The Ultimate Guide To Web3 Security

HOW TO SAFEGUARD EVERYTHING YOU BUILD, RUN, AND OWN IN WEB3

HYPERNATIVE

Foreword

This guide is the result of hundreds of hours we spent in conversations with our 300+ clients ranging from major blockchains and DeFi protocols to asset managers and centralized exchanges. The process of understanding their security needs, expectations, and capabilities revealed something—there is not yet a single standard for security in Web3.

For the past several years, the gap between monetary value of onchain assets and security readiness of Web3 projects has been far too wide, resulting in a bonanza for bad actors. This chapter of crypto history is coming to a close, as blockchain infrastructure, onchain primitives, and use cases reach a new level of maturity. There is also now a security stack and the accumulated industry expertise that Web3 projects and institutions can leverage to proactively guard against the vast majority of risks.

Whether you are launching a DeFi protocol, managing funds in Web3, providing payment rails, launching a stablecoin, operating a blockchain, crypto wallet or an exchange, we believe that the information contained in this guide can help you safeguard everything you build, run, and own in Web3.

Stay safe out there,

Gal Sagie
Co-founder & CEO, Hypernative

Introduction

Evolution of online security: Web2 --> Web3

It seems almost charmingly quaint that 10 years ago the Lazarus Group hacked Sony Pictures solely because of a movie where Seth Rogen assassinates Kim Jong-un. These days, the North Korean hackers have moved on to the less glamorous work of plundering billions of dollars from DeFi protocols and crypto exchanges.

This change tracks the Internet's own evolution, which a16z's General Partner Chris Dixon succinctly describes as "Read > Write > Own". The addition of a value layer has revolutionized online possibilities, enabling new forms of interaction and innovation. It has also supercharged the incentives for bad actors, creating a more complex and risk-prone digital landscape.

The evolution of online security from Web2 to Web3 also reflects a significant shift in both technological architecture and the nature of threats. Web3 introduced a decentralized paradigm powered by blockchain technology, shifting security challenges from protecting central servers to safeguarding decentralized networks and assets.

The emergence of new threat vectors and vulnerabilities underscores the importance of real-time monitoring, code audits, and robust protocol design to secure Web3 ecosystems. The security paradigm that emerges is characterized by a full-stack proactive approach to safeguarding onchain infrastructure and assets.

The Cost of Getting it Wrong

Web3 projects lost more than $2.71B in funds to hacks, exploits, and private key compromises last year. For comparison, total losses in all of in 2024 were $2.21B, a clear sign of how rapidly attackers are evolving. Defending against increasingly sophisticated attackers requires Web3 projects to adopt a comprehensive security posture—one that includes audits, real-time monitoring, operational security, and post-incident response planning.

01

Security Posture

Security Posture

Security is not an end state, it is an ongoing process. There will never be a final victory against bad actors, but you can build a security posture that consistently keeps you a step ahead. Whether you operate a protocol, manage digital assets, or assess partners and integrations, strong security depends on vigilance across every layer of your stack. The following pillars outline the core components of a modern, resilient security posture.

Continuous Monitoring, Real-Time Detection, and Automated Response

Real-time monitoring is the foundation of a modern Web3 security posture, but visibility alone is not enough. Once a protocol or asset is live, risks can emerge within minutes: malicious contract deployments, bridge irregularities, governance proposals, market dislocations, wallet compromises, or user-level phishing attacks. In this environment, security requires both instant detection and **the ability to act immediately.**

Continuous monitoring closes the gap between awareness and response. It gives teams an always-on view of their security, financial, and operational exposure, while **automated response** mechanisms ensure critical issues are contained before they escalate.

KEY FEATURES

- **Continuous Monitoring:** 24/7 observation of onchain activity (smart contracts, transaction events, governance proposals, wallet interactions) and offchain signals (Web apps, price feeds, governance sites, vulnerability databases).
- **Versatile Risk Coverage:** Detection across security (hacks, exploits, smart contract vulnerabilities), operational (private key theft, multisig risks, compliance, fraud), technical (frontend attacks, infrastructure issues, staking or node failures), financial (price or oracle manipulation, large withdrawals, depegs), and governance (suspicious DAO proposals, governance token exposure) categories.
- **Event-driven Alerts:** Instant notifications delivered through preferred channels (Slack, Telegram, Discord, Webhooks, PagerDuty, email) when suspicious or potentially malicious activity is detected.
- **Automation and Response:** The ability to trigger predefined actions (blocking transactions, pausing flows, freezing wallets, invoking emergency controls) to mitigate damage in real time.
- **Integration with Other Security Tools:** Alignment with audits, internal testing, bug bounties, wallet policies, and other operational safeguards to form a comprehensive security framework.

This layer is not passive observability. It is an active control system that enforces your risk tolerance, protects your treasury and users, and ensures your assumptions remain valid as onchain conditions change.

CASE STUDY:

How Kinetic Stopped a Hack and Saved $5M With Hypernative

On Oct. 31, 2024, Kinetic Market, the premier lending protocol on the Flare Network, was targeted by a hacker using flashloans and exploiting a permission vulnerability in a whitelisted liquidator to masquerade as a Uniswap pool. Kinetic was able to save $5M thanks to a detection by Hypernative that alerted the protocol to an exploit in progress.

KEY BENEFITS

- **Prevent Financial Losses:** On transparent blockchains, attackers can easily spot lucrative targets and even modest holdings can attract unwanted attention.
- **Safeguard Against Reputation Damage:** A successful exploit can harm trust and credibility of a project well out of proportion to monetary loss.
- **Stay Ahead of Attackers:** With fast-evolving technology like blockchains, not every threat can be predicted, but many can be detected. Even novel attack vectors often leave early warning signs.
- **Compliance and Risk Management:** Proactive security measures satisfy regulatory requirements and protect against fines, legal liability, and reputational harm.
- **Proactive Defense:** Early detection can trigger responses that prevent exploits entirely. When threats progress, rapid response contains and mitigates harm before it becomes unrecoverable.

VENDOR SELECTION CHECKLIST

When choosing a monitoring provider, projects should consider these key performance metrics:

1. **Breadth of coverage --** it is a multichain world and a monitoring solution must cover not only the chains you are building/operating on now, but also networks you might expand to in the future or where your users may bridge your tokens or interact with dApps;
2. **Detection volume and accuracy --** detecting most hacks is just half the battle; minimizing false positives is equally critical. High signal-to-noise ratio is essential for operational effectiveness, as excessive false alerts lead to alert fatigue, delayed responses, and increased risk of real threats slipping through unnoticed;
3. **Advanced warning --** contrary to popular belief, it is possible to detect hacks before they do damage, but correctly identifying the target of an exploit with enough time to react is what separates a great detection solution from a merely functional one;
4. **Automated response --** the ability to easily create automated triggers can be the difference between a scare and catastrophic loss;
5. **Proven track record --** the radical transparency of blockchain data means that barriers to entry are low, so look for signs of real customer traction.

Pre-Transaction Security and Policy Enforcement

Strong security requires controlling what happens before a transaction reaches the chain. Even with audits and continuous monitoring, onchain activity can still expose your protocol or treasury to malicious logic, hidden approvals, spoofed interfaces, risky counterparties, or subtle behavioral anomalies that only appear at execution time. This is where **pre-transaction security** becomes essential.

A dedicated pre-transaction layer evaluates intent, enforces policy, and blocks actions that fall outside your risk tolerance. By interpreting what a transaction will do, rather than what it claims to do, teams can prevent high-impact mistakes, insider misuse, phishing attempts, and zero-visibility logic embedded deep inside contract calls.

A strong pre-transaction security layer should provide:

- **Transaction Simulation:** Clear interpretation of a transaction's intent and its expected balance changes, token transfers, approvals, contract interactions, and side effects.
- **Policy Enforcement:** Customizable rules that define what is allowed or denied, based on your organizational thresholds (counterparty criteria, token lists, contract constraints, and behavioral norms).
- **Malicious Logic Detection:** Identification of hidden or suspicious behavior embedded inside multi-step interactions (unexpected approvals, impersonated contracts, drainer-like flows, or abnormal execution paths).
- **Protection Against User Error and Social Engineering:** Guardrails that help teams avoid blind signing, spoofed UIs, address poisoning, and interactions with malicious dApps.
- **Integration With Custody Infrastructure:** Compatibility with MPC and multisig wallet systems, enabling pre-signature enforcement across institutional custody stacks and internal key management workflows.
- **Automated Approvals and Denials:** The ability to automatically approve safe transactions, route medium-risk actions to reviewers, or block high-risk flows before any value moves onchain.

Pre-transaction security turns intent analysis into a live control layer. It ensures every transaction aligns with your policies, reduces manual review overhead, and prevents actions that could compromise your users, treasury, or governance. It is a core component of any security posture that aims to stay ahead of rapidly evolving risks.

Codebase Assurance and Pre-Launch Hardening

Before any system goes live, the underlying codebase must be reviewed, stress-tested, and hardened. While no audit can guarantee security, a disciplined approach to code quality significantly reduces the likelihood of critical flaws making it to production. This phase establishes the baseline assumptions that your monitoring and pre-transaction controls will later enforce.

Codebase assurance includes auditing, formal analysis, and extensive testing designed to uncover logic errors, access control issues, unexpected state interactions, and integration risks. The goal is to validate that the system behaves as intended under a variety of conditions and to eliminate weaknesses that attackers commonly exploit.

A strong pre-launch hardening process should include:

- **Independent Audits:** Multiple security reviews by reputable firms, covering smart contract logic, state transitions, authorization pathways, and protocol invariants.
- **Specialized Reviews:** Targeted assessments of complex or high-impact components (bridges, staking modules, oracle integrations, upgradeable contracts).
- **Formal Verification (Where Appropriate):** Use of mathematical methods to confirm that critical contract properties hold under all valid inputs.
- **Fuzzing and Automated Testing:** High-volume, randomized testing to surface unexpected behavior, edge-case failures, or improperly handled states.
- **Testnet and Adversarial Testing:** Deployments in controlled environments, allowing teams and external researchers to interact with the system under realistic conditions.
- **Upgrade and Deployment Controls**: Safeguards for contract initialization, ownership transfers, timelocks, and emergency functions to ensure secure rollout.

Audits and testing do not eliminate the need for continuous monitoring or pre-transaction security. Instead, they establish a stable foundation that reduces the number of unknowns once the system is live. Together, these practices help ensure the protocol begins its lifecycle from a position of strength.

Operational and Governance Controls

Security is not only a technical challenge; it is also an operational one. Even well-audited code can be compromised through weak processes, misconfigured permissions, rushed upgrades, or governance actions that introduce new risks. Robust operational controls ensure that critical changes, privileged access, and organizational workflows cannot be exploited, whether by external attackers or internal misuse.

This layer establishes the guardrails that govern how your system evolves over time. It reduces the surface area for human error, limits the blast radius of potential failures, and ensures that governance remains accountable, transparent, and resistant to manipulation.

A strong operational and governance control framework should include:

- **Privileged Access Management:** Clear policies for multisig configuration, signer rotation, access separation, and key management to minimize the risk of compromise or misuse.
- **Upgrade and Deployment Safeguards:** Use of timelocks, staged rollouts, verifiable builds, and emergency pause mechanisms to ensure code changes are visible, reviewable, and reversible when needed.

- **Governance Oversight:** Monitoring of proposals, voting patterns, delegate activity, and token distribution to detect suspicious behavior or concentration of influence.
- **Change Management Processes:** Documented procedures for contract upgrades, parameter changes, oracle adjustments, and administrative actions, with required approvals and review steps.
- **Operational Redundancy and Fail-safes:** Offchain and onchain contingency plans, including fallback mechanisms and role-based permissions to maintain continuity during incidents.
- **Third-party Risk Controls:** Assessment and monitoring of dependencies such as oracles, bridges, custody providers, relayers, reward distributors, and external contracts to ensure they do not introduce hidden vulnerabilities.

Operational and governance controls strengthen the security posture by reducing the chance that critical decisions can be rushed, subverted, or abused. When paired with real-time monitoring and pre-transaction enforcement, they help ensure that your system evolves safely and remains aligned with security and organizational priorities.

CASE STUDY:

$197M Recovered in Euler Finance Flashloan Attack

On March 13, 2023, Euler Finance fell victim to a flashloan attack that resulted in about $200M of funds lost by the permissionless borrowing and lending protocol on Ethereum. It took the protocol several days to trace the attacker and open lines of communication. Three weeks after the hack, the funds were returned.

User and Transaction Protection

Many security failures do not stem from protocol-level vulnerabilities, but from user interaction risks: blind signing, phishing, malicious dApps, spoofed interfaces, and confusing approval flows. Protecting users requires clear visibility into what a transaction actually does, strong guardrails around high-risk actions, and safeguards that reduce the likelihood of mistakes or social engineering.

User and transaction protection focuses on reducing friction, clarifying intent, and preventing unintentional exposure to malicious actors. These controls help ensure users, whether retail, sophisticated traders, or operational teams, understand the implications of the actions they take.

A strong user and transaction protection layer should include:

- **Transaction Transparency:** Clear, human-readable explanations of balance changes, token transfers, contract interactions, and approvals before any signing occurs.
- **Anti-Phishing Safeguards:** Screening of dApps, URLs, signatures, and RPC interactions to warn users about spoofed frontends, malicious contracts, or dangerous address patterns.
- **Address and Counterparty Checks:** Real-time assessment of reputational risk (scams, hacks, mixers, illicit activity) before a user interacts with another wallet, contract, or pool.
- **Approval Safety:** Guardrails to highlight unexpected or excessive token approvals, delegation requests, or permission escalations.
- **UI/UX Protections:** Trusted signing environments, transaction previews, visual warnings, and friction-increasing steps for high-risk actions such as unstaking, bridging, or invoking admin functions.
- **Support for Organizational Workflows:** Role-based permissions, approval routing, and tiered transaction workflows to prevent accidental or unauthorized actions within teams.

User and transaction protection reduces the likelihood of losses caused by deception, error, or rushed decision-making. When combined with pre-transaction enforcement and real-time monitoring, it provides a comprehensive defense against the social and behavioral risks that remain some of the most common sources of Web3 incidents.

CASE STUDY:

How Venus Saved ~$13M in a Phishing Attack Upon Hypernative's Detection

When a Venus Protocol user with a large BSC position fell victim to a sophisticated phishing attack, Hypernative flagged the attacker's contract as suspicious almost 18 hours before the first loss occurred. The moment funds began moving, the Venus team received Hypernative's alert, despite not being a customer, and paused protocol operations within minutes. That rapid response trapped the attacker's funds, enabled a coordinated recovery effort, and ultimately secured nearly $13M in user assets.

Compliance, Risk, and Treasury Safeguards

Protocols and organizations operate in an environment where financial, regulatory, and counterparty risks can change quickly. Exposure to sanctioned actors, illicit fund flows, market manipulation, depegs, liquidity shocks, bridge issues, and ecosystem dependencies can all threaten operational continuity and asset safety. Treasury operations also require clear oversight to ensure movements align with organizational policy and risk tolerance.

This layer focuses on maintaining **organizational-level resilience**: monitoring the health of your assets and counterparties, enforcing compliance standards, and ensuring treasury processes remain reliable across multiple chains and market conditions.

A strong compliance, risk, and treasury framework should include:

- **Counterparty and Address Screening:** Real-time assessment of reputational, behavioral, and regulatory risks (sanctions, illicit activity, mixers, fraud patterns) to prevent unintended exposure across counterparties and pools.
- **Liquidity and Market Risk Monitoring:** Alerts for significant withdrawals, price dislocations, oracle manipulation, depegs, liquidity shifts, and other market-moving events that can affect positions or collateral.
- **Treasury Governance and Oversight:** Policies defining who can move funds, under what conditions, and with what approvals, supported by clear review processes and internal sign-off requirements.
- **Operational Continuity Monitoring:** Visibility into chain health, bridge reliability, node integrity, and custody infrastructure to ensure that critical treasury functions remain stable.
- **Pool and Token Risk Evaluation:** Ongoing assessment of liquidity pool toxicity, token quality, deployer reputation, historical behavior, and concentration risk to guide deployment and position-sizing decisions.
- **Regulatory Alignment:** Documentation and monitoring practices that support frameworks such as SEC, MiCA, and IOSCO, including audit trails, definable KYT/AML policies, and organization-level compliance reporting.

These safeguards help institutions maintain trust, avoid inadvertent exposure, and operate responsibly across a complex and rapidly shifting environment. When combined with real-time monitoring, pre-transaction controls, and operational governance, they form a resilient foundation for secure digital asset management.

Ecosystem Transparency and Responsible Disclosure

Security does not end with internal controls. Web3 systems operate in open, interdependent environments where users, partners, researchers, and other protocols rely on clear communication. **Transparent practices** strengthen trust, support responsible behavior across the ecosystem, and ensure that issues

can be surfaced and addressed before they become systemic.

This layer focuses on how you communicate, document, and collaborate with the broader community. Strong disclosure practices help users understand risks, help integrators anticipate changes, and help researchers report issues safely and constructively.

A mature transparency and disclosure approach should include:

- **Clear Documentation:** Up-to-date resources covering architecture, upgrade paths, contract addresses, admin rights, timelocks, governance processes, and operational parameters.
- **Public Transparency on Changes:** Notice periods for upgrades, parameter adjustments, migrations, or governance actions that may materially affect users or integrators.
- **Responsible Disclosure Channels:** Well-defined processes for reporting vulnerabilities — including designated contacts, expected response timelines, and clear communication guidelines.
- **Selective Bug Bounty Programs:** Incentive programs launched once the protocol is sufficiently stable, audited, and capable of triaging submissions effectively.
- **Post-Incident Reporting:** Transparent communication following incidents or near misses, including root cause analysis and mitigation steps.
- **Community and Partner Communication:** Ongoing updates to partners, node operators, integrators, and users about relevant risks, security practices, and ecosystem dependencies.

Ecosystem transparency builds trust, improves resilience, and supports safer participation for everyone who interacts with your protocol or organization. When combined with real-time monitoring, pre-transaction enforcement, operational controls, and robust compliance practices, it completes a comprehensive, modern Web3 security posture.

CASE STUDY:

$10M Bounty for a Wormhole Vulnerability

On May 26, 2022, whitehat Satya0x discovered a critical bug in Wormhole that could have been devastating for the bridge if exploited. Satya0x disclosed his find on the Immunefi bounty platform and received a $10M payment. That figure is currently the largest single bug bounty ever paid out in history.

02

Best Practices

Best Practices

Navigating the complex and ever-evolving landscape of Web3 security can be daunting. Whether you're building a decentralized protocol, managing onchain assets, or securing an entire blockchain, the decisions you make about your security posture directly impact your resilience against adversaries. By leveraging the insights and tools outlined in this guide, including Hypernative's monitoring solutions, you can safeguard your Web3 project against potential threats and maintain the trust of your users and stakeholders.

For Protocols

Protocols go through three distinct lifecycle phases, each with its own set of best practices: development, deployment, and operational. Here are some key considerations for safely navigating your launch journey.

1. Development Phase

- Smart Contract Security
  - Conduct multiple audits.
  - Employ formal verification.
  - Use secure libraries.
- Key Management
  - Secure private keys with hardware or multisig wallets.
  - Implement role-based access control for admin operations.
- Development Tools
  - Identify vulnerabilities with static analysis tools.
  - Integrate automated testing & security checks into your development pipeline.

2. Deployment Phase

- Gradual Rollout
  - Deploy the protocol in stages, starting with a testnet or a soft mainnet.
  - Use admin control, circuit breakers, or pausing mechanisms to react quickly to issues.
  - Ensure key admin functions require multisig approval.
- Bug Bounties
  - Consider launching a bug bounty once your protocol is stable, audited, and supported by sufficient triage, monitoring, and incident response capabilities.

3. Operational Phase

- Monitoring & Automated Response
  - Engage in real-time monitoring of onchain and offchain activity across contracts, governance, treasury, and infrastructure.

  - Pair detection with automated responses to contain incidents, such as blocking transactions, pausing contracts, or triggering emergency workflows.
- Pre-Transaction Security & Policy Enforcement
  - Inspect transactions before execution to eliminate blind signing and prevent unintended or malicious actions.
  - Enforce policy-based controls on upgrades, parameter changes, governance execution, and administrative operations.
  - Use pre-transaction security layers, such as Hypernative Guardian, to simulate transactions, evaluate intent, and block unsafe actions before they are signed or executed.
- Operational Security Controls
  - Secure day-to-day protocol operations, including multisig workflows, admin keys, deployment pipelines, and emergency controls.
  - Ensure all operational actions are continuously inspected and unwanted or unauthorized changes are blocked.
- Incident Readiness & Recovery
  - Maintain predefined incident response playbooks and escalation paths.
  - Ensure automated containment mechanisms are in place to reduce blast radius while incidents are investigated and resolved.

KEY TOOLS AND CAPABILITIES FOR PROTOCOL SECURITY

→ **Audits and Formal Verification**
Independent security reviews and formal verification processes used to validate smart contract logic and reduce the risk of critical vulnerabilities before deployment.

→ **Static and Dynamic Analysis**
Automated analysis tools that help identify common vulnerability patterns, unsafe assumptions, and edge cases during development and testing.

→ **Bug Bounty Programs**
Incentivized disclosure mechanisms that encourage responsible reporting of vulnerabilities missed during audits and testing, helping protocols surface issues early.

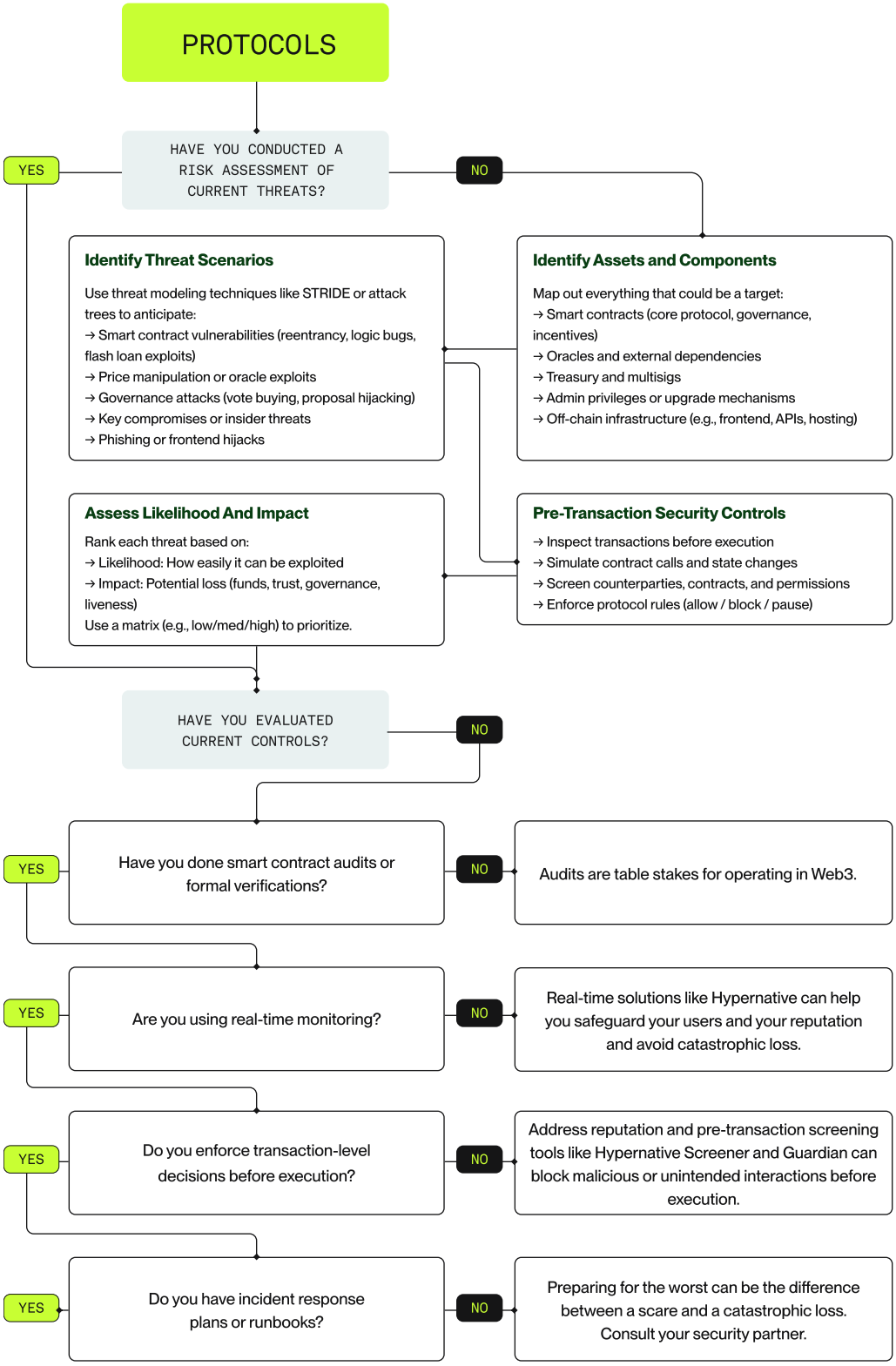
→ **Real-Time Monitoring and Operational Control**
Capabilities to observe live protocol activity and enforce security and governance policies during operation.

REAL-TIME MONITORING & OPERATIONAL CONTROL USE CASE

Protocols want to safeguard their users and reputation, and avoid catastrophic loss. Real-time threat prevention and operational control allows for:

- Threat Detection & Situational Awareness
  - Monitor onchain and offchain activity in real time to identify exploits, abnormal transactions, governance manipulation, oracle anomalies, and infrastructure attacks.
  - Maintain continuous visibility into protocol behavior as conditions change, including periods of market volatility or elevated attack activity.
- Operational Control & Enforcement
  - Use real-time signals to enforce operational controls before incidents escalate into losses.
  - Block, pause, or restrict unsafe actions when monitoring detects behavior that violates security or governance expectations.
- Pre-Transaction Intervention
  - Intercept risky operational actions before execution by inspecting transaction intent and expected outcomes.
  - Prevent blind signing and unintended changes to contracts, parameters, or treasury flows during routine operations and emergency scenarios.
- Automated Response & Containment
  - Trigger predefined automated responses when suspicious or malicious activity is detected.
  - Reduce blast radius by containing incidents immediately, even before human operators intervene.
- Operational Resilience During Live Events
  - Maintain control during fast-moving incidents, such as exploits, flash crashes, or compromised operators.
  - Ensure security tooling supports not only hack prevention, but also operational continuity under stress.

SECURITY DECISION FLOWCHART



Recommendation checklist:

- → Smart contract audits
- → Real-time monitoring
- → Pre-transaction simulation and enforcement
- → Kill switches, circuit breakers, or timelocks

For Blockchains

The complexity, transparency, and constant evolution of modern blockchains present a target rich environment for malicious actors. The first step toward safeguarding your network against attacks is securing the key infrastructure.

1. Consensus Mechanism Security for Proof-of-Stake (PoS)

- Establish slashing mechanisms to penalize validators for malicious or negligent behavior.
- Ensure high staking requirements to make attacks economically unfeasible.
- Promote validator diversity to avoid collusion or centralization.

2. Network Security

- Node Security
  - Run nodes on secure, updated operating systems with minimal services.
  - Use firewalls, rate-limiting, and anti-DDoS services to protect against attacks.
  - Deploy nodes in diverse geographical locations to avoid single points of failure.
- P2P Layer Security
  - Encrypt all peer-to-peer communications using protocols like TLS.
  - Implement sybil attack resistance mechanisms.

3. Smart Contract Layer

- Use real-time monitoring tools to detect and prevent chain-specific exploits.

Additional security checklist:

- Ecosystem Security
  - Secure cross-chain bridge mechanisms
  - Multisig bridges
  - Decentralized oracles
  - External data validation
- Real-Time Monitoring
  - Real-time alerts
  - Automated actions
- Incident Response
  - Response playbook
  - Community alerts
- Key Management
  - HSMs / MPCs

KEY TOOLS AND CAPABILITIES FOR CHAIN SECURITY

→ **Protocol and Client Audits**
Security reviews of consensus logic, client implementations, and upgrade mechanisms to reduce systemic risk.

→ **Validator and Governance Controls**
Safeguards governing validator behavior, governance execution, and privileged chain operations.

→ **Development Framework and Client Hardening**
Secure frameworks and clients used to build, maintain, and upgrade blockchain networks.

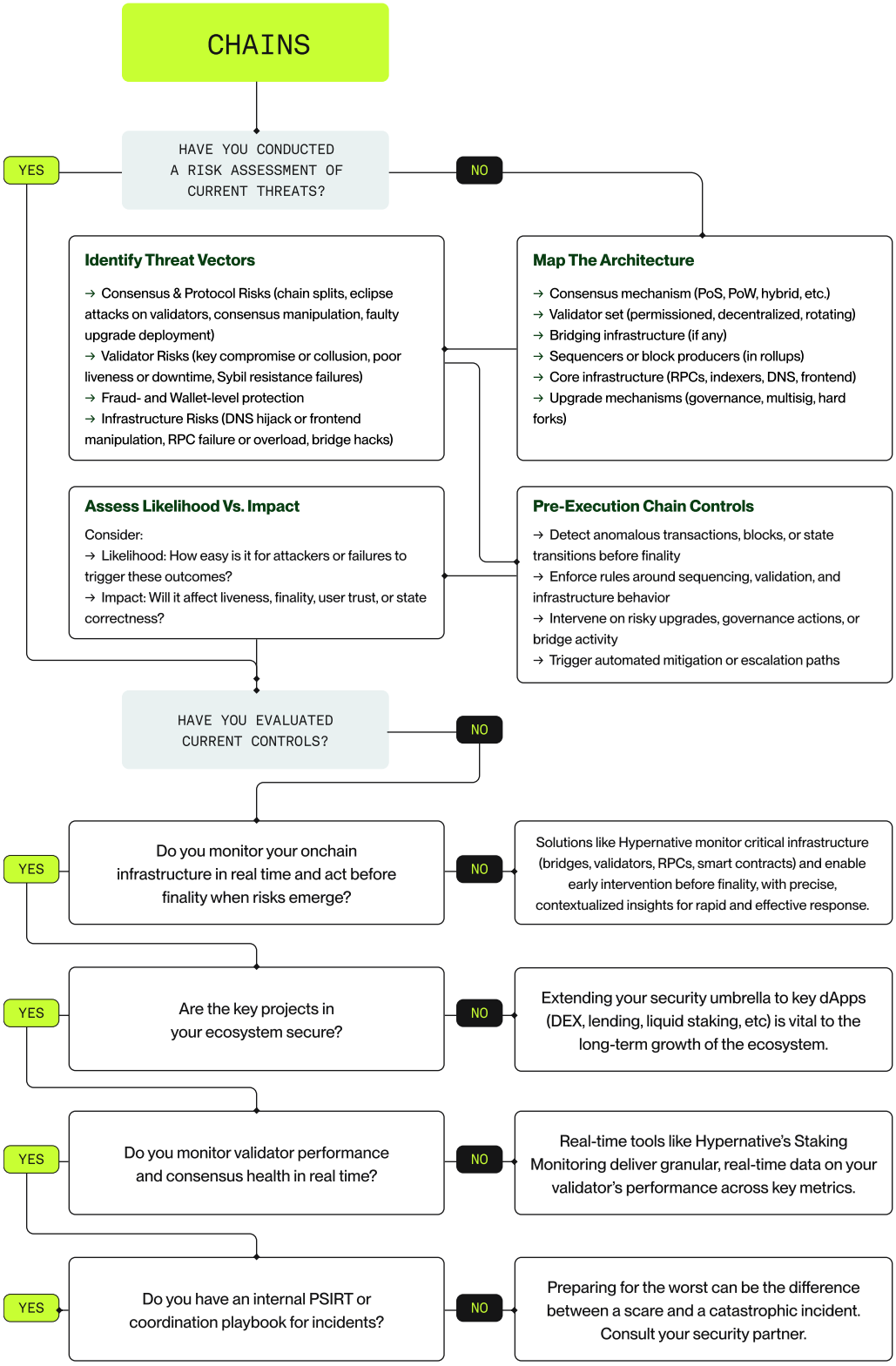
→ **Real-Time Monitoring and Operational Control**
Capabilities to observe network activity and enforce policy on critical chain actions during live operation.

REAL-TIME MONITORING & OPERATIONAL CONTROL USE CASE

Blockchains need to secure their blockspace to build flourishing ecosystems. Real-time threat prevention allows them to:

- Monitoring & Enforcement
  - Continuously monitor chain activity, validator behavior, governance actions, and ecosystem interactions for anomalies and emerging threats.
  - Enforce policies on privileged operations to ensure only expected and authorized actions are executed.
- Pre-Execution Inspection of Critical Actions
  - Inspect high-risk chain operations before execution, including validator configuration changes, bridge administration, sequencer updates, emergency upgrades, and governance execution.
  - Block, delay, or require additional approvals for actions that deviate from expected behavior or defined policy.
- Automated Response and Containment
  - Trigger predefined responses when suspicious or unsafe activity is detected, such as pausing components, isolating affected modules, or restricting further execution.
  - Maintain network integrity and operational continuity during incidents without relying on manual intervention.
- Stop breaches onchain by choosing the integration that fits best to maximize protection:
  - Ecosystem protection
  - Integrated onchain solution
  - Sequencer integration

SECURITY DECISION FLOWCHART



Recommendation checklist:

- → Real-time infrastructure monitoring and alerting
- → Ecosystem protection plan
- → Slashing for validator downtime or malicious behavior
- → Rollback/fork strategy
- → Pre-execution controls for validators

For Fund & Treasury Managers

The first rule of managing money is do not lose the money. Here is a quick checklist to make sure your funds are safe and sound:

- Use cold wallets for storage
- Use hot wallets for operations
- Implement multisig or MPC wallets
- Use hardware wallets
- Back up your keys
- Segment access for specific roles
- Use two-factor authentication

Hackers are increasingly turning to phishing and impersonator scams as improved smart contract security eliminates most common code vulnerabilities. Follow these best practices before you sign a transaction:

- Simulate transactions before execution
- Whitelist trusted addresses
- Set daily or per-transaction limits
- Implement timelocks for large or critical transactions

Even the best-managed protocols sometimes get hacked. But there are things you can do to manage counterparty risk:

- Use real-time monitoring of wallet activities and protocols
- Connect alerts to automated actions to withdraw funds or unwind positions
- Consider decentralized insurance
- Maintain emergency reserves
- Have an incident response plan ready

KEY TOOLS AND CAPABILITIES FOR FUND AND TREASURY SECURITY

→ **Custody and Wallet Infrastructure**
Secure custody setups with approval workflows and segregation of duties for treasury operations.

→ **Treasury Risk Management Controls**
Safeguards governing asset movement, counterparty exposure, and portfolio operations.

→ **Insurance and Risk Transfer Mechanisms**
Coverage structures used to mitigate residual smart contract, custody, and operational risk.

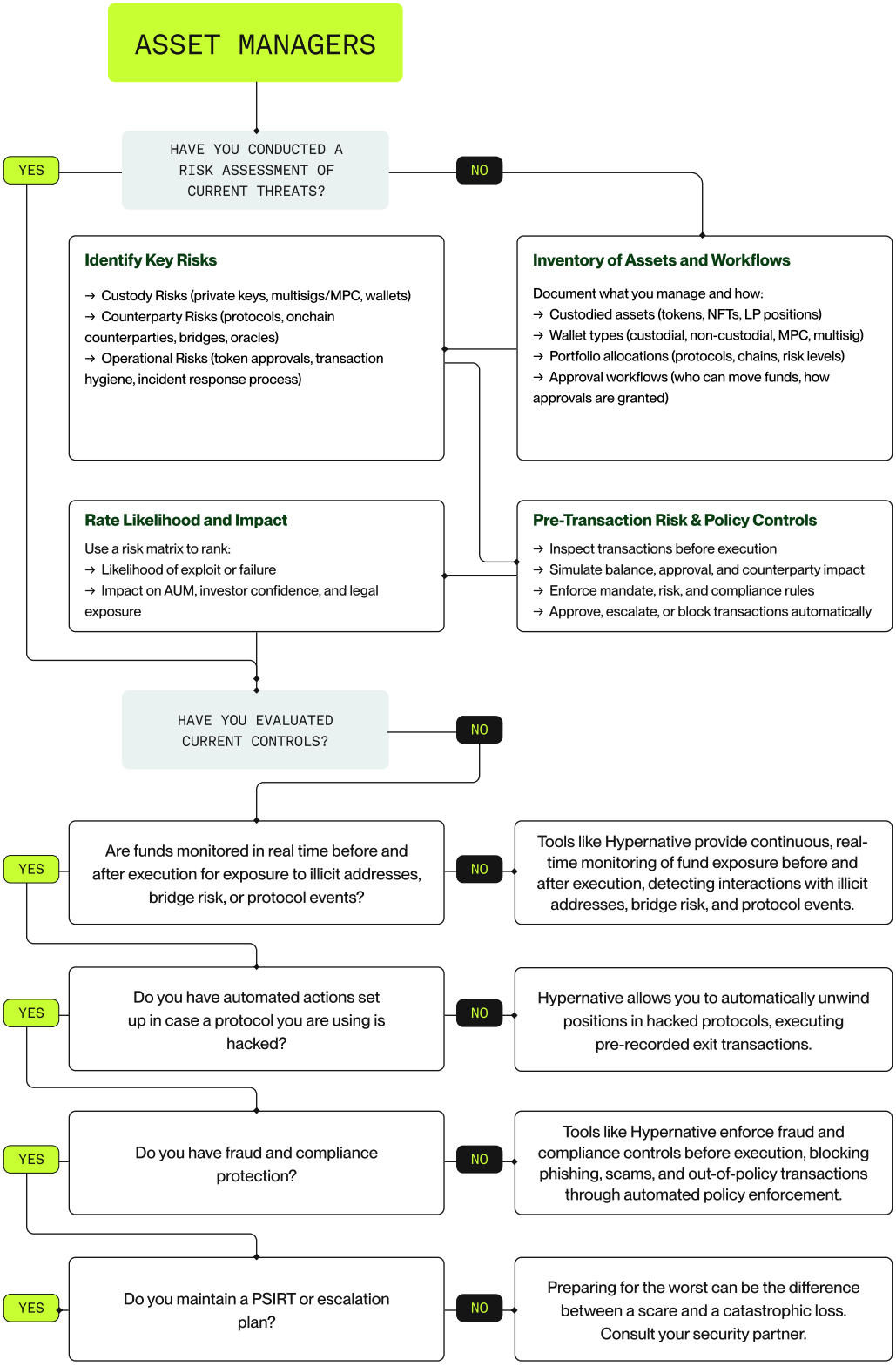
→ **Real-Time Monitoring and Enforcement**
Capabilities to observe treasury activity and apply policy controls during live operations.

REAL-TIME MONITORING & OPERATIONAL CONTROL USE CASE

Asset managers have a fiduciary duty to secure their positions against exploits, systemic, and market risks. Real-time threat prevention allows them to:

- Automatically pull out from hacked protocols
- Manage risk across all of their positions
  - Leverage real-time alerts to monitor all of their positions
  - Optimize their strategy with tailor-made and out-of-the-box push notifications about market risks and market performance
  - Use the rich context embedded in notifications to make swift decisions about their positions
- Manage staking, custody, and yield-generation risks
  - Receive real-time alerts about their staking provider's performance: slashing, missed rewards, initiated exits, and more
  - Monitor operational wallets, MPC wallet activity (such as Fireblocks) and emergency extraction of funds
  - Monitoring of DeFi protocols, liquidity pools etc for possible exploits

SECURITY DECISION FLOWCHART



Recommendation checklist:

- → Real-time monitoring
- → Automated actions
- → Incident handling playbook
- → Scheduled wallet rotation
- → Pre-transaction simulation and policy enforcement

For Centralized Exchanges

Centralized exchanges continue to be a major target for hackers, with some of the biggest recent heists linked to state-based actors. A proper security posture for an exchange has a large Web2 component, which is outside of the scope of this guide. Below are some Web3-specific best practices.

- Protect your funds
  - Use cold storage for the majority of user funds
  - Use hot wallets only for operational funds
  - Use warm wallets as an intermediary between cold and hot wallets
  - Use multisig wallets
  - Use HSMs or MPC for secure key management.
  - Regularly rotate private keys
  - Apply pre-execution inspection and policy controls to withdrawals, internal wallet movements, and administrative actions

- Mind your Web3 interfaces
  - Only integrate with audited smart contracts or protocols
  - Continuously monitor for anomalies
  - Implement withdrawal limits
  - Have an incident response plan ready
  - Implement a freeze mechanism
  - Put out a bug bounty

KEY TOOLS AND CAPABILITIES FOR CENTRALIZED EXCHANGE SECURITY

→ **Wallet and Custody Architecture**
Secure wallet infrastructure supporting controlled asset movement across exchange operations.

→ **Withdrawal and Operational Controls**
Safeguards governing withdrawals, internal transfers, and administrative actions.

→ **Incident Response and Recovery Processes**
Operational processes for investigation, containment, and recovery during security events.

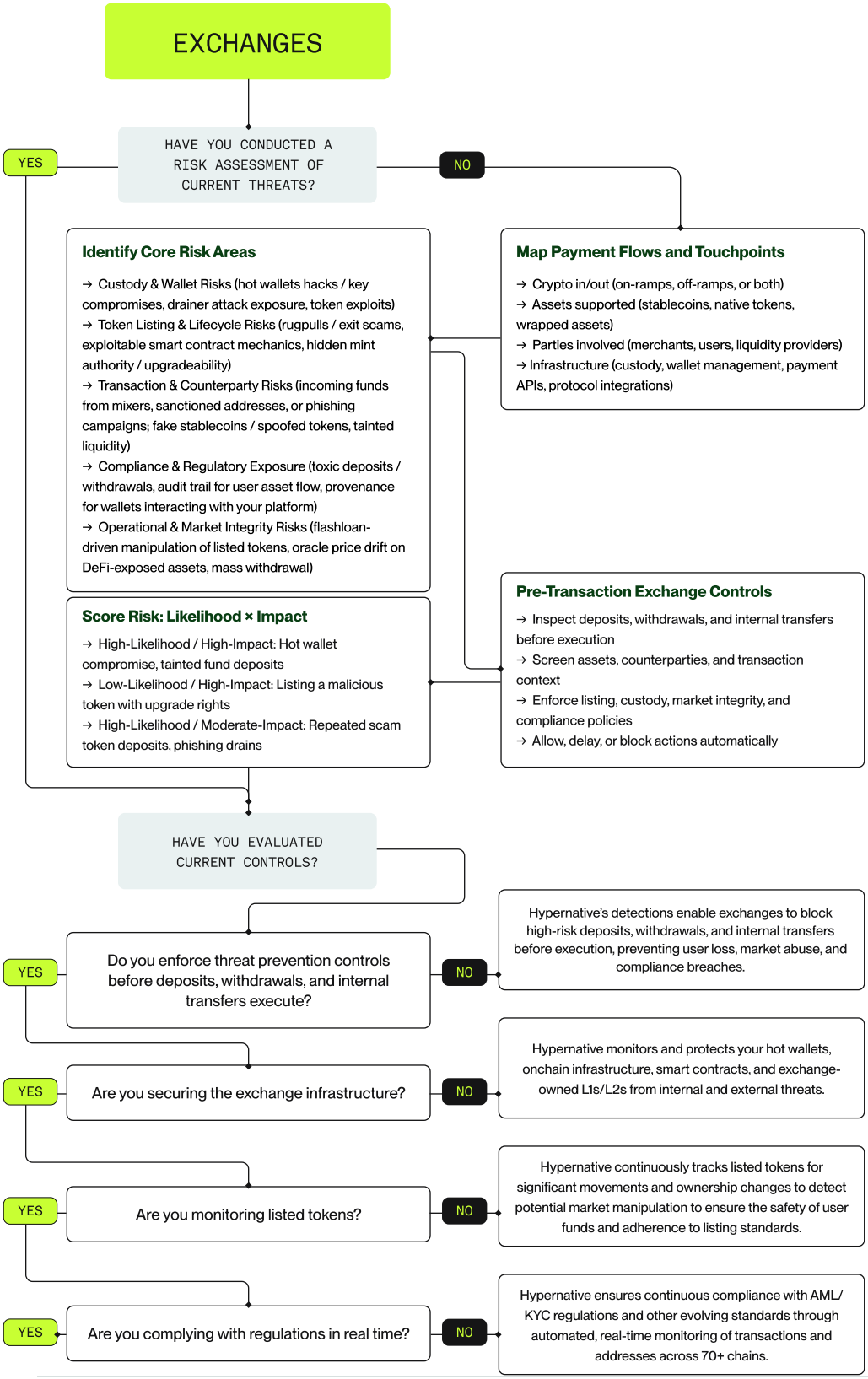
→ **Real-Time Monitoring and Enforcement**
Capabilities to monitor exchange activity and enforce policy across live transaction flows.

REAL-TIME MONITORING USE CASE

CEXs need better security and compliance solutions to onboard the next billion to Web3. Real-time threat prevention offers:

- Automatic, accurate, real-time detection of malicious activity for compliance teams
  - Safeguard users funds from hacks, exploits, phishing, scams, and fraudulent activity.
  - Prevent processing illicit funds with real-time wallet reputation data
  - Address shortcomings in current compliance solutions where it takes days to identify and label bad actors, often missing them altogether
- Increase security with proactive, real-time monitoring that stops hacks before they do damage
  - Monitor the protocols and smart contract risks related to assets you deploy funds to or interact with
  - Detect events and malicious activity across all chains to identify adverse exposure and vulnerabilities that impact their holdings directly or indirectly
  - Monitor hot wallets, onchain infrastructure, invariants, and operational risks like MPC policy for abnormal interactions, transactions and behaviors, private key thefts, access control, internal threats, alerts, and more

SECURITY DECISION FLOWCHART



Recommendation checklist:

- → Pre-transaction controls for deposits, withdrawals, and internal wallet movements
- → Token listing and lifecycle risk monitoring
- → Real-time monitoring of L1 and L2 infrastructure, smart contracts, and market integrity
- → Automated compliance and sanctions screening
- → Incident response and escalation

For Financial Institutions

For financial institutions entering Web3, security is both a trust imperative and a regulatory requirement. The stakes are high: Web3 introduces programmability, decentralization, and real-time settlement, but also exposes institutions to risks they are not accustomed to dealing with. Below are best practices tailored to the Web3 security posture of financial institutions:

1. Custody and Key Management

- Institutional-Grade Custody
  - **Qualified Custodians:** Use regulated, insured custody providers (e.g., Anchorage Digital, Fireblocks, BitGo, Coinbase, Copper, Hex Trust) for client assets.
  - **Multi-Party Computation (MPC):** Prefer MPC wallets for distributed key control without single points of failure.
  - **Cold Storage:** Store the majority of funds in air-gapped, cold environments with layered access control.
- Governance Controls
  - **Multi-Sig Approvals:** Require multiple approvers for large or sensitive transactions.
  - **Transaction Thresholds & Policies:** Enforce spend limits, time-delays, and approval flows that align with internal compliance requirements.

2. Smart Contract and Protocol Risk Management

- Due Diligence on Protocols
  - **Audit Verification:** Only interact with DeFi protocols or bridges that have undergone top-tier audits and have a strong security track record.
  - **Proactive Measures:** Avoid engaging with protocols without proactive defense measures in place, which includes real-time monitoring, transaction simulation, and automated response.
  - **Concentration Risk Monitoring:** Avoid overexposure to any single smart contract, chain, or protocol
- Real-Time Monitoring
  - **Transaction & Contract Monitoring:** Use platforms like Hypernative to watch for abnormal activity, rug pull signals, or exploit patterns.
  - **Risk Scoring:** Classify protocols and assets by risk level (e.g., unaudited contracts, high TVL, low liquidity).

3. Wallet and Access Security

- Segregated Wallets
  - **Per-Client or Per-Strategy Wallets**: Use segregated wallets to isolate risks and simplify forensic investigations.
- Authentication & Device Security
  - **Role-Based Access:** Apply RBAC principles to internal systems and signing operations.
  - **Secure Enclaves:** Use secure enclaves (e.g., AWS Nitro Enclaves, Azure Confidential Compute) for sensitive operations.

4. Compliance and Regulatory Alignment

- KYT / AML
  - **Blockchain Analytics:** Integrate tools like TRM Labs and Elliptic to trace fund provenance and detect high-risk behavior.
  - **Wallet Screening:** Screen all addresses interacting with the institution's infrastructure for sanctions exposure or blacklisted activity using tools like Hypernative Screener.
- Transaction Logging
  - **Immutable Audit Trails:** Record all onchain and internal approvals for compliance audits and dispute resolution.

5. Incident Response and Contingency Planning

- Incident Playbooks
  - Prepare runbooks for wallet compromise, smart contract failure, protocol depegging, or governance hijack.
  - Define escalation paths, internal and public communication templates, and recovery strategies.
- Fail-Safes
  - **Circuit Breakers:** Implement smart contract or operational kill switches for critical functions.
  - **Fallback Wallets:** Maintain secure, pre-authorized backup wallets in case of compromise.

KEY TOOLS AND CAPABILITIES FOR FINANCIAL INSTITUTION SECURITY

→ **Custody and Transaction Infrastructure**
Institutional custody systems with auditable approval and control mechanisms.

→ **Policy, Risk, and Compliance Controls**
Governance and risk frameworks governing onchain activity.

→ **Incident Response and Operational Resilience**
Processes to manage investigation, containment, and recovery during incidents.

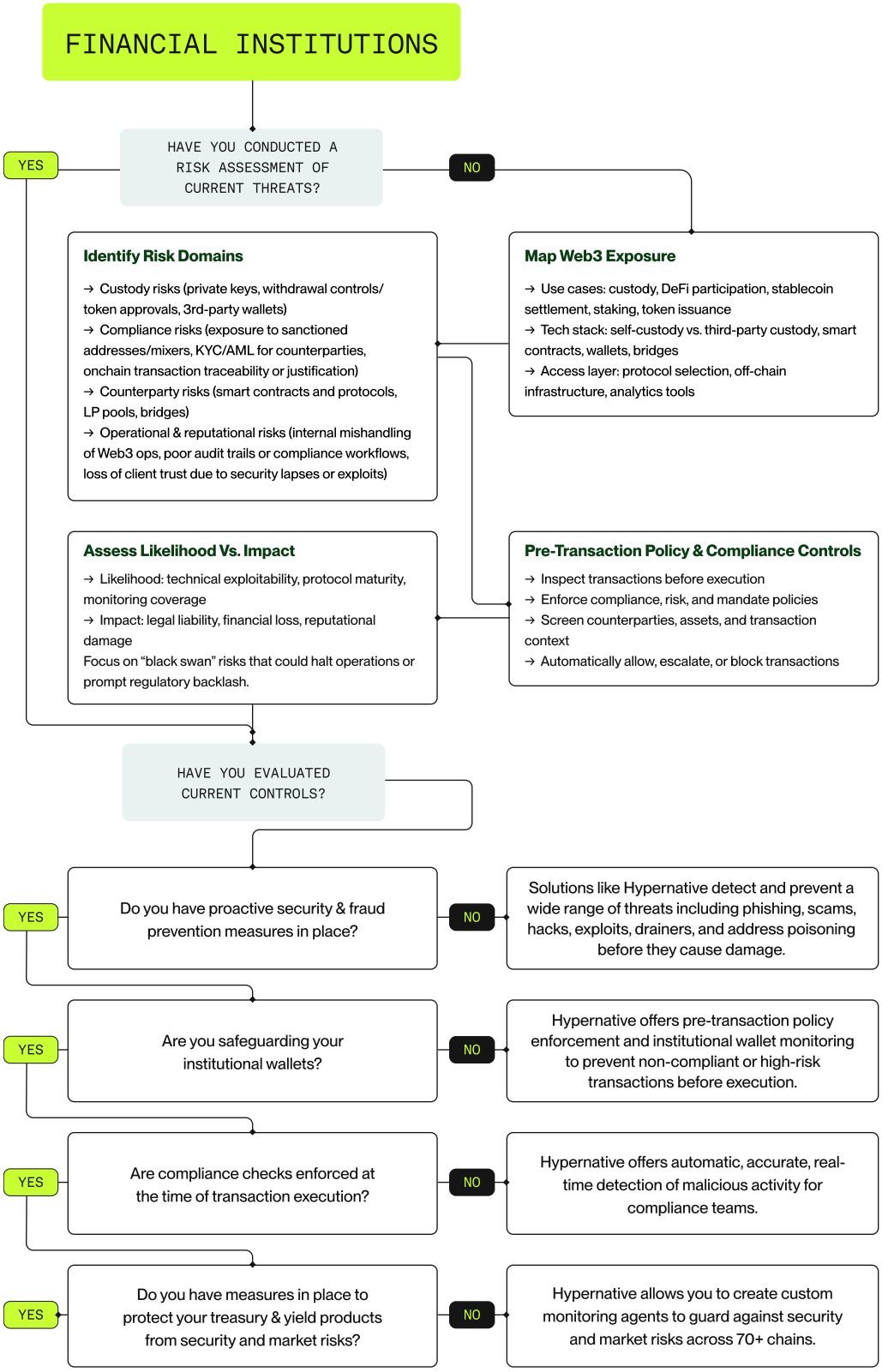
→ **Real-Time Monitoring and Enforcement**
Capabilities to monitor transactions and enforce institutional policies during live operations.

REAL-TIME MONITORING USE CASE

Banks, investment firms, and other financial institutions require a cross-chain security and compliance solution that prevents threats, fraud, and interactions with bad actors in real-time. The key objectives are:

- **Secure smart contracts for stablecoin & tokenization projects** with proactive real-time onchain monitoring and automated response solutions that stop hacks before they do damage;
- **Ensure user safety with wallet and transaction protection** using tools to evaluate if they are dealing with a potentially fraudulent counterparty for payment fraud, such as Authorized Push Payment (APP) scams;
- **Protect treasury & yield products** with custom monitoring agents to guard against security and market risks across multiple chains;
- **Safeguard institutional wallets** with pre-transaction protection and internal wallet monitoring to guard against security and other risks;
- **Comply with regulations in real-time** with automatic, accurate, real-time detection of malicious activity.

SECURITY DECISION FLOWCHART



Recommendation checklist:

- → Pre-transaction policy enforcement (mandates, compliance rules, sanctions screening, signer thresholds)
- → Real-time onchain threat monitoring (hacks, exploits, phishing, scams, fraud)
- → Institutional wallet safeguards (access controls, approvals, segregation of duties)
- → Auditability and compliance visibility (transaction traceability, alerts, evidence for regulators)
- → Incident response and escalation

For Wallet Providers

Wallet providers in Web3 sit at the front lines of user security. They are frequent targets for phishing, malware, supply chain attacks, and exploits at both the UI and smart contract levels. Below are security best practices tailored to wallet providers in Web3:

1. Key Management Security

- Non-Custodial Wallets
  - **Secure Key Generation:** Use strong entropy sources for generating private keys locally (e.g., device-secured RNG).
  - **Client-Side Storage Only:** Never transmit or store user keys/seed phrases on your servers.
  - **Encrypted Storage:** Use device-native secure enclaves (e.g., Secure Enclave on iOS, Android Keystore) for local key encryption.
- Custodial Wallets
  - **MPC or HSMs:** Secure private keys with multi-party computation (MPC) or hardware security modules (HSMs).
  - **Multi-Sig Custody:** For enterprise wallets, enforce multi-sig approvals for fund movement.
  - **Key Rotation:** Enable periodic or event-driven key rotation.

2. Phishing and UI Exploit Defense

- Transaction Clarity
  - **Human-Readable Signing:** Decode transaction data and display clearly what the user is signing (e.g., amount, destination, permissions).
  - **Permission Warnings**: Alert users when signing unlimited approvals or interacting with suspicious contracts.
- Anti-Phishing Measures
  - **Block Malicious URLs:** Maintain and update a list of blocked scam/phishing domains (via services like Chainabuse).
  - **Custom Anti-Phishing Tags:** Let users configure unique phrases in wallet UIs or emails to detect spoofing.

3. Smart Contract Interaction Security

- Simulation Layer: Offer pre-signature transaction simulations or safety scores.
- Contract Risk Scores: Integrate threat intelligence platforms (e.g., Hypernative) to warn users about suspicious or risky contracts before signing.

4. Incident Response & Monitoring

- Behavioral Monitoring: Detect unusual wallet behavior (e.g., out-of-pattern token approvals).
- Threat Intelligence Feeds: Integrate real-time threat detection tools like Hypernative.
- Bug Bounty Programs: Maintain active bounty programs (e.g., via Immunefi) to crowdsource vulnerability detection.

KEY TOOLS AND CAPABILITIES FOR WALLET PROVIDER SECURITY

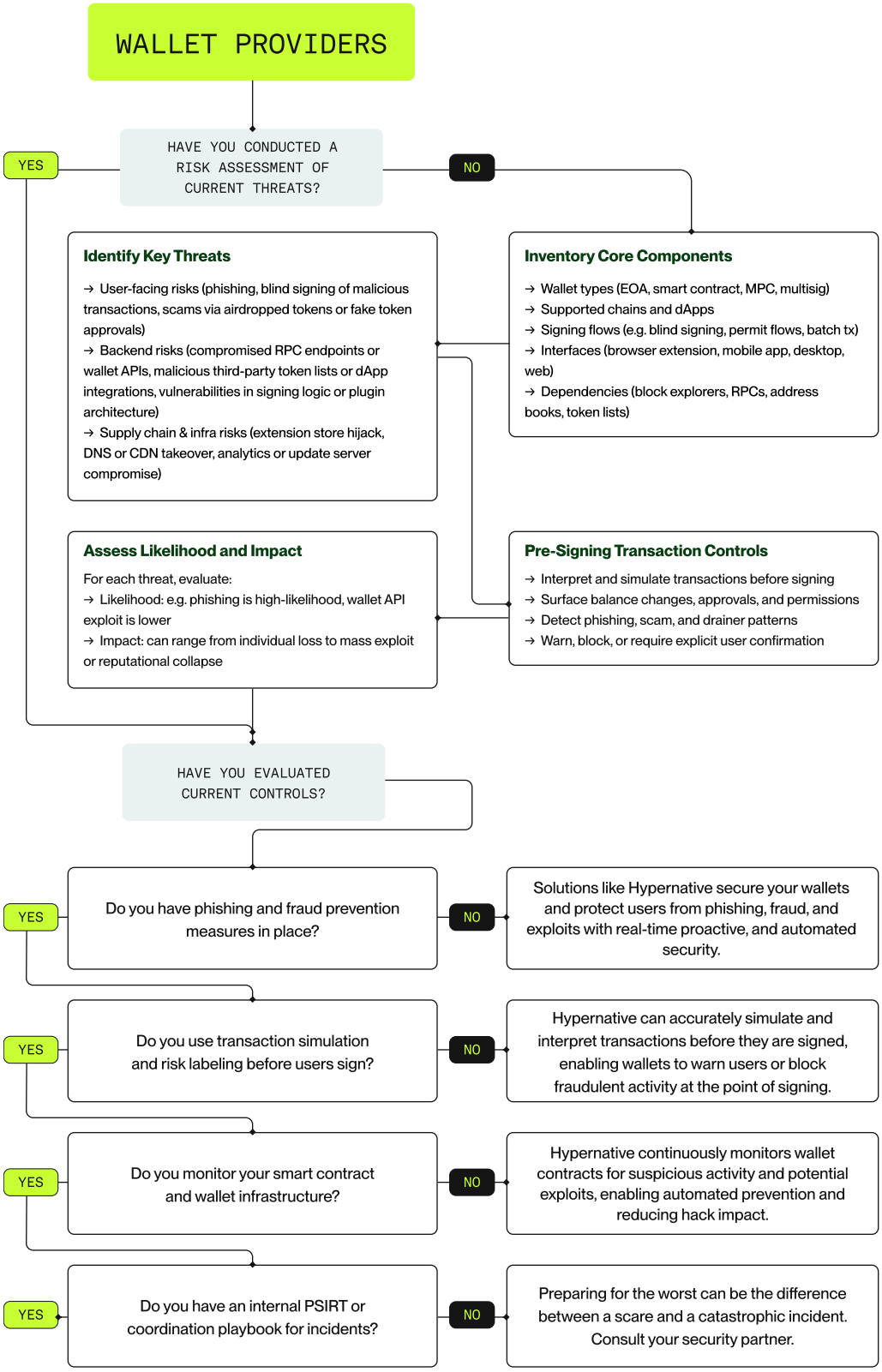
→ **Wallet and Signing Infrastructure**
Systems for key management, transaction construction, and signing workflows.
→ **User Protection Mechanisms**
Controls designed to protect users from unsafe interactions and malicious destinations.
→ **Vulnerability Disclosure Programs**
Bug bounty and responsible disclosure processes to identify wallet-specific vulnerabilities.
→ **Real-Time Monitoring and Enforcement**
Capabilities to assess risk and apply controls at the point of interaction and operation.

REAL-TIME MONITORING & OPERATIONAL CONTROL USE CASE

Securing wallets and protecting users from phishing, fraud, and exploits requires real-time, proactive, and automated security across both user interactions and wallet operations.

- **Real-time phishing and fraud prevention:** Protect users from phishing, pig-butchering scams, and drainer attacks before transaction execution through continuous cross-chain monitoring and real-time detection of malicious infrastructure, destinations, and behavioral patterns.
- **Proactive transaction simulation and intent interpretation:** Simulate and interpret transactions before they are signed to identify and prevent fraudulent activity, including malicious contract interactions, deceptive approvals, and unsafe execution paths.
- **Smart contract and wallet infrastructure security:** Continuously monitor wallet smart contracts and core infrastructure for suspicious activity, abnormal behavior, or potential exploits, enabling automated prevention and reducing the impact of attacks.
- **Operational controls for wallet systems:** Apply policy-based controls to wallet infrastructure changes, signing logic updates, and backend integrations before deployment, ensuring operational changes adhere to defined security and risk requirements.
- **Real-time monitoring and automated response:** Monitor onchain and offchain data sources across a broad range of risk types, with alerts triggering automated actions such as blocking transactions, pausing contracts, restricting access, or isolating affected components, seamlessly integrated into existing systems.
- **Custom risk monitoring and invariants:** Enable technical and non-technical teams to define custom monitoring rules, invariants, and automated responses based on wallet-specific risk profiles, operational workflows, and user protection requirements.

SECURITY DECISION FLOWCHART



Recommendation checklist:

- → Real-time threat monitoring for phishing/ malicious token alerts
- → Pre-signing transaction interpretation and enforcement
- → Blocking known drainer contracts or blacklisted dApps
- → Displaying transaction simulation or risk labels before signing
- → Incident response plan

For Payment Providers

Web3 payment providers sit at the intersection of user trust, financial compliance, and smart contract risk. Because they bridge real-world payments with decentralized systems, attackers see them as high-value targets. Below are security best practices tailored for Web3 payment providers:

1. Transaction Integrity & Authorization

- Secure Transaction Routing
  - **Whitelisted Destinations:** Restrict onchain payments to pre-approved contract addresses or merchants.
  - **Hash-Locking / Nonce Protections:** Use cryptographic locks to prevent replay attacks or duplicate payments.
  - **Smart Contract Escrow:** Use audited contracts to hold funds until delivery conditions are met (escrow/release logic).

2. Wallet and Key Management

- Custodial Models
  - **MPC-Based Wallets:** Use multi-party computation for transaction signing to avoid single key compromise.
  - **Key Rotation Policies:** Regularly rotate private keys and maintain secure backups in HSMs.
- Non-Custodial Models
  - **User-Managed Keys:** Never store or transmit seed phrases. Securely guide users through backup processes.
  - **Smart Contract Wallets:** Support programmable wallets (e.g., Gnosis Safe, Safe Modules) with spending limits and social recovery.

3. Smart Contract and Protocol Security

- Use Audited Infrastructure
  - **Smart Contract Reviews:** Deploy only audited and formally verified contracts for payment processing, invoicing, and escrow.
  - **Minimal Permissions:** Avoid granting protocols unlimited spending rights—use scoped allowances and expiration dates.
- Integration Risk Management
  - **Payment API Protections:** Sanitize and validate all data inputs to prevent contract injection attacks.
  - **Bridge & Oracle Security:** If relying on cross-chain payments or FX rates, choose secure bridges (e.g., LayerZero, Wormhole with insurance) and decentralized oracles (e.g., Chainlink).

4. Real-Time Monitoring & Threat Prevention

- Onchain Risk Detection
  - **Behavioral Monitoring:** Flag sudden surges in transactions, unusual token flows, or use of known malicious contracts.
  - **Exploit Signal Detection:** Integrate with real-time security platforms (e.g., Hypernative) to detect exploit precursors and block affected assets.
- Withdrawal & Payout Safety
  - **Timelocks for High-Value Transfers**: Add programmable delays for large merchant payouts.
  - **Transaction Simulation**: Pre-run transactions to identify errors or unexpected contract behavior before signing.

5. Frontend & UI Security

- Protect Users from Phishing and Blind Signing
  - **Transaction Previews:** Show human-readable descriptions of what users are signing (amount, token, destination).
  - **UI Integrity Checks:** Prevent front-end hijacking by using Subresource Integrity (SRI), CSP headers, and TLS.
- Prevent Typosquatting and UI Spoofing
  - **Domain Whitelisting:** Block transactions that originate from spoofed or malicious dApp domains.
  - **Anti-Phishing Codes:** Show user-set codes in communications to authenticate legitimate payment messages.

6. Backend Infrastructure Security

- API and Payment Gateway Hardening
  - **Rate Limiting and Throttling:** Protect payment APIs against abuse and DDoS.
  - **Zero Trust Networking:** Enforce identity-based access control and mutual TLS between internal services.
  - **Event-Driven Alerts:** Set triggers for anomalous API usage or payment approval patterns.
- Secure Storage and Processing
  - **Encrypted Databases:** Store payment data and merchant information with encryption-at-rest.
  - **Secure Processing Nodes:** Run critical backend services in isolated, hardened environments (e.g., VMs, containers with seccomp).

7. Regulatory & Financial Compliance

- KYC/AML Integration
  - **Partner with RegTech**: Integrate providers like Onfido or Jumio for onboarding users and merchants.
  - **Blockchain Forensics:** Use TRM Labs or Elliptic to detect high-risk addresses and suspicious flows.

- Transaction Screening
  - **OFAC & Sanctions Lists:** Block payments involving wallets flagged under sanctions regimes.
  - **Geofencing**: Enforce access restrictions for users from embargoed or high-risk jurisdictions.

8. Merchant & User Controls

- Granular Access & Permissions
  - **Role-Based Access Control:** Let merchants configure team roles (e.g., view-only, transaction signer).
  - **Spending Limits**: Allow merchants to set daily/weekly payout limits or approval thresholds.
- Refund & Dispute Handling
  - **Programmable Refund Logic**: Automate dispute resolution where possible via smart contract logic.
  - **Audit Trails:** Maintain immutable logs of all payment events and approvals.

9. Recovery & Fallback Mechanisms

- Failover Systems
  - **Redundant Chains / Rails:** Support multiple settlement paths (e.g., USDC on Ethereum and Base) to ensure uptime.
  - **Payout Queueing:** Queue payouts with fallback mechanisms in case a network or contract is down.
- Emergency Off-Switch
  - **Kill Switch or Circuit Breaker:** Allow platform admins to halt all onchain payment activity during active incidents.

KEY TOOLS AND CAPABILITIES FOR PAYMENT PROVIDER SECURITY

→ **Payments and Settlement Infrastructure**
Systems supporting on-ramp, off-ramps, stablecoin flows, and transaction settlement.

→ **Custody and Transaction Authorization**
Controls governing fund custody, approvals, and execution of payment transactions.

→ **Compliance and Screening Controls**
Capabilities for sanctions screening, transaction monitoring, and regulatory alignment.

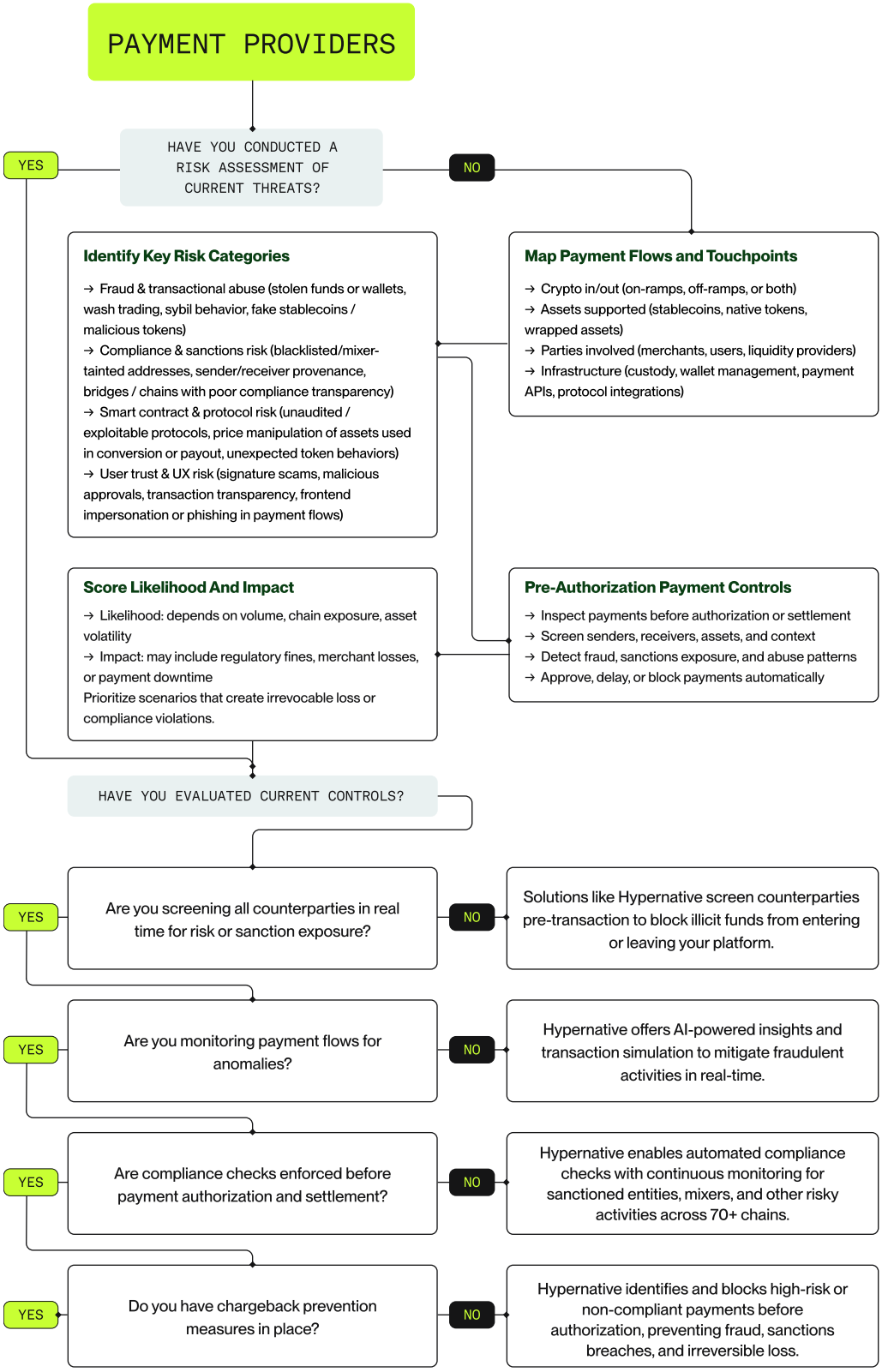
→ **Real-Time Monitoring and Enforcement**
Capabilities to observe payment activity and enforce policy across live payment flows.

REAL-TIME MONITORING & OPERATIONAL CONTROL USE CASE

Secure and reliable crypto payments require proactive fraud prevention and real-time compliance.

- Secure on-ramps & off-ramps
  - Prevent off-ramping tainted funds by screening for fraud, phishing, exploits, and illicit activities.
  - Safeguard on-ramps by preventing funds from being sent to scammers and fraudsters, ensuring clean transactions.
  - Vet counterparties for secure and compliant on/off-ramping.
- Real-time fraud & scam prevention -- Leverage high accuracy ML-driven risk insights that identify threats others miss using:
  - Pre-transaction screening API to detect fraud and scams before transactions are authorized.
  - Real-time detection of phishing, scamming, malicious dApps, drainers, address poisoning, and pig butchering.
- Operational control over payment infrastructure
  - Apply policy-based controls to payment logic, payout configurations, settlement contracts, and administrative actions before execution or deployment.
  - Prevent misconfigurations, unauthorized changes, or unsafe updates from impacting live payment flows.
- Automated compliance & screening -- Ensure compliance with:
  - Real-time, block-level screening for interactions with sanctioned entities, mixers, and illicit addresses.
  - Cross-chain and multi-hop risk analysis, going beyond traditional AML solutions.
  - Customizable risk engine compliant with FATF guidelines and local regulations.
- Effortless integration & scalability
  - Receive real-time alerts about your staking provider's performance: slashing, missed rewards, initiated exits, and more.
  - Monitor hot wallets or custodians (such as Fireblocks) and emergency extraction of funds.
  - Monitoring of DeFi protocols, liquidity pools, and others for possible exploits.

SECURITY DECISION FLOWCHART



Recommendation checklist:

- → Pre-authorization fraud and compliance enforcement
- → On-ramp and off-ramp security
- → Real-time monitoring for payment abuse and anomalies
- → Incident response plan

05

Conclusion

"It takes all the running you can do, to keep in the same place."

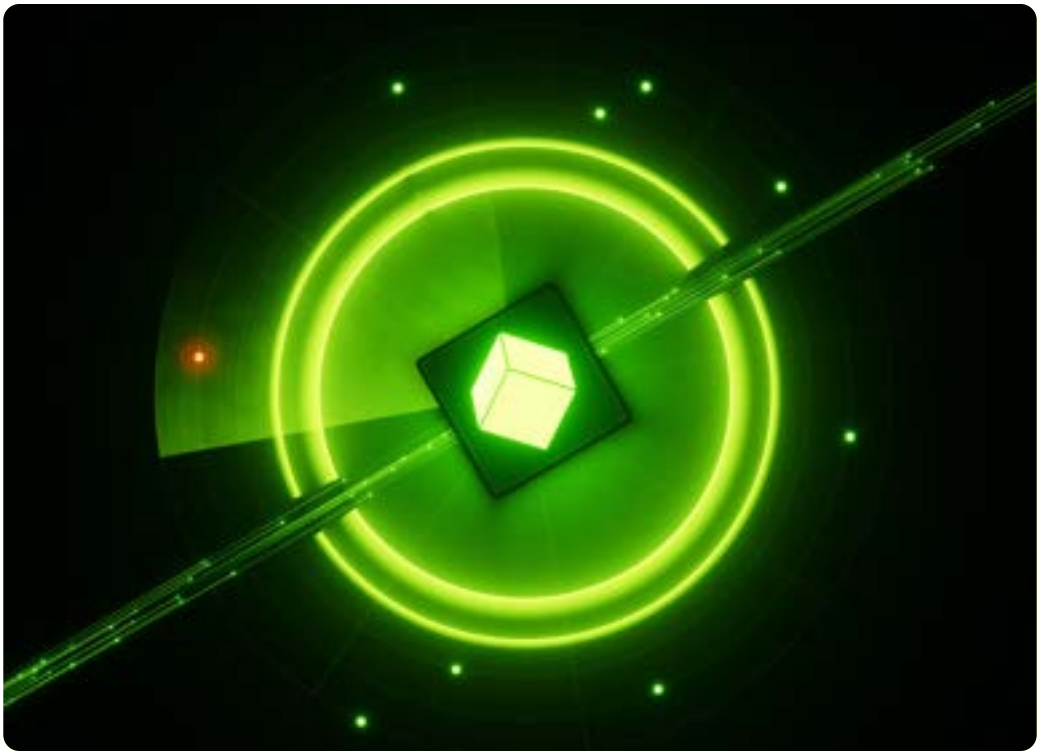
– RED QUEEN, ALICE IN WONDERLAND

Conclusion

Cybersecurity is subject to what is known as the Red Queen Effect, a concept derived from evolutionary biology that describes how species must constantly adapt and evolve to survive in an environment where competitors, predators, and parasites are continually evolving.

The effect is amplified in crypto by the breakneck pace of innovation and adversarial competition. This hyper-accelerated environment also means that many obvious exploits and common vulnerabilities, from reentrancy attacks and flashloans to oracle manipulation and governance exploits, have been addressed.

What emerges as a landscape dominated by more sophisticated and harder-to-predict challenges. To survive, defenders must have impeccable technical and operational security foundations while being continuously on a lookout for new threats.



04

About Hypernative

About Hypernative

Hypernative is the definitive trust and enablement layer for the digital asset economy. We provide institutional-grade infrastructure necessary to manage risk and scale operations with total confidence. Moving beyond simple monitoring, Hypernative delivers an end-to-end platform that unifies real-time defense, transaction simulation, and automated enforcement. By consolidating security, fraud prevention, compliance, and wallet protection into a single cohesive layer, we empower the world's leading enterprises and web3 businesses to eliminate threats before they manifest. Hypernative doesn't just protect assets, it provides the operational resilience and regulatory readiness required to lead in the digital asset era.

Trusted by the best

ethereum foundation · CIRCLE · eToro · bitpanda · galaxy · AMBER · SOLANA · INK · APTOS · consensys · Safe · Wave Digital Assets · reown · NEXO · Chainlink · Movement · Canton · LIDO · Morpho · Ethena · WINTERMUTE

By the numbers

300+
CUSTOMERS SERVED

70+
CHAINS SUPPORTED

$3B+
SAVED TO DATE

$100B+
ASSETS PROTECTED

99.5%
OF HACKS DETECTED

<0.001%
FALSE POSITIVE RATE

Need guidance on next steps?

Whether you're evaluating a new project, responding to a threat, or building a long-term security strategy, our team can help. We combine deep in-house expertise with a trusted network of top security partners to support you across audits, bounties, monitoring, and beyond.

GET IN TOUCH

CONTACT@HYPERNATIVE.IO →

Request a DEMO

BOOK A DEMO →

05

Appendix

Risk Categories

Security Risks

Definition: Risks arising from vulnerabilities in the system that can be exploited by attackers to steal funds, disrupt operations, or compromise data integrity.

TYPE	DEFINITION	MECHANISMS	MITIGATION
Smart contract vulnerabilities	Weaknesses or flaws in the code of smart contracts that attackers can exploit, potentially leading to significant financial losses or disruptions in dApps.	• Reentrancy Attacks • Integer Overflow and Underflow • Unchecked External Calls • Logic Errors	• Code Reviews: Conduct thorough internal reviews. • Audits: Hire third-party experts to audit contracts. • Bug Bounties: Incentivize white-hat hackers to find vulnerabilities. • Testing: Utilize unit tests, integration tests, and fuzz testing. • Real-Time Monitoring: Use tools like Hypernative for threat detection.
Malicious contracts	Smart contracts designed with the intent to exploit vulnerabilities, deceive users, or perform unauthorized operations. Characteristics of malicious contracts are: • Intentional Exploits: Malicious contracts often exploit known vulnerabilities in target contracts or protocols. • Deceptive Interfaces: Designed to appear legitimate to trick users or interacting systems. • Unauthorized Operations: Perform harmful actions such as draining funds, altering state variables, or stealing sensitive data	• Reentrancy Exploit Contracts • Phishing Contracts • Rug-Pull Contracts • Proxy Contracts with Hidden Logic • Gas Manipulation Contracts • Flashloan Exploit Contracts • Oracle Manipulation Contracts • Self-Destruct Contracts • Infinite Approval Exploits • Storage Overwriting Contracts	• Contract Audits: Regularly audit smart contracts for vulnerabilities that could be exploited by malicious actors. • Use Reputable Protocols: Interact only with well-audited and reputable contracts. • Implement Security Patterns: Use established patterns like Checks-Effects-Interactions (CEI) and guard against common vulnerabilities like reentrancy. • User Awareness: Educate users about phishing and fake contracts. Encourage verifying contract addresses through official channels. • Real-Time Threat Monitoring: Employ tools like Hypernative to monitor for malicious activity and potential attacks in real time. • Multi-Signature Wallets: Use multisig setups to add a layer of security to transaction approvals. • Limit Approvals: Encourage users to set specific approval limits for token interactions.
Bridge hacks	Attacks targeting cross-chain bridges, which are a critical component in the Web3 ecosystem. The complexity and the inherent risks of cross-chain interactions make them vulnerable to various types of attacks.	• Smart Contract Vulnerabilities • Validator Exploits • Key Management Failures • Replay Attacks • Oracle Manipulation • Fake Deposits and Withdrawals • Denial of Service (DoS) Attacks • Economic Exploits	• Smart Contract Audits: Conduct thorough audits by multiple reputable firms. Use formal verification to ensure correctness of bridge logic. • Decentralized Validation: Adopt decentralized and robust validator mechanisms, such as proof-of-authority or zero-knowledge proofs. • Enhanced Key Management: Use multi-party computation (MPC) for signing and private key management. Regularly rotate keys and implement secure key storage solutions. • Rate Limiting and Caps: Limit the maximum amount that can be transferred or withdrawn within a certain timeframe. • Multi-Layer Security: Use layered defenses, including real-time monitoring tools like Hypernative, to detect and mitigate attacks early. • Monitoring and Alerts: Deploy monitoring systems to detect unusual activities in real-time. • Use anomaly detection tools to identify potential exploits. • Bug Bounty Programs: Incentivize white-hat hackers to identify vulnerabilities before malicious actors exploit them. • Regular Upgrades: Continuously update bridge software to address vulnerabilities and incorporate the latest security practices.

Operational Risks

Definition: Risks arising from inadequate processes, human errors, or organizational inefficiencies that disrupt the functioning of Web3 projects.

TYPE	DEFINITION	MECHANISMS	MITIGATION
Multisig risks	Multi-signature wallets are a critical security feature in Web3, requiring multiple approvals (signatures) from designated parties to execute transactions. While they enhance security by reducing reliance on a single keyholder, they also introduce specific risks. These risks arise from technical, operational, and human factors associated with managing multisig wallets.	• Key Loss: If a signer loses access to their private key and the wallet requires a quorum of signatures, funds may become inaccessible. • Key Theft or Compromise: If a key is compromised, an attacker can collude with other compromised or malicious signers to execute unauthorized transactions. • Single Point of Failure: If a single entity controls multiple keys, the multisig loses its security advantage.	• Create a granular multisig setup to separate the roles of owner, upgrader, operator, and pauser • Use distinct signers for each multisig • Implement multisig Guard modules to limit / preapprove possible transactions • Track multisig activity for initiated transactions, changes to owners/ thresholds, and executed transactions • Monitor multisig signers for any non-multisig related activity
Private key compromise	Occurs when an unauthorized party gains access to the private key of a wallet or account, granting them full control over the associated blockchain assets or functionalities.	• Phishing Attacks • Malware and Keyloggers • Social Engineering • Insider Threats • Insecure Storage • Man-in-the-Middle (MITM) Attacks • Weak Passwords or PINs • Smart Contract Exploits • Physical Theft • Replay Attacks • Compromised Seed Phrases	• Secure Storage: Use hardware wallets or cold wallets for private keys. Store backups in secure, offline locations, such as physical safes. • Encryption and Passwords: Encrypt private keys with strong, unique passwords. Use password managers to generate and store complex credentials. • Multi-Signature Wallets: Require multiple signers for high-value transactions, reducing single points of failure. • Seed Phrase Management: Never share seed phrases. • Write seed phrases on physical paper and store them securely. • Two-Factor Authentication (2FA): Use 2FA on all platforms interacting with wallets. • Avoid Sharing Keys Online: Never input private keys or seed phrases into websites or apps unless absolutely necessary and verified. • Use MPC (Multi-Party Computation): Distribute private key computation across multiple parties to minimize risks. • Real-Time Monitoring: Use tools like Hypernative to monitor wallets and flag unauthorized transactions. • Regular Key Rotation: Periodically change private keys or use new wallets for critical funds. • Educate Users: Train teams and users on best practices for key management and phishing avoidance.
Phishing & Scamming	Malicious tactics designed to deceive users into divulging sensitive information (e.g., private keys, seed phrases, or wallet credentials) or tricking them into transferring funds to attackers. These attacks exploit the decentralized nature, anonymity, and high-value assets of Web3 to prey on users' trust or lack of technical understanding.	• Fake Wallet Interfaces • Phishing Emails and Messages • Malicious Smart Contracts • Impersonation on Social Media • Typosquatting • Mimicry of dApps • Rug pulls • Ponzi schemes • Fake airdrops	• Verify URLs and Platforms: Double-check URLs before entering sensitive information. Bookmark official platforms like wallet providers and exchanges. • Be Cautious with Wallet Connections: Avoid connecting wallets to unknown or suspicious dApps. Always read transaction details before approving. • Enable Two-Factor Authentication (2FA): Use 2FA on platforms that support it. • Use Trusted Sources: Download wallets and dApps only from official websites or app stores. • Never Share Private Keys or Seed Phrases: Legitimate platforms will never ask for these details. • Educate Yourself: Learn about common scams and phishing techniques to recognize red flags. • Use Hardware Wallets: Keep funds in hardware wallets for added security. • Leverage Security Tools: Use tools like Hypernative to monitor for phishing attempts and detect suspicious activities. • Social Media Vigilance: Verify accounts and avoid interacting with unsolicited messages.

TYPE	DEFINITION	MECHANISMS	MITIGATION
Ownership changes	The transfer of control or responsibility over a project, protocol, or key assets from one entity or group to another. These changes can introduce operational risks due to disruptions, mismanagement, or uncertainty arising during or after the transition.	• Acquisition of Web3 Projects: A popular DeFi protocol is acquired by a large corporation, leading to fears of centralization and higher fees. • Team Turnover in DAOs: Founders of a DAO resign, handing over governance to new participants, causing delays in decision-making. • Private Key Transfers: New owners inherit control of the protocol's treasury, but mismanage private keys, leading to a hack or loss of funds. • Token Distribution and Majority Control: A single party accumulates a majority of governance tokens during ownership changes, threatening decentralization.	• Transparent Communication: Ensure ownership changes are announced clearly and in advance. Publish detailed plans outlining the transition process. • Documented Processes: Maintain thorough documentation of operational, technical, and governance workflows to ensure smooth transitions. • Gradual Transitions: Implement phased handovers, allowing new owners to gradually assume control while minimizing disruptions. • Decentralized Governance Mechanisms: Use multi-signature wallets or DAO structures to distribute control and avoid centralized decision-making. • Independent Audits: Conduct audits before and after ownership changes to ensure compliance and security. • Escrow Arrangements: Use escrow mechanisms for key assets during the transition to prevent misuse. • Community Involvement: Involve the community in decision-making to maintain trust and ensure alignment with stakeholders' interests. • Training and Handover Procedures: Provide comprehensive training to new teams to minimize operational and technical disruptions. • Contingency Planning: Prepare fallback strategies for scenarios where ownership changes fail or new owners abandon the project. • Legal Safeguards: Draft clear agreements defining roles, responsibilities, and obligations of new owners.
Compliance (Sanctioned Addresses)	Compliance with sanctioned addresses refers to ensuring that blockchain-based platforms, protocols, and participants do not interact with or facilitate transactions involving addresses that are blacklisted due to regulatory sanctions, criminal activity, or other legal reasons. Failing to comply can result in operational risks, including legal penalties, reputational damage, and loss of access to essential services.	• OFAC Sanctions • EU and Global Sanctions • Anti-Money Laundering (AML) and Counter-Terrorism Financing (CTF) Requirements	• Automated Sanction Screening: Use blockchain analytics tools like Hypernative, Chainalysis, Elliptic, or CipherTrace to screen wallet addresses against sanction lists. • Smart Contract-Level Controls: Implement functionality to block transactions from known sanctioned addresses directly in smart contracts using tools like Hypernative Oracle. • Real-Time Monitoring: Deploy tools like Hypernative Screener to monitor transactions and flag suspicious or sanctioned activities in real time. • Layered Due Diligence: Perform Know Your Customer (KYC) and Know Your Transaction (KYT) checks where applicable. Use decentralized identity (DID) solutions for compliance without compromising user privacy. • Clear Governance Policies: For DAOs, establish voting mechanisms and governance rules to address compliance measures effectively. • Community Education: Educate users and contributors about the importance of compliance and the risks of interacting with flagged addresses. • Collaborate with Regulators: Proactively engage with regulators to demonstrate good faith compliance efforts and gain clarity on requirements. • Legal Safeguards: Maintain legal counsel to navigate jurisdiction-specific compliance challenges.

Technical Risks

Definition: Risks arising from flaws or limitations in the underlying technology stack, including the blockchain network, protocol design, or development tools.

TYPE	DEFINITION	MECHANISMS	MITIGATION
Frontend and infrastructure attacks	Technical exploits targeting the interfaces and underlying systems that support blockchain applications. These attacks aim to compromise users' trust, manipulate data, or disrupt operations. Such attacks represent significant technical risks as they affect the accessibility, security, and reliability of Web3 services.	Frontened attacks: • Domain Hijacking • DNS Spoofing (DNS Cache Poisoning) • Code Injection (Supply Chain Attack) • Phishing Through UI Manipulation • Session Hijacking • Cross-Site Scripting (XSS) Infrastructure attacks: • Node Exploits • API Abuse • Cloud Infrastructure Compromise • DDoS (Distributed Denial of Service) • Load Balancer Attacks • Data Breaches • Relay Server Exploits • Man-in-the-Middle (MITM) Attacks	Frontend Security Measures: • Secure Domain Management: Use domain registrar features like two-factor authentication and domain locking. • Dependency Management: Regularly audit third-party libraries and dependencies for vulnerabilities. • Content Security Policy (CSP): Restrict the sources of executable scripts on the frontend. • Regular Code Audits: Perform regular security reviews of frontend code. • User Education: Educate users about recognizing phishing attempts and verifying URLs. Infrastructure Security Measures • API Security: Use authentication, rate limiting, and encryption for API endpoints. • Decentralized Hosting: Distribute infrastructure across multiple providers or nodes to reduce single points of failure. • DDoS Protection: Implement DDoS mitigation tools and services to handle large-scale attacks. • Secure Node Operations: Use firewalls, private networks, and secure configurations for blockchain nodes. • Zero-Trust Architecture: Minimize access to critical infrastructure through robust access control policies.
Staking Risks	Technical security vulnerabilities and challenges associated with locking assets in a blockchain network to support its operations (e.g., consensus mechanisms like Proof-of-Stake (PoS)). These risks can compromise the integrity of the staking process, the security of staked funds, or the proper functioning of the blockchain network.	• Smart Contract Risks • Slashing Risks • Centralization Risks • Custodial Risks • Infrastructure Risks • Liquid Staking Risks	• Smart contracts: Conduct thorough audits and implement formal verification. Use established, well-audited staking platforms. Implement upgradeable smart contract patterns cautiously. • Slashing: Use redundant setups for validator nodes. Regularly monitor validator performance. Join staking pools with reliable infrastructure. • Centralization: Promote and support decentralized staking options. Diversify validators and staking pools. Use decentralized liquid staking protocols. • Custodial: Use non-custodial staking platforms. Choose well-audited and insured custodial providers. Use hardware wallets for added security. • Infrastructure: Run redundant validator nodes with failover systems. Store keys securely using hardware security modules (HSMs) or hardware wallets. Use DDoS protection services for validator infrastructure. • Liquid staking: Choose established and well-audited liquid staking protocols. Monitor liquidity and peg stability. Diversify across different liquid staking providers.
Chain nodes risks	Vulnerabilities and challenges that can compromise the functionality, security, or reliability of these nodes, threatening the integrity of the blockchain ecosystem.	• Network Connectivity Risks • Node Exploits and Attacks • Consensus-Related Risks • Software Vulnerabilities • Hardware Risks • Data Storage Risks • Geographic and Jurisdictional Risks	• Network Connectivity: Use reliable internet connections and redundant systems. Deploy nodes in multiple geographic locations to reduce latency and avoid regional outages. Implement monitoring tools to detect and address connectivity issues in real time. • Node Exploits and Attacks: Use DDoS protection tools and secure firewalls. Enable encrypted communication protocols (e.g., TLS) for node communications. Regularly update node software and enforce strong access controls. • Consensus-Related Risks: Monitor and audit validator node activity regularly. Use redundant setups to prevent unintentional double signing. Support decentralized and well-distributed node participation. • Software Vulnerabilities: Keep node software up to date and apply patches promptly. Use only officially supported client software versions. Conduct regular audits of node configurations. • Hardware Risks: Use enterprise-grade hardware with sufficient capacity for the blockchain's requirements. Regularly back up node data to prevent data loss. Monitor system performance and address resource bottlenecks. • Data Storage Risks: Implement secure data encryption for stored data. Regularly back up blockchain data in multiple locations. Use full nodes or archive nodes for critical operations requiring complete blockchain history. • Geographic and Jurisdictional Risks: Distribute nodes geographically across multiple regions. Use decentralized hosting services or self-hosted infrastructure. Monitor regulatory developments in jurisdictions where nodes are deployed.

Financial Risks

Definition: Risks related to the economic model or financial operations of a protocol, including liquidity, tokenomics, and user funds.

TYPE	DEFINITION	MECHANISMS	MITIGATION
Oracle & Pricing manipulations	Attacks that exploit the mechanisms by which blockchains obtain and use off-chain data (e.g., asset prices, exchange rates, or other real-world inputs). These manipulations can result in significant financial losses for decentralized finance (DeFi) protocols and users, as they undermine the reliability of the data that protocols depend on for operations like liquidations, collateral management, and automated market making.	• Flashloan Attacks • Low-Liquidity Manipulation • Single-Source Oracle Exploits • Time-Weighted Average Price (TWAP) Manipulation • Malicious Oracle Nodes • Front-Running Oracles	• Use Decentralized Oracles: Employ decentralized oracle networks (e.g., Chainlink) that aggregate data from multiple sources, reducing reliance on a single provider. • Implement Oracle Diversity: Use multiple independent oracles and require consensus among them for critical operations. • Introduce Circuit Breakers: Set thresholds for price changes that trigger protocol pauses or manual reviews in case of extreme price fluctuations. • Increase Liquidity in Pools: Ensure that DEXs and pools used for pricing have sufficient liquidity to resist manipulation. • Use Time-Weighted Price Feeds: Rely on long-term time-weighted average prices (TWAP) or volume-weighted average prices (VWAP) to smooth out short-term volatility. • Enhance Flashloan Protections: Implement measures to detect and mitigate flashloan-based manipulations, such as delaying oracle updates. • Continuous Monitoring: Employ real-time monitoring and analytics tools (e.g., Hypernative) to detect anomalies in price feeds or oracle data. • Decentralized Governance Oversight: Engage communities in reviewing oracle providers and pricing mechanisms to ensure accountability.
Significant withdrawals	Financial and operational threats that arise when large amounts of assets are withdrawn from a blockchain-based protocol or platform over a short period. These withdrawals can destabilize the protocol, disrupt liquidity, and lead to cascading effects, particularly in decentralized finance (DeFi).	• Market Downturns • Loss of Trust • Yield Migration • Whale Activity • Smart Contract Exploits • Regulatory Actions:	• Implement Withdrawal Limits: Introduce caps on the amount that can be withdrawn in a single transaction or timeframe to prevent liquidity shocks. • Diversify Liquidity Sources: Use multiple liquidity providers or pools to ensure sufficient reserves are available to meet withdrawal demands. • Create Dynamic Incentives: Use tiered or time-locked withdrawal fees to discourage sudden, large withdrawals. • Maintain Reserve Buffers: Hold a portion of assets in easily accessible reserves to meet unexpected withdrawal demands. • Automated Monitoring and Alerts: Deploy monitoring tools like Hypernative to detect large withdrawals or unusual activity in real time. • Foster Community Trust: Maintain transparent communication about the protocol's financial health and operations to prevent panic. • Encourage Decentralized Governance: Distribute governance tokens widely to reduce the influence of large stakeholders and mitigate risks of whale exits. • Stress Testing: Conduct simulations to assess the protocol's resilience to large withdrawals or market shocks. • Integration of Circuit Breakers: Pause certain functionalities or withdrawals temporarily during extreme market
Depegging	Financial and operational threats arising when a pegged asset, such as a stablecoin or a tokenized synthetic asset, loses its intended value parity (peg) to its underlying reference, such as fiat currency (e.g., USD), another asset (e.g., gold), or a basket of assets. Depegging events can lead to significant financial losses, liquidity crises, and systemic risks in decentralized finance (DeFi).	• Market Volatility • Insufficient Collateralization • Smart Contract Exploits • Liquidity Mismatch • Governance Failures • Oracle Manipulation • Regulatory Actions • Panic Selling	• Overcollateralization: Maintain reserves exceeding the value of the pegged asset to ensure stability. • Decentralized and Resilient Oracles: Use robust and tamper-resistant oracles to provide accurate and reliable price feeds. • Dynamic Peg Mechanisms: Employ mechanisms to adjust the peg dynamically during high market volatility. • Reserve Transparency: Ensure that reserve backing is verifiable, auditable, and accessible to the public. • Liquidity Management: Maintain deep liquidity in pools to absorb large trades without affecting the peg. • Circuit Breakers and Withdrawal Caps: Implement mechanisms to pause operations or cap withdrawals during periods of extreme volatility. • Diversification of Collateral: Use a basket of diversified assets as collateral to reduce dependency on any single asset. • Community and Governance Oversight: Involve community governance in decision-making to ensure quick responses to emerging risks.

TYPE	DEFINITION	MECHANISMS	MITIGATION
Large Flashloans	Flashloans are uncollateralized loans that must be borrowed and repaid within the same transaction. While they enable innovative use cases in decentralized finance (DeFi), such as arbitrage and liquidity provisioning, large flashloans pose significant financial risks due to their potential for exploitation. These risks arise primarily from the ability of malicious actors to execute large-scale operations with minimal upfront capital.	• Market Manipulation Risks • Protocol Exploitation Risks • Liquidity Risks • Arbitrage Exploits • Systemic Risk Amplification • Operational and Compliance Risks	• Use Secure Oracles: Employ tamper-resistant, decentralized oracles to prevent price manipulation (e.g., Chainlink oracles). • Implement Circuit Breakers: Temporarily halt protocol operations or certain functionalities if suspicious activity is detected. • Introduce Transaction Limits: Cap the amount that can be borrowed, swapped, or withdrawn in a single transaction. • Require Collateral for Critical Actions: Restrict key protocol actions, like governance voting, to staked or time-locked assets. • Audit Smart Contracts: Regularly audit protocol code to identify and mitigate vulnerabilities that could be exploited using flashloans. • Real-Time Monitoring and Alerts: Deploy monitoring tools like Hypernative to detect and respond to unusual loan or transaction activity. • Strengthen Liquidity Pools: Maintain deep liquidity to reduce the effectiveness of flashloan-based exploits. • Dynamic Pricing Mechanisms: Use algorithms to adjust prices dynamically and minimize the impact of large transactions.
Lending risks	Financial vulnerabilities and challenges associated with decentralized lending protocols. These protocols allow users to borrow and lend assets without intermediaries, relying on smart contracts and collateral mechanisms. While they provide accessibility and efficiency, they also expose participants to significant financial security risks.	• Smart Contract Risks • Collateral Risks • Oracle Risks • Liquidity Risks • Interest Rate Risks • Liquidation Risks • Counterparty Risks • Systemic Risks	• Smart Contract Security: Conduct regular audits and use bug bounties to identify vulnerabilities. • Collateral Management: Require overcollateralization and diversify accepted collateral types. • Robust Oracle Systems: Use decentralized, reliable oracles with multiple data sources. • Liquidity Reserves: Maintain protocol reserves to address sudden liquidity demands. • Dynamic Interest Rates: Implement models that adjust interest rates based on real-time supply and demand. • Efficient Liquidation Mechanisms: Automate liquidations with minimal price impact and ensure efficient auction systems. • Governance Safeguards: Use time-locked and community-reviewed governance proposals to prevent malicious actions. • User Education: Inform users about the risks of borrowing and lending, especially in volatile markets.
Sandwich attacks	A type of front-running attack in Web3, primarily targeting users executing trades on decentralized exchanges (DEXs). It involves an attacker manipulating the price of a token by strategically placing transactions before and after the victim's trade. These attacks exploit the transparent and public nature of blockchain transactions, where attackers monitor pending trades in the mempool to execute their strategy.	• Exploitation of Slippage • Gas Fee Manipulation • Mempool Monitoring	• Implement Anti-Front-Running Solutions: Use commit-reveal schemes where users first commit to a trade and reveal the details later, preventing real-time front-running. • Batch Transactions: Aggregate multiple trades into a single transaction to obscure individual user trades from attackers. • Adjust Fee Models: Introduce mechanisms that penalize attackers using priority gas fees to manipulate transaction order. • Decentralized Order Matching: Use off-chain order books or other matching systems to hide transaction details from attackers

Governance Risks

Definition: Risks related to the decision-making processes in decentralized governance structures, such as DAOs or protocols with token-based voting.

TYPE	DEFINITION	MECHANISMS	MITIGATION
Suspicious DAO proposals	Governance actions or initiatives submitted in a decentralized autonomous organization (DAO) that may compromise the protocol's integrity, manipulate funds, or undermine the interests of the community. These proposals exploit vulnerabilities in the DAO's governance mechanisms, posing risks to the security, fairness, and sustainability of the organization.	• Financial Exploitation • Centralization of Power • Ambiguous or Opaque Language • Manipulative Incentives • Exploitation of Governance Loopholes • Conflicts of Interest • Massive Token Dilution • Targeted Attacks • Rapid or Emergency Proposals	Improve Governance Processes: • Increase Transparency • Set Minimum Quorums • Lengthen Voting Periods Implement Safeguards: • Proposal Vetting Committees • Multi-Signature Approval • Require multi-signature validation for high-value treasury transactions • Time-Locked Proposals Strengthen Community Engagement: • Educate Members • Encourage Participation Use Advanced Tools: • Onchain Analytics • Reputation Systems • Simulations and Audits
Governance token risks	Governance tokens in Web3 enable decentralized decision-making for protocols, projects, and DAOs (Decentralized Autonomous Organizations). However, these tokens also introduce specific security risks that can undermine governance processes, protocol stability, and the broader decentralized ecosystem.	• Centralization of Governance Power • Token Concentration Risks • Token Volatility Risks • Token Bribery and Incentive Manipulation • Governance Capture	Centralization of Governance Power: • Implement quadratic voting or weight caps on voting power. • Encourage diverse token distribution during token sales and airdrops. Token Concentration Risks: • Require token staking or time-locking for governance participation. • Monitor large token transfers and flag suspicious activity. Token Volatility Risks: • Pair governance tokens with stable collateral mechanisms for critical operations. • Encourage long-term holding through staking rewards. Token Bribery and Incentive Manipulation: • Use anti-bribery mechanisms in governance protocols. • Implement slashing mechanisms for fraudulent or malicious voting behavior. Governance Capture: • Monitor token distribution and voting behavior for signs of collusion. • Encourage active participation from a broad community of token holders.