancellation Attack Tags: `runtime`, Weaknesses: GHSA ID: GHSA-qppj-fm5r-hxr3 CWE-400

File HalbornCTF_Golang_Cosmos/go.mod#L246-L250

```
    golang.org/x/mod v0.6.0-dev.0.20220106191415-9b9b3d81d5e3 // indirect
    golang.org/x/net v0.0.0-20211208012354-db4efeb81f4b // indirect
    golang.org/x/sync v0.0.0-20210220032951-036812b2e83c // indirect
    golang.org/x/sys v0.0.0-20220209214540-3681064d5158 // indirect
    golang.org/x/term v0.0.0-20201126162022-7de9c90e9dd1 // indirect
```

=== HTTP/2 Rapid reset attack The HTTP/2 protocol allows clients to indicate to the server that a previous stream should be canceled by sending a RST_STREAM frame. The protocol does not require the client and server to coordinate the cancellation in any way, the client may do it unilaterally. The client may also assume that the cancellation will take effect immediately when the server receives the RST_STREAM frame, before any other data from that TCP connection is processed.

Abuse of this feature is called a Rapid Reset attack because it relies on the ability for an endpoint to send a RST_STREAM frame immediately after sending a request frame, which makes the other endpoint start working and then rapidly resets the request. The request is canceled, but leaves the HTTP/2 connection open.

The HTTP/2 Rapid Reset attack built on this capability is simple: The client opens a large number of streams at once as in the standard HTTP/2 attack, but rather than waiting for a response to each request stream from the server or proxy, the client cancels each request immediately.

The ability to reset streams immediately allows each connection to have an indefinite number of requests in flight. By explicitly canceling the requests, the attacker never exceeds the limit on the number of concurrent open streams. The number of in-flight requests is no longer dependent on the round-trip time (RTT), but only on the available network bandwidth.

In a typical HTTP/2 server implementation, the server will still have to do significant amounts of work for canceled requests, such as allocating new stream data structures, parsing the query and doing header decompression, and mapping the URL to a resource. For reverse proxy implementations, the request may be proxied to the backend server before the RST_STREAM frame is processed. The client on the other hand paid almost no costs for sending the requests. This creates an exploitable cost asymmetry between the server and the client.

Multiple software artifacts implementing HTTP/2 are affected. This advisory was originally ingested from the `swift-nio-http2` repo advisory and their original conent follows.

=== swift-nio-http2 specific advisory swift-nio-http2 is vulnerable to a denial-of-service vulnerability in which a malicious client can create and then reset a large number of HTTP/2 streams in a short period of time. This causes swift-nio-http2 to commit to a large amount of expensive work which it then throws away, including creating entirely new `` Channel`s to serve the traffic. This can easily overwhelm an `EventLoop `` and prevent it from making forward progress.

swift-nio-http2 1.28 contains a remediation for this issue that applies reset counter using a sliding window. This constrains the number of stream resets that may occur in a given window of time. Clients violating this limit will have their connections torn down. This allows clients to continue to cancel streams for legitimate reasons, while constraining malicious actors.

=== Impact: Medium 5.3 / 10

Table 38. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:L

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	Low

=== References

- https://github.com/apple/swift-nio-http2/security/advisories/GHSA-qppj-fm5r-hxr3
- https://nvd.nist.gov/vuln/detail/CVE-2023-44487

- https://github.com/alibaba/tengine/issues/1872
- https://github.com/caddyserver/caddy/issues/5877
- https://github.com/eclipse/jetty.project/issues/10679

=== Recommendation

Consider alternatives of this dependency.

== M-9 HTTP/2 Stream Cancellation Attack Tags: `runtime`, Weaknesses: GHSA ID: GHSA-qppj-fm5r-hxr3 CWE-400

File HalbornCTF_Golang_Cosmos/go.mod#L27-L31

```
	google.golang.org/genproto v0.0.0-20211223182754-3ac035c7e7cb
	google.golang.org/grpc v1.45.0
	google.golang.org/protobuf v1.27.1
	gopkg.in/yaml.v2 v2.4.0
)
```

=== HTTP/2 Rapid reset attack The HTTP/2 protocol allows clients to indicate to the server that a previous stream should be canceled by sending a RST_STREAM frame. The protocol does not require the client and server to coordinate the cancellation in any way, the client may do it unilaterally. The client may also assume that the cancellation will take effect immediately when the server receives the RST_STREAM frame, before any other data from that TCP connection is processed.

Abuse of this feature is called a Rapid Reset attack because it relies on the ability for an endpoint to send a RST_STREAM frame immediately after sending a request frame, which makes the other endpoint start working and then rapidly resets the request. The request is canceled, but leaves the HTTP/2 connection open.

The HTTP/2 Rapid Reset attack built on this capability is simple: The client opens a large number of streams at once as in the standard HTTP/2 attack, but rather than waiting for a response to each request stream from the server or proxy, the client cancels each request immediately.

The ability to reset streams immediately allows each connection to have an indefinite number of requests in flight. By explicitly canceling the requests, the attacker never exceeds the limit on the number of concurrent open streams. The number of in-flight requests is no longer dependent on the round-trip time (RTT), but only on the available network bandwidth.

In a typical HTTP/2 server implementation, the server will still have to do significant amounts of work for canceled requests, such as allocating new stream data structures, parsing the query and doing header decompression, and mapping the URL to a resource. For reverse proxy implementations, the request may be proxied to the backend server before the RST_STREAM frame is processed. The client on the other hand paid almost no costs for sending the requests. This creates an exploitable cost asymmetry between the server and the client.

Multiple software artifacts implementing HTTP/2 are affected. This advisory was originally ingested from the `swift-nio-http2` repo advisory and their original conent follows.

=== swift-nio-http2 specific advisory swift-nio-http2 is vulnerable to a denial-of-service vulnerability in which a malicious client can create and then reset a large number of HTTP/2 streams in a short period of time. This causes swift-nio-http2 to commit to a large amount of expensive work which it then throws away, including creating entirely new ``Channel`s to serve the traffic. This can easily overwhelm an `EventLoop`` and prevent it from making forward progress.

swift-nio-http2 1.28 contains a remediation for this issue that applies reset counter using a sliding window. This constrains the number of stream resets that may occur in a given window of time. Clients violating this limit will have their connections torn down. This allows clients to continue to cancel streams for legitimate reasons, while constraining malicious actors.

=== Impact: Medium 5.3 / 10

Table 39. CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:L

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	Low

=== References

- https://github.com/apple/swift-nio-http2/security/advisories/GHSA-qppj-fm5r-hxr3
- https://nvd.nist.gov/vuln/detail/CVE-2023-44487
- https://github.com/alibaba/tengine/issues/1872
- https://github.com/caddyserver/caddy/issues/5877
- https://github.com/eclipse/jetty.project/issues/10679

=== Recommendation

Consider alternatives of this dependency.

= Low == L-1 rootless: `/sys/fs/cgroup` is writable when cgroupns isn't unshared in runc Tags: `runtime`, Weaknesses: GHSA ID: GHSA-m8cg-xc2p-r3fc CWE-281

File HalbornCTF_Golang_Cosmos/go.mod#None

```
module github.com/cosmos/gaia/v7

go 1.17

require (
```

=== Impact It was found that rootless runc makes `/sys/fs/cgroup` writable in following conditons: 1. when runc is executed inside the user namespace, and the `config.json` does not specify the cgroup namespace to be unshared (e.g.., `(docker|podman|nerdctl) run --cgroupns=host`, with Rootless Docker/Podman/nerdctl) 2. or, when runc is executed outside the user namespace, and `/sys` is mounted with `rbind, ro` (e.g., `runc spec --rootless`; this condition is very rare)

A container may gain the write access to user-owned cgroup hierarchy

`/sys/fs/cgroup/user.slice/…` on the host . Other users's cgroup hierarchies are not affected.

=== Patches v1.1.5 (planned)

=== Workarounds - Condition 1: Unshare the cgroup namespace (`(docker|podman|nerdctl) run --cgroupns=private)`. This is the default behavior of Docker/Podman/nerdctl on cgroup v2 hosts. - Condition 2 (very rare): add `/sys/fs/cgroup` to `maskedPaths`

=== Impact: Low 2.5 / 10

Table 40. CVSS:3.1/AV:L/AC:H/PR:H/UI:N/S:C/C:N/I:N/A:L

CVSS base metrics	
Attack vector	Local
Attack complexity	High
Privileges required	High
User interaction	None
Scope	Changed
Confidentiality	None
Integrity	None
Availability	Low

=== References

- https://github.com/opencontainers/runc/security/advisories/GHSA-m8cg-xc2p-r3fc
- https://nvd.nist.gov/vuln/detail/CVE-2023-25809
- https://github.com/opencontainers/runc/commit/0d62b950e60f6980b54fe3bafd9a9c608dc1df17
- https://github.com/advisories/GHSA-m8cg-xc2p-r3fc

=== Recommendation

Consider alternatives of this dependency.

== L-2 Go package github.com/cosmos/cosmos-sdk module x/crisis does NOT cause chain halt Tags: `runtime`, Weaknesses: GHSA ID: GHSA-qfc5-6r3j-jj22

File HalbornCTF_Golang_Cosmos/go.mod#None

```
module github.com/cosmos/gaia/v7

go 1.17

require (
```

=== x/crisis does NOT cause chain halt

=== Impact If an invariant check fails on a Cosmos SDK network and a transaction is sent to the `x/crisis` module to halt the chain, the chain does not halt. All versions of the `x/crisis` module is affected on all versions of the Cosmos SDK.

=== Details The x/crisis module is supposed to allow anyone to halt a chain in the event of a violated invariant by sending a MsgVerifyInvariant with the name of the invariant. Processing this message is supposed to cause the nodes to panic. However, because the panic is within a transaction, it is caught by the SDK's built-in panic-recovery machinery and just treated as a normal "invalid" transaction (ie. it returns a non-zero abci Code). Thus the x/crisis transactions don't actually cause chains to halt. If there is an invariant violation, it can be confirmed with an x/crisis transaction, but it won't cause any nodes to halt, they will just continue processing blocks.

That said, any node running with `start --inv-check-period X` will actually panic when it runs the periodic check (though it will still not panic just by processing an x/crisis transaction). Since this panic is located in EndBlock, it is not caught by the panic-recovery machinery and does actually crash the node. Presumably few if any nodes actually run with this in production because of how long the invariant checks take, and this runs all of them every X blocks.

=== Patches No patches will be released.

The x/crisis module was originally intended to allow chains to halt rather than continue with some unknown behaviour in the case of an invariant violation (safety over liveness). However, as chains mature, and especially as the potential cost of halting increases, chains should consider carefully what invariants they really want to halt for, and what invariants are just sort of helpful sanity checks, but may not be worth halting for.

In some cases, chains have already broken the invariant calculations but have dealt with the consequences off-chain or during development. Halting these chains would be counter-productive.

The SDK team is working on new modules that allow chain developers to fine-tune the chain invariants and the necessary actions.

Hence, the decision was made that the x/crisis module will not be patched for chain halts. The module will be deprecated when new modules take over its responsibilities.

=== Workarounds In case of a valid invariant check failure that requires a chain halt, the network validators are encouraged to coordinate off-chain for network halts. This has been an already established process for security patches.

=== References SDK developer epic about invariant checking: https://github.com/cosmos/cosmos-sdk/issues/15706 Public report: https://github.com/cosmos/cosmos-sdk/issues/15325

=== References

- https://github.com/cosmos/cosmos-sdk/security/advisories/GHSA-qfc5-6r3j-jj22
- https://github.com/cosmos/cosmos-sdk/issues/15325
- https://github.com/cosmos/cosmos-sdk/issues/15706
- https://github.com/advisories/GHSA-qfc5-6r3j-jj22

=== Recommendation

Consider alternatives of this dependency.

== L-3 github.com/cosmos/cosmos-sdk's x/crisis does not charge ConstantFee Tags: runtime, Weaknesses: GHSA ID: GHSA-w5w5-2882-47pc

File HalbornCTF_Golang_Cosmos/go.mod#None

```
module github.com/cosmos/gaia/v7
```

```

go 1.17

require (
```

=== x/crisis does not charge ConstantFee === Impact If a transaction is sent to the `x/crisis` module to check an invariant, the ConstantFee parameter of the chain is NOT charged. All versions of the `x/crisis` module are affected on all versions of the Cosmos SDK.

=== Details The `x/crisis` module is supposed to allow anyone to halt a chain in the event of a violated invariant by sending a `MsgVerifyInvariant` with the name of the invariant. Processing this message takes extra processing power hence a `ConstantFee` was introduced on the chain that is charged as extra from the reporter for the extra computational work. This is supposed to avert spammers on the chain making nodes do extra computations using this transaction. By not charging the `ConstantFee`, the transactions related to invariant checking are relatively cheaper compared to the computational need and other transactions.

That said, the submitter still has to pay the transaction fee to put the transaction on the network, hence using this weakness for spamming is limited by the usual mechanisms.

Synthetic testing showed up to a 20% increase in CPU usage on a validator node that is spammed by hundreds of `MsgVerifyInvariant` messages which still makes this an expensive operation to carry out on a live blockchain network.

=== Patches The `ConstantFee` charge of the `x/crisis` module will either be fixed or disabled in an upcoming regular release of the Cosmos SDK.

The `x/crisis` module was originally intended to allow chains to halt rather than continue with some unknown behavior in the case of an invariant violation (safety over liveness). However, as chains mature, and especially as the potential cost of halting increases, chains should consider carefully what invariants they really want to halt for, and what invariants are just sort of helpful sanity checks.

The SDK team is working on new modules that allow chain developers to fine-tune the chain invariants and the necessary actions.

Hence, the decision was made that the `x/crisis` module will be deprecated when new modules take over its responsibilities.

=== Workarounds There is no workaround posted. Validators are advised to leave some extra computing room on their servers for possible spamming scenarios. (This is a good measure in any case.)

=== References SDK developer epic about invariant checking: https://github.com/cosmos/cosmos-sdk/issues/15706

=== References

- https://github.com/cosmos/cosmos-sdk/security/advisories/GHSA-w5w5-2882-47pc
- https://github.com/cosmos/cosmos-sdk/issues/15706
- https://github.com/advisories/GHSA-w5w5-2882-47pc

=== Recommendation

Consider alternatives of this dependency.

== L-4 ASA-2024-003: Missing `BlockedAddressed` Validation in Vesting Module Tags: `runtime`, Weaknesses: GHSA ID: GHSA-4j93-fm92-rp4m CWE-20

File HalbornCTF_Golang_Cosmos/go.mod#None

```
module github.com/cosmos/gaia/v7

go 1.17

require (
```

=== ASA-2024-003: Missing `BlockedAddressed` Validation in Vesting Module

Component: Cosmos SDK **Criticality**: Low **Affected Versions**: Cosmos SDK versions ¬ 0.50.3; ¬ 0.47.8 **Affected Users**: Chain developers, Validator and Node operators **Impact**: Denial of Service

=== Description

A vulnerability was identified in the `x/auth/vesting` module, which can allow a user to create a periodic vesting account on a blocked address, for example a non-initialized module account. Additional validation was added to prevent creation of a periodic vesting account in this scenario.

If this case is triggered, there is the potential for a chain halt if the uninitialized account in question is called by `GetModuleAccount` in `Begin`/`EndBlock` of a module. This combination of an uninitialized blocked module account is not common.

=== Next Steps for Impacted Parties

If your chain has uninitialized blocked module accounts, it is recommended to proactively initialize them, as they are often initialized during a chain migration or during init genesis.

If you are a chain developer on an affected version of the Cosmos SDK, it is advised to update to the latest available version of the Cosmos SDK for your project. Once a patched version is available, it is recommended that network operators upgrade.

A Github Security Advisory for this issue is available in the Cosmos-SDK repository. For more information about Cosmos SDK, see https://docs.cosmos.network/.

This issue was found by Dongsam who reported it to the Cosmos Bug Bounty Program on HackerOne on January 30, 2024. If you believe you have found a bug in the Interchain Stack or would like to contribute to the program by reporting a bug, please see https://hackerone.com/cosmos.

=== Impact: Low 3.5 / 10

Table 41. CVSS:3.1/AV:A/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:L

CVSS base metrics	
Attack vector	Adjacent
Attack complexity	Low
Privileges required	Low
User interaction	None

CVSS base metrics	
Scope	Unchange
Confidentiality	None
Integrity	None
Availability	Low

=== References

- https://github.com/cosmos/cosmos-sdk/security/advisories/GHSA-4j93-fm92-rp4m
- https://github.com/cosmos/cosmos-sdk/releases/tag/v0.47.9
- https://github.com/cosmos/cosmos-sdk/releases/tag/v0.50.4
- https://github.com/advisories/GHSA-4j93-fm92-rp4m

=== Recommendation

Consider alternatives of this dependency.

== L-5 ASA-2024-005: Potential slashing evasion during re-delegation Tags: `runtime`, Weaknesses: GHSA ID: GHSA-86h5-xcpx-cfqc CWE-372

File HalbornCTF_Golang_Cosmos/go.mod#None

```
module github.com/cosmos/gaia/v7

go 1.17

require (
```

=== ASA-2024-005: Potential slashing evasion during re-delegation

Component: Cosmos SDK **Criticality**: Low **Affected Versions**: Cosmos SDK versions ¬ 0.50.4; ¬ 0.47.9 **Affected Users**: Chain developers, Validator and Node operators **Impact**: Slashing Evasion

=== Summary

An issue was identified in the slashing mechanism that may allow for the evasion of slashing penalties during a slashing event. If a delegation contributed to byzantine behavior of a validator, and the validator has not yet been slashed, it may be possible for that delegation to evade a pending slashing penalty through re-delegation behavior. Additional validation logic was added to restrict this behavior.

=== Next Steps for Impacted Parties

If you are a chain developer on an affected version of the Cosmos SDK, it is advised to update to the latest available version of the Cosmos SDK for your project. Once a patched version is available, it is recommended that network operators upgrade.

A Github Security Advisory for this issue is available in the Cosmos-SDK repository. For more information about Cosmos SDK, see https://docs.cosmos.network/.

This issue was found by cat shark (Khanh) who reported it to the Cosmos Bug Bounty Program on

HackerOne on December 6, 2024. If you believe you have found a bug in the Interchain Stack or would like to contribute to the program by reporting a bug, please see https://hackerone.com/cosmos.

=== References

- https://github.com/cosmos/cosmos-sdk/security/advisories/GHSA-86h5-xcpx-cfqc
- https://github.com/cosmos/cosmos-sdk/commit/7dbed2fc0c3ed7c285645e21cb1037d8810372ae
- https://github.com/cosmos/cosmos-sdk/commit/d1b5b0c5ae2c51206cc1849e09e4d59986742cc3
- https://github.com/advisories/GHSA-86h5-xcpx-cfqc

=== Recommendation

Consider alternatives of this dependency.

== L-6 Lenient Parsing of Content-Length Header When Prefixed with Plus Sign

Tags: `runtime`, Weaknesses: CWE-444, CVE ID: CVE-2021-32715, GHSA ID: GHSA-f3pg-qwvg-p99c

File HalbornCTF_Rust_Substrate/Cargo.lock#L2258-L2262

```
[[package]]
name = "hyper"
version = "0.12.36"
source = "registry+https://github.com/rust-lang/crates.io-index"
checksum = "5c843caf6296fc1f93444735205af9ed4e109a539005abb2564ae1d6fad34c52"
```

To determine if vulnerable, all these things must be true:

- **Using hyper as an HTTP server**. While the lenient decoder also exists in the client, a vulnerability does not exist around *responses*.
- **Using HTTP/1**. The HTTP/2 code uses a stricter parser.
- **Using a vulnerable HTTP proxy upstream to hyper**. If an upstream proxy correctly rejects the illegal `Content-Length` header, **OR** can parse the length with the plus sign, the desync attack cannot succeed.

=== Impact: Low 3.1 / 10

Table 42. CVSS:3.1/AV:N/AC:H/PR:N/UI:R/S:U/C:L/I:N/A:N

CVSS base metrics	
Attack vector	Network
Attack complexity	High
Privileges required	None
User interaction	Required
Scope	Unchange

CVSS base metrics	
Confidentiality	Low
Integrity	None
Availability	None

=== Patches

We have released the following patch versions:

- v0.14.10 (to be released when this advisor is published)

<RECOMMENDATION>

=== References

== L-7 `tokio::io::ReadHalf<T>::unsplit` is Unsound Tags: `runtime`, Weaknesses: GHSA ID: GHSA-4q83-7cq4-p6wg

File HalbornCTF_Rust_Substrate/Cargo.lock#L7836-L7840

```
[[package]]
name = "tokio"
version = "0.1.22"
source = "registry+https://github.com/rust-lang/crates.io-index"
checksum = "5a09c0b5bb588872ab2f09afa13ee6e9dac11e10a0ec9e8e3ba39a5a5d530af6"
```

`tokio::io::ReadHalf<T>::unsplit` can violate the `Pin` contract

The soundness issue is described in the tokio/issues#5372

Specific set of conditions needed to trigger an issue (a !Unpin type in ReadHalf) is unusual, combined with the difficulty of making any arbitrary use-after-free exploitable in Rust without doing a lot of careful alignment of data types in the surrounding code.

The `tokio` feature `io-util` is also required to be enabled to trigger this soundness issue.

Thanks to zachs18 reporting the issue to Tokio team responsibly and taiki-e and carllerche appropriately responding and fixing the soundness bug.

Tokio before 0.2.0 used `futures` 0.1 that did not have `Pin`, so it is not affected by this issue.

=== References

- https://github.com/tokio-rs/tokio/issues/5372
- https://rustsec.org/advisories/RUSTSEC-2023-0005.html
- https://github.com/advisories/GHSA-4q83-7cq4-p6wg

=== Recommendation

Consider alternatives of this dependency.

== L-8 Race Condition Enabling Link Following and Time-of-check Time-of-use (TOCTOU) Race Condition in remove_dir_all Tags: `runtime`, Weaknesses: GHSA ID: GHSA-mc8h-8q98-g5hr CWE-366, CWE-367

File HalbornCTF_Rust_Substrate/Cargo.lock#L5299-L5303

```
[[package]]
name = "remove_dir_all"
version = "0.5.3"
source = "registry+https://github.com/rust-lang/crates.io-index"
checksum = "3acd125665422973a33ac9d3dd2df85edad0f4ae9b00dafb1a05e43a9f5ef8e7"
```

The `remove_dir_all` crate is a Rust library that offers additional features over the Rust standard library `fs::remove_dir_all` function. It suffers the same class of failure as the code it was layering over: TOCTOU race conditions, with the ability to cause arbitrary paths to be deleted by substituting a symlink for a path after the type of the path was checked.

Thanks to the Rust security team for identifying the problem and alerting us to it.

=== References

- https://github.com/XAMPPRocky/remove_dir_all/security/advisories/GHSA-mc8h-8q98-g5hr
- https://github.com/XAMPPRocky/remove_dir_all/commit/7247a8b6ee59fc99bbb69ca6b3ca4bfd8c809ead
- https://rustsec.org/advisories/RUSTSEC-2023-0018.html
- https://github.com/advisories/GHSA-mc8h-8q98-g5hr

=== Recommendation

Consider alternatives of this dependency.

== L-9 Undefined Behavior in Rust runtime functions

Tags: `runtime`, Weaknesses: CWE-758, CVE ID: CVE-2023-30624, GHSA ID: GHSA-ch89-5g45-qwc7

File HalbornCTF_Rust_Substrate/Cargo.lock#L8589-L8593

```
[[package]]
name = "wasmtime"
version = "0.22.0"
source = "registry+https://github.com/rust-lang/crates.io-index"
checksum = "7426055cb92bd9a1e9469b48154d8d6119cd8c498c8b70284e420342c05dc45d"
```

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.

=== Impact: Low 3.9 / 10

Table 43. CVSS:3.1/AV:N/AC:H/PR:H/UI:R/S:U/C:L/I:L/A:L

CVSS base metrics	
Attack vector	Network
Attack complexity	High
Privileges required	High
User interaction	Required
Scope	Unchange
Confidentiality	Low
Integrity	Low
Availability	Low

=== Patches

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.

<RECOMMENDATION>

=== References

- GitHub Advisory
- Mailing list announcement

== L-10 atty potential unaligned read Tags: `runtime`, Weaknesses: GHSA ID: GHSA-g98v-hv3f-hcfr

File HalbornCTF_Rust_Substrate/Cargo.lock#L389-L393

```
[[package]]
name = "atty"
version = "0.2.14"
source = "registry+https://github.com/rust-lang/crates.io-index"
checksum =
"d9b39be18770d11421cdb1b9947a45dd3f37e93092cbf377614828a319d5fee8
"
```

On windows, `atty` dereferences a potentially unaligned pointer.

In practice however, the pointer won't be unaligned unless a custom global allocator is used.

In particular, the `System` allocator on windows uses `HeapAlloc`, which guarantees a large enough alignment.

=== atty is Unmaintained

A Pull Request with a fix has been provided over a year ago but the maintainer seems to be unreachable.

Last release of `atty` was almost 3 years ago.

=== Possible Alternative(s)

The below list has not been vetted in any way and may or may not contain alternatives;

- std::io::IsTerminal - Stable since Rust 1.70.0
- is-terminal - Standalone crate supporting Rust older than 1.70.0"

=== References

- https://github.com/softprops/atty/issues/50
- https://github.com/softprops/atty/pull/51
- https://rustsec.org/advisories/RUSTSEC-2021-0145.html
- https://github.com/advisories/GHSA-g98v-hv3f-hcfr

=== Recommendation

Consider alternatives of this dependency.

== L-11 wasmtime_trap_code C API function has out of bounds write vulnerability

Tags: `runtime`, Weaknesses: CWE-787, CVE ID: CVE-2022-39394, GHSA ID: GHSA-h84q-m8rr-3v9q

File HalbornCTF_Rust_Substrate/Cargo.lock#L8589-L8593

```
[[package]]
name = "wasmtime"
version = "0.22.0"
source = "registry+https://github.com/rust-lang/crates.io-index"
checksum =
"7426055cb92bd9a1e9469b48154d8d6119cd8c498c8b70284e420342c05dc45d
"
```

There is a bug in Wasmtime's C API implementation where the definition of the `wasmtime_trap_code` does not match its declared signature in the `wasmtime/trap.h` header file. This discrepancy causes the function implementation to perform a 4-byte write into a 1-byte buffer provided by the caller. This can lead to three zero bytes being written beyond the 1-byte location provided by the caller.

=== Impact: Low 3.8 / 10

Table 44. CVSS:3.1/AV:L/AC:H/PR:H/UI:R/S:U/C:L/I:L/A:L

CVSS base metrics	
Attack vector	Local
Attack complexity	High
Privileges required	High
User interaction	Required
Scope	Unchange
Confidentiality	Low
Integrity	Low
Availability	Low

=== Patches

This bug has been patched and users should upgrade to Wasmtime 2.0.2.

<RECOMMENDATION>

=== References

- Definition of `wasmtime_trap_code`
- Mailing list announcement
- Patch to fix for `main` branch

== Tools Used

=== SARIF

=== Certora

=== Wake

=== Foundry

== Challenges

=== Context

= Terminology

EVM

also known as Etherum virtual machine, is a turing-complete virtual machine that executes smart contract code on a stack with a depth of 1024 items

Opcodes

Operations codes that run instructions on the Ethereum Virtual Machine

Intrinsic Gas Costs

amount of gas paid prior to execution of a transaction, the gas paid by the initiator of a transaction, which will always be an externally-owned account, before any state updates are made or any code is executed

.DS_Store

In the Apple macOS operating system, `.DS_Store` is a file that stores custom attributes of its containing folder, such as folder view options, icon positions, and other visual information. The name is an abbreviation of Desktop Services Store, reflecting its purpose. It is created and maintained by the Finder application in every folder, and has functions similar to the file desktop.ini in Microsoft Windows.

= References

- [] ETHEREUM: A Secure Decentralised Generalised Transaction Ledger: Dr. Gavin Wood Founder, Ethereum & Parity
- [] https://github.com/gehaxelt/Python-dsstore
- [] https://0day.work/parsing-the-ds_store-file-format/