ON AUTOMATED RESPONSE, BLAST RADIUS AND THE SECOND HUMAN

Delegation is a design problem, not a setting

Teams do not decline automated response because it is slow to configure. They decline it because a wrong action at machine speed is an outage they caused themselves, and nothing in the product told them where the edges were.

IN BRIEF

Most platforms ship automated response and most estates run it in report-only mode. The reason is rational: the operator cannot predict what an action will touch, cannot undo some of them, and cannot approve one when the incident has taken the approval channel with it. This paper treats those as three engineering requirements — declared blast radius, a reversibility classification the action carries rather than the operator sets, and out-of-band authorisation — plus the case where the correct response on a laptop is the wrong response on a controller. The claim is narrow and, we think, load-bearing: autonomy is adopted in proportion to how well its limits are expressed.

1 · Report-only is a verdict

Ask a security team why the response playbooks are enabled but not armed and the answer is rarely about the detection quality. It is some version of: we do not know what it will do. Not what it is supposed to do — the description is clear enough — but what it will actually touch on the night it fires, given an estate whose ownership records are eighteen months stale.

This is a design failure being read as a trust failure. The team is behaving correctly: asked to authorise an action of unstated scope with unstated reversibility, refusing is the only defensible position. Every product feature that follows from "make the automation smarter" misses it, because the obstacle is not the quality of the decision but **the illegibility of its consequences**.

The consequence is measurable and rarely measured: the mean time to contain that appears in a board pack is the time taken by a human who was paged, opened three consoles and typed a command — while the platform that could have done it in four seconds watched and wrote a record. The gap between those

two numbers is the entire value of response automation, and it is being declined for reasons the vendor could have removed.

FIGURE 1 · WHAT THE OPERATOR IS ACTUALLY BEING ASKED

WHAT IS OFFERED

"Isolate host" · confidence 94% · enable automatic execution [toggle]

WHAT IS NEEDED

This action will sever 1 host, 3 active sessions and 1 replication link. Reversible in ~8s. Owner: platform-eng. Two similar actions ran last month; both were reversed within an hour.

A confidence percentage answers a question nobody asked. The operator is not adjudicating the detection; they are estimating the cost of being wrong, and the platform holds every fact that estimate requires.

2 · Blast radius is computed, not documented

Every automated action should be able to state, before it runs, what it will affect: which hosts, which sessions, which identities, which rules, and what depends on them. Not a description written by the playbook's author — an evaluation against the estate as it is at the moment of execution.

The distinction matters because authored descriptions age. "Isolates one workstation" was true when the playbook was written and false after the workstation became a jump host for a payments batch. A computed radius reads the current graph — the sessions terminating on that host, the services that answer from it, the replication that flows through it — and produces a number the operator can act on.

This has a strong corollary that is easy to state and unpopular to implement: **an action whose scope cannot be computed does not run unattended.** Not because it is dangerous, but because nobody can say whether it is. Where the graph lacks ownership or dependency data, the honest behaviour is to escalate rather than to assume the radius is small — and the resulting escalations become the most useful inventory-gap report an estate will ever get.

Scope is also a containment boundary. An action that can compute its radius can be refused for exceeding one. "No automated action may sever more than N sessions or touch more than one production tier" is a policy the system can enforce, and it is the difference between delegating a task and delegating unlimited authority to perform it.

3 · Reversibility belongs to the action

Blocking an address, quarantining a file and suspending a session are reversible: the estate returns to its prior state and the cost of a mistake is minutes of disruption. Revoking a certificate, deleting a workload, rotating a credential another system depends on, or wiping a device are not: the cost of a mistake is a recovery project.

These should not be governed by the same control. Reversible actions can be delegated broadly; irreversible ones require a second human — and, critically, **the classification must be a property of the action rather than a checkbox on it**. A setting that can be relaxed will be relaxed, at 03:00, by someone under pressure who intends to restore it and does not.

REVERSIBLE — DELEGABLE

Block an address · quarantine a file · suspend a session · rate-limit an identity · isolate a host at the network layer. Undo is a recorded action with its own latency, published and tested.

IRREVERSIBLE — SECOND HUMAN

Revoke a certificate · delete or re-image a workload · rotate a shared credential · wipe a device · disable a break-glass identity. The approval is part of the action's record, not a message beside it.

One nuance deserves stating, because glossing it would be dishonest: reversibility is about the estate, not about the incident. Suspending a session is reversible; the meeting it interrupted is not, and the operator who does it twice to the same executive will lose the argument for automation regardless of the technical classification. Blast radius and reversibility are separate axes, and a small irreversible action is often safer to delegate than a large reversible one.

4 · Approval must not depend on the estate

The requirement for a second human introduces a dependency that is usually left implicit: the approver has to be reachable. In an incident severe enough to warrant an irreversible action, the reachable channels are exactly the ones under contest — the identity provider, the chat platform, the ticketing system, the corporate mail flow.

An approval path that runs through the compromised estate is not a control. It is a control-shaped object that fails at the only moment it is needed, and it fails silently: the request goes out, nobody answers, and the action does not happen. The design requirement is an out-of-band channel with an independent trust root and its own enrolled devices — used often enough in ordinary operation that it works when it matters, rather than a break-glass envelope in a drawer.

FIGURE 2 · ONE ACTION, FOUR GATES

T+0.0s **credential theft on a build host, legitimacy unsupported** · no change record matches

T+0.4s **radius computed** · 1 host · 3 sessions · 0 production tiers · within policy

T+1.1s **reversible actions execute unattended** · sessions suspended, host isolated, undo latency ~8s

T+3m **credential rotation held for a second human** · requested out-of-band; the shared credential has 6 dependants

The containment happened in a second; the consequential act waited for a person, and waited on a channel the incident could not reach. Splitting one response into a delegable part and an authorised part is what makes arming it possible at all.

5 · Where the right answer is not a security answer

In operational environments the calculus inverts. Isolating a compromised engineering workstation is prudent; isolating a controller mid-sequence can be a safety event, and the correct response may be to allow the compromise to continue under observation while a person with process authority decides.

A platform that treats every asset as an endpoint will eventually make that mistake, and the mistake is not recoverable by a rollback. The requirement is that the response engine reads a safety classification from the asset itself, and that in the classified zones it proposes rather than acts — with the proposal carrying the same computed radius, so the person deciding has the security case in the form they need it.

> **The unifying idea.** Bounded autonomy is not a weaker automation. It is the only kind that gets armed — and an armed narrow automation contains more incidents than a broad one running in report-only mode.

6 · Questions worth asking, whoever you are buying from

01 Before an action runs, can the platform tell me what it will touch — computed now, not described by its author?

02 What happens when the radius cannot be computed because ownership data is missing?

03 Is reversibility a property of the action or a setting an administrator can change?

04 What is the measured undo latency for each reversible action, and where is it published?

05 If our identity provider and chat platform are both compromised, how does an approval reach you?

06 Can I cap automated authority — no more than N sessions, never across a production boundary — and is the cap enforced?

07 How does the response engine know an asset is safety-critical, and what does it do differently there?

08 Six months on, can I retrieve why a specific automated action ran — with its radius, its basis and its approver?

7 · Where this sits in the series

Paper 01 argued that the sensor and the enforcement point must be one process. Paper 02 gave the action an explanation it could be authorised on. Paper 03 supplied the judgement that nobody meant to do this. This paper is the constraint that makes the three of them deployable: a bound on what the system may do with all of it.

Read together, they describe a specific trade. The platform is permitted to act without asking, in exchange for stating its scope, classifying its reversibility, escalating what it cannot bound, and recording every one of those decisions as an event in the same graph as the signal that prompted it. That is a contract an organisation can sign. A confidence percentage is not.

> *Nobody withholds authority from a system because it is too fast. They withhold it because they cannot describe what it will do.*

EventLake

ABOUT THIS PAPER

Paper 05 in a series on the architecture of active defense, published by **EventLake Research**. Papers 01–04 cover enforcement at the edge, explainable alerts, the legitimacy judgement, and compiling findings into enforced rules. The series is deliberately vendor-neutral, and the questions in §6 are meant to be asked of us as readily as of anyone else.

In EventLake, every automated action computes its blast radius before it runs, carries a reversibility classification it cannot be talked out of, escalates what it cannot bound, and records its basis and approver as events. eventlakes.com