A FIELD MANUAL

They Log In. They Don’t Break In.

Identity as the perimeter — human and machine, joiner-mover-leaver, privilege, and killing the password without bringing the building down.

Bhavin G Patel

C|CISO · CISSP-ISSAP · CISM

cybersecleader.com

Twelve runbooks · Nine templates · Built to be adapted, not admired

Why I wrote this

Pull the thread on almost any modern breach and you land on identity. A stolen credential. An over-privileged account. A service token nobody rotated. A former employee whose access never got turned off. The attacker rarely breaks in. They log in, with something that was supposed to belong to someone else, and they walk past defences built to spot intruders that were never taught to question a legitimate login.

None of the fixes are exotic. They're boring, and boring is exactly why they don't get done. So this manual is mostly plumbing: how to find every identity you have, how to stop them accumulating permissions like sediment, how to take access away the day it stops being needed, and how to retire the password without breaking a batch job at three in the morning.

Where the right answer depends on your environment rather than on principle, I say so. Appendix H collects the calls I don't think anyone can make for you.

Who it's for

Whoever owns identity, whether or not that's their title. A CISO, an IAM lead, a platform engineer who inherited the directory, or the person who just discovered how many service accounts have domain admin. Parts I and VI read at executive level. Parts II through V assume you're going to build this.

How to use it

If you have limited time, three things in this order. Run **ID-01** to find out how many identities you actually have, human and machine, because the machine number will surprise you. Run **ID-07** to find the non-human ones nobody owns. Then run **ID-04** against your last two years of leavers.

That third one is where I have found live accounts belonging to people who left years ago in almost every environment I have ever reviewed. It is not an edge case, and finding it yourself is considerably better than the alternative.

> **ON THE PASSWORDLESS PART**
>
> Part V is deliberately unfashionable. Anyone selling overnight passwordless is selling a headline. The version that works is a patient migration where, at every step, users are safer than they were the day before — MFA first so the password stops mattering, then retirement system by system. Slower than the slogan, and the one that actually finishes.

First published 2026 at cybersecleader.com. *They Log In. They Don't Break In.: A Field Manual for Identity, Human and Machine.*

Disclaimer

This manual is general information, not legal, regulatory or audit advice, and reading it does not create an advisory relationship between us. Identity changes touch authentication for every user and system you have; test in a non-production environment, stage your rollouts, and keep a rollback path. Obligations vary by jurisdiction, sector and size. I accept no liability for decisions taken on the basis of this document. Verify before you rely.

Contents

PART ONE

The Position

1 · The Wall Is Gone

For years we drew security as a wall. Inside, good. Outside, bad. Spend the budget on the perimeter, trust what's within it, and sleep at night.

Then the data moved to the cloud, everyone started working from their kitchen, and the wall quietly stopped existing. There is no inside any more. Your data lives in a dozen SaaS platforms, your people log in from everywhere, and the network you used to defend is one of many roads to the things that matter.

So what is left to defend? Identity. Who someone is, and what they're allowed to reach. That's the perimeter now, and a lot of teams are still funding a wall that isn't there.

What actually changes when you accept that

The reframe is not rhetorical. It moves budget and it moves ownership. Authentication strength becomes a board-level control rather than a helpdesk setting. The joiner-mover-leaver process stops being an HR workflow and becomes a security one. Service accounts stop being infrastructure trivia and become the largest untracked population in the environment. And the access review, that most despised of quarterly rituals, turns out to be the thing standing between one compromised account and all of them.

> **THE UNCOMFORTABLE VERSION**
>
> If identity is the perimeter, then your perimeter is currently defended by a directory nobody has fully enumerated, a set of service accounts nobody owns, and a leaver process that depends on a manager remembering to file a ticket. That's not a criticism of your team. It's where almost every organisation actually is.

2 · Every Breach Is an Identity Story

The attacker rarely breaks in. They log in, with something that was supposed to belong to someone else, through a door you left unlocked.

It is far less dramatic than the movies and far more common. No exotic exploit. Valid credentials in the wrong hands, walking past defences built to spot intruders and never taught to question a legitimate login. That is the whole problem in one sentence, and it explains why so much detection spend produces so little protection: the behaviour is authorised, the source is plausible, and nothing in the telemetry says intrusion.

WHAT WE BUILT FOR, AND WHAT ACTUALLY HAPPENS

WHAT THE DEFENCES EXPECT

Exploit against a service. Unfamiliar source. Malware on disk. Lateral movement that looks nothing like normal work.

WHAT THEY ACTUALLY DO

Sign in with a valid credential. Correct password. Sometimes an approved MFA push. Then behave like an employee with access.

WHERE THE CREDENTIAL CAME FROM

- Phishing or a look-alike sign-in page
- Reused password exposed in someone else's breach
- MFA fatigue: enough prompts until one gets approved
- A token or API key left in a repository
- A service account with a password set in 2019
- An account belonging to someone who left

> Six routes, none of them exotic. Note that the right-hand column is entirely machine and lifecycle failure, which is where the least attention goes.

3 · The Four Failures

When I assess an identity programme I look for the same four failures, because they're the ones that show up in the post-mortems.

Failure	What it looks like, and why it survives
Weak authentication	A stolen password alone is enough to get in. Survives because MFA coverage is reported as a percentage of users rather than a percentage of *authentication paths*, and the gaps are always the legacy protocol nobody counted.
Too much privilege	Accounts that reach far more than the job requires, so one compromise becomes total. Survives because granting is fast, revoking is slow, and nobody is measured on the difference.
Stale secrets	Tokens, keys and service-account passwords set once and never rotated, sitting in the environment like unattended keys. Survives because rotation risks breaking something and nobody knows what.
Broken deprovisioning	Access that outlives the need for it. Survives because it produces no symptom until the day it does.

Almost every identity-driven breach is one or more of those four. None of them are exotic. They're boring, and boring is exactly why they don't get fixed: none has a demo, none has a conference talk, and none makes a good slide.

The unglamorous fixes, in order of leverage

Strong authentication first, so a stolen password isn't game over. This is the single highest-leverage control available right now, and it's the one most attacks are explicitly designed to get around, which tells you exactly how much it matters.

Least privilege second, so a compromised account can't reach the whole estate. The difference between an incident and a catastrophe is usually just how far the stolen account could go.

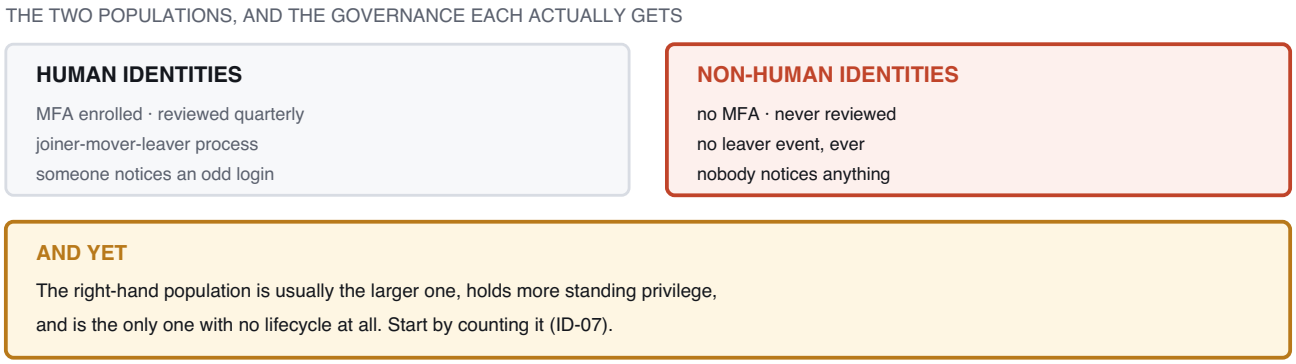
Deprovisioning third, and it's the one everyone skips. I have walked into companies with active accounts for people who left years ago. Live logins, real access, attached to nobody. That is not an edge case. It is the front door left open with a welcome mat, and it is in almost every environment I have ever reviewed.

4 · The Machines Outnumber the Humans

Here’s the part that catches teams off guard: most of the identities in your environment aren’t people.

Service accounts, API keys, tokens, automated processes. They usually outnumber human users by a wide margin, often by an order of magnitude. Nobody rotates them, nobody reviews them, and they frequently hold far more standing access than any human would be allowed, because it was easier that way when someone set up the integration at 2 a.m. under a deadline.

Attackers know this. A leaked API key with broad permissions is often a cleaner path in than any human account: no MFA, nobody watching it, and no expectation that it will ever sign in from somewhere strange. Machine identity is the quiet frontier, and it is where a lot of the next decade’s breaches are going to come from.



> Most identity programmes govern the smaller, better-behaved population thoroughly and the larger one not at all.

PART TWO

Human Identity

5 · Strong Authentication, and What Attackers Do About It

MFA is the highest-leverage control you have. It is also the one attackers have spent the most effort learning to defeat, and not all of it is equal.

Reporting MFA as a single percentage hides the two things that matter: which authentication *paths* are covered, and which factor is in use. A firm at 98% MFA enrolment with a legacy protocol still accepting password-only authentication has a gap that enrolment numbers will never show, because the attacker will use the path you didn't count.

Factor	What defeats it
SMS / voice code	SIM swap, interception, and any real-time phishing proxy. Better than nothing, and should not be your target state.
App-generated code (TOTP)	Real-time phishing: the look-alike page collects the code and replays it within the window.
Push approval	MFA fatigue. Enough prompts at 2 a.m. and one gets approved. Number matching helps and does not eliminate it.
Phishing-resistant: passkeys, FIDO2 security keys, certificate-based	Bound to the origin, so a look-alike domain cannot use them. This is the target state, and where privileged accounts should already be.

The practical sequence is in ID-02: close the uncovered paths first, since a single uncovered legacy endpoint undoes everything else; then upgrade privileged accounts to phishing-resistant factors; then move the general population as the estate allows.

6 · Least Privilege Without Stopping the Business

Least privilege fails in practice for an entirely predictable reason: granting takes minutes and revoking takes a meeting.

Every organisation agrees with the principle. Very few have a mechanism that removes access at anything close to the rate it grants it, and the arithmetic of that asymmetry is what produces an environment where a mid-level employee can quietly touch half the company. Nobody decided that. It accumulated.

Three mechanisms that actually reduce privilege

Make access expire by default. Time-bound grants convert the problem from “someone must remember to remove this” to “someone must justify keeping it”. Same outcome, opposite default, and only one of them survives contact with a busy quarter.

Show usage at review time. A recertification that lists role names produces rubber-stamping. One that says “can read all customer records; last used fourteen months ago” produces removals. The data is almost always available and almost never shown.

Make removal the low-friction path. If retaining access requires a justification and removing it requires one click, the numbers move. If it is the other way round, your recertification is decoration and its removal rate will tell you so.

> **THE METRIC THAT EXPOSES THE THEATRE**
>
> Track the percentage of access *removed* at each recertification campaign. Near zero across a large campaign does not mean the access was all justified. It means nobody read it. That single number is the most honest measure of an identity programme I know.

7 · Joiner, Mover, Leaver

Identity isn’t a one-time grant, it’s a lifecycle. Get the plumbing right and it’s worth more than most of the shiny tools.

The three events are not equally handled, and the pattern is consistent everywhere. **Joiner** is usually solid, because someone is blocked and complaining until it works. **Mover** is usually broken, because nothing is blocked. **Leaver** is usually assumed, because nobody notices when it fails.

Mover is where the sediment forms

Someone moves to a new team and picks up the new access. Whether they keep the old is the entire question, and in most organisations the answer is yes, because removal was nobody’s task and the person is not complaining. Do that three times over five years and you have an employee whose access reflects every role they have ever held rather than the one they hold now.

The fix is unglamorous: treat a move as a leaver from the old role and a joiner to the new one, with removal as the default and retention requiring the new manager to actively ask for it. That is more work than most JML processes do today, and it is the single highest-value change available to a mature identity programme.

Leaver has to be same-day, and it has to include the machines

Access should vanish the day the person leaves. Two things make that hard in practice. First, the trigger is usually an HR record change that may lag reality by days. Second, the human accounts are the easy part: the leaver may also own service accounts, hold API keys, be the named contact on a vendor integration, and appear in a break-glass custody arrangement. ID-04 covers both halves, and Appendix C is the matrix.

8 · Standing Privilege and Break-Glass

The target for permanently-held administrative access is zero. The number you actually have is a metric worth reporting, and it is usually higher than anyone expects.

Standing privilege is the difference between a compromised administrator account and a compromised administrator account *that was already privileged at the moment of compromise*. Just-in-time elevation does not prevent the compromise. It shrinks the window in which the compromise is useful, and that is frequently the whole game.

What replaces it: elevation that is requested, justified, time-boxed, logged and approved by someone other than the requester. The friction is the point, and it should be tuned so that routine work does not require elevation at all — if administrators are elevating twenty times a day, the role definitions are wrong, not the process.

Break-glass, and the mistake everyone makes

You need a path that works when the identity provider is down. That path must not depend on the identity provider, which sounds obvious and is violated constantly, usually by an MFA requirement that cannot be satisfied when the directory is unavailable.

The other common mistake is reviewing break-glass use quarterly. Every use should be reviewed, individually, within a business day, against the stated justification. A break-glass account used without anyone noticing for eleven weeks is just an unmonitored administrator account with a dramatic name. ID-06 is the procedure.

PART THREE

Machine Identity

9 · The Population Nobody Owns

You cannot govern what you have never counted, and almost nobody has counted this.

Start with enumeration, not policy. A policy for non-human identities written before you know how many exist, where they live and what they can reach will be written against an imagined environment. ID-07 is the discovery procedure, and the first run of it is usually enough on its own to secure funding for everything that follows.

Where they hide

Location	What you find
Directory service accounts	The obvious ones. Frequently with passwords set years ago and never expiring, because expiry broke something once.
Cloud IAM	Roles, service principals, managed identities, and long-lived access keys attached to users nobody uses interactively.
SaaS integrations and OAuth grants	Consent granted by an individual, often years ago, frequently with broad scopes and no owner.
CI/CD systems	Deployment credentials, usually privileged, usually shared across pipelines.
Source code and config	Hard-coded keys. Scan history as well as current state; a secret removed from the current file is still in the history and still valid.
Scripts and scheduled jobs	The 3 a.m. batch job with stored credentials that nobody has touched in years and everybody is afraid of.

Every one needs an owner, and that is the hard part

The technical work in non-human identity governance is straightforward. The organisational work is finding a person willing to be named against a service account whose purpose nobody remembers. Expect a substantial tail of identities with no owner, and treat “unowned identity in production” as its own finding rather than folding it into a general count — because the remediation is different. An unowned identity cannot be rotated safely, cannot be recertified, and cannot be retired without someone accepting the risk of what breaks.

10 · Secrets and the 2 a.m. Integration

Rotation is the control everyone agrees with and nobody performs, for one honest reason: nobody knows what will break.

That fear is rational. A secret with unknown consumers cannot be rotated confidently, and a failed rotation at the wrong moment takes down a payment flow. The way through is not courage, it is instrumentation: establish who consumes a secret before you rotate it, and the fear goes away.

The sequence that makes rotation safe

Centralise before you rotate. A secret in a manager can be rotated. A secret in six config files cannot, because you will miss one. Migration into a secrets manager is the prerequisite, not an improvement to do later.

Instrument consumption. Log every retrieval, with the identity of the retriever, for a full business cycle including month-end. That log is your consumer list, and it is almost always longer than the documentation.

Support two valid secrets at once. Dual-secret rotation — issue the new one, let consumers pick it up, then revoke the old — turns rotation from a synchronised cutover into a background task. Systems that cannot do this are the ones that need a maintenance window, and knowing which those are is itself valuable.

Then rotate on a schedule, automatically. Manual rotation decays the moment the person who owned it moves on.

> **A NOTE ON THE ONES YOU CANNOT ROTATE**
>
> You will find secrets that genuinely cannot be rotated without unacceptable risk — an embedded credential in a device fleet, a partner integration with a contractual change process. Record them explicitly as accepted risk with a named owner and a review date. An enumerated list of un-rotatable secrets is a defensible position. Discovering them during an incident is not.

11 · Workload Identity: Killing the Static Key

The best rotation strategy is not needing one. A workload that authenticates with a short-lived, federated credential has no secret to leak.

This is where machine identity is genuinely improving. Cloud platforms can issue identity to a workload based on where it is running rather than on a secret it holds: federated tokens, managed identities, service accounts bound to a pod. The credential lives minutes, is scoped to the workload, and cannot be copied out and used elsewhere in any useful way.

The migration is mechanical and slow rather than difficult. Inventory the static keys, identify which workloads can federate, migrate in waves, and treat every remaining static key as a documented exception with an owner. ID-09 is the procedure. The end state most organisations can realistically reach is federated identity for anything running on their own platforms, and a small, known, owned set of static credentials for third-party integrations that cannot federate.

12 · Agents Are Joiners Too

An autonomous agent that calls tools is an identity with permissions. Everything in this manual applies to it, and almost nobody is applying it yet.

It needs its own identity, not a shared service account, so its actions are attributable. It needs permissions scoped to what it actually does rather than to what was convenient. It needs to appear in your inventory, get recertified, and have an owner. And it needs a leaver process for the day it is retired, which is the part nobody builds, because agents are new and nothing has been retired yet.

The one thing that is different: an agent’s instructions can arrive from content it reads. That makes prompt injection an identity problem rather than a content-filtering one, and the mitigation is the same as for any privileged actor — scope the credential tightly, put a human gate in front of irreversible actions, and never let the model’s output be the thing that authorises the action.

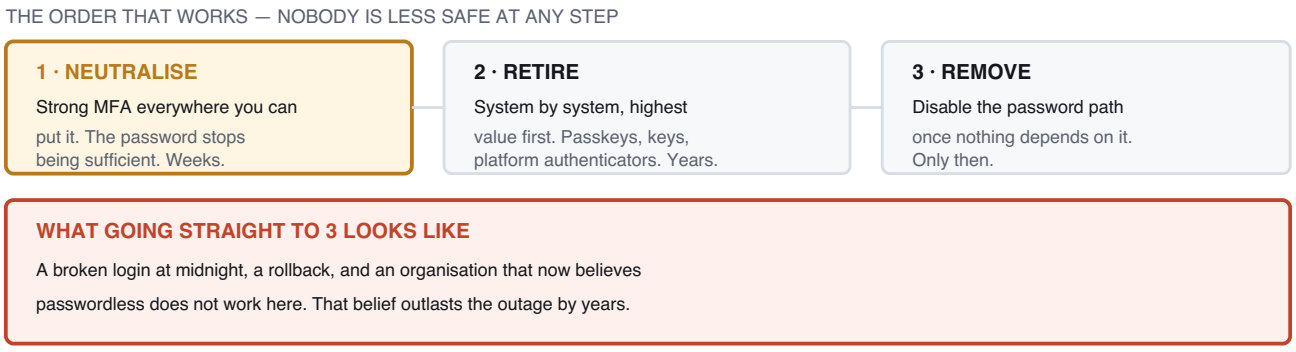
PART FOUR

Killing the Password

13 · Layer First, Remove Later

Killing the password is less a technology project and more a demolition job. Swing the wrecking ball too fast and you take out load-bearing walls.

Passwords are bad security: reused, guessable, phishable, endlessly leaked in breaches you will never hear about. Nobody defends them on the merits. The problem is not that people are attached to them, it is that they are wired into everything — every app, every legacy system, every integration, and two decades of muscle memory. “This is bad and we all hate it” and “we can remove it quickly” are not the same sentence.



> Step 1 removes most of the risk in weeks. Steps 2 and 3 take years and remove the rest. Doing them in the wrong order is how the programme dies.

Step one is the fast, boring, high-impact move and it is the one I always do first. Strong MFA everywhere you can put it means a stolen or guessed password on its own is worthless. That neutralises the vast majority of the risk passwords carry, and you can do it now, without waiting for a multi-year migration to finish.

Then you migrate the systems that can go passwordless, one at a time, starting with the highest-value targets. Admin accounts, crown-jewel systems, the things an attacker wants most. The password dies slowly, on purpose, as the safer thing takes over underneath it — and at no point in that migration is anyone less safe than they were the day before.

14 · The Walls That Hold Up the Roof

The reason the overnight approach fails isn't ambition. It's the load-bearing dependencies you didn't see.

The legacy application that can only speak password. The batch job that authenticates with stored credentials at 3 a.m. The partner integration, the field device, the service account buried in a script nobody has touched in years. Try to remove passwords everywhere at once and you will find these the hard way, when something critical breaks and you are rolling back at midnight.

A patient migration finds them one at a time, on your schedule, with a plan for each. The discovery work is the same either way — the only variable is whether you do it deliberately in advance or involuntarily during an outage.

Dependency	The realistic plan
Legacy app, password-only	Front it with an identity-aware proxy or federate it. Where neither is possible, isolate it, apply compensating controls, and put it on the application roadmap with a date.
Batch job with stored credentials	This is a machine identity problem, not a password one. Route it to ID-09 and federate it, or move the secret into a manager with rotation.
Partner integration	Contractual and slow. Start the conversation early; the technical change is usually the easy part.
Field or embedded devices	Frequently cannot change at all within the hardware lifecycle. Isolate, monitor, and record as accepted risk with a replacement date.
Shared or generic accounts	Usually a process problem wearing a technical costume. Resolve the process and the account disappears.

15 · The Recovery Path Is the Real Front Door

Here’s the trap even good passwordless projects fall into: they harden the front door and leave the back door wide open.

If your shiny new passkey system falls back to “forgot my key? here’s a password reset by email”, you have not killed the password. You have moved it to the account-recovery flow, which is exactly where attackers will now aim. Attackers do not attack the strong door. They attack the weak one you built next to it and forgot about.

The recovery and enrolment paths have to be as strong as the primary one, or the whole effort is theatre. That is uncomfortable, because recovery is where usability pressure is highest and where the helpdesk is measured on speed.

What a hardened recovery path looks like

Enrolment of a new authenticator requires an existing strong factor, or an in-person or video-verified identity check for genuine loss. Helpdesk-assisted resets require verification the caller cannot socially engineer, which in practice means something better than knowledge-based questions. Recovery events are alerted on, logged and reviewed, particularly for privileged accounts. And there is a deliberate, documented decision about what happens when someone genuinely loses their only factor — because if that path is not designed, it will be improvised by whoever is on the phone.

> **TEST THE BACK DOOR, NOT THE FRONT**
>
> Assess your recovery flow the way an attacker would: attempt to take over an account through it, with only publicly available information about the target. Appendix F is the assessment. Most organisations that have invested heavily in strong primary authentication discover their recovery path is the weakest link in the estate, and they discover it either this way or the other way.

PART FIVE

Runbooks

RUNBOOK ID-01

Identity Inventory

PURPOSE	Establish how many identities exist, human and machine, and who owns each. Everything else depends on this.
TRIGGER	Once to establish the baseline, then quarterly reconciliation.
OWNER	Identity team. Business owners confirm their own.
EVIDENCE	Inventory (Appendix A), unowned-identity list, reconciliation record.

1. **Enumerate from the authoritative sources, not from a single console.**
Directory, cloud IAM in every tenancy, SaaS platforms, CI/CD, database engines, network devices. Any source you skip is a population you have decided not to govern.

2. **Split the result into human and non-human at the outset.**
The two need different lifecycles and different reviews. If your directory does not distinguish them reliably, classify by naming pattern and interactive-login history, and correct as owners confirm.

3. **Reconcile human accounts against the HR record.**
Accounts with no matching active employee or contractor are the highest-priority finding in this runbook. Expect leavers, and expect the count to be higher than anyone predicted.

4. **Attach a named owner to every identity.**
An individual, not a team mailbox. Identities that end this step with no owner go on their own list and get treated as a finding in their own right, because an unowned identity cannot be rotated, recertified or retired safely.

5. **Record what each can reach, not just that it exists.**
Permissions in terms of what they permit. This is what makes the inventory useful during an incident rather than only at audit.

6. **Flag the ones with no recent authentication.**
Dormant identities are cheap to remove and expensive to explain after a breach. Ninety days with no use is a reasonable first threshold.

7. **Publish two numbers.**
Total identities, and the ratio of non-human to human. That ratio is the single most effective way to explain to leadership why machine identity needs funding.

RUNBOOK ID-02

MFA Rollout and Phishing-Resistance Upgrade

PURPOSE	Make a stolen password insufficient, then make the second factor unphishable.
TRIGGER	Programme. Re-run the path audit after any authentication change.
OWNER	Identity team, with application owners for legacy paths.
EVIDENCE	Authentication path audit, coverage by path, factor mix, exception register.

1. **Audit authentication paths, not user enrolment.**
Every way a credential can be presented: modern protocols, legacy protocols, VPN, mail clients, API endpoints, admin consoles, break-glass. Coverage is measured against this list, and the gaps are always the paths nobody counted.

2. **Close or block the paths that cannot carry MFA**
before celebrating enrolment numbers. A legacy protocol still accepting password-only authentication makes the enrolment percentage meaningless, because that is the path the attacker will use.

3. **Enrol privileged accounts on phishing-resistant factors first.**
Passkeys, FIDO2 keys, or certificate-based. Not SMS, and not push. These are the accounts worth an attacker's effort.

4. **Move the general population to the strongest factor the estate supports,**
in waves by business unit, with a support path staffed ahead of each wave.

5. **Retire the weak factors deliberately.**
SMS and voice remain in place at most organisations long after they were meant to be temporary. Set a date, communicate it, and enforce it.

6. **Turn on number matching and prompt suppression**
wherever push is still in use, and alert on repeated denied prompts — that pattern is an attack in progress, not a user with a flaky phone.

7. **Register every exception with an expiry.**
The system that cannot do MFA is a known risk with an owner and a date, or it is an unknown risk.

RUNBOOK ID-03

Access Recertification

PURPOSE	Remove access that is no longer needed, and produce a removal rate that proves the review was real.
TRIGGER	Privileged quarterly, standard annually, and on any manager change.
OWNER	Business owner decides. Identity executes removals.
EVIDENCE	Campaign record, decisions, removals executed and verified, removal rate.

1. **Pull entitlements from the systems, not from the request records.**
   What was granted and what exists have diverged, and the divergence is the interesting part.

2. **Translate entitlements into business language before showing anyone.**
   A reviewer presented with role names approves everything. A reviewer told “can approve payments up to £50,000” does not.

3. **Attach last-used data to every line.**
   This single addition moves removal rates more than any other change.

4. **Default to revoke.**
   Require justification to retain rather than to remove. Same review, opposite default, and only one of them survives a busy quarter.

5. **Give reviewers a bounded, prioritised set.**
   A manager sent four hundred lines will approve all of them in one sitting. Send the privileged and the dormant first.

6. **Execute removals inside a defined window and verify each one.**
   A decision to remove that is never executed is worse than no review, because the record now says the access is gone.

7. **Report the removal rate, by reviewer.**
   Near-zero means the campaign was decoration. Publishing it by reviewer changes behaviour faster than any policy.

RUNBOOK ID-04

Joiner, Mover, Leaver

PURPOSE	Keep access matched to what people need today, and remove it the day they leave.
TRIGGER	HR event, and a weekly reconciliation to catch events the trigger missed.
OWNER	Identity team. HR owns the trigger data quality.
EVIDENCE	Event log with elapsed time to completion, exception list, reconciliation results.

1. **Joiner: grant from a role definition, never by copying an existing user.**
   Copying is how one over-privileged account becomes forty. It is fast, it is common, and it is the origin of most privilege sprawl.

2. **Mover: treat it as a leaver from the old role and a joiner to the new one.**
   Old access is removed by default; retention requires the new manager to ask for it explicitly. This is the step most organisations skip and the one that prevents permission sediment.

3. **Leaver: disable immediately, then delete on a defined delay.**
   Immediate disable stops access. The delay preserves data and mailbox access for the business and keeps the audit trail intact.

4. **Do not stop at the human accounts.**
   The leaver may own service accounts, hold API keys, be the named contact on a vendor integration, hold break-glass custody, or be the sole owner of a scheduled job. Reassign every one of them before the account is deleted. Appendix C is the checklist.

5. **Reconcile weekly against HR, independently of the event trigger.**
   Triggers fail silently. A weekly diff between active accounts and active people catches what the workflow missed, and it is the control that actually holds.

6. **Handle contractors and non-employees explicitly.**
   They frequently sit outside the HR system entirely, which is why their access is the access most likely to outlive them. Give every one an expiry date at creation.

7. **Measure elapsed time from departure to access removal,**
   and report the tail rather than the average. The average is always reassuring; the tail is where the risk lives.

RUNBOOK ID-05

Standing Privilege Reduction

PURPOSE	Reduce permanently-held administrative access towards zero.
TRIGGER	Programme, then quarterly measurement.
OWNER	Identity team, with platform owners.
EVIDENCE	Standing privilege count by tier over time, elevation logs, approval records.

1. **Count what you have, by privilege tier.**
   Per Appendix D. Publishing the starting number is usually what secures the mandate for the rest.
2. **Separate administrative identities from daily-driver accounts.**
   Nobody reads email as an administrator. This is old advice and it remains widely unimplemented.
3. **Introduce just-in-time elevation for the highest tier first,**
   with request, justification, time-box, approval by someone other than the requester, and full logging.
4. **Tune the role definitions until routine work needs no elevation.**
   If administrators elevate twenty times a day, the roles are wrong and the process will be routed around. Watch the elevation rate as a design signal, not just an audit trail.
5. **Alert on elevation outside normal patterns**
   — unusual hours, unusual approver, unusual frequency.
6. **Work down the tiers,**
   and re-measure quarterly. This is a multi-year reduction, not a project with an end date.

RUNBOOK ID-06

Break-Glass Account Management

PURPOSE	Maintain an emergency access path that works when the identity provider does not, without creating a standing back door.
TRIGGER	Quarterly test; review after every single use.
OWNER	Identity team holds custody. Security reviews every use.
EVIDENCE	Custody record, retrieval alerts, per-use review sign-off, quarterly test result.

1. **Keep the number small and fixed,**
and document exactly what each account is for.

2. **Remove every dependency on the system it exists to recover.**
No federation, no dependency on the corporate directory, and an MFA method that still works when the directory is down. This is the requirement most commonly violated.

3. **Split custody.**
Credential and second factor held separately by different people. A break-glass account one person can use alone is an unmonitored administrator account with a dramatic name.

4. **Alarm on retrieval from custody, not on use.**
By the time it is used, the early warning is gone. Alert out of band, to people who will see it.

5. **Review every use individually, within a business day,**
against the stated justification. Not quarterly.

6. **Rotate immediately after any use**
and confirm the old credential no longer authenticates.

7. **Test quarterly, as a deliberate exercise.**
Untested break-glass fails on the day you need it: expired credentials, rotated MFA seed, a custodian who left. Ask when the last successful test was, and treat a vague answer as a failure.

RUNBOOK ID-07

Non-Human Identity Discovery

PURPOSE	Find the machine identities nobody has counted, and give each one an owner.
TRIGGER	Once to baseline, then continuous discovery.
OWNER	Identity team, with platform and application owners.
EVIDENCE	NHI register (Appendix B), unowned list, privilege distribution.

1. **Sweep every location, not just the directory.**
Directory service accounts, cloud IAM users and roles, service principals and managed identities, SaaS OAuth grants, CI/CD credentials, database logins, scheduled jobs, and secrets in source control.

2. **Scan repository history, not only the current tree.**
A key removed from the current file is still in the history and still valid until rotated. This is one of the highest-yield steps in the runbook.

3. **Record what each can reach.**
Expect to find machine identities with more standing privilege than any human, granted because it was expedient during a deployment.

4. **Attribute an owner to every one, and accept that this is the slow part.**
Trace through the consuming system, the pipeline that uses it, or the last person to modify the config.

5. **Treat “unowned identity in production” as a distinct finding class.**
It needs different handling: it cannot be rotated safely, cannot be recertified, and cannot be retired without someone accepting the risk of what breaks.

6. **Identify dormant machine identities**
— no authentication in ninety days. These are usually safe to disable and are the cheapest risk reduction available here. Disable before deleting, so you can reverse quickly.

7. **Make discovery continuous.**
A one-off inventory decays from the day it is produced. Alert on creation of new machine identities, particularly privileged ones.

RUNBOOK ID-08

Secret Rotation

PURPOSE	Rotate credentials safely, including the ones nobody dares touch.
TRIGGER	Scheduled by sensitivity; immediately on suspected exposure or a leaver who held it.
OWNER	Secret owner executes. Identity team owns the schedule.
EVIDENCE	Rotation record, consumer list, un-rotatable exceptions with accepted risk.

1. **Centralise into a secrets manager before attempting rotation.**
   A secret in six config files cannot be rotated reliably, because you will miss one. This is a prerequisite, not a later improvement.
2. **Instrument retrieval for a full business cycle,**
   including month-end. The retrieval log is your consumer list, and it is almost always longer than the documentation.
3. **Prefer dual-secret rotation.**
   Issue the new credential, let consumers pick it up, verify, then revoke the old. This turns a synchronised cutover into a background task.
4. **Where dual-secret is impossible, schedule a window and prepare rollback.**
   Knowing which systems these are is itself a useful output.
5. **Verify the old credential is dead.**
   Attempt authentication with it. Rotation that issues a new secret without revoking the old has doubled your exposure rather than removed it.
6. **Automate the schedule.**
   Manual rotation decays the moment its owner changes role.
7. **Record what genuinely cannot be rotated**
   as accepted risk, with a named owner and a review date. An enumerated list of un-rotatable secrets is defensible. Discovering them during an incident is not.

RUNBOOK ID-09

Workload Identity Migration

PURPOSE	Replace static long-lived credentials with short-lived federated identity, so there is no secret to leak.
TRIGGER	Programme, and for every new workload by default.
OWNER	Platform engineering, with application owners.
EVIDENCE	Static credential count over time, migration waves, remaining exceptions with owners.

1. **Inventory static credentials from ID-07**
   and sort by privilege and blast radius, not by ease of migration.
2. **Establish the federation pattern for each platform**
   once, as a paved path, so application teams consume it rather than designing it.
3. **Block new static credentials by policy first.**
   Stopping the inflow before draining the pool is what makes the programme finish. Without it you migrate at the same rate new keys are created.
4. **Migrate in waves, highest privilege first,**
   running old and new in parallel where the platform allows, then removing the static key.
5. **Verify by revoking the static credential and confirming the workload still functions.**
   A key left in place “just in case” has not been migrated; it has been duplicated.
6. **Scope the federated identity tightly.**
   Federation without scoping trades a leakable credential for an over-privileged one.
7. **Record the residue.**
   Third-party integrations that cannot federate stay static: register them, own them, rotate them under ID-08, and report the count.

RUNBOOK ID-10

Passwordless Migration by System

PURPOSE	Retire passwords system by system without anyone being less safe than the day before.
TRIGGER	Programme, after ID-02 has established MFA coverage.
OWNER	Identity team, with application owners per wave.
EVIDENCE	Wave plan (Appendix E), dependency register, per-system completion record.

1. **Confirm MFA coverage first.**
   Passwordless before broad MFA is the wrong order: it delivers less risk reduction, much later. Neutralise, then retire.

2. **Inventory the load-bearing dependencies before touching anything.**
   Legacy applications, batch jobs with stored credentials, partner integrations, embedded devices, shared accounts. Chapter 14 has the plan for each type. Finding these deliberately is the entire difference between a migration and an outage.

3. **Sequence by value to an attacker, not by ease.**
   Administrative accounts and crown-jewel systems first, because that is where passwordless buys the most.

4. **Run each wave with the password path still available,**
   until enrolment is confirmed complete for that population. Overlap is what keeps the safety property true at every step.

5. **Harden recovery before disabling the password, not after.**
   Per ID-11. Disabling the password while recovery still resets by email moves the weakness rather than removing it.

6. **Disable the password path only when nothing depends on it,**
   and verify by attempting password authentication.

7. **Track the residue honestly.**
   Systems that will keep passwords for years belong on a register with compensating controls and a review date, not in a footnote.

RUNBOOK ID-11

Account Recovery Hardening

PURPOSE	Ensure the recovery path is not the weakest way into an account.
TRIGGER	Before any passwordless wave; annually thereafter.
OWNER	Identity team, with the service desk.
EVIDENCE	Recovery path assessment (Appendix F), attempted-takeover test result, alerting configuration.

1. **Map every route to regaining access.**
   Self-service reset, service desk, manager attestation, backup codes, secondary email, secondary phone, and any vendor-side support path. The vendor path is the one most often forgotten and least controlled.

2. **Rate each route against the primary authentication it bypasses.**
   Any route weaker than the primary is your real authentication strength for that account.

3. **Require an existing strong factor to enrol a new one**
   wherever possible.

4. **Replace knowledge-based verification at the service desk.**
   Anything an attacker can research or phish is not verification. Use a callback to a known device, an in-person or video check, or manager attestation through a separate channel.

5. **Design the genuine-total-loss path deliberately**
   — higher assurance, slower, and documented. If it is not designed, it will be improvised by whoever is on the phone, under pressure, at the worst moment.

6. **Alert on every recovery event for privileged accounts,**
   and review them.

7. **Test it adversarially.**
   Attempt an account takeover through the recovery flow using only publicly available information about the target. This is the step that finds the gap, and organisations that skip it tend to find the gap the other way.

RUNBOOK ID-12

Credential Compromise Response

PURPOSE	Respond to a compromised credential, human or machine, without leaving the attacker a way back.
TRIGGER	Suspected or confirmed credential compromise, or exposure in a public repository or breach dump.
OWNER	Incident commander, with the identity team.
EVIDENCE	Timeline, actions taken, session invalidation confirmation, scope assessment.

1. **Invalidate sessions, not just the credential.**
   Disabling an account does not terminate tokens already issued. This is the single most commonly missed step, and it is why attackers persist through a password reset.
2. **Revoke refresh tokens, application passwords and API keys**
   associated with the identity. Long-lived tokens survive a password change by design.
3. **Establish what the identity could reach,**
   from the inventory. If ID-01 was done, this is a lookup. If it was not, this is the slowest hour of the incident.
4. **Review everything the identity did across the full plausible window,**
   not just since detection. Assume the compromise predates the alert.
5. **Check for persistence created by the identity.**
   New MFA methods enrolled, mail forwarding rules, OAuth grants issued, new service accounts, changed recovery details, added group memberships. Attackers establish a second way in before using the first.
6. **Re-enrol authentication from a trusted path,**
   and confirm any attacker-registered factor is removed.
7. **For machine identities, rotate and confirm the old credential is dead,**
   then find how it leaked — repository, log, config, or a third party.
8. **Feed the root cause back into the programme.**
   A phished credential is an ID-02 finding. A leaked key is an ID-09 finding. An account belonging to a leaver is an ID-04 finding.

PART SIX

Templates and Reference

A · Identity Inventory Schema

Field	Notes
Identifier	The account or principal name as it exists in the source system.
Type	Human / non-human. Drives which lifecycle applies.
Source system	Directory, cloud tenancy, SaaS platform, CI/CD, database.
Owner	Named individual. Blank is a finding, not a gap.
Purpose	One sentence. If nobody can write it, that is itself the answer.
Privilege tier	Per Appendix D.
What it can reach	In terms of what the permissions permit, not role names.
Authentication method	Factor type for humans; credential type for machines.
Last authentication	Drives dormancy review.
Last recertified	
HR match	Human accounts only. No match = priority finding.
Expiry	Mandatory for contractors and machine identities.

B · Non-Human Identity Register

Field	Notes
Identity	
Consuming system	What actually uses it. Often the hardest field to complete.
Credential type	Federated / short-lived / static key / password. Static is a migration candidate under ID-09.
Secret location	Secrets manager path, or the honest answer: config file, pipeline variable, source code.
Rotation	Automated / manual / not possible. “Not possible” requires accepted risk and a review date.
Last rotated	
Consumers	From retrieval logging. Populate before rotating.
Privilege tier	Machine identities frequently exceed human tiers. Record it honestly.
Owner	Named individual.
Created / expires	Machine identities should expire too.
Retirement plan	What happens when the consuming system is decommissioned.

C · Joiner-Mover-Leaver Matrix

Item	Joiner	Mover	Leaver
Directory account	Create from role	Retain	Disable same day
Role entitlements	Grant from definition	**Remove old, add new**	Remove all
Privileged access	Only if role requires	Re-justify, do not carry	Remove immediately
MFA enrolment	Before first access	Retain	Deregister factors
SaaS applications	Provision via SSO	Re-evaluate each	Deprovision, incl. non-SSO
Owned service accounts	—	Reassign ownership	**Reassign before deletion**
Held secrets / API keys	—	Review	**Rotate** — they had the value
Vendor contact roles	—	Reassign	Reassign and notify vendor
Break-glass custody	—	Reassign	Reassign and rotate
Devices and certificates	Issue	Retain	Recover and revoke

D · Privilege Tiering

Tier	Definition	Treatment
Tier 0	Control of the identity system itself, or of the platform hosting everything.	Phishing-resistant MFA mandatory. Zero standing privilege. Dedicated admin identity. Every action logged and reviewed. Separate workstation where feasible.
Tier 1	Administrative control of critical business systems or bulk sensitive data.	Phishing-resistant MFA. Just-in-time elevation. Quarterly recertification. Dedicated admin identity.
Tier 2	Elevated access within one application or dataset.	Strong MFA. Quarterly recertification. Elevation preferred.
Tier 3	Standard business access.	MFA. Annual recertification.

E · Passwordless Migration Wave Plan

Wave	Population	Prerequisites and notes
0	MFA coverage across all authentication paths	Not a passwordless wave. The prerequisite that makes the rest safe, and where most of the risk reduction happens.
1	Tier 0 administrators	Recovery hardened first (ID-11). Small population, highest value, most attacker interest.
2	Tier 1 administrators and crown-jewel system users	Dependency register complete for those systems.
3	General population, by business unit	Support staffed ahead of each unit. Password path retained until enrolment confirmed.
4	Disable password authentication where nothing depends on it	Verified by attempting password authentication, not by configuration review.
Residue	Systems that keep passwords	Registered with compensating controls, owner and review date. Expect this list to be non-empty for years.

F · Recovery Path Assessment

One row per route back into an account. Any route weaker than the primary authentication is your real authentication strength.

Route	Verification required	Strength vs primary	Action
Self-service reset			
Service desk assisted			Knowledge-based questions are not verification
Manager attestation			Confirm via a separate channel
Backup codes			Where are they stored?
Secondary email / phone			Inherits that account's security, not yours
Vendor support path			Outside your control. Ask what their process is
Genuine total loss			Must be designed, or it will be improvised

G · Board Reporting: Identity

Measure	Current	Commentary prompt
Non-human to human identity ratio		Which population gets governance, and which is larger?
Authentication paths without MFA		Not enrolment percentage. Uncovered paths.
Tier 0 / Tier 1 accounts with standing privilege		Target is zero. What is the trajectory?
Access removed at last recertification (%)		Near zero means nobody read it.
Days from departure to access removal (tail, not average)		The tail is where the risk lives.
Machine identities with no owner		Cannot be rotated, recertified or retired.
Static long-lived credentials remaining		Is the inflow blocked, or are we bailing?

H · Where This Gets Hard

The calls I have not made for you, because they depend on your estate rather than on principle.

How hard do you push phishing-resistant MFA? Passkeys and hardware keys are the right answer and they are not free: hardware costs money, enrolment costs support time, and a workforce without company devices makes platform authenticators awkward. Every organisation should have Tier 0 and Tier 1 on phishing-resistant factors. Beyond that, where you draw the line is a budget and workforce question, and anyone who tells you it is purely a security question has not run the rollout.

What do you do about the legacy application that cannot change? Chapter 14 says isolate, compensate and put it on a roadmap. Sometimes the roadmap is five years and the application runs a revenue line. That is a real accepted risk, not a failure, and it should be written down as one with a named owner rather than quietly carried by the identity team.

How aggressive should dormant-identity cleanup be? Disabling a dormant machine identity is cheap risk reduction right up until it is the quarterly job that only runs in March. Ninety days is a reasonable threshold; disable before delete, and keep the reversal fast. Your tolerance depends on how well you can reverse.

Do you separate administrative identities in a small organisation? The advice is unambiguous and the practicality is not, when the same six people do everything. The compromise I would defend is separation for Tier 0 only, with the daily account genuinely unprivileged, rather than a four-tier model nobody follows.

When is the recovery path strong enough? There is a point where hardening recovery locks out real users who genuinely lost their phone on holiday. The right answer differs for an internal workforce and a consumer base, and it is a product decision as much as a security one. What is not defensible is not knowing where your line currently sits.

I · Glossary

Break-glass	Emergency access that works when the identity provider does not. Must not depend on the system it recovers.
Entitlement	A specific permission held by an identity, as distinct from the role that granted it.
Federated credential	Short-lived identity issued based on where a workload runs rather than on a secret it holds. Nothing to leak.
JML	Joiner, mover, leaver. Mover is the one usually broken, because nothing is blocked when it fails.
MFA fatigue	Repeated push prompts until one is approved. Number matching reduces it; it does not eliminate it.
NHI	Non-human identity. Service accounts, API keys, tokens, workload identities. Usually the larger population.
Phishing-resistant	An authenticator bound to the origin, so a look-alike site cannot use it. Passkeys, FIDO2, certificate-based.
Standing privilege	Administrative access held permanently rather than elevated on demand. Target is zero.
Tier 0	Control of the identity system itself. Compromise here is compromise everywhere.

USING THIS MANUAL

Take it apart. Rename the runbooks, renumber the tiers, delete what does not apply, and argue with Appendix H in particular — that is where your estate should produce different answers than mine would.