Audit Report

Prism Auto Compounding cAsset

v1.0

November 4, 2022

Table of Contents

License

Disclaimer

This audit has been performed by

Oak Security

https://oaksecurity.io/
info@oaksecurity.io

Introduction

Purpose of This Report

Oak Security has been engaged by Terraform Labs Limited to perform a security audit of the Prism Auto Compounding cAsset smart contracts.

The objectives of the audit are as follows:

1. Determine the correct functioning of the protocol, in accordance with the project specification.
2. Determine possible vulnerabilities, which could be exploited by an attacker.
3. Determine smart contract bugs, which might lead to unexpected behavior.
4. Analyze whether best practices have been applied during development.
5. Make recommendations to improve code safety and readability.

This report represents a summary of the findings.

As with any code audit, there is a limit to which vulnerabilities can be found, and unexpected execution paths may still be possible. The author of this report does not guarantee complete coverage (see disclaimer).

Codebase Submitted for the Audit

The audit has been performed on the following GitHub repository:

https://github.com/prism-finance/prism-auto-compounding-casset

Commit hash: `005fafa5be0003875bdcb0a56ad5c0b53a5ea40c`

Methodology

The audit has been performed in the following steps:

1. Gaining an understanding of the code base's intended purpose by reading the available documentation.
2. Automated source code and dependency analysis.
3. Manual line by line analysis of the source code for security vulnerabilities and use of best practice guidelines, including but not limited to:
    a. Race condition analysis
    b. Under-/overflow issues
    c. Key management vulnerabilities
4. Report preparation

Functionality Overview

Prism Auto Compounding cAsset is a protocol that lets users delegate coins to validators with liquid staking and reward auto compounding features.
The audit scope includes the Prism Hub smart contract, which is responsible for protocol operations both for users and administrators.

How to Read This Report

This report classifies the issues found into the following severity categories:

Severity	Description
Critical	A serious and exploitable vulnerability that can lead to loss of funds, unrecoverable locked funds, or catastrophic denial of service.
Major	A vulnerability or bug that can affect the correct functioning of the system, lead to incorrect states or denial of service.
Minor	A violation of common best practices or incorrect usage of primitives, which may not currently have a major impact on security, but may do so in the future or introduce inefficiencies.
Informational	Comments and recommendations of design decisions or potential optimizations, that are not relevant to security. Their application may improve aspects, such as user experience or readability, but is not strictly necessary. This category may also include opinionated recommendations that the project team might not share.

The status of an issue can be one of the following: **Pending, Acknowledged**, or **Resolved**.

Note that audits are an important step to improving the security of smart contracts and can find many issues. However, auditing complex codebases has its limits and a remaining risk is present (see disclaimer).

Users of the system should exercise caution. In order to help with the evaluation of the remaining risk, we provide a measure of the following key indicators: **code complexity**, **code readability**, **level of documentation**, and **test coverage**. We include a table with these criteria below.

Note that high complexity or low test coverage does not necessarily equate to a higher risk, although certain bugs are more easily detected in unit testing than in a security audit and vice versa.

Summary of Findings

No	Description	Severity	Status
1	`UpdateExchangeRate` message leads to delegation of incorrect amount of funds	Major	Resolved
2	State update not stored	Minor	Resolved
3	Custom access control implementation is error-prone and decreases maintainability	Minor	Resolved
4	Lack of configuration parameter validation	Minor	Partially Resolved
5	`uluna` coin denom is used instead of `underlying_coin_denom`	Minor	Partially Resolved
6	Pseudorandom validator selection can be predicted and gamed by validators	Minor	Acknowledged
7	WithdrawUnbonded transaction requires the execution of multiple unbounded loops	Minor	Acknowledged
8	Updating `token_contract` will affect previously bonded funds	Minor	Resolved
9	Outdated and unmaintained dependencies in use	Informational	Acknowledged
10	Custom functionality for checking additional funds	Informational	Acknowledged
11	Canonical address transformations are inefficient	Informational	Acknowledged
12	Missing tax deductions	Informational	Acknowledged
13	Typographical errors	Informational	Resolved
14	Duplicated code can negatively impact maintainability	Informational	Resolved
15	Unused code	Informational	Resolved
16	Comment contradicts implementation	Informational	Resolved
17	Lack of address validation upon querying	Informational	Resolved
18	Inefficiency in querying a specific validator	Informational	Acknowledged
19	Lack of pausing mechanism	Informational	Partially Resolved

Code Quality Criteria

Criteria	Status	Comment
Code complexity	**Medium**	-
Code readability and clarity	**Medium-High**	-
Level of documentation	**Low**	No documentation was provided
Test coverage	**High**	Code coverage of 94.19% according to `cargo-tarpaulin`

Detailed Findings

1. `UpdateExchangeRate` message leads to delegation of incorrect amount of funds

Severity: Major

The `execute_update_exchange_rate` function tries to delegate more than it should, as it doesn't take into account the `protocol_fee` that needs to be accrued in case it is not zero. This causes `ExecuteMsg::UpdateExchangeRate` to error if the contract balance is less than the amount to be delegated plus the protocol fee.

The variable `claimed_rewards` is incorrectly used in `contracts/prism_hub/src/autho_compounding.rs:87`, since it includes the `protocol_fee` that is previously transferred to the `fee_collector` in `contracts/prism_hub/src/autho_compounding.rs:66`. Therefore, the message moving fee funds will be executed first and then the delegation message will try to delegate an amount bigger than the actual balance, as `protocol_fee` has already been transferred.

A proof of concept test case for this security issue can be found at Appendix 1.

Recommendation

We recommend substituting the value of `amount` in the `StakingMsg::Delegate` message with `user_rewards`, which is set in `contracts/prism_hub/src/autho_compounding.rs:74` as `user_rewards = claimed_rewards - protocol_fee`.

Status: Resolved

2. State update not stored

Severity: Minor

The `execute_update_global` function uses the incorrect instance of the loaded `STATE` to update the `principle_balance_before_exchange_update` element of the struct. Although at the moment the code base does not make use of this element, it could dramatically affect future features as an outdated value would result in incorrect results when used as part of calculations.

Instead of changing the value of `last_state` in `contracts/prism_hub/src/contract.rs:213`, the code changes the value that is loaded in the initial state but never stored. Consequently, `principle_balance_before_exchange_update` will never be updated.

Please note that we consider this issue to be a minor issue since the affected variable is not used. In a future release, this issue might have critical consequences.

Recommendation

We recommend modifying `contracts/prism_hub/src/contract.rs:213`, substituting `state.principle_balance_before_exchange_update` for `last_state.principle_balance_before_exchange_update`.

Status: Resolved

3. Custom access control implementation is error-prone and decreases maintainability

Severity: Minor

The Hub contract implements custom access controls, which introduce a few issues:

The transfer of the `creator` role to a different account is implemented in a one-step fashion without confirmation from the receiving party. This could potentially cause a loss of access to the role in case a mistake is made during the role transfer.

In addition, the Access Control logic that enforces these restrictions is duplicated across the handlers of each function, which negatively impacts the code's readability and maintainability, as it is error-prone.

Recommendation

We recommend making use of a well-known access control implementation such as `cw_controllers::Admin` (https://docs.rs/cw-controllers/0.14.0/cw_controllers/struct.Admin.html) and to implement the mentioned two-step role transfer. The flow can be as follows:

1. The current `creator` proposes a new `creator` address that is validated and stored.
2. The new `creator` account claims ownership, which replaces the previous `creator`.

Status: Resolved

4. Lack of configuration parameter validation

Severity: Minor

The Hub contract lacks validation steps on most configuration parameters upon instantiation or update. Although the values are supplied by the owner, this could affect the well-functioning of the protocol if an unexpected value were to be assigned by mistake or if a rate parameter were assigned a value outside of the 0 to 1 range. For example, a fee rate of 1 will not make the protocol usable, while a fee rate greater than 1 will lead to the protocol losing value.

The following lines are affected:

- `contracts/prism_hub/src/config.rs:35-40`
- `contracts/prism_hub/src/contract.rs:70-75`

Recommendation

Thorough value validation is recommended on all the configuration parameters, especially for values that need to be within certain bounds.

Status: Partially Resolved

A validation function has been implemented, however, the `peg_recovery_fee` parameter is still not validated.

In addition, this function is called within the `instantiate` function but not on `execute_update_params`. Therefore any updated detail will not undergo the intended validation.

5. `uluna` coin denom is used instead of `underlying_coin_denom`

Severity: Minor

The Hub contract defines in the `Parameters` struct the `underlying_coin_denom` field. This parameter represents the `denom` of the coin that is delegated by the contract to validators.

However in lines:

- `contracts/prism_hub/src/autho_compounding.rs:68`
- `contracts/prism_hub/src/autho_compounding.rs:89`
- `contracts/prism_hub/src/contract.rs:44`

instead of using the correct denom saved in the store, `uluna` is hardcoded.

This implies that the contract will work on chains with a `uluna` denom, such as Terra.

Recommendation

We recommend removing the hardcoded denoms and using the stored `underlying_coin_denom` instead.

Status: Partially Resolved

Two of the affected instances have been successfully fixed. However, the last one that is part of an error message still includes a hardcoded “uluna” string.

6. Pseudorandom validator selection can be predicted and gamed by validators

Severity: Minor

In order to fairly select a validator during auto compounding, unbonding, and validator delisting, the protocol is using `XorShiftRng` with the current block height as a seed to generate a pseudo-random index.

This calculation at a certain block height is easy to predict. Validators could use this information to try to execute a specific transaction at a specific block height in order to be selected, in case of auto compounding, or not be selected in case of unbonding and validator delisting.

Recommendation

We recommend using external entropy for the random number generator, for example from an oracle or from another protocol the validator cannot influence. Alternatively, we recommend changing the validator selection to pick the validator with the lowest delegation amount, or equalizing the funds across validators.

Status: Acknowledged

The Prism team stated that they were aware of the problem and that it will be fixed in a future release.

7. `WithdrawUnbonded` transaction requires the execution of multiple unbounded loops

Severity: Minor

In `contracts/prism_hub/src/unbond.rs:146`, the `execute_withdraw_unbonded` function handles `WithdrawUnbonded` messages running multiple unbounded loops.

In line `168` the `process_withdraw_rate` function executes two unbounded loops over all not processed yet `UnbondHistory`.

In line `170` the `get_finished_amount` function executes an unbounded loop over the user unbounding waitlist.

In line `180` and `181` the `get_unbond_batches` and `remove_unbond_wait_list` functions execute an unbounded loop over the user unbonding waitlist.

This implies that under specific conditions, for example if the user has a lot of pending batches to unbond in the whitelist or if the user is another contract that does a big number of operations, the execution can run out of gas and the user would not be able to withdraw their funds.

We consider this issue to only be of minor severity since it is unlikely to occur in the short to mid-term.

Recommendation

We recommend removing unbounded iterations from the `execute_withdraw_unbonded` function by re-architecting the logic, using a limit to the number of entries that can be added, or allowing paginated withdrawals.

Status: Acknowledged

8. Updating `token_contract` will affect previously bonded funds

Severity: Minor

The admin can update the `token_contract` parameter in `contracts/prism_hub/src/config.rs:74`.

As the entire protocol state relies on information that depends on that parameter or queries using that parameter, any update will affect users that have their funds previously bonded by not allowing them to unbond.

We classify this issue as minor since only the owner can cause it.

Recommendation

We recommend removing the ability to update the `token_contract` parameter.

Status: Resolved

9. Outdated and unmaintained dependencies in use

Severity: Informational

The Hub contract uses outdated versions of multiple libraries, including a `cosmwasm-vm` version that lacks some important upgrades. In addition, two libraries were affected by publicly known vulnerabilities `CVE-2021-32810` and `CVE-2020-35880`.

Although `bigint` is not maintained and therefore there is no fix available, `crossbeam-deque` affected by `CVE-2021-32810` has an official patch ready.

Further details can be found in Appendix 2.

Recommendation

As a general rule, we recommend using the latest version of the libraries in scope unless there is a clear reason not to do so. This will guarantee that the dependencies' known bugs and vulnerabilities patches are in place.

Specifically, the core CosmWasm libraries and `crossbeam-deque` should be upgraded.

Status: Acknowledged

10. Custom functionality for checking additional funds

Severity: Informational

The Hub contract makes use of custom functionality to check for additional funds being sent in `contracts/prism_hub/src/bond.rs:35-47` and `contracts/prism_hub/src/contract.rs:39-45`. Although not a security issue, well-known community-driven libraries are recommended for this kind of feature.

Recommendation

We recommend using https://docs.rs/cw-utils/latest/cw_utils/fn.must_pay.html instead of the custom logic.

Status: Acknowledged

11. Canonical address transformations are inefficient

Severity: Informational

While previously recommended as a best practice, usage of canonical addresses for storage is no longer encouraged. The background is that canonical addresses are no longer stored in a canonical format, so the transformation just adds overhead without much benefit. Additionally, the codebase is more complicated with address transformations.

Recommendation

We recommend removing any transformation from human to canonical addresses and using the new `Addr` type and address validation instead if CosmWasm 1.0.0 is in place. If not, the address should be lower-cased before storing.

Status: Acknowledged

12. Missing tax deductions

Severity: Informational

While Terra's tax rate has been set to zero, the tax mechanism is still implemented and the rate might be increased again in the future. It is still best practice to include functionality to deduct taxes.

A non-zero tax rate could be reinstated via a governance proposal due to circumstances where the expected income from the tax rewards increases significantly. In this situation, stablecoin transactions on Terra would expand to a state where a meaningful portion of the staking rewards income is derived from tax rewards rather than the vast majority coming from swap fees.

We consider this to only be a minor issue since the contract owner can recover from tax mismatches by simply sending funds back to the contract. Additionally, the likelihood of the Terra team to increase taxes again is low.

See this discussion for more details about the tax rate changes on Terra.

Recommendation

We recommend implementing a tax rate query and deducting taxes from native assets to ensure future compatibility.

Status: Acknowledged

13. Typographical errors

Severity: Informational

In `contracts/prism_hub/src/contract.rs:51` and other places, the wrongly typed `porotcol_fee_collector` occurs, which should be `protocol_fee_collector`.

The same misspelling will also introduce a cascading effect when other protocols query the `config`.

In addition, in `contracts/prism_hub/src/config.rs:139`, the configuration is loaded for a function named `token`, which is potentially another typo.

Recommendation

We recommend correcting these typos.

Status: Resolved

14. Duplicated code can negatively impact maintainability

Severity: Informational

The `read_validators` and `read_valid_validators` functions in `contracts/prism_hub/src/state.rs:176` and `20` are performing the same logic. Additionally, all validators are valid in the Prism Auto Compounding cAsset protocol. Code duplication increases code complexity and can negatively impact maintainability.

Recommendation

We recommend removing one of these duplicated functions.

Status: Resolved

15. Unused code

Severity: Informational

Multiple instances of unused code exist in the codebase, including variables and functions that are only used within tests, but not in the actual contract. Unused code affects the overall code readability, and might confuse users since it is suggestive of any features that are currently not being implemented.

The following list includes the affected lines:

- `contracts/prism_hub/src/state.rs:10,53-63`
- `contracts/prism_hub/src/contract.rs:36,37,58`
- `contracts/prism_hub/src/hub.rs:160,163`
    - The `UnbondHistory` struct includes elements that are not in use. `batch_id` is not used as it is the actual key in the history storage instance. Similarly, `applied_exchange_rate` and `withdraw_rate` seem to reflect the same value here, however, `applied_exchange_rate` is only used inside tests and not the actual contract.

Recommendation

We recommend removing unused code.

Status: Resolved

16. Comment contradicts implementation

Severity: Informational

The Hub contract includes comments describing the required privileges for each function assigned as a handler of an execution message. The comment for `execute_register_validator` in `contracts/prism_hub/src/config.rs:96` states that the functionality is callable only by the `creator`. However, this differs from the implementation as the Hub contract itself is whitelisted too, given that this message is sent by the contract upon instantiation.

Recommendation

We recommend updating the comment to reflect the actual requirements of the functionality.

Status: Resolved

17. Lack of address validation upon querying

Severity: Informational

The Hub contract doesn't validate the input address of the `query_unbond_requests` function. Although not being a security risk, it may lead to a lowered user experience if incomplete or invalid addresses are provided and no address validation error is returned.

Recommendation

We recommend performing validation on input addresses of `QueryMsg` messages too.

Status: Resolved

18. Inefficiency in querying a specific validator

Severity: Informational

In `contract/prism_hub/src/config.rs:112`, the execution is querying the list of all validators in order to check if the provided one is part of the list.

Instead of executing `query_all_validators` and then iterating through the list in order to find the required one, which has a O(n) complexity, `query_validator` could be used to directly query the required one.

Recommendation

We recommend using `query_validator` instead of `query_all_validators`.

Status: Acknowledged

19. Lack of pausing mechanism

Severity: Informational

Currently when the smart contract is deployed, if there is any catastrophic security loophole being discovered there is no quick way to limit its functionality and restrict the potential impact of the vulnerability while the team carries out an investigation/working on an upgrade.

Recommendation

We suggest adding the ability to pause/unpause the contract at any time. This pausing feature should act in a similar way to a time-lock, requiring a duration to be submitted and automatically expiring to avoid leaving the contract unusable in the event of a compromised owner key.

Additionally, it is recommended to follow a Role Based Access Controls philosophy by adding a second privileged pauser role that will be in charge of the feature, instead of restricting access to the owner.

Status: Partially Resolved

New features to Pause/Unpause the contract have been implemented. However, no duration has been included. In addition, the pausing feature is controlled by the admin instead of an additional “pauser” role as mentioned in the recommendation.

Appendix

Appendix 1: Test case for `UpdateExchangeRate` message leads to delegation of incorrect amount of funds

```
pub fn update_exchange_finding() {
    let mut deps = dependencies(&[]);
    let validator = sample_validator(DEFAULT_VALIDATOR.to_string());
    set_validator_mock(&mut deps.querier);

    let alice = "addr1000".to_string();
    let bond_amount = Uint128::new(10);

    let owner = "owner1".to_string();
    let token_contract = "token".to_string();
    let fee_collector_contract = "fee_collector".to_string();

    init2_audit(
        deps.borrow_mut(),
        owner,
        token_contract,
        fee_collector_contract.clone(),
        validator.address.clone(),
    );

    deps.querier.with_token_balances(&[(
        &"token".to_string(),
        &[(&MOCK_CONTRACT_ADDR.to_string(), &INITIAL_DEPOSIT_AMOUNT)],
    )]);

    // register_validator
    do_register_validator(deps.as_mut(), validator.clone());
    // bond
    do_bond(deps.as_mut(), alice.clone(), bond_amount,
validator.clone());

    //set delegation for query-all-delegation
    let delegations: [FullDelegation; 1] =
        [(sample_delegation(validator.address.clone(),
coin(bond_amount.u128(), "uluna")))];

    let validators: [Validator; 1] = [(validator.clone())];

    set_delegation_query(&mut deps.querier, &delegations, &validators);
    // Set balance of 100 uluna
    deps.querier.with_native_balances(&[(
        MOCK_CONTRACT_ADDR.to_string(),
        Coin {
            denom: "uluna".to_string(),
            amount: Uint128::new(100),
        },
    )]);
    let update_exchange_rate = ExecuteMsg::UpdateExchangeRate {};

    let info = mock_info(MOCK_CONTRACT_ADDR, &[]);
    let res = execute(deps.as_mut(), mock_env(), info,
update_exchange_rate).unwrap();

    assert_eq!(res.messages.len(), 2);

    // A 10% fee is transferred to the collector
    assert_eq!(
        res.messages[0],
        SubMsg::new(CosmosMsg::Bank(BankMsg::Send {
            to_address: fee_collector_contract.clone(),
            amount: vec![Coin::new(10u128, "uluna")],
        }))
    );

    // The whole intial balance of 100 is then delegated, which will not
be in the contract after the transfer above
    assert_eq!(
        res.messages[1],
        SubMsg::new(CosmosMsg::Staking(StakingMsg::Delegate {
            validator: validator.address,
            amount: Coin::new(100, "uluna"), //lol?
        }))
    );

}

pub fn init2_audit<S: Storage, A: Api, Q: Querier>(
    deps: &mut OwnedDeps<S, A, Q>,
    owner: String,
    token_contract: String,
    fee_contract: String,
    validator: String,
) {
    let msg = InstantiateMsg {
        epoch_period: 30,
        underlying_coin_denom: "uluna".to_string(),
        unbonding_period: 2,
        peg_recovery_fee: Decimal::zero(),
        er_threshold: Decimal::one(),
        validator,
        protocol_fee:
Decimal::from_ratio(Uint128::new(10),Uint128::new(100)),
    };

    let owner_info = mock_info(owner.as_str(), &[coin(1000000,
"uluna")]);
    instantiate(deps.as_mut(), mock_env(), owner_info.clone(),
msg).unwrap();

    let register_msg = UpdateConfig {
        owner: None,
        token_contract: Some(token_contract),
        protocol_fee_collector: Some(fee_contract),
    };

    let res = execute(deps.as_mut(), mock_env(), owner_info,
register_msg).unwrap();
    assert_eq!(0, res.messages.len());
}
```

Appendix 2: Detailed list of outdated issues for [Outdated and unmaintained dependencies in use]

```
Name                          Project                        Compat  Latest                         Kind         Platform
----                          -------                        ------  ------                         ----         --------
autocfg                       1.0.1                          ---     Removed                        Build        ---
bitflags                      1.2.1                          ---     Removed                        Normal       ---
cfg-if                        1.0.0                          ---     Removed                        Normal       ---
cloudabi                      0.0.3                          ---     Removed                        Normal       cfg(target_os =
"cloudabi")
const-oid                     0.6.0                          ---     0.7.1                          Normal       ---
const-oid                     0.6.0                          ---     Removed                        Normal       ---
cosmwasm-crypto               0.16.0                         ---     1.0.0                          Normal       cfg(not(target_arch
= "wasm32"))
cosmwasm-derive               0.16.0                         ---     1.0.0                          Normal       ---
cosmwasm-schema               0.16.0                         ---     1.0.0                          Development  ---
cosmwasm-std                  0.16.0                         ---     1.0.0                          Normal       ---
cosmwasm-vm                   0.16.0                         ---     1.0.0                          Development  ---
cranelift-bforest             0.74.0                         ---     0.76.0                         Normal       ---
cranelift-codegen             0.74.0                         ---     0.76.0                         Normal       ---
cranelift-codegen-meta        0.74.0                         ---     0.76.0                         Build        ---
cranelift-codegen-shared      0.74.0                         ---     0.76.0                         Normal       ---
cranelift-entity              0.74.0                         ---     0.76.0                         Normal       ---
cranelift-frontend            0.74.0                         ---     0.76.0                         Normal       ---
crc32fast                     1.2.1                          ---     Removed                        Normal       ---
crypto-bigint                 0.2.2                          ---     0.3.2                          Normal       ---
crypto-mac                    0.11.1                         ---     Removed                        Normal       ---
cw-storage-plus               0.13.4                         ---     0.14.0                         Normal       ---
cw-utils                      0.13.4                         ---     0.14.0                         Normal       ---
cw2                           0.13.4                         ---     0.14.0                         Normal       ---
cw20                          0.13.4                         ---     0.14.0                         Normal       ---
cw20-base                     0.13.4                         ---     0.14.0                         Normal       ---
der                           0.4.1                          ---     Removed                        Normal       ---
digest                        0.9.0                          ---     Removed                        Normal       ---
dynasm                        1.1.0                          ---     1.2.3                          Normal       ---
dynasmrt                      1.1.0                          ---     1.2.3                          Normal       ---
ecdsa                         0.12.3                         ---     0.13.4                         Normal       ---
ed25519-zebra                 2.2.0                          ---     3.0.0                          Normal       ---
elliptic-curve                0.10.5                         ---     0.11.12                        Normal       ---
fallible-iterator             0.2.0                          ---     Removed                        Normal       ---
ff                            0.10.0                         ---     0.11.1                         Normal       ---
fuchsia-cprng                 0.1.1                          ---     Removed                        Normal       cfg(target_os =
"fuchsia")
generic-array                 0.14.4                         ---     Removed                        Normal       ---
getrandom                     0.2.3                          ---     Removed                        Normal       ---
gimli                         0.24.0                         ---     0.25.0                         Normal       ---
group                         0.10.0                         ---     0.11.0                         Normal       ---
hashbrown                     0.11.2                         ---     Removed                        Normal       ---
hmac                          0.11.0                         ---     Removed                        Normal       ---
indexmap                      1.7.0                          ---     Removed                        Normal       ---
k256                          0.9.6                          ---     0.10.4                         Normal       ---
libc                          0.2.98                         ---     Removed                        Normal       cfg(unix)
memchr                        2.4.0                          ---     2.5.0                          Normal       ---
memmap2                       0.2.3                          ---     0.5.7                          Normal       ---
memoffset                     0.6.4                          ---     Removed                        Normal       ---
object                        0.25.3                         ---     0.28.4                         Normal       ---
pkcs8                         0.7.5                          ---     Removed                        Normal       ---
proc-macro2                   1.0.28                         ---     1.0.43                         Normal       ---
quote                         1.0.9                          ---     Removed                        Normal       ---
rand                          0.8.4                          ---     0.8.5                          Normal       ---
rand_core                     0.3.1                          ---     0.6.3                          Normal       ---
rand_hc                       0.3.1                          ---     Removed                        Development  ---
region                        2.2.0                          ---     3.0.0                          Normal       ---
rkyv                          0.6.7                          ---     0.7.39                         Normal       ---
rkyv_derive                   0.6.7                          ---     0.7.39                         Normal       ---
schemars                      0.8.3                          ---     0.8.10                         Normal       ---
schemars_derive               0.8.3                          ---     0.8.10                         Normal       ---
serde                         1.0.126                        ---     1.0.144                        Normal       ---
serde-json-wasm               0.3.1                          ---     0.4.1                          Normal       ---
serde_derive                  1.0.126                        ---     1.0.144                        Normal       ---
serde_derive_internals        0.25.0                         ---     0.26.0                         Normal       ---
snafu                         0.6.10                         ---     0.7.1                          Normal       ---
snafu-derive                  0.6.10                         ---     0.7.1                          Normal       ---
spki                          0.4.0                          ---     Removed                        Normal       ---
stable_deref_trait            1.2.0                          ---     Removed                        Normal       ---
subtle                        2.4.1                          ---     Removed                        Normal       ---
syn                           1.0.74                         ---     1.0.99                         Normal       ---
target-lexicon                0.12.1                         ---     0.12.4                         Normal       ---
thiserror                     1.0.26                         ---     1.0.33                         Normal       ---
```

```
thiserror-impl              1.0.26                        ---     1.0.33                        Normal       ---
typenum                     1.13.0                        ---     Removed                       Normal       ---
unicode-xid                 0.2.2                         ---     Removed                       Normal       ---
version_check               0.9.3                         ---     Removed                       Build        ---
wasi                        0.9.0+wasi-snapshot-preview1  ---     0.10.2+wasi-snapshot-preview1 Normal       cfg(target_os =
"wasi")
wasmer                      2.0.0                         ---     2.2.1                         Normal       ---
wasmer-compiler             2.0.0                         ---     2.2.1                         Normal       ---
wasmer-compiler-cranelift   2.0.0                         ---     2.2.1                         Normal       ---
wasmer-compiler-singlepass  2.0.0                         ---     2.2.1                         Normal       ---
wasmer-derive               2.0.0                         ---     2.2.1                         Normal       ---
wasmer-engine               2.0.0                         ---     2.2.1                         Normal       ---
wasmer-engine-dylib         2.0.0                         ---     2.2.1                         Normal       ---
wasmer-engine-universal     2.0.0                         ---     2.2.1                         Normal       ---
wasmer-middlewares          2.0.0                         ---     2.2.1                         Normal       ---
wasmer-object               2.0.0                         ---     2.2.1                         Normal       ---
wasmer-types                2.0.0                         ---     2.2.1                         Normal       ---
wasmer-vm                   2.0.0                         ---     2.2.1                         Normal       ---
winapi                      0.3.9                         ---     Removed                       Normal       cfg(windows)
winapi-i686-pc-windows-gnu  0.4.0                         ---     Removed                       Normal       i686-pc-windows-gnu
winapi-x86_64-pc-windows-gnu 0.4.0                        ---     Removed                       Normal
x86_64-pc-windows-gnu
```