PRODUCTION INCIDENT RESPONSE RUNBOOK

The 3AM Runbook

What to have in place before your trading or fintech system fails in production - severity classification, escalation, monitoring, kill switches, and the playbooks for the night everything goes wrong.

Modulus Global, Inc. • Scottsdale, Arizona

24/7 Production Support: +1 (480) 525-7940 • modulusglobal.com

Contents

1. Introduction

This runbook describes how to prepare for, respond to, and learn from production incidents in live trading and fintech systems: exchanges, matching engines, trading platforms, payment systems, treasury and liquidity infrastructure, and any other software where a failure at the wrong moment moves real money in the wrong direction.

It is written for the operator of such a system — the founder, CTO, lead developer, or the person who will be woken up when it breaks. It assumes your system is live, or about to be. It does not assume you have a dedicated operations team. Most of the firms this document is written for do not.

Production incidents do not schedule themselves during business hours. They cluster at the worst possible times: Sunday nights, holidays, the first minutes of a volatility spike, the hour after a deploy. The difference between an incident that costs you an hour and one that costs you your business is rarely the failure itself. It is whether someone knew, in advance, exactly what to do about it.

How to use this document. A runbook that is read once and filed away is worthless. This one is designed to be completed: fill in the directories, assign the roles, test the kill switch, run the drill. Sections marked as templates are meant to be printed, filled in, and kept somewhere you can reach from your phone at 3AM without waking anyone up.

> **THE ONE RULE**
>
> If a procedure in this runbook has never been tested, assume it does not work. An untested backup is a rumor. An untested kill switch is a decoration. An untested phone tree is a list of people who will not answer.

2. The 3AM Problem

Modern development — especially AI-accelerated development — produces working systems faster than at any time in the history of software. A trading platform that once took a team eighteen months can now be built in weeks. This is genuinely remarkable, and this document is not an argument against it.

It is an observation about what speed does not change: **production is where code meets reality**, and reality arrives with conditions no one specified. Order flow spikes fifty times baseline. A venue sends malformed data. A clock drifts. A certificate expires on a holiday. A retry loop that looked reasonable in code review becomes a cascade under load. These conditions do not appear in development, in staging, or in the prompt that generated the code.

Systems built quickly, by small teams, with heavy AI assistance tend to share a recognizable profile:

- **Untested edge paths.** The happy path works beautifully. The failure paths exist but have never executed once against real conditions.
- **Error handling that looks right.** Exceptions are caught, logged, and swallowed. The system appears healthy while quietly doing the wrong thing.

- **No single person understands the whole system.** Everyone understands the parts they prompted for. Nobody has ever traced a full order lifecycle under failure conditions.
- **Missing operational machinery.** No reconciliation, no runbooks, no rehearsed rollback — because nothing about building the system required them.
- **Fragile deploy culture.** Changes ship directly to production because they always worked before.

None of this is a character flaw. It is what "built fast" means, everywhere, for everyone. But it produces a specific situation that every operator of an AI-built system eventually recognizes: **there is a system running the business that nobody in the building fully wants to support at 3AM.**

You cannot prevent every failure. You can decide, today and in daylight, how the next one goes. That decision is this document.

3. Emergency Contact & Coverage

Keep this section completed and within reach of every person on the on-call rota. During an incident is the wrong time to search for a phone number.

Table 1 Internal emergency contacts (complete and print)

Role	Name	Phone	Backup contact
Primary on-call	_____	_____	_____
Secondary on-call	_____	_____	_____
Manager / executive	_____	_____	_____
Incident bridge URL/number	_____	_____	_____

Table 2 Modulus production support (external escalation)

Channel	Detail
24/7 incident line	+1 (480) 525-7940
Web	modulusglobal.com/production-support
Coverage	24/7 including holidays (24/7 plan); business days 9am–5pm MST (Business Hours plan)

Table 3 Modulus response targets by severity and plan

Severity	24/7 Plan	Business Hours Plan
P1 — production down, no workaround	Immediate response, around the clock	Immediate response, business days
P2 — materially degraded, workaround exists	1 hour	2 business hours
P3 — everything else	3 hours	6 business hours

Modulus support is advisory: our engineers diagnose, find root causes, and tell you exactly what must be done. Your systems stay in your hands and your credentials stay yours — where access is needed, it is least-privilege, named, revocable, and every change is executed by you or expressly authorized by you.

4. Severity Classification

Every incident gets a severity. The severity decides who gets woken up, how fast the response must be, and whether clients are notified. Classify immediately, on incomplete information, and re-classify as you learn. When in doubt between two levels, take the higher one — downgrading an over-response costs an hour of sleep; upgrading a late response can cost the business.

Table 4 Severity definitions with fintech examples

Level	Definition	Examples in trading & fintech systems
P1	Production down or money moving incorrectly. No workaround. Client funds, balances, or order integrity at risk.	Matching engine halted during market hours; balances or positions wrong; duplicate or phantom executions; orders accepted but never routed; reconciliation break involving client funds; trading halted by the venue on your error; suspected unauthorized access with fund movement capability.
P2	Materially degraded, but a workaround exists and money is not moving incorrectly.	Elevated order reject rates; market data delayed or stale on one feed with a second feed healthy; one venue or symbol set down while others operate; elevated latency; client-facing UI down while trading API functions.
P3	Everything else: issues with no immediate client or funds impact.	Reporting failures; cosmetic defects; a non-critical internal tool down; a backup job failed once with a successful retry.

> **Declaration authority.** Anyone on the on-call rota may declare a P1. You do not need permission to treat an incident as serious. The incident commander (Section 5) may re-classify once scope is understood. A false P1 is a drill; a missed P1 is a disaster.

5. Roles & the Escalation Chain

Incidents fail in two ways: nobody is clearly in charge, or the wrong people are doing the wrong jobs. Assign roles at the start of every P1 and P2, even if some roles are held by the same person.

Table 5 Incident roles

Role	Responsibility
Incident Commander (IC)	Owns the incident. Decides severity, approves every change to production, decides on halt/rollback, and declares resolution. One person. Not a committee.
Technical Lead(s)	Execute diagnosis and changes under the IC’s direction. Hands on keyboards.
Communications Lead	Owns internal updates and client notifications (Section 14). Keeps the IC and engineers free of “any update?” interruptions.
Scribe	Maintains the incident timeline: every observation, decision, and action with a timestamp. The timeline is the raw material of the post-incident review (Section 15) and, in regulated environments, a compliance artifact.

5.1 The on-call chain and the ten-minute rule

Alerts page the primary on-call by a channel that actually wakes a human — a phone call or pager app, not email and not a chat message. If the primary does not acknowledge within ten minutes, the alert escalates to the secondary, then to the manager. The chain must be short, tested, and documented:

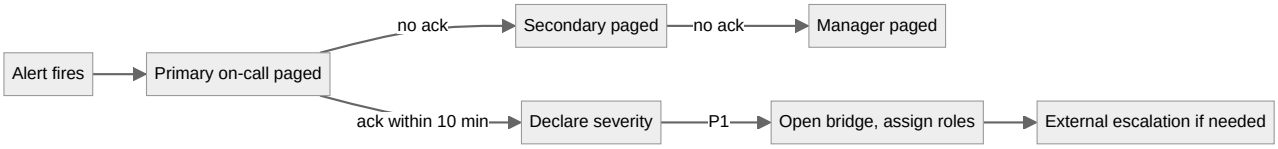

Figure 1 Escalation chain: alert to response

- **Two people minimum on any P1 bridge.** No solo heroics on systems that move money. The second person catches the mistake the first person is about to make.
- **Rotate the rota.** A permanent on-call of one person is a single point of failure with a phone.
- **Test the chain monthly.** Fire a test page at an inconvenient hour. If it does not wake someone, fix it before the night it has to.

6. Access Readiness: The Break-Glass Checklist

At 3AM, access failures compound technical failures. The person on call cannot reach the database because the MFA device belongs to a co-founder who is asleep in another time zone. The cloud console session expired. The venue portal password lives in someone’s personal password manager. Every one of these has turned a forty-minute incident into a four-hour one.

If access depends on one specific person being awake, you do not have access. Complete and test every row below.

Table 6 Break-glass access checklist (test each item quarterly)

Done	Item	Standard
[ ]	Cloud console break-glass account	Stored in a sealed, shared location; MFA not tied to one person’s phone; tested monthly
[ ]	VPN / network access	Works from home and mobile hotspot; credentials on the rota
[ ]	Database credentials	Read-only and admin tiers documented; reachable by on-call
[ ]	Venue / exchange / custodian portals	Logins for every venue, with emergency contact numbers for each
[ ]	DNS registrar & CDN	Account access documented; failover procedures rehearsed
[ ]	Monitoring dashboards & logs	On-call can reach dashboards, log search, and alerting config
[ ]	Deployment & rollback tooling	On-call can deploy and roll back without a developer present
[ ]	Offline copy of this runbook	Printed or saved locally; not dependent on the systems it describes

[ ]	Certificate & key inventory	Expiry dates monitored; renewal owner named for each
[ ]	Secrets storage	On-call can retrieve API keys and service credentials without waking a founder

7. Golden Rules Before You Touch Anything

Most production damage during incidents is self-inflicted: well-meaning changes made in the first minutes that destroy evidence, corrupt state, or make the original problem worse. These rules exist to slow your hands down.

> **1. Capture state first.** Before any change: snapshot logs, dashboards, error traces, database state, and the exact symptoms. Evidence destroyed at 3:05AM cannot be recovered at 9AM when you need to know what actually happened.

> **2. Do not restart blindly.** A restart may clear the evidence, lose in-flight state, and — in money systems — turn a stall into duplicated or lost work. Restart only when you know what state the restart preserves and what it discards.

> **3. One change at a time.** Change one thing, observe, then decide. Two simultaneous changes leave you unable to say which one helped — or which one hurt.

> **4. Log every action with a timestamp.** The scribe records who did what and when. In a money system this is not bureaucracy; it is your audit trail.

> **5. Announce before executing.** Every action on production is announced on the bridge before it runs. No silent changes, ever.

> **6. Know the rollback before the change.** If you cannot state how to undo what you are about to do, you are not ready to do it.

> **7. In money systems, halt beats wrong.** Stopping trading is recoverable — the market will be there tomorrow. Incorrect balances, duplicate executions, or mispriced fills may not be. When the choice is between downtime and wrongness, choose downtime.

8. Minimum Monitoring & Alerting Baseline

You cannot respond to what you cannot see. The signals below are the minimum for any system that moves money. Each needs a threshold, an owner, and a paging rule. Two principles govern the whole table:

- **Alerts page humans only when a human must act now.** Everything else goes to a dashboard or a daily digest. A pager that cries wolf gets silenced — and then the one real page goes unheard.

- **Alert on user-visible symptoms, not just machine metrics.** High CPU that harms no one is noise. Rising order rejects is a symptom.

Table 7 Minimum alerting baseline for trading & fintech systems

Signal	Example paging condition	Why it matters
Order reject rate	> 2× baseline for 5 minutes	First visible sign of matching, risk, or venue connectivity trouble
Order / execution latency (p99)	> 3× baseline for 5 minutes	Degradation before failure; staleness against the market
Order throughput vs. baseline	< 20% of expected for the session/time	A silent halt looks exactly like low volume
Market data staleness	No tick for N seconds during market hours	Trading on stale prices manufactures losses
Sequence gaps on feeds	Any unrecovered gap	Missed messages mean wrong books, wrong prices, wrong risk
WebSocket / API disconnect rate	> threshold per minute sustained	Clients silently losing their live view of the market
Reconciliation break	Any break past daily tolerance	The ledger and reality have diverged — find out which is wrong
Risk / position limit breach	Any breach	Exposure beyond what the firm decided to carry
Database replication lag	> N seconds or growing	Your failover copy is only as good as its lag
Queue depth growth	Monotonic growth over 15 minutes	A consumer is stalled; backlog becomes data loss or delay
Disk / memory headroom	< 15% free on critical hosts	The cheapest failure to prevent, the most embarrassing to suffer
Failed login / API auth spikes	> 3× baseline	Attack or misconfigured client — both need eyes
Deployment events	Annotate all dashboards	Half of incidents follow a change; know when changes happened
Backup job success	Any failure, plus weekly restore test result	A failed backup that never pages is a future P1
Certificate / domain expiry	Page at 30, 14, and 3 days out	Expiry outages are 100% preventable and always public

9. Daily Reconciliation Discipline

Reconciliation is how you know your system's view of the world matches reality: the venue's ledger, the custodian's balances, the bank's statement. Systems that skip reconciliation do not avoid discrepancies — they merely postpone discovering them, usually at maximum cost and in front of clients or regulators.

- **Automate it daily, including weekends and holidays.** Markets may rest; drift does not. The small break that appears on Friday is a large one by Tuesday.
- **Compare every source of truth.** Internal ledger vs. venue statements, custodian balances, bank movements, trade counts, fees, positions, and open-order state.
- **Define tolerances and page on breach.** Zero tolerance for client-fund discrepancies. Documented, small tolerances elsewhere — with every break investigated same-day.
- **Breaks are P2 by default, P1 if client funds are involved.** A reconciliation break means one of your two sources of truth is wrong. Until you know which, assume it is you.
- **Keep break history.** Recurring small breaks in the same place are a latent defect announcing itself politely, in advance, for free.

> **The silent killer.** The worst production failures in financial systems are not crashes — crashes are loud and get fixed. They are slow divergences: balances drifting a few units a day, fees miscalculated for a quarter, positions mislabeled since the last deploy. Reconciliation is the only machinery that catches them.

10. The Kill Switch: Graceful Halt Procedures

Every trading system needs the ability to stop, on purpose, quickly, and in a controlled way. A graceful halt you choose beats an ungraceful one the market chooses for you — and it is the correct answer to more incidents than any engineer's pride wants to admit. Define these levels now, implement them now, and test them quarterly.

Table 8 Halt levels

Level	Effect	Authority to invoke
Cancel-only mode	New orders rejected; cancels and amendments accepted; clients can flatten risk	Any on-call engineer, announced on bridge
Full halt	All open orders cancelled at venues; all order flow blocked; state preserved for review	Incident Commander
Disconnect	Venue connectivity severed (know what each venue does with your resting orders on disconnect — some cancel, some do not)	Incident Commander, with executive notification

- **Write the exact commands or console paths for each level** and keep them in this runbook, not in someone's memory.

- **Test every level quarterly in staging**, and test cancel-only in production at a quiet time at least once. The first execution of a halt procedure must not be during the incident that needs it.
- **Rehearse the restart as well.** Know how the system re-enters the market cleanly: order state reload, sequence resumption, client notification.
- **Decide now what halting means for clients.** What they see, what they are told, and who tells them (Section 14).

11. The First Fifteen Minutes of a P1

The opening minutes of a P1 set its total cost. Follow the sequence; do not improvise it.

Table 9 P1 first-fifteen-minutes sequence

Window	Actions
0–2 min	Acknowledge the page. Declare P1. Wake the second person — no solo work on money systems.
2–5 min	Open the incident bridge. Assign Incident Commander and scribe. Start the timeline: first entry is the observed symptoms and the time they began.
5–8 min	Capture state before touching anything: logs, dashboards, error traces, recent deploys, database snapshot where feasible (Golden Rule 1).
8–12 min	Assess scope with one question first: *is money moving incorrectly right now?* If yes or unknown — invoke the kill switch (Section 10). Halt beats wrong.
12–15 min	Stabilize. Send the first stakeholder communication: we are aware, we are investigating, next update at a stated time (Section 14). Then work the problem: one hypothesis, one change, observe, repeat (Golden Rule 3).

After stabilization, the incident shifts from firefighting to diagnosis. Keep the timeline going. Keep the update cadence. Escalate externally (Section 13) the moment the problem is beyond the room — an hour of pride is an expensive thing to bill your clients for.

12. Incident Playbooks

Each playbook gives symptoms, first checks, immediate actions, and the trigger for external escalation. They assume the Golden Rules: state captured first, one change at a time, everything logged.

12.1 Matching Engine Stalled or Degraded

Element	Procedure
Symptoms	Order accepts without responses; rejects spiking; execution latency climbing; throughput collapsing against normal volume.
First checks	Recent deploy? Queue depth growing? Disk or memory exhaustion? Downstream dependency (market data, risk, database) stalled and back-pressuring the engine?

Immediate actions	Invoke cancel-only mode. Capture engine and queue state. If money is moving incorrectly or state is unknown: full halt. Fail over to the standby engine only if its state is verified current — failing over to a stale standby manufactures duplicates.
Escalate when	Not stabilized within 30 minutes; state integrity uncertain; or any doubt about order state on restart. This is the canonical 3AM call: +1 (480) 525-7940.

12.2 Market Data Anomaly (Stale or Bad Ticks)

Element	Procedure
Symptoms	No ticks for N seconds; prices frozen or implausible; sequence gaps; venue reports “no issue” while your feed disagrees.
First checks	One feed or all? Sequence gap recoverable by re-request/replay? Cross-check against a second venue or public reference price.
Immediate actions	Stop consuming the bad feed for pricing and risk. Switch to the healthy feed or widen/halt affected symbols. **Never trade on prices you do not trust.** If executions occurred on bad data, quantify them before resuming.
Escalate when	Executions occurred on corrupted data; venue disputes your evidence; or the corruption source is unclear.

12.3 WebSocket / API Disconnect Storm

Element	Procedure
Symptoms	Mass client disconnections; reconnect floods; connection counts oscillating; clients reporting blank or frozen screens.
First checks	Recent deploy? Load balancer or proxy change? Certificate expiry? A single abusive client reconnect-looping? Rate limiter misfiring?
Immediate actions	Cap reconnect rates (backoff enforcement) to stop the self-inflicted DDoS. Shed load by shedding connections cleanly, not by crashing. If one client is the source, isolate it. Communicate early — a silent outage breeds panic trading.
Escalate when	The storm is masking a deeper failure; client impact exceeds one hour; or the cause resists identification.

12.4 Database Failure or Replication Lag

Element	Procedure
Symptoms	Write failures; connections exhausted; replication lag growing; replica divergence errors.
First checks	Disk full? Connection leak from a recent deploy? Lag cause: long transaction, DDL, network? Is the replica actually caught up enough to trust?
Immediate actions	For the ledger of a money system: halt writes that cannot be verified rather than write them into an uncertain state. Fail over only with verified replication position; a promoted-but-stale replica is a silent balance corruption machine. Snapshot before any repair attempt.

Escalate when	Any doubt about ledger integrity; any need to reconcile writes that may have landed twice or not at all.

12.5 Reconciliation Break

Element	Procedure
Symptoms	Daily recon reports a discrepancy beyond tolerance: balances, positions, trade counts, or fees disagree between your ledger and an external source.
First checks	Timing artifact (unsettled, in-flight)? Timezone or cut-off bug? New fee schedule? A specific account, symbol, or time window where the break concentrates?
Immediate actions	If client funds are involved, treat as P1: freeze affected flows until you know which side is wrong. Isolate the break to the smallest set of accounts and transactions. Never “adjust” the ledger to match without a documented root cause.
Escalate when	Root cause not found same-day; break is growing; or the discrepancy is in your favor — those are the ones clients and regulators ask about later.

12.6 Failed Deployment and Rollback

Element	Procedure
Symptoms	Error rates, latency, or behavioral changes beginning at or shortly after a deploy.
First checks	Confirm correlation with the deploy timestamp (dashboards should annotate deploys). Is the failure in the new code or in a migration it ran? Is the database still compatible with the old version?
Immediate actions	Roll back — quickly, and without debate about whose change it was — if the rollback is safe (Golden Rule 6). If a database migration makes rollback unsafe: halt the affected flow, and do not improvise schema surgery on a live money system at 3AM.
Escalate when	Rollback is unsafe or has already caused state questions; any data written by the bad version needs audit.

12.7 Cloud Region Outage and Failover

Element	Procedure
Symptoms	Provider status page confirms regional degradation; your health checks failing across an entire region; network partitions.
First checks	Is this truly regional or a provider-wide identity/DNS event? What is the verified state of your secondary region? When did it last replicate, and when did you last *test* failover?
Immediate actions	Execute the documented failover runbook — this is why it exists and why it was drilled. Failover to an unverified secondary trades a known outage for an unknown state. Decide the acceptable data-loss window (RPO) in advance, in writing, so no one improvises it under pressure.
Escalate when	Failover involves money-state uncertainty; DNS/traffic cutover stalls; or the secondary reveals itself untested (it will, the first time).

12.8 Credential Compromise / Security Event

Element	Procedure
Symptoms	API keys used from unknown locations; unexpected withdrawals or transfers; auth log anomalies; a key committed to a repository; a report from a third party.
First checks	Which credentials, what can they do, from where, since when? Preserve logs before rotation wipes sessions.
Immediate actions	Revoke and rotate the compromised credentials immediately — this is the one change you make before full diagnosis. Freeze fund movement if withdrawal-capable credentials are involved. Preserve all evidence. Engage legal counsel on notification obligations; financial systems carry them in most jurisdictions.
Escalate when	Immediately. Security events in money systems are P1 by definition and are the one category where outside help is never optional.

13. External Escalation Directory

During an incident you will need other people: your cloud provider, your venues, your data vendors. Find these numbers now, verify they reach a human, and print this page. **If you complete nothing else in this document, complete this one.**

Table 10 External escalation directory (complete, verify quarterly, print)

Provider	Account / customer ID	24×7 support contact	Named contact / TAM
Cloud provider	_____	_____	_____
Exchange / venue 1	_____	_____	_____
Exchange / venue 2	_____	_____	_____
Market data vendor	_____	_____	_____
Custodian / bank	_____	_____	_____
DNS registrar / CDN	_____	_____	_____
ISP / network provider	_____	_____	_____
Legal counsel	_____	_____	_____
Modulus 24/7 production support	—	**+1 (480) 525-7940**	modulusglobal.com

14. Communication Templates

In an incident, silence is the message clients remember. Communicate early, state what you know and what you do not, commit to the next update time — and never miss a committed update time, even if the update is “no

change.”

INTERNAL UPDATE (every 30 minutes during a P1)

[Time] — [Severity] — [System]
Status: Investigating / Identified / Mitigating / Monitoring / Resolved.
Impact: Who and what is affected. Is trading halted? Are funds at risk?
Actions since last update: One line each.
Next update: [exact time]. IC: [name].

CLIENT NOTICE — INCIDENT OPEN

“We are currently investigating an issue affecting [system/function]. [New orders are temporarily paused / some clients may experience X]. Client funds and balances are [unaffected / being verified]. Our team is actively working on this, and we will provide the next update by [time]. We apologize for the disruption.”

CLIENT NOTICE — RESOLUTION

“The issue affecting [system/function] was resolved at [time]. Service has been fully restored. [If applicable: all balances and positions have been verified and are correct.] The cause was [plain-language description]. A full post-incident review is underway; we will share its findings and the preventive measures we are taking by [date]. Thank you for your patience.”

15. Post-Incident Review

Every P1 and P2 gets a written review within 48–72 hours, while memory is fresh. In regulated environments this document may be requested by clients, auditors, or regulators — write it as if they will read it, because they may.

1. **Timeline.** Reconstruct from the scribe’s log: detection, declaration, every action, every decision, resolution. Include what people believed at each point, not just what was true.
2. **Root cause.** Ask “why” until the answer is a process or system property, not a person. “Engineer made a typo” is never the root cause; “no staging environment validates this class of change” is.
3. **Contributing factors.** The deploy without a rollback, the alert that fired but didn’t page, the undocumented dependency. Incidents are rarely single-cause.
4. **What went well.** Name it and keep it. The halt that worked, the recon break caught early.
5. **Action items.** Each with an owner and a date. Track them to completion. The review that changes nothing is a eulogy.
6. **Blameless rule.** Blame systems, not people. A team that fears blame hides incidents, delays escalation, and learns nothing. You are buying information with every incident; do not also pay with your engineers’ candor.

16. Production Readiness Self-Assessment

Score each item honestly: 1 if true and tested, 0 otherwise. Do this today, in daylight.

Table 11 Readiness scorecard

Score	Item
[ ]	Severity levels defined in writing; anyone on call can declare a P1
[ ]	On-call rota of at least two people, with a tested 10-minute escalation chain
[ ]	Paging channel that actually wakes a human — tested at an inconvenient hour
[ ]	Break-glass access checklist (Table 6) complete and tested within the last quarter
[ ]	Minimum monitoring baseline (Table 7) in place, with paging thresholds set
[ ]	Order reject rate, latency, and throughput alert against baselines
[ ]	Market data staleness and sequence-gap alerting in place
[ ]	Daily automated reconciliation runs seven days a week and pages on breaks
[ ]	Kill switch levels defined (cancel-only, full halt, disconnect) with exact procedures written
[ ]	Kill switch tested within the last quarter — including the restart procedure
[ ]	Rollback tested for the last three production deploys
[ ]	Backups verified by an actual restore within the last month
[ ]	Failover to secondary region/environment drilled within the last six months
[ ]	External escalation directory (Table 10) complete, verified, and printed
[ ]	Client communication templates agreed in advance; spokesperson named
[ ]	Every P1/P2 in the last year has a written post-incident review with closed action items
[ ]	Certificate and domain expiries monitored with advance paging
[ ]	Deployment events annotated on monitoring dashboards
[ ]	Someone besides the founder can execute every procedure in this document
[ ]	This runbook exists offline, current, and reachable at 3AM

Table 12 Scoring

Score	Reading
18–20	**Ready.** Incidents will still happen; they will cost you hours, not sleep and clients.
14–17	**Exposed.** You have partial machinery. Your next P1 will find the gaps before you do.
Below 14	**You have a 3AM problem.** Not someday — on a schedule you do not control. Close the gaps or put someone on call who has done this before: +1 (480) 525-7940.

17. Conclusion

Nobody plans to have a 3AM incident. The firms that survive them are not the ones with the fewest failures — they are the ones that decided, in daylight, what to do in the dark: who gets woken up, what gets captured first, when to halt, who to call, and what to say.

Complete this document. Test the parts that claim to work. Print the directories. If your score on the self-assessment is not where it should be, you now know exactly which rows to fix — and that is the entire value of a runbook.

And if the night comes anyway and the problem is bigger than the room: this is the call we have been answering since 1997, for exchanges, trading platforms, and financial infrastructure in more than 100 countries. Modulus engineers diagnose, find root causes, and tell you exactly what must be done — while your systems stay in your hands and your credentials stay yours.

MODULUS 24/7 PRODUCTION SUPPORT

Instant P1 response, around the clock, month to month, cancel anytime.
+1 (480) 525-7940 • modulusglobal.com/production-support • Scottsdale, Arizona

18. Disclaimer