GUIDE

Incident Response Playbook for Web3 Teams

Compressing the time between something is wrong and the right person acting

Prepared for	Web3 teams	Classification	Public
Prepared by	Oak Security	Date	2026-06-17

Oak Security GmbH

1. What This Guide Is For

This guide is for anyone on a Web3 team who might be in the room when an incident hits: engineers, operators, communications, founders, and the signers who hold the keys. It assumes incidents will happen. Audits, monitoring, multisig discipline, and signer hygiene reduce the probability of a bad day. They do not eliminate it. Every mature protocol (Compound, Aave, Uniswap, Euler, MakerDAO) has had at least one war-room moment. The ones that survive prepared for it.

The clock in Web3 runs against irreversibility. The press cycle is compressed to hours, not days. Part of the response itself is executed in public, on chain. The first hour is the expensive one: minutes 0 through 60 decide whether funds are at risk or already gone, and the remaining hours largely allocate the consequences. There are no do-overs in that hour, and no time to invent the response while running it.

> **NOTE**
>
> **Baseline rule:** every control in this playbook exists to compress the time between "something is wrong" and "the right person is doing the right thing," and to make that right thing one that was written down, drilled, and pre-approved before the incident began.

The lifecycle follows NIST SP 800-61 (Prepare, Detect, Analyse, Contain, Eradicate, Recover, Learn), adapted for the Web3 context. This guide walks it in operational order: prevention, detection, pre-wired containment, the first 60 minutes, legal, communication, the Euler case study, post-mortem, recovery, the plan structure, and drills.

2. Defence in Depth: the Swiss-Cheese Model

The frame for prevention is the Swiss-cheese model from aviation safety. Each defensive layer is a slice of cheese with holes. No single slice catches every bug, misconfiguration, or social-engineering attempt. The bugs that survive a layer are the ones that align with its holes. Stack independent slices and the holes have to align across all of them for an incident to land. The work is not perfecting one slice. It is stacking enough slices with sufficiently independent failure modes that the alignment becomes rare.

Layer	Covers	Catches
Design-time	Specification, threat modelling, architectural simplicity, code-quality standards, internal review before outsiders are invited in.	Design flaws and unnecessary complexity.
Development-time	Unit, integration, and end-to-end tests; non-happy-path coverage; fuzzing and invariant testing for stateful logic; formal verification on value-bearing functions.	Implementation bugs, especially in the paths where most bugs live.
Release-time	Multiple independent audits, public contests on material releases, supply-chain monitoring, strict code-review and deploy policy, a funded bug bounty live on day one.	Bugs that survived design and development; introduced-at-release defects.

Each layer catches a different class of bug. The layers are stacked, not ranked.

> **NOTE**
>
> **Re-audit on every material upgrade.** An audit is of a commit, not of a codebase. A protocol that audits version 1.0 and ships version 1.7 has, in security terms, an unaudited protocol. Drift between audited state and deployed state is a recurring post-launch failure mode.

> Upgradeability without re-audit discipline is a slow-motion incident waiting for a vector.

3. Detection: Monitors That Actually Work

Detection catches the bugs the prevention stack missed. The defining property of a working detection layer: an alert reaches a human who is on the pager, empowered to act, fast enough that there is still time to act.

> **NOTE**
> An alert that pages nobody is not an alert. It is a log entry.

- **Invariant monitors.** The highest-value detection control for on-chain protocols. An invariant is a property the state must satisfy at all times: total supply equals the sum of balances; no vault falls below collateral ratio X; PnL across markets sums to zero. Most exploits violate an invariant before they drain. Encode invariants as runtime monitors (Forta detection bots, Tenderly alerts, custom on-chain checkers) and page the on-call when they break. This catches a substantial fraction of incidents in the first minute or two.
- **On-chain anomaly alerts.** Fill the gap invariants do not cover: large transactions relative to baseline, unusual call-graph patterns, unexpected admin actions outside change windows, oracle deviations beyond a threshold, mempool monitors watching for attacker-shaped bundles before they land. Heuristic, not provable. The goal is a tunable signal that catches enough real incidents early, not zero false positives.
- **Dry-run and simulation tooling.** Tenderly, Foundry's `forge simulate`, custom mempool replayers fork mainnet, replay pending transactions, and watch for revert patterns or state changes that indicate exploitation. Sophisticated teams run this as a continuous monitoring surface, not just a dev tool.
- **Cooldowns and rate limits.** The structural complement. They do not detect the incident; they buy the human time to react. A one-hour withdrawal cooldown on large amounts gives the on-call a chance to pause before funds leave. Cheap, standard, uncontroversial. Every protocol should ship with them.

The pager is the part most easily neglected. A 24/7 on-call rota with a primary and a backup, a rotation tested at least quarterly, and a runbook the on-call has actually read is

the difference between a 3-minute response and a 30-minute one. The cost of being woken for a false positive is much smaller than the cost of an alert that fired at 03:00 and was found at 09:00.

4. Pre-Wired Damage-Limitation Controls

The controls that limit damage during a live incident are the controls that exist before it. Generally-accepted controls (rate limits, circuit breakers, withdrawal cooldowns) are uncontroversial and should ship with every protocol. Debated controls (pause and kill switches, upgradeability, drain mechanisms) trade centralisation against operational defensibility and must be designed deliberately, not added in haste.

Control	Upside	Downside	Mature compromise
Pause / kill switch	Stops the bleeding instantly.	Centralisation risk; can be abused; undermines credible neutrality.	Scoped pause (per-market, per-function), multisig-gated, timelocked unpause, documented scope and activation criteria.
Upgradeability	Lets you ship the fix.	One compromised admin key from a rug-shaped incident.	Timelocked path, diffs published in advance, re-audit every upgrade, long public window.
Drain mechanism	Pre-empts an exploit by moving funds first.	Mechanically a rug vector.	Only with transparent scope, signer discipline, pre-published authorisation. Rarely the right control before the simpler ones are exhausted.
Whitehat front-run	Saves funds mid-flight.	On chain, looks identical to the team draining its own protocol.	SEAL Safe Harbor pre-authorisation; never executed without written counsel sign-off.

Whole-protocol kill switches operated by a single key are how protocols turn an exploit into a governance crisis. On whitehat front-runs, the [SEAL Safe

Harbor](https://securityalliance.org/safe-harbor) framework is a public legal scaffold that lets protocols pre-authorise intervention with defined scope, ruleset, compensation, and publication of recovered funds. Pre-publication is the point: it gives the defender legal and reputational cover during the incident. Even under Safe Harbor, no team member drains without explicit written counsel sign-off. The policy is the framework; the call is still made by lawyers.

5. The First 60 Minutes: a Flat-Out Runbook

This is the period in which the structural decisions are made: is this real, who is leading, what is paused, who is told, what is captured. Every later phase depends on this hour. The structure is the same for every severity. Speed and scope shift with the level.

1. **Minutes 0 to 5: triage.** Confirm the alert is not a monitor false positive. A second pair of eyes on the transaction trace. A quick calldata decode against the expected template. One named person says "go," not a committee. The fastest way to lose the first ten minutes is to spend them debating whether you have an incident.
2. **Minutes 5 to 15: name the Incident Commander.** The IC is one person with explicit role authority who decides who is on and who is off the response. The IC is not the technical lead. The IC's job is coordination and decision rights, not debugging. Say it out loud: "I am IC, [name] is tech lead, [name] is comms." The ritual has an outsized effect on how cleanly the hour runs.
3. **Minutes 5 to 15: stand up the war room.** Small and named. Tech response lead, communications lead, legal counsel, law-enforcement liaison. Usually four people, not forty. Keeping everyone else out is a control, not a slight: a leaked screenshot from a twelfth person is how attackers track your state and how press stories break early. Move to an out-of-band channel. A pre-built Signal group is the standard pattern. Assume Slack is compromised, monitored, or failing under load from the start.
4. **Minutes 5 to 30: pause decision.** Pause first if the protocol supports it. The cost of an incorrect pause is user inconvenience; the cost of a missing pause is fund loss. Pause the narrowest possible scope (the affected market, function, or module). Halting the whole protocol when one vault is at risk makes recovery harder and damages trust further. Draining to a safe vault is higher-stakes: it needs Safe Harbor cover, counsel sign-off, and the operational capability to do it without fat-fingering a second incident.
5. **Minutes 0 to 60: evidence discipline.** Snapshot everything before anyone touches it: infrastructure state, signer-machine disks, Slack history, git state, container images, log volumes. Capture attacker transactions by hash, with block number, calldata, event logs, and explorer screenshots, into the war-room record, not a personal DM. Engage a

chain-analysis firm (Chainalysis, TRM, Elliptic) early. Maintain a war-room log with timestamped decisions and their rationale.

NOTE

Do not rush an irreversible action in the first fifteen minutes. And beware the rush to conclude: "it was X" said too early becomes the press narrative. If the team is not sure, the team does not speculate.

6. Legal and Law Enforcement

The incident is a legal event from the moment it begins. Counsel comes before statements. Anything published during an incident is evidence, in civil litigation, in regulatory action, and potentially in criminal cases. A team that drafts its first statement before its first call to counsel is making evidentiary decisions without the legal frame to make them safely.

- **Jurisdiction matters.** Where the entity is incorporated, where the users are, where the funds moved, where team members reside. A protocol with US users, a Cayman foundation, a Swiss operating entity, and Korean attackers may need three or four lawyers on call before a single public statement.
- **Report to law enforcement early.** FBI IC3 in the US, Europol and the National Crime Agency in Europe, equivalents elsewhere. Chain-tracing is dramatically more effective when law enforcement can request exchange-side cooperation within hours, before funds reach a mixer or cross a bridge several times. Warm the relationship before the incident, not from inside one.
- **Negotiate with the attacker only through counsel.** Never through a team Slack. Public on-chain messages to the attacker are part of the legal record the second they confirm. Threats, offers, and timelines published unilaterally during an incident have created problems in later cases that legal then spent months unwinding.

> **NOTE**
> A specific warning for decentralised projects: taking on the "response coordinator" role for a DAO, or for a protocol with no clear legal entity, can transfer personal liability onto the coordinator. Counsel guides this, not a sense of duty.

7. Communication During an Incident

Four rules survive every iteration of the lifecycle.

- **Do not go dark.** Silence is read as guilt or chaos. A short holding statement within the first hour ("we are aware, we are investigating, an update will follow within X hours") buys time to investigate without the vacuum filling itself with speculation.
- **Single named spokesperson.** One voice. Not the CEO's 3am tweet, the CTO's contradictory Discord reply, and a community moderator speculating in parallel. The spokesperson's brief is in the runbook before the incident; the people behind the role rotate with vacations and turnover.
- **Only say what you know.** "Investigation ongoing." "User funds are paused." "Update at 14:00 UTC." Speculation becomes the press narrative and is hard to walk back. Corrections during an incident are survivable. Discovered lies end protocols.
- **No threats and no negotiation offers without counsel.** Public messages to the attacker are evidence. The communications lead and counsel co-author them; the IC signs off.

The exchange-notification template, the user-facing email, the social-media holding statement, the press contact list: write all of them before the incident. Under stress, no team composes them cleanly.

8. Case Study: Euler, March 2023, $197 Million Recovered

The textbook example of an incident response that worked. On 13 March 2023 at 08:56 UTC, a flash-loan exploit against Euler's `donateToReserves` function drained approximately $197 million across multiple tokens. The bug was a missed health-check interaction between two features that had each been audited, but never together in the specific configuration the attacker assembled.

The response began within hours. The team paused the affected markets. A small war room (tech lead, communications lead, counsel, law-enforcement liaison) was convened. Chain-analysis firms (TRM, Chainalysis) were engaged early. Counsel was retained in multiple jurisdictions. A dialogue channel with the attacker was opened through on-chain messages drafted with counsel.

The negotiation phase took twenty-three days. The team published calm, professional on-chain messages, offered a bounty, and noted that law-enforcement pressure was being applied through proper channels. No threats, no public shaming, no escalating ultimatums. The tone was patient and procedural. By 4 April 2023 the attacker had returned 100% of the funds. Users were made whole. Euler published a detailed post-mortem and rebuilt.

What made it work was preparation. The war-room structure existed before 13 March. Counsel was already retained. Chain-analysis relationships were warm. The communications lead had drafted templates. The team had run scenarios. The playbook existed before it was needed.

The contrasting case is Nomad, August 2022, $190 million lost. An initialisation bug in a contract upgrade allowed message verification to be bypassed. After the first attacker drained funds, the exploit was openly copyable: hundreds of subsequent transactions by dozens of independent actors drained the bridge in a public free-for-all, because there was no pause path drilled and no coordinated response in place.

> **NOTE**
> Euler and Nomad had bugs of comparable severity. The difference in outcome was three orders of magnitude. The difference in preparation is why.

9. Post-Mortem and Disclosure

Publish the post-mortem within thirty days. Late or evasive post-mortems signal to the ecosystem that the team is hiding something, even when it is not. The post-mortem is blameless (it focuses on systems and processes, not individuals), technical, honest, and structured for utility to other teams as well as your own users.

Section	Content
Timeline	Timestamped events from first signal to final state.
Root cause	The technical mechanism of the bug, in enough detail that an engineer learns something.
Contributing factors	Process and human dimensions: what design decisions, review gaps, and alerting gaps let the bug survive prevention.
What worked	The controls, decisions, and people that made the response succeed.
What did not work	Honest and specific.
Changes made and planned	A concrete list, with owners and target dates.

Be specific about the fix. ”We added more tests” is not a fix. ”We added invariant X monitored by Forta with alerts paging the on-call rota” is a fix. Generic post-mortems are unread post-mortems.

Credit the bug discovery fairly. If a whitehat reported it, the advertised bounty is paid, publicly. If law enforcement helped recover funds, the agency is thanked publicly. Cheapness on disclosure poisons future incidents: whitehats who are not paid the first time do not report the next time. A post-mortem that does not update the runbook, the monitors, and the next drill is a document, not a lesson.

10. Recovery: Redeploy, Rotate, Reimburse

The operational unwind. Three principles guide it.

- **Re-audit before relaunch.** The pressure to ship the fix is intense. The same pressure produced the original bug. A bug-fix audit on a hot patch is the floor; a full re-audit on the next deployable commit is the ceiling.
- **Rotate everything the incident touched.** Deployer keys, multisig seats, operational access tokens, cloud IAM credentials, CI tokens, npm tokens, VPN certificates, SSH keys. Anything live during the incident is rotated, on the assumption it may have been observed by the attacker or by someone downstream.
- **Reimburse transparently.** A protocol that makes users whole keeps its users. One that does not, even for a defensible reason such as a bounty-to-attacker arrangement, rarely recovers brand. Outline reimbursement mechanics before the incident; adapt them to specifics during recovery.

Stage the relaunch: governance gate, deposit caps on restart, monitoring dialled up for the first weeks. The community will wait if the team explains why. Teams lose users on relaunch when they try to skip the staging. Document the full recovery, not just the fix.

11. The IR Plan: Structure, Severity, Decision Rights

The plan is the document that makes everything above possible. Clear, executable, authoritative. Nobody chases a VP for approval at 03:00. The plan pre-approves the actions at each severity level.

Severity	Trigger	Posture
Level 1	Vulnerability discovered, not exploited.	Planning mode. Scoped fix, deliberate communication.
Level 2	Exploit ongoing or funds at immediate risk.	All-hands. War room stood up, pause activated.
Level 3	Exploit underway and funds being moved.	Pause now, ask later. Drain mechanism considered, counsel notified within minutes.

Use named roles, not named people. The names behind each role rotate with vacations and turnover; the role is the durable unit.

Role	Owns
Incident Commander	Coordination, decision rights, who is on the response.
Tech Response Lead	Technical analysis and containment execution.
Communications Lead	Holding statements, spokesperson coordination, external messaging.
Legal Counsel	Evidentiary frame, jurisdiction, attacker negotiation.
Law-Enforcement Liaison	Agency reporting and chain-analysis coordination.
Signer Coordinator	Multisig pause, drain, and key-rotation execution.

The decision ladder, for each severity, documents who decides pause, who decides public communications, who decides negotiation, who decides reimbursement, who has unilateral authority in the first 30 minutes, and who must be looped in within an hour. Written down before the incident, it eliminates the worst failure mode: the right call delayed because no one was sure they were authorised to make it.

12. Drills, Red Team, and War-Games

> **NOTE**
> The plan that is not drilled is the plan that does not work.

- **War-games.** Realistic scenarios (flash-loan exploit, signer device seized, key leak on GitHub, oracle manipulation, governance attack) run on wall-clock execution, with a post-drill review and a list of runbook updates. Drill the technical response, the communications response, the legal-engagement path, and the war-room logistics. All four are equally likely to be the surprise that costs the first hour.
- **Red team, annually, external.** Not just code: full opsec, including social engineering, phishing, recruiter pretexts, signer-device compromise. The findings feed the next round of prevention.
- **Test the pause path.** When did a human last press the pause button on a test network with a stopwatch running? If the answer is "we have not," the pause path is a hypothesis, not a control.
- **Drill the comms path.** Write the holding statement, the user email, and the exchange-notification template now; test retrieving, editing, and sending them under stress in the drill.
- **Close the loop.** Every drill produces new runbook items. The plan is a living document. The cadence (quarterly drill, annual external red-team, post-mortem feedback into the runbook within 30 days) is what keeps it honest. A plan untouched in a year no longer matches reality.

13. IR Readiness Checklist

Check	Ready
Prevention layers stacked: design-time, development-time, release-time, with re-audit on every material upgrade.	
Invariant monitors encoded and paging the on-call rota.	
On-chain anomaly alerts and simulation monitoring in place.	
Rate limits, circuit breakers, and withdrawal cooldowns shipped.	
24/7 on-call rota with primary and backup, tested quarterly.	
Scoped, multisig-gated pause with timelocked unpause, documented and drilled.	
Upgradeability (if any) timelocked, diffs published, re-audited.	
SEAL Safe Harbor adopted and published before any whitehat front-run is contemplated.	
First-60-minutes runbook written and rehearsed; out-of-band Signal war-room pre-built.	
Counsel retained across relevant jurisdictions; law-enforcement contacts warm.	
Chain-analysis relationship established before an incident.	
Communications templates (holding statement, user email, exchange notification, press list) pre-written.	
Single named spokesperson role assigned in the runbook.	
Severity classification, named roles, and decision	

ladder documented and pre-approved.	
Post-mortem structure and 30-day publication commitment agreed.	
Recovery plan covers re-audit, rotate-everything-touched, transparent reimbursement, staged relaunch.	
Quarterly war-games and annual external red-team scheduled; pause path drilled with a stopwatch.	

14. References

- NIST SP 800-61 (Computer Security Incident Handling Guide): https://csrc.nist.gov/pubs/sp/800/61/r3/final
- SEAL Safe Harbor framework: https://securityalliance.org/safe-harbor
- Security Alliance (SEAL) frameworks: https://frameworks.securityalliance.org
- Euler Finance incident post-mortem: https://www.euler.finance/blog/euler-exploit-post-mortem
- Forta detection bots: https://docs.forta.network/
- Tenderly alerts and simulation: https://docs.tenderly.co/
- FBI Internet Crime Complaint Center (IC3): https://www.ic3.gov/