CAIRO SECURITY CLAN

PRAGMA HYPERLANE

SECURITY ASSESMENT REPORT

NOVEMBER 2024

Prepared for

PRAGMA

Contents

1 About Cairo Security Clan

Cairo Security Clan is a leading force in the realm of blockchain security, dedicated to fortifying the foundations of the digital age. As pioneers in the field, we specialize in conducting meticulous smart contract security audits, ensuring the integrity and reliability of decentralized applications built on blockchain technology.

At Cairo Security Clan, we boast a multidisciplinary team of seasoned professionals proficient in blockchain security, cryptography, and software engineering. With a firm commitment to excellence, our experts delve into every aspect of the Web3 ecosystem, from foundational layer protocols to application-layer development. Our comprehensive suite of services encompasses smart contract audits, formal verification, and real-time monitoring, offering unparalleled protection against potential vulnerabilities.

Our team comprises industry veterans and scholars with extensive academic backgrounds and practical experience. Armed with advanced methodologies and cutting-edge tools, we scrutinize and analyze complex smart contracts with precision and rigor. Our track record speaks volumes, with a plethora of published research papers and citations, demonstrating our unwavering dedication to advancing the field of blockchain security.

At Cairo Security Clan, we prioritize collaboration and transparency, fostering meaningful partnerships with our clients. We believe in a customer-oriented approach, engaging stakeholders at every stage of the auditing process. By maintaining open lines of communication and soliciting client feedback, we ensure that our solutions are tailored to meet the unique needs and objectives of each project.

Beyond our core services, Cairo Security Clan is committed to driving innovation and shaping the future of blockchain technology. As active contributors to the ecosystem, we participate in the development of emerging technologies such as Starknet, leveraging our expertise to build robust infrastructure and tools. Through strategic guidance and support, we empower our partners to navigate the complexities of the blockchain landscape with confidence and clarity.

In summary, Cairo Security Clan stands at the forefront of blockchain security, blending technical prowess with a client-centric ethos to deliver unparalleled protection and peace of mind in an ever-evolving digital landscape. Join us in safeguarding the future of decentralized finance and digital assets with confidence and conviction.

2 Disclaimer

Disclaimer Limitations of this Audit:

This report is based solely on the materials and documentation provided by you to Cairo Security Clan for the specific purpose of conducting the security review outlined in the Summary of Audit and Scoped Files. The findings presented here may not be exhaustive and may not identify all potential vulnerabilities. Cairo Security Clan provides this review and report on an "as-is" and "as-available" basis. You acknowledge that your use of this report, including any associated services, products, protocols, platforms, content, and materials, occurs entirely at your own risk.

Inherent Risks of Blockchain Technology:

Blockchain technology remains in its developmental stage and is inherently susceptible to unknown risks and vulnerabilities. This review is specifically focused on the smart contract code and does not extend to the compiler layer, programming language elements beyond the reviewed code, or other potential security risks outside the code itself.

Report Purpose and Reliance:

This report should not be construed as an endorsement of any specific project or team, nor does it guarantee the absolute security of the audited smart contracts. No third party should rely on this report for any purpose, including making investment or purchasing decisions.

Liability Disclaimer:

To the fullest extent permitted by law, Cairo Security Clan disclaims all liability associated with this report, its contents, and any related services and products arising from your use. This includes, but is not limited to, implied warranties of merchantability, fitness for a particular purpose, and non-infringement.

Third-Party Products and Services:

Cairo Security Clan does not warrant, endorse, guarantee, or assume responsibility for any products or services advertised by third parties within this report, nor for any open-source or third-party software, code, libraries, materials, or information linked to, referenced by, or accessible through this report, its content, and related services and products. This includes any hyperlinked websites, websites or applications appearing on advertisements, and Cairo Security Clan will not be responsible for monitoring any transactions between you and third-party providers. It is recommended that you exercise due diligence and caution when considering any third-party products or services, just as you would with any purchase or service through any medium.

Disclaimer of Advice:

FOR THE AVOIDANCE OF DOUBT, THIS REPORT, ITS CONTENT, ACCESS, AND/OR USE, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHOULD NOT BE CONSIDERED OR RELIED UPON AS FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER PROFESSIONAL ADVICE.

3 Executive Summary

This document presents the security review performed by Cairo Security Clan on the Starknet implementation of Hyperlane protocol by Pragma.

Vesu, DeFi's latest progression in the on-chain lending space, is a pioneering platform designed to facilitate fully permissionless, over-collateralized lending agreements. With its ambitious design, Vesu looks to combine the best aspects of both worlds: a liquidity monolith with permissionless, multi-asset lending compartments aka lending pools. Learn more from docs.

The audit was performed using

- manual analysis of the codebase,
- automated analysis tools,
- simulation of the smart contract,
- analysis of edge test cases

14 points of attention, where 3 are classified as Critical, 4 are classified as High, 1 is classified as Medium,5 are classified as Low,1 is classified as Informational and 0 is classified as Best Practices. The issues are summarized in Fig. 1.

This document is organized as follows. Section 1 About Cairo Security Clan. Section 2 Disclaimer. Section 3 Executive Summary. Section 4 Summary of Audit. Section 5 Risk Classification. Section 6 Issues by Severity Levels. Section 7 Test Evaluation.

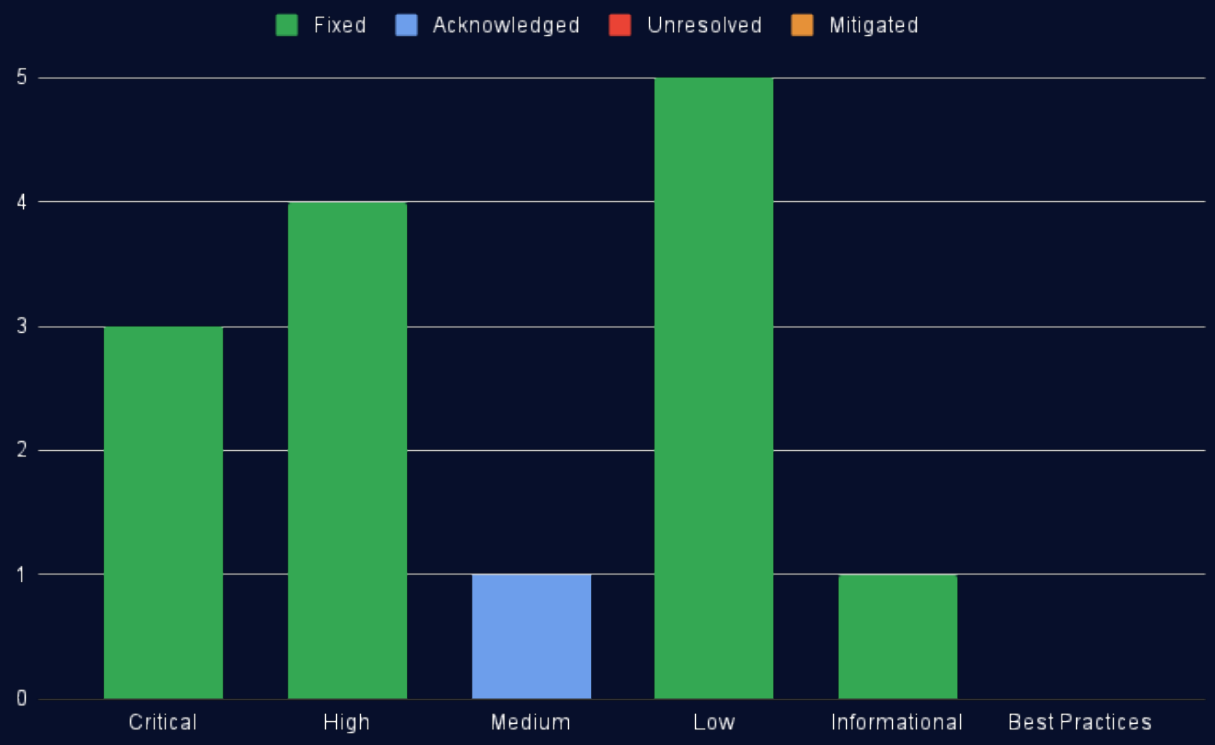

Fig 1: Distribution of issues: Critical (3), **High** (4), **Medium** (1), **Low** (5), **Informational** (1), **Best Practices** (0). **Distribution of status: Fixed** (13), **Acknowledged** (1), **Mitigated** (0), **Unresolved** (0).

4 Summary of Audit

Audit Type	Security Review
Cairo Version	2.6.5
Final Report	08/11/2024
Repository	astraly-labs/hyperlane_starknet
Initial Commit Hash	f0b05d4ea8df6a444b1672cec0388a86f6ac8201
Final Pull Request	#116
Documentation	Website documentation
Test Suite Assessment	High

4.1 Scoped Files

	Contracts
1	/src/client/gas_router_component.cairo
2	/src/client/router_component.cairo
3	/src/lib.cairo
4	/src/libs/enumerable_map.cairo
5	/src/libs/math.cairo
6	/src/components/erc721_enumerable.cairo
7	/src/components/erc721_uri_storage.cairo
8	/src/components/fast_token_router.cairo
9	/src/components/hyp_erc20_collateral_component.cairo
10	/src/components/hyp_erc20_component.cairo
11	/src/components/hyp_erc721_collateral_component.cairo
12	/src/components/hyp_erc721_component.cairo
13	/src/components/hyp_native_component.cairo
14	/src/components/token_message.cairo
15	/src/components/token_router.cairo
16	/src/extensions/fast_hyp_erc20.cairo
17	/src/extensions/fast_hyp_erc20_collateral.cairo
18	/src/extensions/hyp_erc20_collateral_vault_deposit.cairo
19	/src/extensions/hyp_erc20_vault.cairo
20	/src/extensions/hyp_erc20_vault_collateral.cairo
21	/src/extensions/hyp_erc721_URI_collateral.cairo
22	/src/extensions/hyp_erc721_URI_storage.cairo
23	/src/extensions/hyp_fiat_token.cairo
24	/src/extensions/hyp_native_scaled.cairo
25	/src/extensions/hyp_xerc20.cairo
26	/src/extensions/hyp_xerc20_lockbox.cairo
27	/src/hyp_erc20.cairo
28	/src/hyp_erc20_collateral.cairo
29	/src/hyp_erc721.cairo
30	/src/hyp_erc721_collateral.cairo
31	/src/hyp_native.cairo
32	/src/interfaces/ierc4626.cairo
33	/src/interfaces/ifiat_token.cairo
34	/src/interfaces/imessage_recipient.cairo
35	/src/interfaces/ixerc20.cairo
36	/src/interfaces/ixerc20_lockbox.cairo

4.2 Issues

	Findings	Severity	Update
1	hyp_native and hyp_native_scaled components malfunction	Critical	Fixed
2	Public _dispatch() function allows users to bridge tokens without locking or burning on source chain	Critical	Fixed
3	Missing access control check in function handle()	Critical	Fixed
4	Users can use the transfer_remote() function without scaling in hyp_native_scaled	High	Fixed
5	Mailbox ETH fee not transferred by users when calling mailbox.dispatch()	High	Fixed
6	Incorrect token recipient in function _get_token_recipient() of FastTokenRouterComponent	High	Fixed
7	Missing logic in function `transfer_to_hook()` of `HypERC721URICollateral`	High	Fixed
8	Tokens minted in constructor of hyp_erc20 and hyp_erc721 not backed by collateral	Medium	Acknowledged
9	hyp_native and hyp_native_scaled contract should not implement ERC20Component	Low	Fixed
10	Constructor of contract HypErc721 does not call erc721_enumerable.initializer()	Low	Fixed
11	Duplicate definition of wrapped_token in hyp_erc721_collateral	Low	Fixed
12	Unused storage variable gas_router	Low	Fixed
13	Unused storage variables in `hyp_erc721_URI_collateral`	Low	Fixed
14	Unchecked return values of ERC20 transfer	Informational	Fixed

5 Risk Classification

The risk rating methodology used by Cairo Security Clan follows the principles established by the CVSS risk rating methodology. The severity of each finding is determined by two factors: **Likelihood** and **Impact**.

Likelihood measures how likely an attacker will uncover and exploit the finding. This factor will be one of the following values:

a) **High**: The issue is trivial to exploit and has no specific conditions that need to be met;

b) **Medium**: The issue is moderately complex and may have some conditions that need to be met;

c) **Low**: The issue is very complex and requires very specific conditions to be met.

When defining the likelihood of a finding, other factors are also considered. These can include but are not limited to Motive, opportunity, exploit accessibility, ease of discovery, and ease of exploit.

Impact is a measure of the damage that may be caused if an attacker exploits the finding. This factor will be one of the following values:

a) **High**: The issue can cause significant damage such as loss of funds or the protocol entering an unrecoverable state;

b) **Medium**: The issue can cause moderate damage such as impacts that only affect a small group of users or only a particular part of the protocol;

c) **Low**: The issue can cause little to no damage such as bugs that are easily recoverable or cause unexpected interactions that cause minor inconveniences.

When defining the impact of a finding other factors are also considered. These can include but are not limited to Data/state integrity, loss of availability, financial loss, and reputation damage. After defining the likelihood and impact of an issue, the severity can be determined according to the table below.

		Likelihood		
		High	**Medium**	**Low**
Impact	**High**	Critical	High	Medium
	Medium	High	Medium	Low
	Low	Medium	Low	Info/Best Practices

To address issues that do not fit a High/Medium/Low severity, Cairo Security Clan also uses three more finding severities: **Informational**, **Best Practices** and **Gas**

a) **Informational** findings do not pose any risk to the application, but they carry some information that the audit team intends to formally pass to the client;

b) **Best Practice** findings are used when some piece of code does not conform with smart contract development best practices;

b) **Gas** findings are used when some piece of code uses more gas than it should be or have some functions that can be removed to save gas.

6 Issues by Severity Levels

6.1 Critical

6.1.1 hyp_native and hyp_native_scaled components malfunction

File(s): hyp_native_component.cairo, hyp_native_scaled.cairo

Description: In Starknet, there is no native token like in EVM; all tokens are represented as smart contracts, such as ETH and STRK.

The `hyp_native` component is intended to manage the transfer of native tokens (ETH in this case). Due to Starknet's architecture, its functionality should align with that of a standard `hyp_erc20`. However, despite the `eth_token` variable being defined in the storage, it is not utilized in the contract. The `_transfer_to()` function mistakenly uses the contract address of `hyp_native` instead of the intended ERC20 token address for transferring to the recipient.

```
fn _transfer_to(ref self: ComponentState<TContractState>, recipient: u256, amount: u256) {
    let contract_address = starknet::get_contract_address(); // this address or eth address // @audit not working
    let erc20_dispatcher = ERC20ABIDispatcher { contract_address };
    let recipient_felt: felt252 = recipient.try_into().expect('u256 to felt failed');
    let recipient: ContractAddress = recipient_felt.try_into().unwrap();
    erc20_dispatcher.transfer(recipient, amount);
}
```

Additionally, the `_transfer_from_sender()` function, which is responsible for retrieving tokens from users, currently lacks any implementation. Similarly, in the `hyp_native_scaled`, the function `transfer_from_sender_hook()` does not implement any logic.

Recommendation(s):

- Use the `eth_token` storage variable in the `_transfer_to()` function to correctly transfer tokens.
- Implement the logic for the `_transfer_from_sender()` and `transfer_from_sender_hook()` functions to ensure they effectively retrieve tokens from users.

Status: Fixed

Update from client: Fixed in commit 22dc1d0 and d4989a7

6.1.2 Public _dispatch() function allows users to bridge tokens without locking or burning on source chain

File(s): mailboxclient_component.cairo

Description: The function `_dispatch()` is intended to be an internal function. However, it is implemented inside `MailboxClient`, which is exposed as external functions in token contracts. This makes `_dispatch()` public and allows it to be called directly.

This function enables anyone to call the `mailbox` contract with arbitrary input, allowing users to perform token bridging without locking or burning any tokens on the source chain.

```
fn _dispatch( // @audit this function can be called directly
    self: @ComponentState<TContractState>,
    _destination_domain: u32,
    _recipient: u256,
    _message_body: Bytes,
    _fee_amount: u256,
    _hook_metadata: Option<Bytes>,
    _hook: Option<ContractAddress>
) -> u256 {
    self
        .mailbox
        .read()
        .dispatch(
            _destination_domain,
            _recipient,
            _message_body,
            _fee_amount,
            _hook_metadata,
            _hook
        )
}
```

Recommendation(s): Consider moving the `_dispatch()` function into the internal implementation `MailboxClientInternalImpl` instead.

Status: Fixed

Update from client: Fixed in 1858691

6.1.3 Missing access control check in function handle()

File(s): router_component.cairo

Description: The public function `handle()` in the destination contract is designed to unlock or mint tokens for users. This function should only be callable by the mailbox contract after necessary checks and verifications are completed within the mailbox contract. However, there is currently no access control in place to enforce this restriction. As a result, any external caller can invoke this function, potentially allowing them to mint an unlimited number of hyp tokens for any address or deplete the token balance.

```
fn handle(
    ref self: ComponentState<TContractState>, origin: u32, sender: u256, message: Bytes
) { // @audit no check caller is mailbox
    let router = self._must_have_remote_router(origin);
    assert!(router == sender, "Enrolled␣router␣does␣not␣match␣sender");

    Hook::_handle(ref self, origin, sender, message);
}
```

Recommendation(s): Implement an access control check in the `handle()` function to ensure that the caller is the designated mailbox contract.

Status: Fixed

Update from client: Fixed in 1858691

6.2 High

6.2.1 Users can use the transfer_remote() function without scaling in hyp_native_scaled

File(s): hyp_native_scaled.cairo

Description: The `hyp_native_scaled` module embeds the `HypNativeComponent::TokenRouterImpl`, which includes the `transfer_remote()` function. However, this implementation is intended to be used by `hyp_native` instead of `hyp_native_scaled`, which lacks the necessary scaling logic.

```
#[abi(embed_v0)]
impl HypNativeTokenRouterImpl =
    HypNativeComponent::TokenRouterImpl<ContractState>;
```

```
fn transfer_remote(
    ref self: ComponentState<TContractState>,
    destination: u32,
    recipient: u256,
    amount_or_id: u256,
    value: u256,
    hook_metadata: Option<Bytes>,
    hook: Option<ContractAddress>
) -> u256 {
    assert!(value >= amount_or_id, "Native: amount exceeds msg.value");
    let hook_payment = value - amount_or_id;

    let mut token_router_comp = get_dep_component_mut!(ref self, TokenRouterComp);
    TokenRouterTransferRemoteHookDefaultImpl::_transfer_remote(
        ref token_router_comp,
        destination,
        recipient,
        amount_or_id,
        hook_payment,
        Option::None,
        Option::None
    )
}
```

Recommendation(s): Consider removing the embedding of `HypNativeTokenRouterImpl` in `hyp_native_scaled` contract.

Status: Fixed

Update from client: Fixed in 292a727

6.2.2 Mailbox ETH fee not transferred by users when calling mailbox.dispatch()

File(s): router_component.cairo, hyp_erc20_vault_collateral.cairo

Description: The `mailbox.dispatch()` function includes a `_fee_amount` parameter that represents the ETH fee users must pay to initiate a dispatch. This fee is intended to be deducted from the caller's address using the `transferFrom()` method, as shown below:

```
fn dispatch(
    ref self: ContractState,
    _destination_domain: u32,
    _recipient_address: u256,
    _message_body: Bytes,
    _fee_amount: u256,
    _custom_hook_metadata: Option<Bytes>,
    _custom_hook: Option<ContractAddress>
) -> u256 {
    // ...
    assert(_fee_amount >= required_fee + default_fee, Errors::NOT_ENOUGH_FEE_PROVIDED);
    let caller_address = get_caller_address();
    let contract_address = get_contract_address();
    let token_dispatcher = ERC20ABIDispatcher { contract_address: ETH_ADDRESS() };
    let user_balance = token_dispatcher.balanceOf(caller_address);
    assert(user_balance >= required_fee + default_fee, Errors::INSUFFICIENT_BALANCE);
    assert(
        token_dispatcher.allowance(caller_address, contract_address) >= _fee_amount,
        Errors::INSUFFICIENT_ALLOWANCE
    );
    if (required_fee > 0) {
        token_dispatcher.transferFrom(caller_address, required_hook_address, required_fee);
    }
    // ...
}
```

However, in the context of warp routes, the caller address in the `mailbox.dispatch()` function is typically the warp route contract (e.g., `HypERC20`, `HypERC721Collateral`). This means that the ETH fee is transferred from these warp route contracts instead of the actual users. Consequently, there is no mechanism in the warp route contracts to collect the ETH fee from users, leading to transaction failures since not all warp routes hold ETH.

```
fn _Router_dispatch(
    self: @ComponentState<TContractState>,
    destination_domain: u32,
    value: u256,
    message_body: Bytes,
    hook_metadata: Bytes,
    hook: ContractAddress
) -> u256 {
    let router = self._must_have_remote_router(destination_domain);
    let mut mailbox_comp = get_dep_component!(self, MailBoxClient);
    let value = mailbox_comp
        .mailbox
        .read()
        .dispatch(
            destination_domain,
            router,
            message_body,
            value, // @audit `value` fee is not transferred in
            Option::Some(hook_metadata),
            Option::Some(hook),
        );
    value
}
```

Recommendation(s): Consider implementing a mechanism to collect the fee `value` from the user's wallet to the warp route contract before invoking `mailbox.dispatch()` function.

Status: Fixed

Update from client: Fixed in 5ca86c5

6.2.3 Incorrect token recipient in function _get_token_recipient() of FastTokenRouterComponent

File(s): fast_token_router.cairo

Description: The Fast Token Router allows liquidity providers to fulfill transfer requests before message processing. After processing on the destination chain, liquidity providers receive their funds back. The `_get_token_recipient()` function determines where tokens should be sent during this process. If fulfilled by a liquidity provider, the recipient is the provider's address; otherwise, it defaults to the original `recipient` address.

However, the implementation contains an error. Specifically, when `filler_address` equals 0, the function incorrectly returns `filler_address` as the token recipient. This results in tokens being sent to the zero address, leading to a loss of funds. In other cases, liquidity providers who fulfill the message do not receive their tokens back.

```
fn _get_token_recipient(
    self: @ComponentState<TContractState>,
    recipient: u256,
    amount: u256,
    origin: u32,
    metadata: Bytes
) -> u256 {
    if metadata.size() == 0 {
        return recipient;
    }

    let (_, fast_fee) = metadata.read_u256(0);
    let (_, fast_transfer_id) = metadata.read_u256(2);

    let filler_address = self
        ._get_fast_transfers_key(origin, fast_transfer_id, amount, fast_fee, recipient);
    if filler_address == 0 { // @audit should be != instead of ==
        return filler_address;
    }

    recipient
}
```

Recommendation(s): Change the condition to check if `filler_address != 0`.

Status: Fixed

Update from client: Fixed in b3c8396

6.2.4 Missing logic in function `transfer_to_hook()` of `HypERC721URICollateral`

File(s): hyp_erc721_URI_collateral.cairo

Description: In the `HypERC721URICollateral`, the `transfer_from_sender_hook()` function correctly transfers the ERC721 collateral token from the user's wallet to the contract address. However, the `transfer_to_hook()` function in `HypERC721URICollateral` is defined but currently lacks the implementation necessary to transfer the ERC721 token back to the user. As a result, the ERC721 token of users will be stuck in the `HypERC721URICollateral` contract.

```
fn transfer_to_hook(
    ref self: TokenRouterComponent::ComponentState<ContractState>,
    recipient: u256,
    amount_or_id: u256,
    metadata: Bytes
) {}
```

Recommendation(s): Implement the logic in `transfer_to_hook()` to ensure the ERC721 token is transferred to the designated recipient.

Status: Fixed

Update from client: Fixed in 7911e96

6.3 Medium

6.3.1 Tokens minted in constructor of `hyp_erc20` and `hyp_erc721` not backed by collateral

File(s): hyp_erc20.cairo, hyp_erc721_component.cairo

Description: The `hyp_erc20` and `hyp_erc721` tokens are designed as synthetic tokens, intended to be backed by original tokens on the source chain. When users bridge tokens, their assets on the source chain are locked, and synthetic tokens are minted on the destination chain.

However, during the construction of the synthetic token, the entire `total_supply` is minted to the caller without any corresponding collateral locked on the source chain. This poses a risk: if all synthetic tokens are bridged back to the source chain, the source chain contract will not possess sufficient tokens to cover the total supply.

```
fn constructor(
    ref self: ContractState,
    decimals: u8,
    mailbox: ContractAddress,
    total_supply: u256,
    name: ByteArray,
    symbol: ByteArray,
    hook: ContractAddress,
    interchain_security_module: ContractAddress,
    owner: ContractAddress
) {
    self.ownable.initializer(owner);
    self
        .mailbox
        .initialize(mailbox, Option::Some(hook), Option::Some(interchain_security_module));
    self.hyp_erc20.initialize(decimals);
    self.erc20.initializer(name, symbol);
    self.erc20.mint(starknet::get_caller_address(), total_supply); // @audit these tokens are not backed by any
     collateral
}
```

Similarly, in the `hyp_erc721` token, all token IDs from `0` to `mint_amount - 1` are minted to the caller upon initialization. This means that these token IDs cannot bridge back to the source chain, as they are not backed by any collateral.

```
fn initialize(
    ref self: ComponentState<TContractState>,
    mint_amount: u256,
    name: ByteArray,
    symbol: ByteArray,
) {
    let mut erc721_comp = get_dep_component_mut!(ref self, ERC721);
    erc721_comp.initializer(name, symbol, "");

    let caller = starknet::get_caller_address();

    let mut i = 0;
    while i < mint_amount {
        erc721_comp.mint(caller, i.into()); // @audit not backed by collateral
        i += 1;
    };
}
```

Recommendation(s): Consider not minting tokens to caller directly in the constructor of `hyp_erc20` and `hyp_erc721`.

Status: Acknowledged

Update from the client: The concern is acknowledged. The Solidity version also exhibits similar behavior. In the opinion of the speaker, it would be advisable to reach out to the team at Hyperlane for clarification on whether there is a specific reason for this approach.

6.4 Low

6.4.1 hyp_native and hyp_native_scaled contract should not implement ERC20Component

File(s): hyp_native.cairo, hyp_native_scaled.cairo

Description: The `hyp_native` and `hyp_native_scaled` components manage ETH tokens received from senders as collateral. Since these components do not mint or burn tokens, implementing the `ERC20Component` is unnecessary. The current implementation does not align with the functionality provided by ERC20, as no tokens are created or transferred.

```
// ERC20
#[abi(embed_v0)]
impl ERC20Impl = ERC20Component::ERC20MixinImpl<ContractState>;
```

Recommendation(s): Consider removing the `ERC20Component` implementation from both contracts.

Status: Fixed

Update from the client: Fixed in 18f852f, 6789abd and 0ab4368

6.4.2 Constructor of contract HypErc721 does not call erc721_enumerable.initializer()

File(s): hyp_erc721.cairo

Description: The `HypErc721` contract embeds `ERC721EnumerableImpl`, but the constructor fails to call `erc721_enumerable.initializer()`. This initializer is essential for calling `register_interface()`, which declares support for the `IERC721Enumerable` interface ID.

```
#[constructor]
fn constructor(
    ref self: ContractState,
    mailbox: ContractAddress,
    name: ByteArray,
    symbol: ByteArray,
    mint_amount: u256,
    hook: ContractAddress,
    interchain_security_module: ContractAddress,
    owner: ContractAddress
) { // @audit not call erc721_enumerable.initializer()
    self.ownable.initializer(owner);
    self
        .mailboxclient
        .initialize(mailbox, Option::Some(hook), Option::Some(interchain_security_module));
    self.hyp_erc721.initialize(mint_amount, name, symbol);
}
```

Recommendation(s): Consider calling `erc721_enumerable.initializer()` within the constructor of the `HypErc721` contract.

Status: Fixed

Update from the client: Fixed in 31f7f25

6.4.3 Duplicate definition of wrapped_token in hyp_erc721_collateral

File(s): hyp_erc721_collateral.cairo

Description: In the `hyp_erc721_collateral`, the storage variable `wrapped_token` is defined in both `HypErc721Collateral` and `HypErc721CollateralComponent`. This redundancy can lead to confusion, as assigning a value to one will not affect the other, potentially leaving one uninitialized.

```
// HypErc721Collateral
#[storage]
struct Storage {
    wrapped_token: ERC721ABIDispatcher, // @audit also defined in HypErc721CollateralComponent storage
    #[substorage(v0)]
    ownable: OwnableComponent::Storage,
    #[substorage(v0)]
    token_router: TokenRouterComponent::Storage,
    #[substorage(v0)]
    mailboxclient: MailboxclientComponent::Storage,
    #[substorage(v0)]
    router: RouterComponent::Storage,
    #[substorage(v0)]
    gas_router: GasRouterComponent::Storage,
    #[substorage(v0)]
    hyp_erc721_collateral: HypErc721CollateralComponent::Storage,
    #[substorage(v0)]
    upgradeable: UpgradeableComponent::Storage
}

// HypErc721CollateralComponent
#[storage]
struct Storage {
    wrapped_token: ERC721ABIDispatcher, // @audit already defined in `hyp_erc721_collateral`
}
```

Recommendation(s): Consider removing one of the definitions for `wrapped_token`.

Status: Fixed

Update from the client: Fixed in 21d8d09

6.4.4 Unused storage variable gas_router

File(s): router_component.cairo

Description: In the `Storage` struct of `RouterComponent`, there is a state variable `gas_router` defined, but it is not initialized or utilized anywhere in the code. This may indicate that the variable is unnecessary and could lead to confusion.

```
struct Storage {
    routers: EnumerableMap<u32, u256>,
    gas_router: ContractAddress, // @audit not set anywhere
}
```

Recommendation(s): If `gas_router` is not intended for future use, consider removing it from the `Storage` struct to streamline the code and improve clarity.

Status: Fixed

Update from the client: Fixed in 9afb1d0

6.4.5 Unused storage variables in `hyp_erc721_URI_collateral`

File(s): hyp_erc721_URI_collateral.cairo

Description: In the Storage struct of `HypERC721URICollateral`, there are state variables `erc721` and `mailbox` defined, but it is not initialized or utilized anywhere in the code. This may indicate that the variable is unnecessary and could lead to confusion.

```
struct Storage {
    erc721: ContractAddress,
    mailbox: ContractAddress,
    // ...
}
```

Recommendation(s): Consider removing them from the Storage struct to streamline the code and improve clarity.

Status: Fixed

Update from client: Fixed in 0ab4368

6.5 Informational

6.5.1 Unchecked return values of ERC20 transfer

File(s): hyp_erc20_collateral_component.cairo

Description: In the codebase, there are two ERC20 transfers to move tokens between the users' wallet and the `HypErc20Collateral` contract using `transfer()` and `transfer_from()`. These functions return a boolean value indicating whether the transfer succeeded or not. However, these return values are not currently checked, potentially resulting in unexpected behavior, especially if the token does not revert on failure.

```
fn _transfer_from_sender(ref self: ComponentState<TContractState>, amount: u256) -> Bytes {
    self
        .wrapped_token
        .read()
        .transfer_from(
            starknet::get_caller_address(), starknet::get_contract_address(), amount
        ); // @audit unsafe ERC20 operation
    BytesTrait::new_empty()
}

fn _transfer_to(ref self: ComponentState<TContractState>, recipient: u256, amount: u256) {
    self
        .wrapped_token
        .read()
        .transfer(recipient.try_into().expect('u256 to ContractAddress failed'), amount); // @audit unsafe ERC20 operation
}
```

Recommendation(s): Consider adding a check for the boolean return values.

Status: Fixed

Update from the client: Fixed in f446767

7 Test Evaluation

7.1 Compilation Output

```
 scarb build
   Compiling lib(contracts) contracts v0.0.6 (./cairo/crates/contracts/Scarb.toml)
   Compiling starknet-contract(contracts) contracts v0.0.6 (./cairo/crates/contracts/Scarb.toml)
   Compiling lib(mocks) mocks v0.0.6 (./cairo/crates/mocks/Scarb.toml)
   Compiling starknet-contract(mocks) mocks v0.0.6 (./cairo/crates/mocks/Scarb.toml)
   Compiling lib(token) token v0.0.1 (./cairo/crates/token/Scarb.toml)
   Compiling starknet-contract(token) token v0.0.1 (./cairo/crates/token/Scarb.toml)
    Finished release target(s) in 3 minutes
```

7.2 Tests Output

```
snforge test
   Compiling lib(contracts) contracts v0.0.6 (./cairo/crates/contracts/Scarb.toml)
   Compiling starknet-contract(contracts) contracts v0.0.6 (./cairo/crates/contracts/Scarb.toml)
   Compiling lib(mocks) mocks v0.0.6 (./cairo/crates/mocks/Scarb.toml)
   Compiling starknet-contract(mocks) mocks v0.0.6 (./cairo/crates/mocks/Scarb.toml)
   Compiling lib(token) token v0.0.1 (./cairo/crates/token/Scarb.toml)
   Compiling starknet-contract(token) token v0.0.1 (./cairo/crates/token/Scarb.toml)
    Finished release target(s) in 3 minutes


Collected 120 test(s) from contracts package
Running 10 test(s) from src/
[PASS] contracts::utils::keccak256::tests::test_u64_word_size (gas: ~6)
[PASS] contracts::utils::keccak256::tests::test_reverse_endianness (gas: ~2)
[PASS] contracts::libs::message::tests::test_append_u128_to_byte_array (gas: ~21)
[PASS] contracts::utils::keccak256::tests::test_down_bytes (gas: ~1)
[PASS] contracts::hooks::libs::standard_hook_metadata::tests::test_standard_hook_metadata_default_value (gas:
    ~49)
[PASS] contracts::libs::aggregation_ism_metadata::test::test_aggregation_ism_metadata (gas: ~37)
[PASS] contracts::libs::aggregation_ism_metadata::test::test_aggregation_ism_has_metadata (gas: ~14)
[PASS] contracts::utils::keccak256::tests::test_up_bytes (gas: ~68)
[PASS] contracts::hooks::libs::standard_hook_metadata::tests::test_standard_hook_metadata (gas: ~106)
[PASS] contracts::utils::keccak256::tests::test_compute_keccak (gas: ~457)
Running 110 test(s) from tests/
[PASS] tests::test_mailbox::test_owner (gas: ~1112)
[PASS] tests::test_mailbox::test_set_default_hook (gas: ~1118)
[PASS] tests::test_mailbox::test_local_domain (gas: ~1112)
[PASS] tests::test_mailbox::test_set_default_ism (gas: ~1118)
[PASS] tests::test_mailbox::test_set_required_hook_fails_if_not_owner (gas: ~1112)
[PASS] tests::test_mailbox::test_set_default_hook_fails_if_not_owner (gas: ~1112)
[PASS] tests::test_mailbox::test_set_default_ism_fails_if_not_owner (gas: ~1112)
[PASS] tests::test_mailbox::test_transfer_ownership (gas: ~1119)
[PASS] tests::test_validator_announce::test_digest_computation (gas: ~596)
[PASS] tests::isms::test_messageid_multisig::
    test_message_id_multisig_verify_with_4_valid_signatures_fails_if_duplicate_signatures (gas: ~4216)
[PASS] tests::isms::test_aggregation::test_aggregation_module_type (gas: ~361)
[PASS] tests::test_validator_announce::test_announce_fails_if_replay (gas: ~2804)
[PASS] tests::isms::test_aggregation::test_aggregation_initialize_with_too_many_modules (gas: ~267)

Success data:
    0x526573756c743a3a756e77726170206661696c65642e ('Result::unwrap␣failed.')

[PASS] tests::test_mailbox::test_dispatch_with_protocol_fee_hook_fails_if_provided_fee_lower_than_required_fee (
    gas: ~2050)
[PASS] tests::isms::test_merkleroot_multisig::test_merkle_root_multisig_module_type (gas: ~296)
[PASS] tests::test_mailbox::test_process (gas: ~2190)
[PASS] tests::test_mailbox::test_set_required_hook (gas: ~1118)
[PASS] tests::isms::test_aggregation::test_setup_aggregation_with_null_module_address (gas: ~230)

Success data:
    0x526573756c743a3a756e77726170206661696c65642e ('Result::unwrap␣failed.')
```

```
[PASS] tests::test_mailbox::test_dispatch_with_protocol_fee_hook_fails_if_insufficient_allowance (gas: ~2052)
[PASS] tests::test_mailbox::test_dispatch_with_two_fee_hook (gas: ~2332)
[PASS] tests::isms::test_aggregation::test_get_modules (gas: ~363)
[PASS] tests::routing::test_domain_routing_ism::test_remove_domain_check_module (gas: ~501)
[PASS] tests::isms::test_default_ism::test_pause_pausable_ism_fails_if_not_owner (gas: ~167)
[PASS] tests::isms::test_default_ism::test_unpause_pausable_ism_fails_if_not_owner (gas: ~167)
[PASS] tests::routing::test_default_fallback_routing_ism::test_remove_domain_fails_if_caller_not_owner (gas:
    ~1409)
[PASS] tests::isms::test_default_ism::test_verify_noop_ism (gas: ~105)
[PASS] tests::isms::test_default_ism::test_pause_unpause_pausable_ism (gas: ~173)
[PASS] tests::routing::test_default_fallback_routing_ism::test_remove_domain_fails_if_domain_not_found (gas:
    ~1804)
[PASS] tests::test_mailbox::test_dispatch_with_two_fee_hook_fails_if_greater_than_required_and_lower_than_default
     (gas: ~2176)
[PASS] tests::routing::test_default_fallback_routing_ism::test_set_domain_and_module (gas: ~1949)
[PASS] tests::isms::test_default_ism::test_verify_trusted_relayer_ism (gas: ~2601)
[PASS] tests::isms::test_default_ism::test_verify_pausable_ism (gas: ~170)
[PASS] tests::routing::test_default_fallback_routing_ism::test_set_domain_and_module_fails_if_caller_is_not_owner
     (gas: ~1409)
[PASS] tests::routing::test_default_fallback_routing_ism::test_module_type (gas: ~1408)
[PASS] tests::isms::test_merkleroot_multisig::test_set_validators (gas: ~363)
[PASS] tests::routing::test_default_fallback_routing_ism::test_route_ism (gas: ~1817)
[PASS] tests::routing::test_domain_routing_ism::test_initialize (gas: ~571)
[PASS] tests::routing::test_domain_routing_ism::get_module_fails_if_origin_not_found (gas: ~167)
[PASS] tests::routing::test_domain_routing_ism::test_initialize_fails_if_caller_not_owner (gas: ~168)
[PASS] tests::routing::test_domain_routing_ism::test_initialize_fails_if_module_is_zero (gas: ~429)
[PASS] tests::isms::test_merkleroot_multisig::test_merkleroot_ism_metadata (gas: ~1430)
[PASS] tests::isms::test_merkleroot_multisig::test_set_threshold (gas: ~297)
[PASS] tests::isms::test_merkleroot_multisig::test_set_validators_fails_if_null_validator (gas: ~294)

Success data:
    0x526573756c743a3a756e77726170206661696c65642e ('Result::unwrap␣failed.')

[PASS] tests::routing::test_domain_routing_ism::test_module_type (gas: ~167)
[PASS] tests::isms::test_aggregation::test_aggregation_verify (gas: ~3333)
[PASS] tests::routing::test_domain_routing_ism::test_remove_domain (gas: ~503)
[PASS] tests::routing::test_domain_routing_ism::test_remove_domain_fails_if_caller_not_owner (gas: ~168)
[PASS] tests::isms::test_default_ism::test_veriy_pausable_ism_fails_if_paused (gas: ~236)
[PASS] tests::routing::test_domain_routing_ism::test_initialize_fails_if_length_mismatch (gas: ~169)
[PASS] tests::routing::test_domain_routing_ism::test_remove_domain_fails_if_domain_not_found (gas: ~563)
[PASS] tests::routing::test_default_fallback_routing_ism::test_verify (gas: ~5450)
[PASS] tests::routing::test_domain_routing_ism::test_set_domain_and_module (gas: ~708)
[PASS] tests::routing::test_domain_routing_ism::test_route_ism_fails_if_origin_not_found (gas: ~562)
[PASS] tests::isms::test_messageid_multisig::test_message_id_multisig_module_type (gas: ~296)
[PASS] tests::routing::test_domain_routing_ism::test_set_domain_and_module_fails_if_caller_is_not_owner (gas:
    ~168)
[PASS] tests::routing::test_domain_routing_ism::test_route_ism (gas: ~569)
[PASS] tests::isms::test_messageid_multisig::test_message_id_ism_metadata (gas: ~263)
[PASS] tests::isms::test_merkleroot_multisig::test_merkle_root_multisig_verify_with_empty_metadata (gas: ~556)
[PASS] tests::isms::test_messageid_multisig::test_message_id_multisig_verify_with_empty_metadata (gas: ~556)
[PASS] tests::test_mailbox::test_process_fails_if_version_mismatch (gas: ~1303)
[PASS] tests::isms::test_messageid_multisig::test_message_id_multisig_verify_with_insufficient_valid_signatures (
    gas: ~4400)
[PASS] tests::test_validator_announce::test_announce (gas: ~2808)
[PASS] tests::test_mailbox::test_process_fails_if_destination_domain_does_not_match_local_domain (gas: ~1304)
[PASS] tests::isms::test_messageid_multisig::test_message_id_multisig_verify_with_4_valid_signatures (gas: ~3662)
[PASS] tests::isms::test_merkleroot_multisig::test_merkle_root_multisig_verify_with_insufficient_valid_signatures
     (gas: ~10252)
[PASS] tests::isms::test_merkleroot_multisig::
    test_merkle_root_multisig_verify_with_4_valid_signatures_fails_if_duplicate_signatures (gas: ~9638)
[PASS] tests::test_validator_announce::test_announce_fails_if_wrong_signer (gas: ~2349)
[PASS] tests::routing::test_default_fallback_routing_ism::test_remove_domain_module_check (gas: ~1747)
[PASS] tests::hooks::test_merkle_tree_hook::test_merkle_tree_hook_type (gas: ~1407)
[PASS] tests::hooks::test_merkle_tree_hook::test_supports_metadata (gas: ~1425)
[PASS] tests::hooks::test_merkle_tree_hook::test_quote_dispatch (gas: ~1414)
[PASS] tests::libs::test_enumerable_map::test_initialize_empty_map (gas: ~102)
[PASS] tests::hooks::test_merkle_tree_hook::test_post_dispatch (gas: ~2141)
[PASS] tests::test_validator_announce::test_double_announce (gas: ~1034)
[PASS] tests::hooks::test_protocol_fee::test_collect_protocol_fee_fails_if_insufficient_balance (gas: ~917)
```

```
[PASS] tests::hooks::test_merkle_tree_hook::test_post_dispatch_fails_if_invalid_variant (gas: ~1414)
[PASS] tests::test_mailbox::test_dispatch_with_protocol_fee_hook_fails_if_user_balance_lower_than_fee_amount (gas: ~2051)
[PASS] tests::hooks::test_merkle_tree_hook::test_post_dispatch_fails_if_message_not_dispatching (gas: ~1532)
[PASS] tests::test_mailbox::test_dispatch (gas: ~1702)
[PASS] tests::hooks::test_merkle_tree_hook::test_quote_dispatch_fails_if_invalid_variant (gas: ~1413)
[PASS] tests::test_mailbox::test_process_fails_if_already_delivered (gas: ~2386)
[PASS] tests::hooks::test_protocol_fee::test_hook_type (gas: ~425)
[PASS] tests::routing::test_domain_routing_ism::test_verify (gas: ~4221)
[PASS] tests::hooks::test_protocol_fee::test_set_beneficiary (gas: ~429)
[PASS] tests::hooks::test_protocol_fee::test_set_protocol_fee_fails_if_not_owner (gas: ~426)
[PASS] tests::hooks::test_protocol_fee::test_set_protocol_fee (gas: ~430)
[PASS] tests::isms::test_messageid_multisig::test_set_threshold (gas: ~296)
[PASS] tests::hooks::test_protocol_fee::test_set_protocol_fee_fails_if_higher_than_max (gas: ~427)
[PASS] tests::test_mailbox::test_dispatch_with_protocol_fee_hook (gas: ~2196)
[PASS] tests::isms::test_messageid_multisig::test_set_validators_fails_if_null_validator (gas: ~294)

Success data:
    0x526573756c743a3a756e77726170206661696c65642e ('Result::unwrap failed.')

[PASS] tests::isms::test_merkleroot_multisig::test_merkle_root_multisig_verify_with_4_valid_signatures (gas: ~9088)
[PASS] tests::routing::test_default_fallback_routing_ism::test_initialize_fails_if_module_is_zero (gas: ~1669)
[PASS] tests::routing::test_default_fallback_routing_ism::test_initialize_fails_if_caller_not_owner (gas: ~1409)
[PASS] tests::routing::test_default_fallback_routing_ism::test_initialize_fails_if_length_mismatch (gas: ~1409)
[PASS] tests::hooks::test_protocol_fee::test_post_dispatch_fails_if_invalid_variant (gas: ~432)
[PASS] tests::hooks::test_merkle_tree_hook::test_count (gas: ~1408)
[PASS] tests::hooks::test_protocol_fee::test_supports_metadata (gas: ~442)
[PASS] tests::routing::test_default_fallback_routing_ism::test_initialize (gas: ~1812)
[PASS] tests::routing::test_default_fallback_routing_ism::test_remove_domain (gas: ~1743)
[PASS] tests::hooks::test_protocol_fee::test_set_beneficiary_fails_if_not_owner (gas: ~426)
[PASS] tests::hooks::test_protocol_fee::test_collect_protocol_fee (gas: ~1001)
[PASS] tests::routing::test_default_fallback_routing_ism::get_default_module (gas: ~1413)
[PASS] tests::hooks::test_protocol_fee::test_quote_dispatch (gas: ~432)
[PASS] tests::hooks::test_protocol_fee::test_quote_dispatch_fails_if_invalid_variant (gas: ~431)
[PASS] tests::hooks::test_merkle_tree_hook::test_insert_node_into_merkle_tree_hook (gas: ~13494)
[PASS] tests::isms::test_messageid_multisig::test_set_validators (gas: ~363)
[PASS] tests::libs::test_enumerable_map::test_fuzz_get_keys (runs: 256, gas: {max: ~715, min: ~587, mean: ~713.00, std deviation: ~12.57})
[PASS] tests::libs::test_enumerable_map::test_fuzz_contains (runs: 256, gas: {max: ~427, min: ~102, mean: ~275.00, std deviation: ~161.92})
[PASS] tests::libs::test_enumerable_map::test_fuzz_set (runs: 256, gas: {max: ~438, min: ~310, mean: ~437.00, std deviation: ~8.92})
[PASS] tests::libs::test_enumerable_map::test_fuzz_should_remove (runs: 256, gas: {max: ~190, min: ~126, mean: ~189.00, std deviation: ~4.06})
Tests: 120 passed, 0 failed, 0 skipped, 0 ignored, 0 filtered out
Fuzzer seed: 10015187276832038134


Collected 0 test(s) from mocks package
Running 0 test(s) from src/
Tests: 0 passed, 0 failed, 0 skipped, 0 ignored, 0 filtered out


Collected 48 test(s) from token package
Running 0 test(s) from src/
Running 48 test(s) from tests/
[PASS] tests::hyp_erc20::common::test_hyp_erc20_setup (gas: ~1)
[PASS] tests::hyp_erc721::hyp_erc721_test::test_erc721_owner_of (gas: ~8474)
[PASS] tests::hyp_erc20::hyp_erc20_test::test_erc20_local_transfer (gas: ~5938)
[PASS] tests::hyp_erc20::hyp_fiat_token_test::test_fiat_token_handle (gas: ~6841)
[PASS] tests::hyp_erc20::hyp_erc20_lockbox_test::test_erc20_lockbox_approval (gas: ~7544)
[PASS] tests::hyp_erc20::hyp_erc20_test::test_erc20_remote_transfer (gas: ~7411)
[PASS] tests::hyp_erc20::hyp_fiat_token_test::test_fiat_token_remote_transfer (gas: ~8040)
[PASS] tests::hyp_erc20::hyp_xerc20_test::test_remote_transfer (gas: ~8393)
[PASS] tests::hyp_erc20::hyp_erc20_collateral_test::test_remote_transfer_with_custom_gas_config (gas: ~8096)
[PASS] tests::hyp_erc721::common::test_erc721_setup (gas: ~8146)
[PASS] tests::hyp_erc20::hyp_xerc20_test::test_handle (gas: ~7153)
[PASS] tests::hyp_erc721::hyp_erc721_test::test_erc721_total_supply (gas: ~8474)
[PASS] tests::hyp_erc721::hyp_erc721_collateral_test::test_erc721_collateral_remote_transfer (gas: ~11439)
```

```
[PASS] tests::hyp_erc721::hyp_erc721_collateral_test::
    test_erc721_collateral_remote_transfer_revert_invalid_token_id (gas: ~10175)

Success data:
    0x4552433732313a20696e76616c696420746f6b656e204944 ('ERC721: invalid token ID')

[PASS] tests::hyp_erc721::hyp_erc721_collateral_test::test_erc721_collateral_remote_transfer_revert_unowned (gas:
     ~10247)

Success data:
    0x4552433732313a20756e617574686f72697a65642063616c6c6572 ('ERC721: unauthorized caller')

[PASS] tests::hyp_erc20::hyp_erc20_test::test_erc20_remote_transfer_with_custom_gas_config (gas: ~7470)
[PASS] tests::hyp_erc721::hyp_erc721_collateral_uri_storage_test::
    test_erc721_collateral_uri_storage_remote_transfer_revert_burned (gas: ~11608)
[PASS] tests::hyp_erc20::hyp_erc20_test::test_erc20_decimals (gas: ~5859)
[PASS] tests::hyp_erc20::hyp_erc20_test::test_erc20_total_supply (gas: ~5859)
[PASS] tests::hyp_erc20::hyp_erc20_lockbox_test::test_erc20_lockbox_transfer (gas: ~8831)
[PASS] tests::hyp_erc20::hyp_erc20_lockbox_test::test_erc20_lockbox_handle (gas: ~7588)
[PASS] tests::hyp_erc721::hyp_erc721_test::test_erc721_local_transfer (gas: ~8549)
[PASS] tests::hyp_erc20::hyp_erc20_collateral_test::test_remote_transfer (gas: ~8037)
[PASS] tests::hyp_erc721::hyp_erc721_test::test_erc721_local_transfer_invalid_token_id (gas: ~8475)

Success data:
    0x4552433732313a20696e76616c696420746f6b656e204944 ('ERC721: invalid token ID')

[PASS] tests::hyp_erc721::hyp_erc721_uri_storage_test::test_erc721_uri_storage_remote_transfer_revert_burned (gas
    : ~12497)

Success data:
    0x4552433732313a20696e76616c696420746f6b656e204944 ('ERC721: invalid token ID')

[PASS] tests::vault_extensions::hyp_erc20_vault_test::test_collateral_domain (gas: ~11644)
[PASS] tests::vault_extensions::hyp_erc20_vault_test::test_withdrawal_with_yield (gas: ~15281)
[PASS] tests::vault_extensions::hyp_erc20_vault_test::test_cyclic_transfers (gas: ~20141)
[PASS] tests::vault_extensions::hyp_erc20_vault_test::test_rebase_with_transfer (gas: ~15258)
[PASS] tests::hyp_erc20::hyp_erc20_collateral_test::test_remote_transfer_invalid_allowance (gas: ~6819)

Success data:
    0x45524332303a20696e73756666696963696656e7420616c6c6f77616e6365 ('ERC20: insufficient allowance')

[PASS] tests::hyp_erc721::hyp_erc721_test::test_erc721_remote_transfer_revert_unowned (gas: ~9685)

Success data:
    "Caller is not owner"

[PASS] tests::vault_extensions::hyp_erc20_vault_test::test_withdrawal_after_yield (gas: ~16791)
[PASS] tests::vault_extensions::hyp_erc20_vault_test::test_withdrawal_without_yield (gas: ~15139)
[PASS] tests::vault_extensions::hyp_erc20_vault_test::test_withdrawal_in_flight (gas: ~18567)
[PASS] tests::vault_extensions::hyp_erc20_vault_test::test_remote_transfer_rebase_after (gas: ~15186)
[PASS] tests::vault_extensions::hyp_erc20_vault_test::test_exchange_rate_set_only_by_collateral (gas: ~18392)
[PASS] tests::hyp_erc721::hyp_erc721_test::test_erc721_remote_transfer_revert_invalid_token_id (gas: ~9361)

Success data:
    0x4552433732313a20696e76616c696420746f6b656e204944 ('ERC721: invalid token ID')

[PASS] tests::vault_extensions::hyp_erc20_vault_test::test_withdrawal_after_drawdown (gas: ~16647)
[PASS] tests::vault_extensions::hyp_erc20_vault_test::test_synthetic_transfers_with_rebase (gas: ~15335)
[PASS] tests::vault_extensions::hyp_erc20_collateral_vault_deposit_test::
    test_fuzz_erc4626_vault_deposit_remote_transfer_deposits_into_vault (runs: 256, gas: {max: ~8937, min: ~8553,
     mean: ~8935.00, std deviation: ~23.96})
[PASS] tests::hyp_erc721::hyp_erc721_test::test_erc721_remote_transfer (runs: 256, gas: {max: ~10802, min:
    ~10435, mean: ~10599.00, std deviation: ~182.55})
[PASS] tests::vault_extensions::hyp_erc20_collateral_vault_deposit_test::
    test_erc4626_vault_deposit_remote_transfer_sweep_excess_shares_multiple_deposit (runs: 256, gas: {max:
    ~11236, min: ~11236, mean: ~11236.00, std deviation: ~0.00})
[PASS] tests::vault_extensions::hyp_erc20_collateral_vault_deposit_test::
    test_erc4626_vault_deposit_remote_transfer_sweep_excess_shares_12312 (runs: 256, gas: {max: ~9928, min:
    ~9928, mean: ~9928.00, std deviation: ~0.00})
[PASS] tests::hyp_erc20::hyp_erc20_test::test_erc20_remote_transfer_invalid_amount (gas: ~5861)
```

```
234
235 Success data:
236     0x45524332303a20696e737566666963696e742062616c616e6365 ('ERC20:␣insufficient␣balance')
237
238 [PASS] tests::vault_extensions::hyp_erc20_collateral_vault_deposit_test::
        test_fuzz_erc4626_vault_deposit_remote_transfer_withdraw_less_shares (runs: 256, gas: {max: ~10039, min:
        ~10039, mean: ~10039.00, std deviation: ~0.00})
239 [PASS] tests::vault_extensions::hyp_erc20_collateral_vault_deposit_test::
        test_fuzz_erc4626_vault_deposit_remote_transfer_sweep_no_excess_shares (runs: 256, gas: {max: ~8979, min:
        ~8595, mean: ~8977.00, std deviation: ~23.96})
240 [PASS] tests::vault_extensions::hyp_erc20_collateral_vault_deposit_test::
        test_fuzz_erc4626_vault_deposit_remote_transfer_sweep_revert_non_owner (runs: 256, gas: {max: ~10070, min:
        ~10070, mean: ~10070.00, std deviation: ~0.00})
241
242 Success data:
243     0x43616c6c6572206973206e6f7420746865206f776e6572 ('Caller␣is␣not␣the␣owner')
244
245 [PASS] tests::vault_extensions::hyp_erc20_collateral_vault_deposit_test::
        test_fuzz_erc4626_vault_deposit_remote_transfer_withdraws_from_vault (runs: 256, gas: {max: ~10043, min:
        ~9785, mean: ~10041.00, std deviation: ~16.12})
246 Tests: 48 passed, 0 failed, 0 skipped, 0 ignored, 0 filtered out
247 Fuzzer seed: 12236578350745337443
```